MOHAMED ABDULLAHI MOHAUD

WEB BASED REPOSITORY SYSTEM FOR GRADUATION PROJECTS

NEU 2021

WEB BASED REPOSITORY SYSTEM FOR GRADUATION PROJECTS

A THESIS SUBMITTED TO THE GRADUATE SCHOOL OF APPLIED SCIENCES OF NEAR EAST UNIVERSITY

By
MOHAMED ABDULLAHI MOHAUD SHURIE

In Partial Fulfilment of the Requirements for the Degree of Master of Science In Software Engineering

NICOSIA, 2021

A THESIS SUBMITTED TO THE GRADUATE SCHOOL OF APPLIED SCIENCES OF NEAR EAST UNIVERSITY

By
MOHAMED ABDULLAHI MOHAUD SHURIE

In Partial Fulfilment of the Requirements for the Degree of Master of Science In Software Engineering

NICOSIA, 2021

Mohamed Abdullahi Mohamud SHURIE: WEB BASED REPOSITORY SYSTEM FOR GRADUATION PROJECTS.

Approval of Director of Graduate School of Applied sciences

Assist. Prof. Dr. Nadire ÇAVUŞ

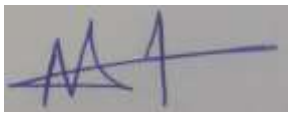
We certify this thesis is satisfactory for the award of the degree of Masters of Science in Software Engineering

Examining Committee in Charge:

Assist. Prof. Dr. Yöney Kırsal EVER	Committee Chairman, Department of Software Engineering, NEU
Assist. Prof. Dr. Ümit İLHAN	Department of Computer Engineering, NEU
Assist. Prof. Dr. Kaan Uyar	Supervisor, Department of Computer Engineering, NEU

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: Mohamed Abdullahi
Signature:
Date: 09/02/2021



ACKNOWLEDGEMENTS

First and foremost, I would like to thank Allah s.w.t for giving me the strength and patience to complete this mission.

I would like to express my deep and sincere gratitude to Assist. Prof. Dr. Kaan UYAR for the expertise and intelligence he has displayed while supervising this project. I declare this great work is a result of his great guidance and support.

I am extremely gratitude to my advisor Assoc. Prof. Dr. Boran ŞEKEROĞLU for his recommendations and suggestions. Without his suggestions I would not have successfully completed my study with this short time.

Above all, I would like to thank my family members and my parents for their unwavering support in helping me with this research. Without their assistance, I would not have successfully completed this work done on time.

I would also like to thank my friends and all the individuals who have been involved in helping me directly or indirectly. Thanks for your support.

Lastly, I would like to express my sincere gratitude to the lecturers in our faculty for their great job done during the study period of our course.

I love you all.

To my mother…

ABSTRACT

The aim of this study is to develop a web-based repository system for graduation projects of Jamhuriya University of Science & Technology, Somalia. The developed project may increase the ability of the involved parties to manage their graduation projects via designed website, and enable them to arrange their tasks with minimum effort. The website may help student supervisors to easily keep track of their students, performance, and give them feedback through a mailing system (mailing system is implemented in our website to ensure that each and every one to contact each other). The examiners have a permission to monitor all tasks of their assigned students and ultimately give grades.

In this work, ASP.net is used together with VB.net as a programming language and MS-SQL Server was used as a database management system. The developed system may help to the visitors of the website who has no username to access the system, to search graduation project titles by department, or by academic year, or both of them. The visitors can also view the proposal of the projects.

Another motive of this research is to provide a mean for protecting saved information, and archiving them. The website may increase the availability of the project's information anytime anywhere.

Keywords: Web-based Repository System, Graduation Projects; Projects; Management; Repository

ÖZET

Bu çalışmanın amacı, Somali Jamhuriya Bilim ve Teknoloji Üniversitesi bitirme projeleri için web tabanlı bir depo sistemi geliştirmektir. Geliştirilen proje, ilgili tarafların bitirme projelerini tasarlanan web sitesi üzerinden yönetme becerilerini artırabilir ve görevlerini minimum çabayla düzenlemelerini sağlayabilir. Tasarlanan web sitesi, öğrenci danışmanlarının öğrencilerini, performanslarını kolayca takip etmelerine ve bir posta sistemi aracılığıyla onlara geri bildirimde bulunmalarına yardımcı olabilir (web sitemizde her birinin birbiriyle iletişim kurmasını sağlamak için posta sistemi uygulanmaktadır). Sınav görevlilerinin, atanmış öğrencilerinin tüm görevlerini izleme ve nihayetinde not verme izni vardır.

Bu çalışmada programlama dili olarak VB.net ile birlikte ASP.net, veritabanı yönetim sistemi olarakta MS-SQL Server kullanılmıştır. Geliştirilen sistem, kullanıcı adı olmayan web sitesi ziyaretçilerinin sisteme erişmesine, mezuniyet proje başlıklarını bölüm, akademik yıl veya her ikisine göre aramalarına yardımcı olabilir. Ziyaretçiler ayrıca proje önerilerini de görebilirler.

Bu araştırmanın bir başka amacı da, kaydedilen bilgilerin korunması ve arşivlenmesi için bir yol sağlamaktır. Web sitesi, proje bilgilerinin her an her yerde kullanılabilirliğini artırabilir.

Anahtar Kelimeler: Web Tabanlı Arşiv Sistemi, Bitirme Projeleri; Projeler; Yönetim; Depo

TABLE OF CONTENTS

CHAPTER 3: METHODOLOGY

CHAPTER 4: SYSTEM ANALYSIS AND DESIGN

CHAPTER 5: IMPLEMENTATION AND TESTING

CHAPTER 6: CONCLUSION & RECOMMENDATIONS

LIST OF FIGURES

LIST OF ABBREVIATIONS

IS	Information system
CIO	chief information officer
TPS	Transaction Processing System
DSS	Decision Support System
MIS	Management Information System
WBE	Web-based education
E-INQUIRY	Electronic Inquiry
E-ADMISSION	Electronic Admission
E-ENROLLMENT	Electronic Enrollment
E-COURSE	Electronic Course
E-GRADUATION MANAGEMENT	Electronic Graduation management
ASP	Active server page
WWW	World Wide Web
MS SQL Server	Microsoft SQL server
PC	Personal computer
HDD	Hard disk
GB	Gigabyte
RAM	Random access memory
GHz	Gigahertz
VB.NET	Visual Basic .net
HTTPS	Secured hyper-text transfer protocol
HTTP	hyper-text transfer protocol
SQL	Structured Query Language
UML	Unified Modeling Language

CHAPTER 1

INTRODUCTION

Jamhuriya University of Science & Technology is a higher educational institution in Somalia that established in 2012. The university has six registered faculties. Every faculty works with a manual system for managing final year projects.

1.1. Problem Statement of the Thesis

The information about graduation projects currently records into spreadsheet. Because of several issues, this information has been lost by the university. To avoid this problem, the faculties pinpointed the need for a computerized management system to manage the students' graduation projects.

In this work the Graduation Project System process analyzed in detail, and the following problems identified: time-consuming, manual tracking of due projects, lack of centralized storage and backup, and poor searching and finding capabilities.

The main aim of this work is to develop an online repository system for graduation projects management for Jamhuriya University.

Moreover, the website will improve the capacity of concerned users to deal and arrange their tasks with least effort, allow supervisors to effortlessly monitor graduates and their tasks, and eventually enable graduation projects panel individuals to screen all tasks occurring to guarantee that everything is each assignment is most likely handled.

The web-based repository System may help Jamhuriya University staff easily recognize about the projects that are already taken to avoid plagiaristic acts of trying to present a project already implemented by other groups, and also as a guide for other people, who searches for a title for his/her graduation project from the future work part in which the researcher recommends a few things identified with the subject covered which deserve further research as well as expected improvements to the proposed functionalities of the system.

1.2. Project Objectives

1) The main objective of automation in any field is to make the University's job of managing graduation projects easier and available anytime anywhere.
2) The computer can compute, store, and retrieve the data quickly and effectively. It is an effective tool to solve the problems of manual entry and computation.
3) Furthermore, the System will automate the project tasks automatically and efficiently.
4) The system will give the student a chatting area with his/her supervisor.
5) The system will help university administration to detect any plagiaristic acts of trying to present a project already implemented by other groups.
6) It will provide information about the Graduation Project Titles.
7) The system saves and retrieves the information about the student marks.
8) The failed students can be known easily from the report.
9) The system will keep track marks of students.
10) The system will allow printing facilities.

1.3. Project Scope

This study is, in terms of geography, limited to Mogadishu-Somalia and will be focused on Jamhuriya University of Science and Technology. The project will record transactions provided for storing information of students, Supervisors, Faculties, the marks, generate various reports, and help managers in their decision process.

1.4. Significance of the Project

This project will improve all necessary information related to graduation project management. The significance of this project is to save the universities' resources by improving the efficiency of the decision-making process. The project will also be useful for the students for getting easily accurate information about graduation projects.

1.5. Current System

The current system used by the academic staff is Microsoft Excel as its backbone for entering, editing, and storing graduation project data. After we have analyzed the system,

we found that there are many limitations. These limitations were detected in different phases of the system like data gathering, editing, and searching for the information.
In this work we found that the system needs a greater re-engineering and redesign to meet the required standards of today's technologies. Because of the importance of the transaction processing system to the whole system, the work gave a considerable effort to zone those limitations to resolve them and to increase the efficiency and consistency of the system.

1.6. Data Gathering

There are several techniques to make data gathering such as interviews, observations, questionnaires, surveys, and so on. This research uses mainly the first two techniques.

1.7. Research Question

How can we solve these problems? How can we re-engineer and redesign this system to meet the required standards of today's technologies?

How we can help Jamhuriya University to automate the tasks associated with the management of the Graduation projects?

The answer is to develop a web-based repository system to manage and automate the graduation project tasks.

1.8. Organization of the Study

The remaining chapters of this study will be coordinated as follows:

- Chapter 2 gives a general idea of the document management systems. It contains information related to different types of document management systems and some related things to this topic.
- Chapter 3 describes the methodology used in this study. It defines the system hardware and software requirements of the website development, the research question answer, the methods chosen to use, and the system architecture.

- Chapter 4 gives the analysis and design of the proposed system; it analyzes the current system and its security needs. The chapter finally describes the important system components and describes the functions of each and every component.
- Chapter 5 which is the implementation and testing of new system, it first states the implementation, the test environment and set up of the lab. The chapter researches the details of the main functions of online we based repository system.
- Chapter 6 contains the conclusion and future work part in which the researcher recommends a few areas related to the study covered which needs further researcher as well as possible upgrades to the proposed system functionalities.

CHAPTER 2
DOCUMENT MANAGEMENT SYSTEM

2.1. Introduction

Universities and schools continue to make incremental steps into the digital era, but a significant part of what they manage is still in hard-copy form. For higher education institutions, it's critical that they must be able to arrange their information storage for easier, more effective access, information which can include: personal information records of students, academic records, graduation projects' information, and administrative information on housing, classes, instructors, security and other services.

There is a great possibility that in the recent past, they have used a desktop program. Right now, they could even have one on their computer. There are programs like Adobe Photoshop, Microsoft Word, and Notepad. Almost anywhere, you may find them offering almost any form of possible usage to help users obtain the results they want while operating.

Another choice has also started to emerge for computer users over the last two decades. They are web applications in which users communicate having a browser on their desktop. They work like a desktop program, but people do not need to download and install software to use the program into their computer. They are used with the web browser on the server. (exoft, 2017).

2.1.1. Paper-based document management system

Think of all the important records that serve as every universities' lifeblood. A few that come to mind are accounting accounts, invoices, non-disclosure contracts, and graduation project documents.

Now, think about how to treat these important papers. Do you store them in a cabinet for filing? Are some of them crawling all over the desk? Do the documents have to be manually transported and signed by hand?

As you can see, the use of a paper-based document processing framework has quite a few disadvantages. According to Teach ICT (2018), here are some of the problems that every paper-based document management system has:

- Limited storage capacity.
- Searching for all the documents manually will consume time and effort.
- It is impossible to retrieve at a certain criterion; any document would have to be looked at manually.
- The data is very difficult to analyze.
- It is difficult to sort data according to more than one criterion.
- Modifications have to be performed manually. If scribbled out, documents can look dirty.
- Locking up the documents would be the only security.
- It's impossible to make a backup since it will be important to re-write or photocopy any page/card. This means it requires more storage space.

2.1.2. Computerized document management system

There are two types of computerized system when managing documents which are: Desktop application and Web management systems.

2.1.2.1. Desktop applications

The Desktop Application is a software application that can be deployed independently and installed on a computer (laptop or desktop). Via any data storage facility, you can install one or download an installation from the Internet. The desktop app acts as stand-alone software, which means that it can be used offline and does not require Internet or web browser access.

Desktop apps provide more security and do not need support from third parties for backups. In contrast to the benefits, it also has drawbacks that are often not portable and limited to one place, which is your computer, requiring manual installation of the user as well as somewhere on the hard drive, and sometimes space on your hard drive may not be

appropriate for an app, and improving this issue may cause inconvenience (Brandon Gaille, 2019).

2.1.2.2. Web document management systems

On the contrary, web document management system or simply web application is "a software program that is placed on a remote server and the users use a web browser to run it over the Internet". Web applications ask about the content server and create web documents automatically for their users. Since these applications are accessed through web browsers, the format of these web documents is standard in order for all browsers to provide support. Web browser is apparently important here-it interprets and runs all the scripts as well as displays the look and feel of application (exoft, 2017).

Web-based software actually do not need any process of installation, and are often cross-platform, so they can be deployed on multiple platforms of computing. Web applications are quickly portable and available globally (Brandon Gaille, 2019).

In general, web applications encounter more security threats than those on the desktop. Everyone recognizes that the Internet is not the safest place on earth. So, if all web application data is stored in the cloud, you are at risk of a breach of security.

2.2. Information Systems

The term of (IS) is "any organized combination of people, computers, apps, communication networks, data resources, and processes that input, store, query, transform, and generate information in an organization". (O'Brien, and Marakas, 2007).

People benefit from information systems to interconnect with each other using a different type of equipment, guidelines and policies, communication lines and warehoused data.

Nowduri and Al-Dossary (2012) said that it is a group of organized elements that work together to perform actions for input, preparation, data storage, processed data and validate as a way to convert data into a useful information which will be used in an organization to help prediction, preparation, monitoring, decision-making process and organizational

events. In this era, any commercial enterprise requires an information system (IS) to keep records of all functional departments.

Technically the term Information system is “a collection of interconnected elements that obtain process, keep and spread information in an organization to make administrative decision, communication and control easier”.

It can also assist administrators and employees to evaluate and assess problems, imagine complex concepts and develop new things, with the exception of supporting administrative decision-making, management and control. (Laudon, K., and Laudon, J., 2006)

It can also concentrate on the distribution of information about key persons, locations, and things inside or in the world around the organization. When we are saying “information” it is not the raw data, it is the “data that has been shaped into a type that is valuable and helpful to persons”.

In opposition to, “data are streams of raw facts describing events that exist in organizations before they have been converted into a shape that every person can understand”.

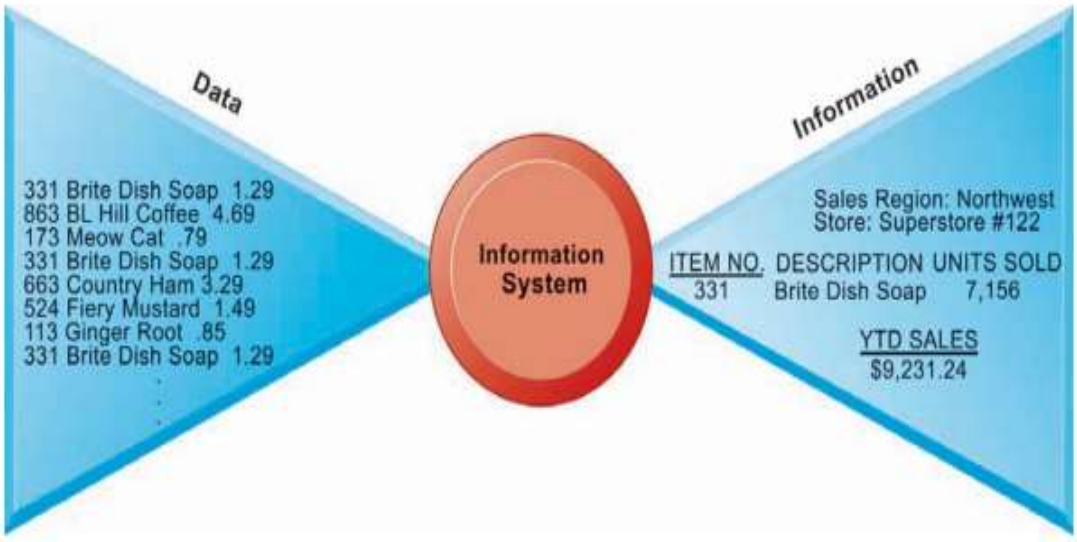


Figure 2.1: Data and information (Laudon, and Laudon, 2006)

An “Information System” is an application which processes and generates information from data (facts). The term process is described as a processing cycle of information. The processing stage for information comprises of 4 functions. Input, procedure, outcome, and storage. (Khanore, et al, 2011).

2.2.1. Information systems function in a company

The department of Information Systems is the institutional corporate entity responsible for services of information technology. It is responsible for managing the hardware, applicatio ns, storage of information and networks that compose the IT infrastructure of the organizati on. The department is made up of experts, such as programmers, system analysts, project leaders, and administrators of information systems, and are also led by a chief information officer (CIO).

2.2.2. Types of information system

We talked about the information systems (IS) definition and it is role in the institution. Now we are going to deep into the different types of information system.

According to geeksforgeeks (2019), the following are the categories of IS:

a) Transaction Processing System (TPS):

TPS is a database system which processes information produced from institutional transaction events.

The main objectives of this type are to include transactions, refresh records and generate reports, such as billing software, and payroll application.

b) Decision Support System (DSS):

DSS is an interactive information system that offers tools to assist in semi-structured and non - structured decision-making using information, prototypes and manipulation of data.

The DSS provides tools and strategies to support in collecting related information and evaluating solutions and alternatives, such as financial planning software, Bank loan management programs.

c) Experts System:

This type provide experience to help administrators identify issues or fix problems. The ideals of artificial intelligence science are the foundation of these programs.

Expert Systems is a scientific framework for intelligence. It practices its expertise of a specific form to serve as a professional user advisor.

The elements of an expert system are the knowledge base and software modules. These modules include knowledge inference and answers to the query of a customer.

d) Management Information System (MIS):

The MIS is the implementation of business systems and procedures. It's nothing more than file structures and file processing for a programmer. Yet, it does require much more complexity.

2.3. Management Information System

The 'MIS' is a programmed collection system for storing and distributing in the procedure of the information required to be passed the management functions (Tutorials point, 2014).

A more complete and focused definition is provided by the three components of MIS, where integration and full perspective are suggested by the system, information stands for Data that has been analyzed and interpreted, and management is the final user.

Management term refers to the planning, monitoring, and implementation of the functions. The executives manage planning; the middle management works on control; and the administration concerns the employee management.

The data that is processed which supports the management in planning, guidance and processes is called information. In contrast, data is the facts that arise out of the concern's

processes. We process the data, record it, summarize it, and finally submit it to the management in a report format.

Using a system, data is processed into a useful information. A system is composed of different parts of input, processing, output and comment.

This type of system implies a data processing system to provide the management with proper information to perform its functions.

2.3.1. Objectives of MIS

The objectives of MIS are to effectively manage the organization and to identify the scope for competitive advantage of the information system, to apply the organizational structure and functions of the industry.

According to tutorials point (2014), the following are some of the basic objectives:

- Capturing Data: collecting appropriate data, or usable information which take part in decision-making from many inside sources and outside sources of the company.

- Data Processing: Collected data is processed into useful information which is required at strategic, tactical and operational levels to schedule, organize, coordinate, direct and monitor functionalities.

- Information Storage: Information must be stored in a centralized database for future use.

- Information Searching: when the user needs to search and retrieve information, the system must be able to do.

- Information spread: the processed data or the information should be distributed to the users constantly.

2.3.2. Characteristics of computerized MIS

A well-designed and developed computerized MIS characteristics are below:

- Process data correctly and with less time.
- Should obtain, organize, operate, and update huge quantity of raw data of both similar and unconnected nature, driving from inside and outside sources at different times.
- Providing actual information on daily tasks without any delay.
- Providing structured and appropriate information.
- Extreme flexibility in data storage and retrieval.

2.4. Web-based Education (WBE)

According to IDC (2001), Web-based education (WBE) is rising exponentially, and the demand is estimated to hit nearly 28.6 billion by 2006.

The advances in technology and student preferences have demanded a change from a “synchronous environment” to a “click and learn” environment. Since the learners expect connectivity everywhere, anywhere, and universities are forced to give students knowledge.

The teacher’s role changes from "lecturing" to "facilitating," and the role of the scholar changes from "recipient" to "participant." These online scholars need online accessibility from enrollment to graduation, opt for a "full" online education.

Scholars are going to be more like clients, and schools are going be more like corporations competing for these clients. A lot of profit oriented and conventional colleges are seeking to be among the first to deliver web-based education. By the end of this year, it is expected that almost 200 online courses of some type or another may be delivered by universities (IDC, 2001).

Since students are like clients, their happiness is essential, and with online convenience, they demand online education. This is understood by colleges and their offerings are distinguished by revamping syllabuses, providing all the time online support facilities. Efficiency and effectiveness are going to be crucial to existence, raising obstacles to provide smooth processes for staff, faculty, and support workers.

However, according to Aggarwal (2001). There are many actors, "students, faculty, and technical personnel perform a major role in WBE".

These major players are actively engaged in the daily tasks and development of the system. The stakeholders expect form WBE operations should help the whole process, from student registering to graduation. The mechanism is not only the difference between distance learning and face-to-face education, but the style, nature or situation, and managing of the distribution method.

2.4.1. Steps of web-based in education

The Common steps in a complete WBE are:

2.4.1.1. E-Inquiry

Before settling on the actual college for their studies, the E-Inquiry process consists of tasks that prospective students undertake. As students are beyond the domain of the academic community, having a useful and exciting website that provides its mission, academic activities, support facilities, and general knowledge is the smartest thing a university can do. It should provide a powerful and robust search engine and above all, contact information for email and voicemail.

2.4.1.2. E-Admission

The step of E-Admission consists of all processes that are required for a student to be accepted to a certain program or let in. This requires a supporting evidence application, such as transcripts, standardized test scores, recommendation letters, letter of intent, bank and financial statements, and other records, as required.

2.4.1.3. E-Enrollment

The E-Enrollment process consists of all steps associated with the admission activities of a semester. This contains tasks such as fee payment, meeting the advisor, collection of classes, access to the course portal, getting familiar with the program of the online course, and any other activity required before the class begins.

2.4.1.4. E-Course

Critical thinking and continuous learning are the ultimate aims of education. Both of them are done by classes and programs supported by the institution. Before, after, and after its Web offering, an E-Course contains of events associated to the course.

2.4.1.5. E-Graduation management (Web Based Repository System)

According to (Aadamsoo, 2010) web-based repository systems are designed to maintain and store information used as web-based software for graduation projects.

The purpose of this website is to handle various activities involved in the graduation project, which makes it very useful for all stakeholders and for the educational process. Stakeholders include adviser to the faculty, graduate student, coordinator, and examiner for the faculty (Awad ,2017).

The roles in education differ from one participant to another. Graduates has duties include searching for topics, choosing a thesis title, working on a project, uploading documents, and grades for viewing.

Responsibilities for the administrator include calling for student projects, managing users, notices, assigning advisors and examiners to the students, review of the system, monitoring, and analytics of data.

For faculty advisors' tasks include viewing projects, giving suggestions, monitoring, evaluating documents, and marking. For examiners tasks include grading projects and viewing documents and notices.

In other words, other groups are still having difficulty finding the correct files and wasting useful time to obtain and search project records, identifying and specifying this project, and actions that need to be taken before continuing to implement improvements in graduation projects.

To handle these activities for all stakeholders, web-based repository system provides an easy-to-use, flexible, and an online solution.

For consuming less time, keeping all inconveniences less, and to arrange all reports into one place and in particular, to monitor graduation projects that are in execution for students or for watching out for errors or mistakes that happen during the work cycle, at that point a good web-based repository for graduation project management was recommended. To think about everyday use and needs, the point was to make an inside system for the university (Aadamsoo, 2010).

2.5. Web-based Repository System for Graduation Project Services

According to (Lip et al, 2012) Web based repository system for graduation project services are:

- Scalability/Expandability/Upgradability: Referring to the web-based repository system's capacity to extend without major performance degradation
- Reliability/Resilience (Fault Tolerance): Referring to the web-based repository system's ability to manage any unusual situation and if necessary, provide a disaster recovery service.
- System and Data Integrity: Referring to the capacity of the web-based repository system to ensure that the data stored is accurate and correct after a particular processing, such as replication of data, addition, deletion and alteration of data.
- Performance: This refers to the turn-around time of a web-based repository system for a full upload or downloading process.
- System and Data Availability: Refers to be available anytime.

- Accessibility: Refers to be accessible anytime for searching and retrieving information.
- Achievability: The ability of the system to store diverge data types.
- Integrity: The ability of the system to integrate with other web-based system if needed.
- Security: Referring to the ability of the system to avoid any malicious and other security threats from hackers and crackers.

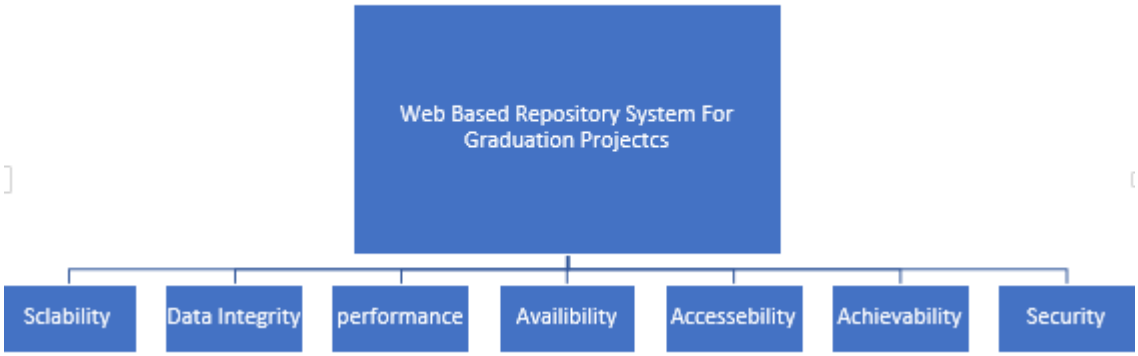


Figure 2.2: Web-based repository system services

2.6. Technological and Higher Institutional Solutions

The university management system is built as the integrated solution, in order to automate all university tasks. Processes use general university databases, standard categorizations. It helps for individual users to avoid the same data from being interpreted differently. All data is kept in a shared database.

Just like that, it allows easier to prevent data replication. Information is gathered, put in the database and handled where the first sources are found, i.e., in the parts of the university. That is why document circulation time reduces, fewer logical errors are made, and error search and removal can be performed efficiently (Komka, 2011).

2.7. Conclusion

After observation and information gathering, I have decided to develop computerization process. The process which are planned to computerize is graduation project tasks.

CHAPTER 3

METHODOLOGY

3.1. System Description

The methodology chosen for answering the research question (see section 1.7) and provide a proof concept for the proposed solution is to develop an online web repository system, a website built in the ASP.NET language (that can create and design a suitable website with validation and security features provided by Microsoft) for graduation projects.

As shown in Figure 3.1, our solution consists of four main components with each of which has different functionalities.

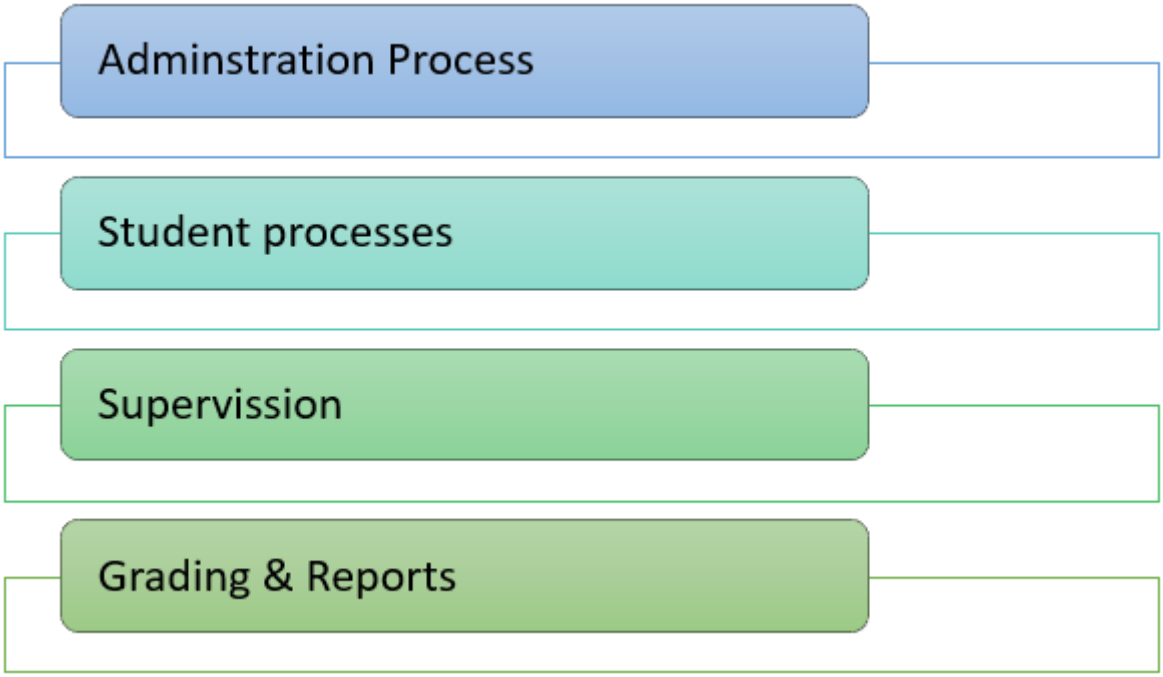


Figure 3.1: online web-based repository system components.

The administration process deals with setting up the system, creating users, registering students, advisors, supervisors, examiners, faculties, departments, and assigning advisors to students.

The admin will keep track of the log file which stores every one's transaction in a file as a footprint.

The Student tasks is the part that is responsible for students' processes. The student enters the system through a valid and registered username and password. After signing to the system, the student will choose a title before supervisor appointment. If the thesis title is accepted, he/she chooses a supervisor, and if the supervisor accepts to be a supervisor the student will submit the following forms: thesis registration form, proposal submission form, and plagiarism and ethical form. The last but not the least, the student will upload the chapters of the thesis, the presentation of the thesis. This process covers from choosing a title until submitting the whole work.

The supervision process is the part where the supervisor accepts to be a supervisor for someone, receives the proposal of the thesis topic, and other documents. The supervisor can download and view the documents. The supervisor will give feedback to the student through mailing system designed for contacting each other.

The last component is the grading part, where the supervisor and other examiners will give the student grades. So, the committee will decide if the project was successful or rejected.

These are the basic components of the web-based repository system for graduation projects.

3.2. Criteria Used to Compare the Current System to The Existing System

The main objective of this work is to provide an integrated solution to security and availability issues for graduation projects through a validated online repository framework.

Since the current system of the university is not fully computerized, the new system will be expected to handle all tasks automatically in managing projects and also the new system

will offer user-friendly, flexible and efficient environment and also have a centralized storage and backup, strong searching capabilities.

According to C.laudon (2014), authentication refers to the ability to check the identity of all individuals who participate in the website of an online repository before enabling sensitive data or system access to them. We developed a validation to achieve authentication, which guarantees all site data.

Confidentiality answers the question of: is the site is accessible to someone other than those authorized to access the site? Confidentiality, which is one of the cornerstones of information security, refers to ensuring that only those allowed to have access can access information. We used a validation concept to achieve confidentiality.

Privacy is ensuring that the personal information is protected. There are some internet problems that affect the personal user details, so how do we manage these challenges?

The solution is that there are certain technological solutions that protect privacy, such as anonymity tools, anti-spyware tools, browser features such as "private browsing and options for do not track."

According to (Easttom, 2012), Securing the server that contains the website's database is the most crucial principle. The foundation of every website is its server database. Typically, this computer will store the most important data. This implies that this machine is a particularly attractive target for intruders and the most important thing is to protect them.

3.2.1. Existing system limitations

After we made the requirement analysis, several problems were related with the manual system (current system). Some of these problems were the following:

Paper and pencil operation: The existing system is utterly paper based. It does not automate the tasks related to the graduation projects.

Information accuracy: The existing system, particularly the paper and pencil operations, makes it simpler for information to be inadequately managed (mishandle).

Data redundancy: Information or Details concerning a particular person can easily be found here and there. Disorganized record handling is credited to errors, unstable, and repetitive student records in several workplaces.

Decision making: Since the efficiency of the existing system is weak, decision making is generally prolonged.

Time Consuming: Searching something for example thesis proposal takes a lot of time and effort.

3.2.2. Existing system and the new system comparison

Roles of users: Instead of the current system, the proposed system gives out various roles to different users, and builds the clients of the system too. Students, supervisors, advisors, examiners, and administration staff, will all communicate with the system and have a role to carry out.

System Features: Arrangement of different roles to various users implies that several features are being added to the proposed system, in contrast to the existing system.

Data Repetition: The proposed system utilizes use of data in a very consistent way. Different users can use on the same data set without conflicting the roles of one another. For now, the data is system based (database) unlike the existing system where data is generally paper based.

Security of the data: data can simply be backed up regularly, consequently making it safer and better than the existing system. Users will encounter validations to satisfy their user roles.

3.3. Web Development Environment

Web development move well after the use of markup codes and a few plug-ins or scripting procedures to deliver great and valuable web sites. The term refers to the use of particular

techniques, tools and approaches that are defined as three-tier, client/server, and information management systems for the development of web pages and websites. (Middle Georgia State University, 2020)

The term Web-based derives from the fact that the information processing systems relies on the Internet technology, especially that section known as the World Wide Web (WWW).

The Golden Grid System (2020), says that the web application development process is important to the success of web-based projects. Proper procedures cannot be implemented, unless technological environments are correctly designed. For development, testing and production, technological environments are needed.

3.3.1. Why ASP.Net

ASP.neT 4 is an incredible technology to create your Web solutions with. In the field of web application creation, when ASP.NET 1.0 was launched in 2000, many considered it a transformative leap forward (Evjen, 2010).

3.4. Architecture of The System

"The architectural design is a structure and plan for how the system (network environment) will be circulated across the networks and what hardware and software will be used for each computer. Three essential system architectures are in use today: server-based, client-based, and client-server-architectures" (Sommerville, 2016).

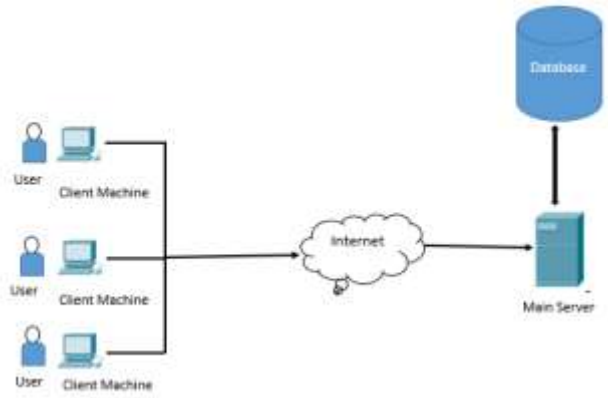


Figure 3.2. Web based repository system architecture (Sommerville, 2016)

“The client-server architecture endeavors to synchronize processing or workloads between the client and the server. This is a typical choice, as it requires decreased overhead and simpler maintenance. Online systems regularly use this architecture, with the introduction of the Internet browser (the client) and only just fundamental application logic utilizing programming languages such as JavaScript, while the server handles the application logic, data access logic and storage” (Sommerville, 2016).

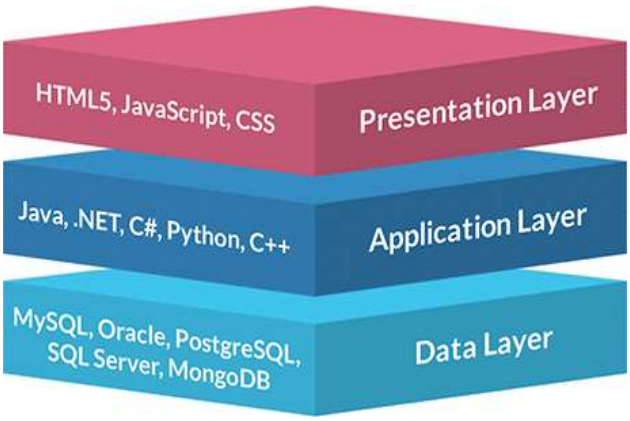


Figure 3.3 3-tier architecture (Jinfonet, 2020)

The “presentation tier” in the three-tier system is the first layer and contains the user interface.

The second tier is the “application tier” and it covers the business process logic that makes the main functionality of the application operates. It is written in programming languages like ASP, Java.

The last part is the “data tier” which contains the database and “data access” layer. Example: MS SQL Server, etc.

3.4.1. Client-Server architecture advantages

The first advantage of the client-server architecture is it supports scalability, which ensures that the storage of the server and the capabilities of processing can be conveniently increased or reduced. Whenever a server gets slow, another server is easily installed, so that many servers can be used to execute program’s functional logic, or database system.

Th second benefit is: it is possible to support a lot of different types of servers and clients, meaning that devices that use different operating systems can be directly connected.

The last advantage is: when it comes to a slim client server architecture (the one which containing a limited part of the program logic) that uses the standards of the Internet, it is a reasonable approach to explicitly differentiate the program logic, data access logic, and presentation logic, and to build individually being rather stand-alone.

In a nutshell, the frontend of the system can be modified without changing the program logic, and the other way around. Because of these motives, I decided to use this architecture.

3.5. Methodology Development

The approach to the answer the research question is to set up a practical lab, for developing an online web-based repository system for graduation projects in ASP.NET. The lab is made up based on client/server architecture.

3.6. Requirements

3.6.1. Hardware requirements

The proposed system development will require: Two or more PCs 2GHz, 1 GB RAM and 40GB HDD, with browsers, one used as development PC as well as Database server, and the other(s) will be used as a client.

3.6.2. Software requirements

The following software packages are required for the development of our system:

a) ASP.NET using VB.NET.

b) JavaScript & other plugins (Bootstrap, and jQuery).

c) Microsoft SQL server.

CHAPTER 4

SYSTEM ANALYSIS AND DESIGN

4.1. Analysis of the System

4.1.1. Manual system

Graduation projects are maintained in Microsoft Excel as its backbone for entering, editing and storing graduation project data. After our analysis of the system, we found that there are many limitations. These limitations have been detected in different phases of the system, such as data collection, editing and information searching.

The researchers found that in order to meet the standards of today's technology, the system needed greater re-engineering and redesign. Due to the relevance of the graduation project information management system to the entire system, the researchers have made a major effort to overcome these shortcomings and to improve the system's efficiency and reliability.

4.1.2. The Proposed system

The proposed system will be developed using ASP.Net, which is an object-oriented language. The new Graduation Projects System will enable to process projects related tasks automatically and efficiently.

The proposed system will have high level security features for controlling unauthorized access to the data.

4.2. System Requirements

> "Requirements Specification is one of the most critical requirement engineering activities through which the elicited and evaluated requirements for their intended stakeholders are properly recorded for usage. Throughout the initial requirement phase of a project the old approach of requirement engineering may produce the document like a word document" (Firesmith, 2003).

To get the target solution, functional requirements and nonfunctional requirements of the system must be indicated.

4.2.1. Types of requirements

A software requirement is mainly divided in two groups which is functional and nonfunctional.

4.2.2. System non-functional requirements

System performance, suitability and data security are examples of Non-functional or quality requirements, and are the success keys of a software system (Affleck et al, 2015).

The following are Nonfunctional requirements of the web-based repository system for graduation projects:

User Interface requirements: it is required that at least one browser can be functional to load the system to the user's device screens.

While every user has it is own unique screen size, the system must be compatible with their devices. The compatibility issue is important and it is required.

The website interface should have buttons and options which is easy to understand and use.

Performance requirements: The user's internet must be fast to get quickly the response of the system. With a good network connection, the system can respond to its users more quickly as well. Our system was developed with simple, but common, flexible, and usable development tools that gives browsers the ability to perform faster.

Unlicensed Users: Different roles are granted to individual users of the website. Users Password security controls would be authenticated to make sure the unauthorized interference is prevented.

Internet security: instead of using HTTP we recommend you to use HTTPS (secured hyper-text transfer protocol) for enhanced Internet security. HTTPS is more secure than the HTTP.

Management Requirements: An administrator will manage the system and maintain it in order to be effective.

4.2.3. System Functional Requirements

The tasks or processes that the system must perform while developing are called functional requirements.

Data Insertion, altering and Storing: The different users with different roles can access the system, insert and alter at the same time without interfering each other.

Data Manipulation: Each website takes input; processes input and produces outputs. The aim of our system is to quickly operate data based on user information, in particular, the system shall take advantage of the specific grading scale given by the university board and used in the current system created.

Grade Report: Graduate student will preview his/her grade and print it.

Thesis List: Visitors or system guests who do not have a username to access the system will search the student titles by department or academic year, or both, and will preview the proposed topic details.

The system will also produce summary information for the administrators, supervisors, advisors, and examiners.

4.3. System Design

The primary goal of designing the system is to get a simple approach that determines the features of the system and also the interconnections of functional units of the system.

File and database design consist of two main stages. To define data, you begin by creating a logical database model which uses a notation corresponding to a data organization used by a database management software. It is the responsibility of this system software to store, retrieve, and secure data like SQL Server.

For a logical database model, the most common type is the relational database model. If a predictable and consistent logical data model is created, the technical specifications for computer files and databases in which the data can actually be stored are ready to be proposed.

This phase covers designing the structure of the database table with data dictionaries that correspond to the database tables. System design can be reviewed or presented in order to allow constraints, specifications and even conditions to be adjusted prior to programming.

4.3.1. Design tools

There are many tools to use when designing a system, we are going to use the most useful ones. We use UML as system design analysis tool to describe the proposed system's activities. UML stands for "Unified Modeling Language; The UML is an object-oriented modeling language for software process modeling" (D.et al, 1999).

Although UML has different types of diagrams, we decided to use two of the most used parts which is "use case and activity diagram".

4.3.1.1. Use cases

This diagram is "commonly referred to as behavior diagrams to describe a set of actions (use cases) that should or can be performed by some system or systems (subject) in collaboration with one or more external system users (actors). Each case of use should provide the actors or other stakeholders of the system with some measurable and useful result" (uml-diagrams, 2014).

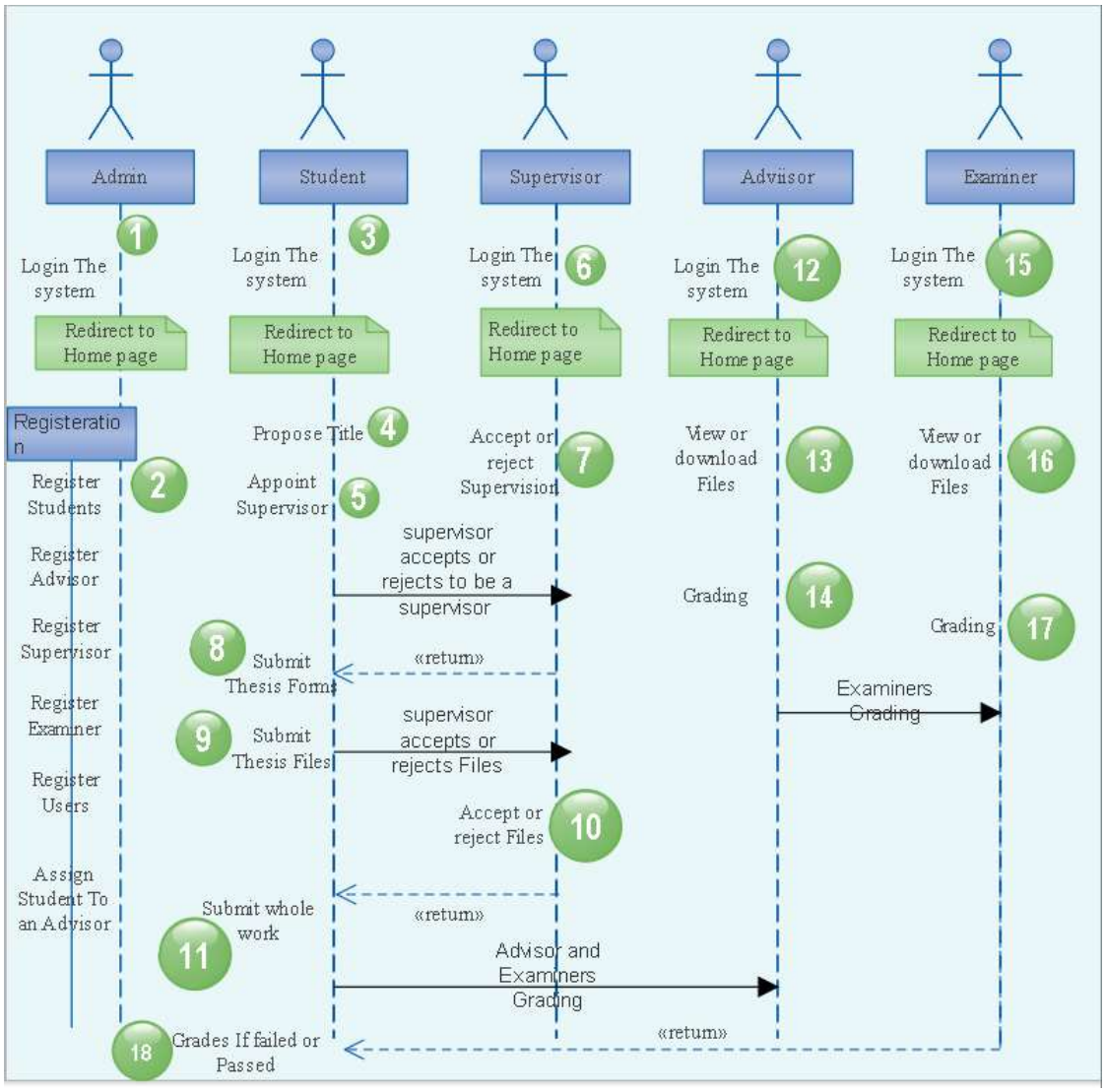


Figure 4.1: System processes

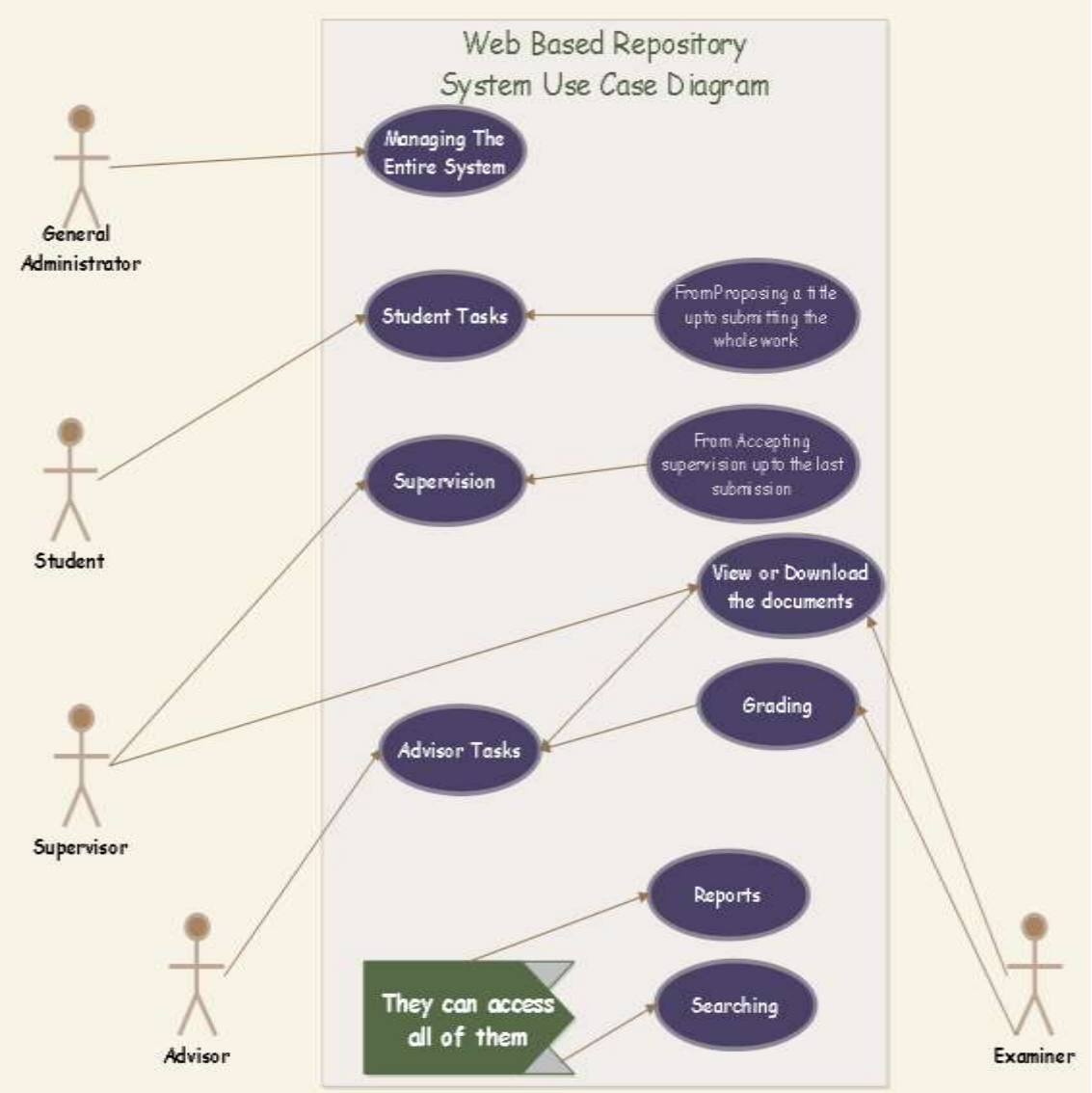


Figure 4.2: System users.

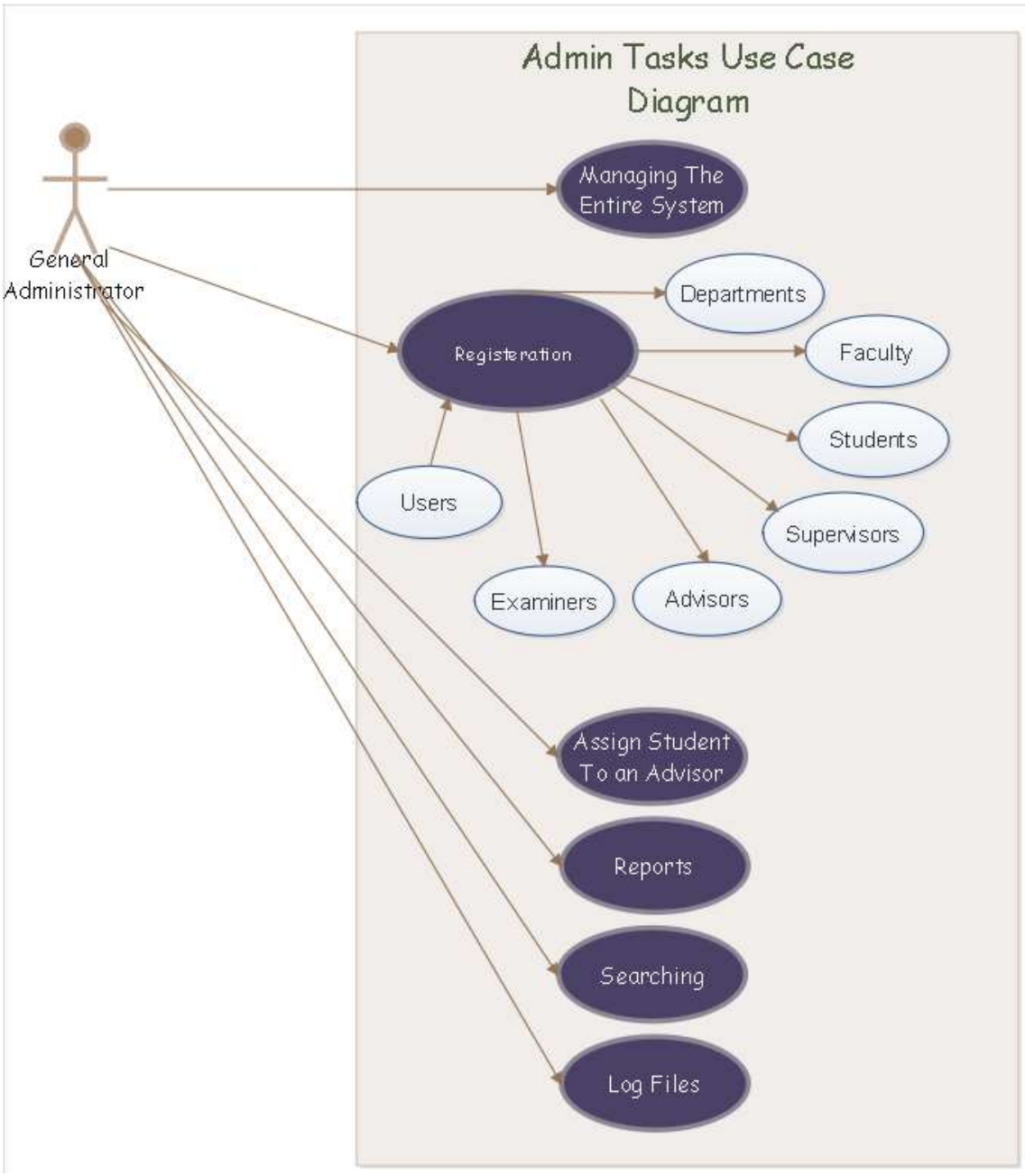


Figure 4.3: Admin tasks.

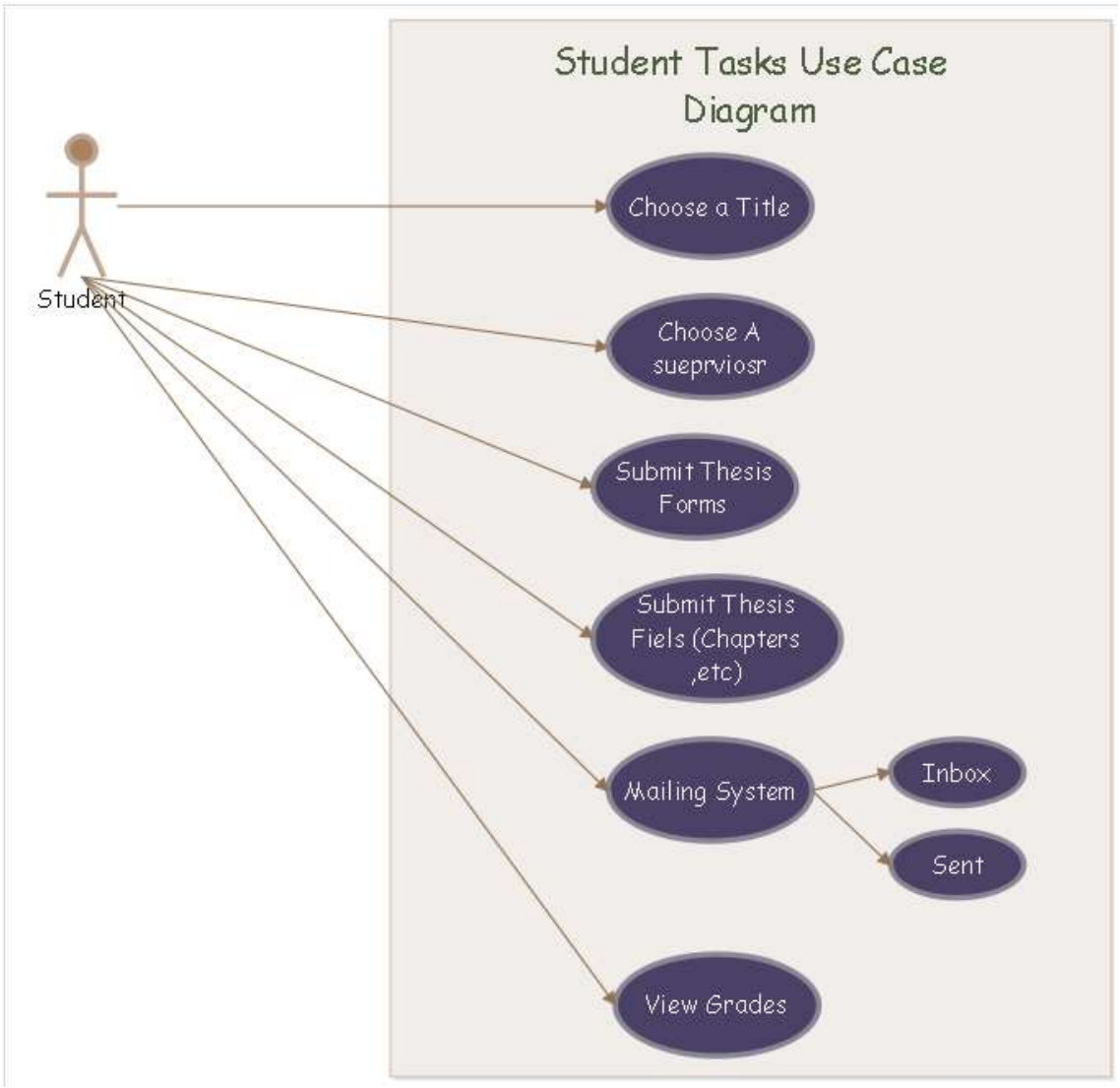


Figure 4.4: Student tasks.

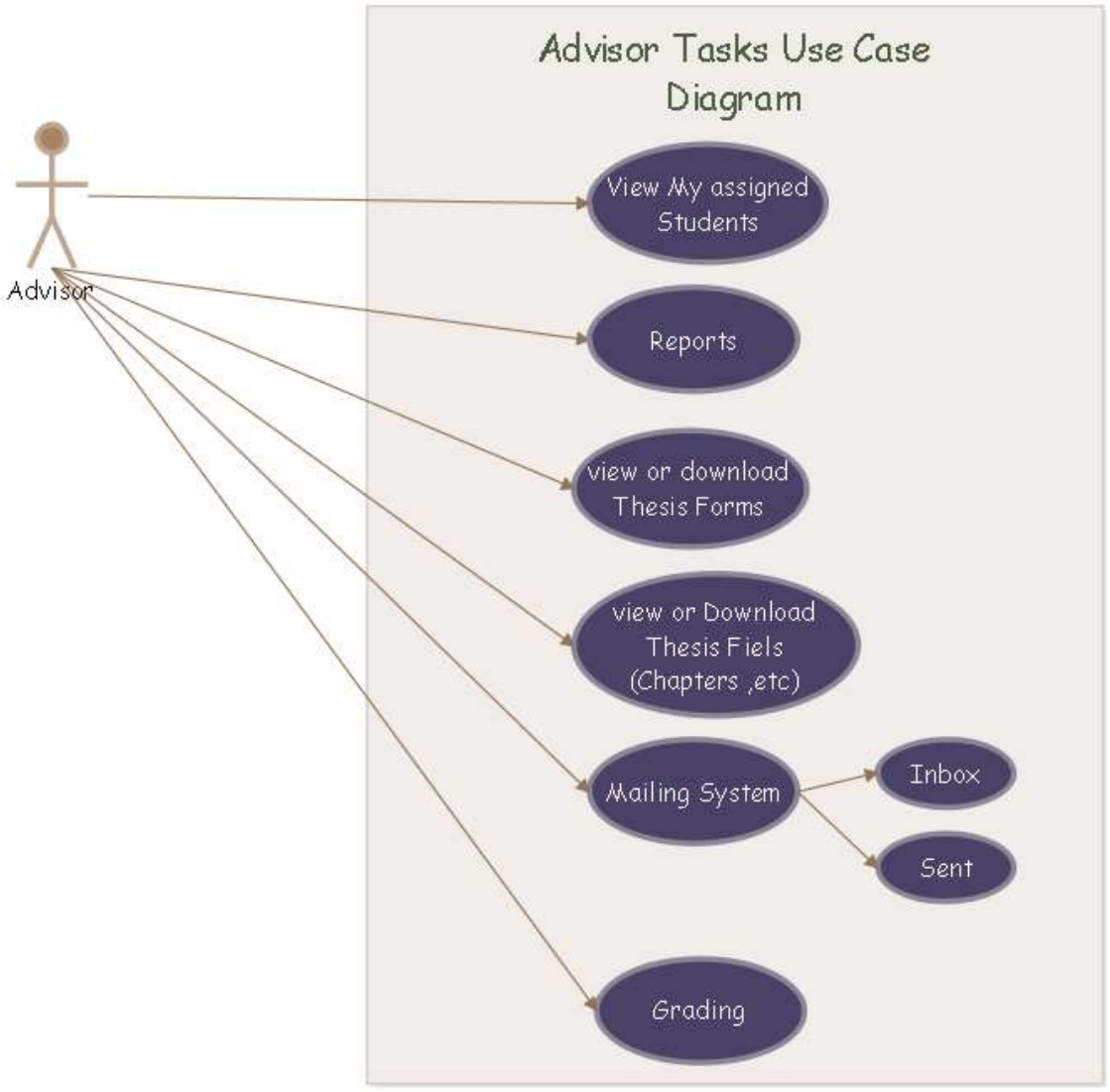


Figure 4.5: Advisor tasks.

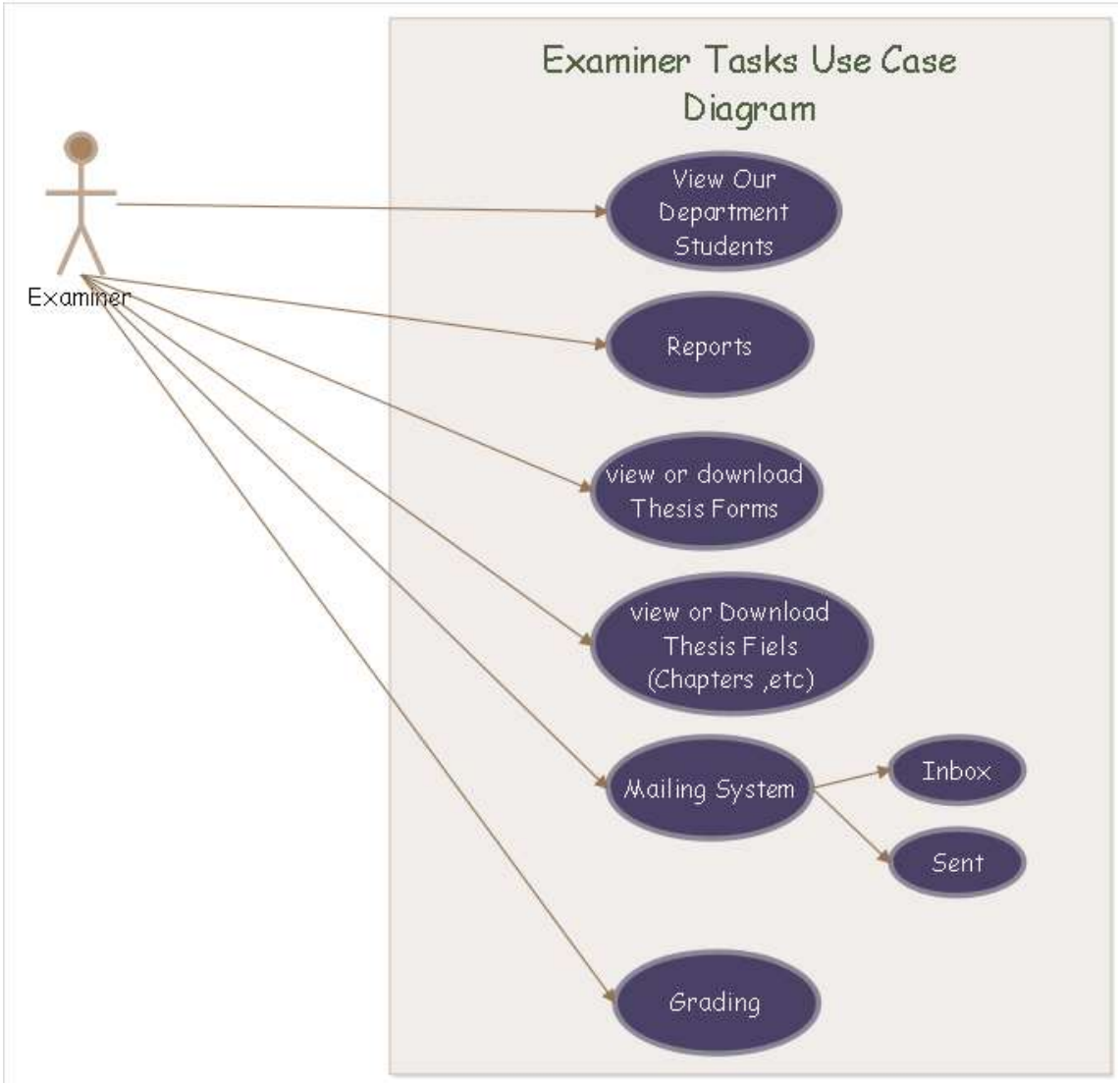


Figure 4.6: Examiner tasks.

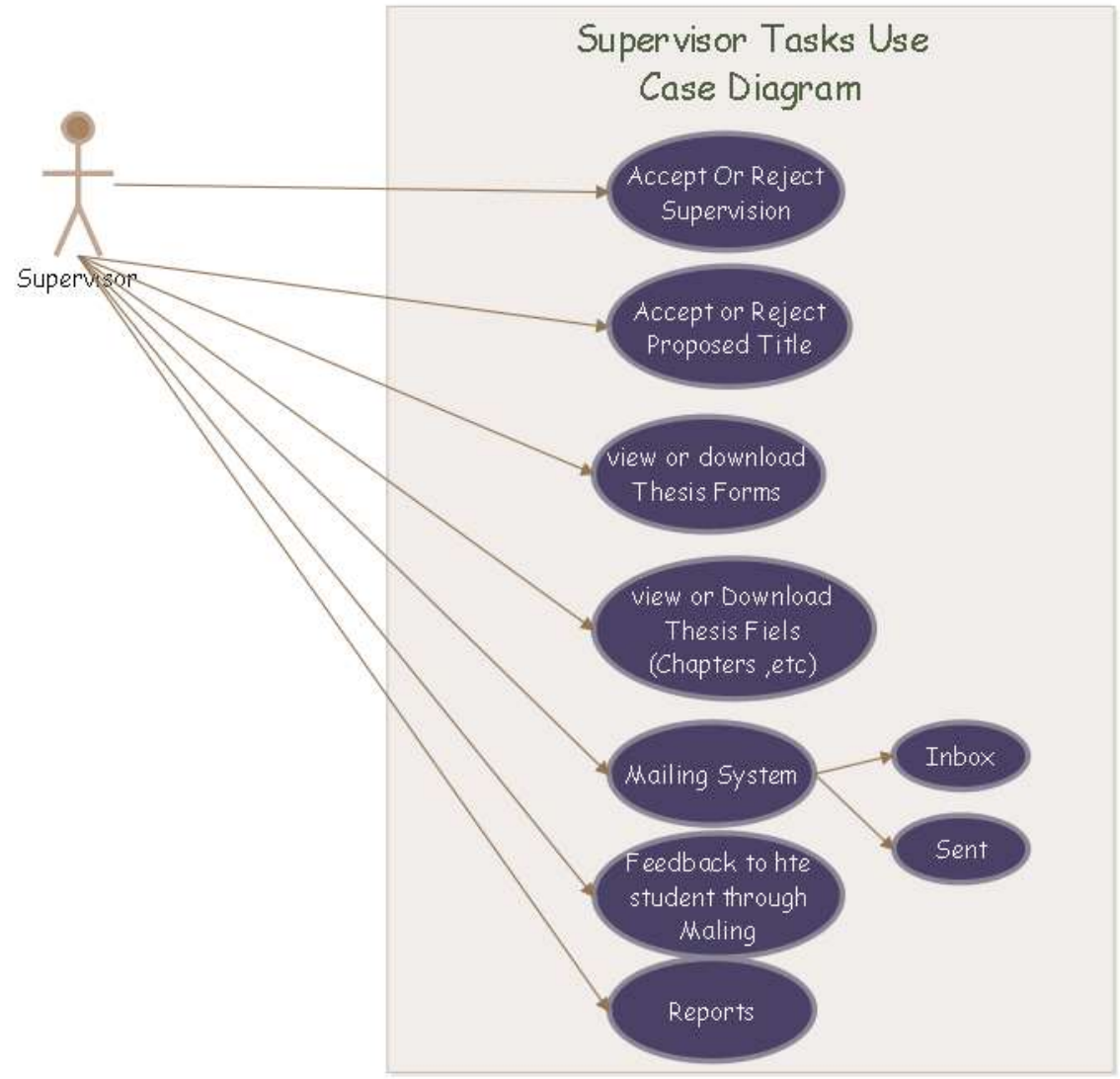


Figure 4.7: Supervisor tasks.

4.3.1.2. Activity Diagrams

This diagram, also referred to as “control flow and object flow diagram is one of the UML (unified modeling language) behavioral diagrams. To identify the sequential, conditional, and parallel composition of lower-level behaviors, they provide a graphical notation” (OMG, n.d.).

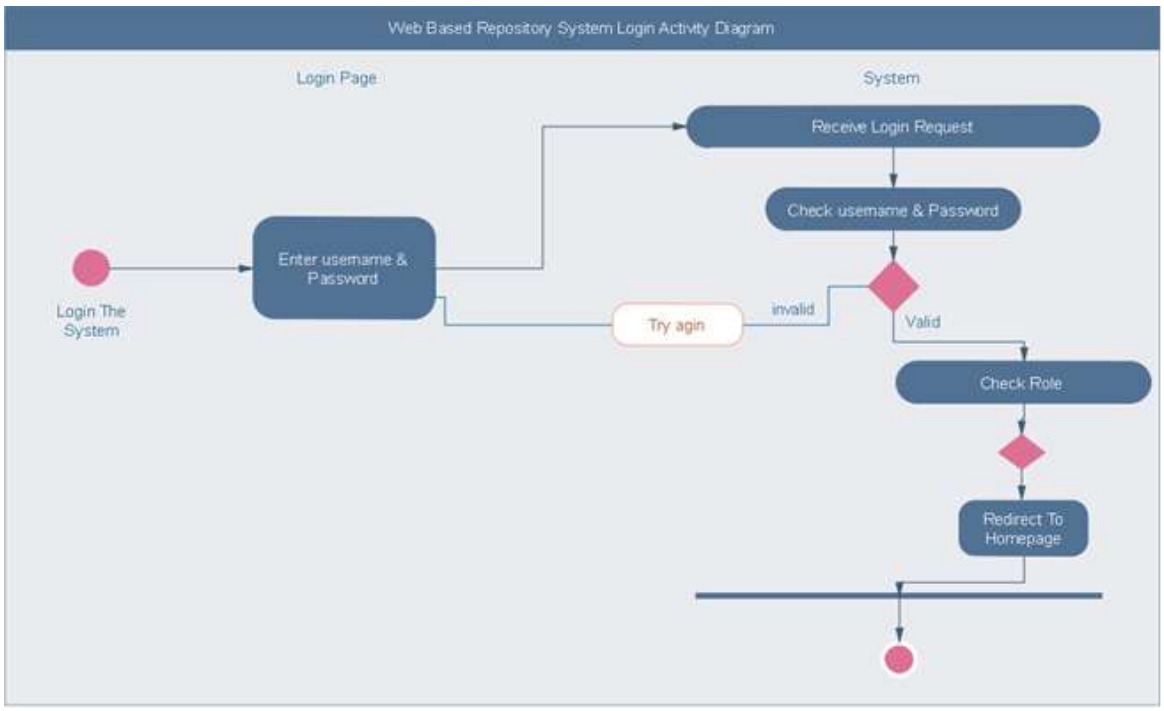


Figure 4.8: Sign in steps.

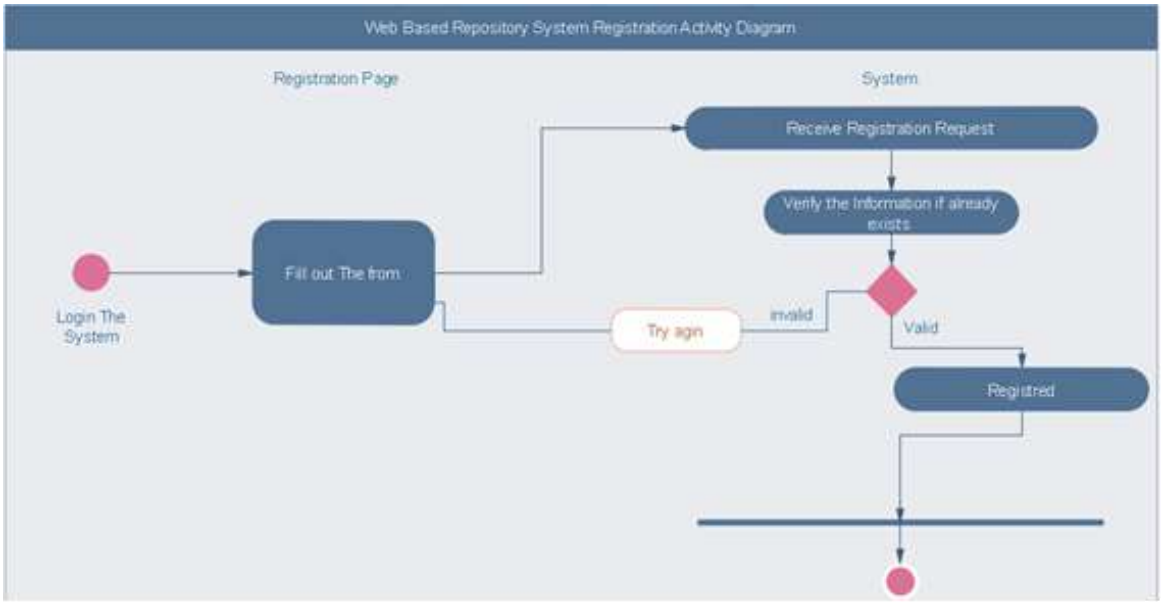


Figure 4.9: Registering steps.

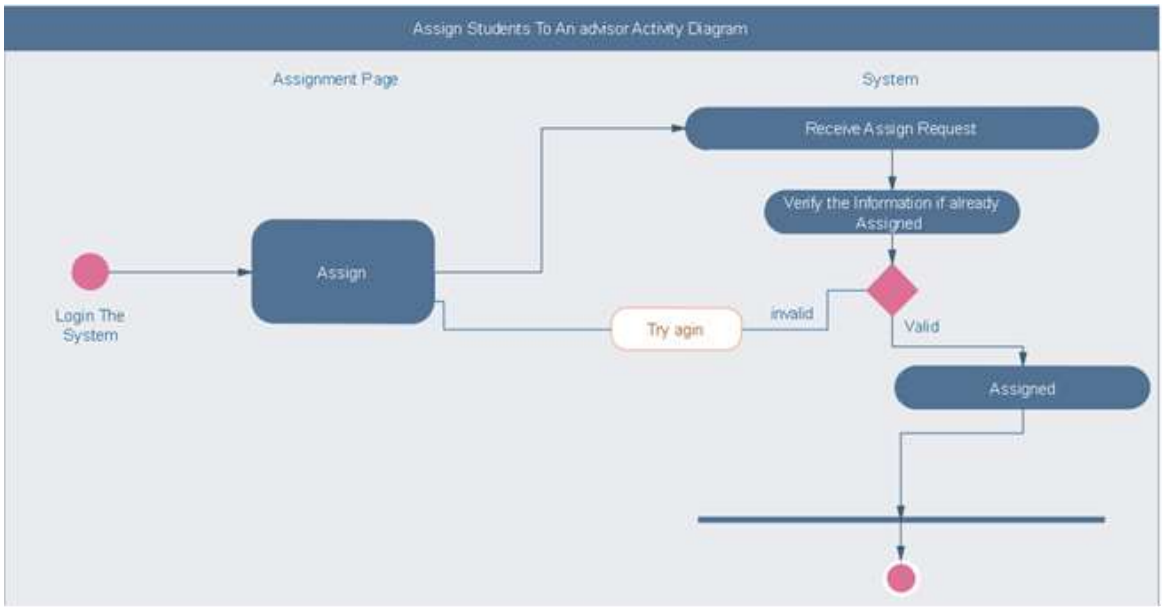


Figure 4.10: Assigning advisor steps

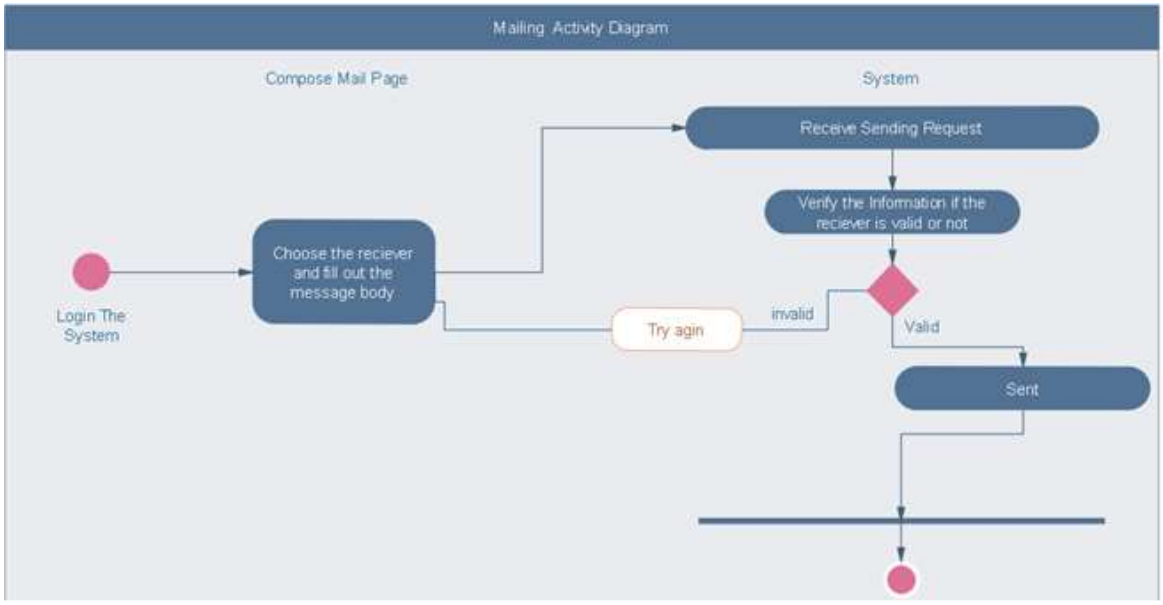


Figure 4.11: Mailing steps.

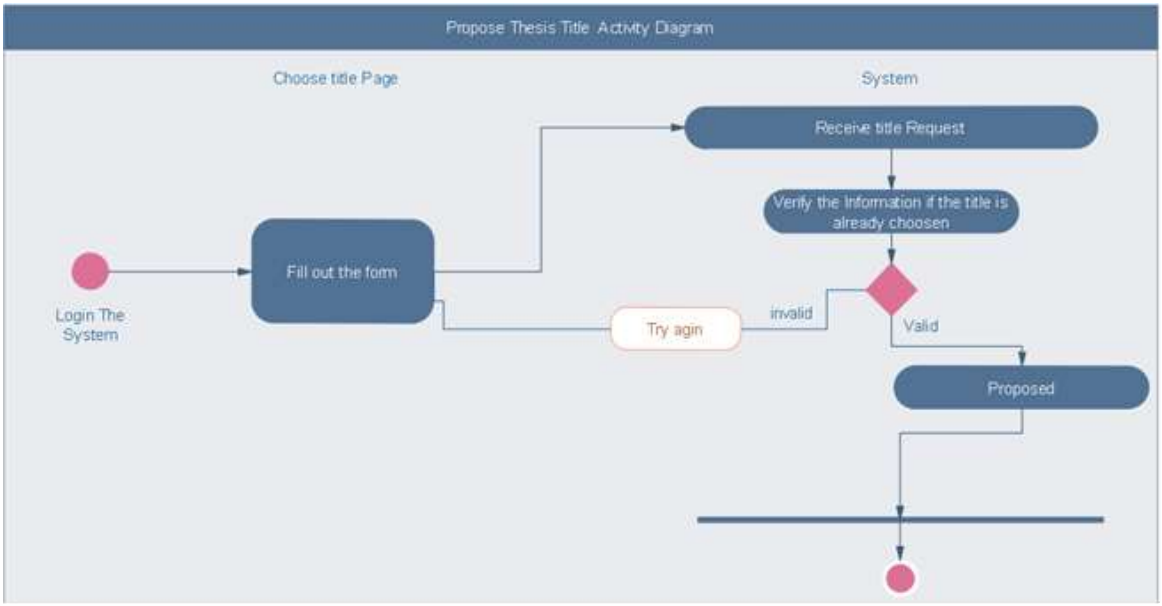


Figure 4.12: Proposing title steps

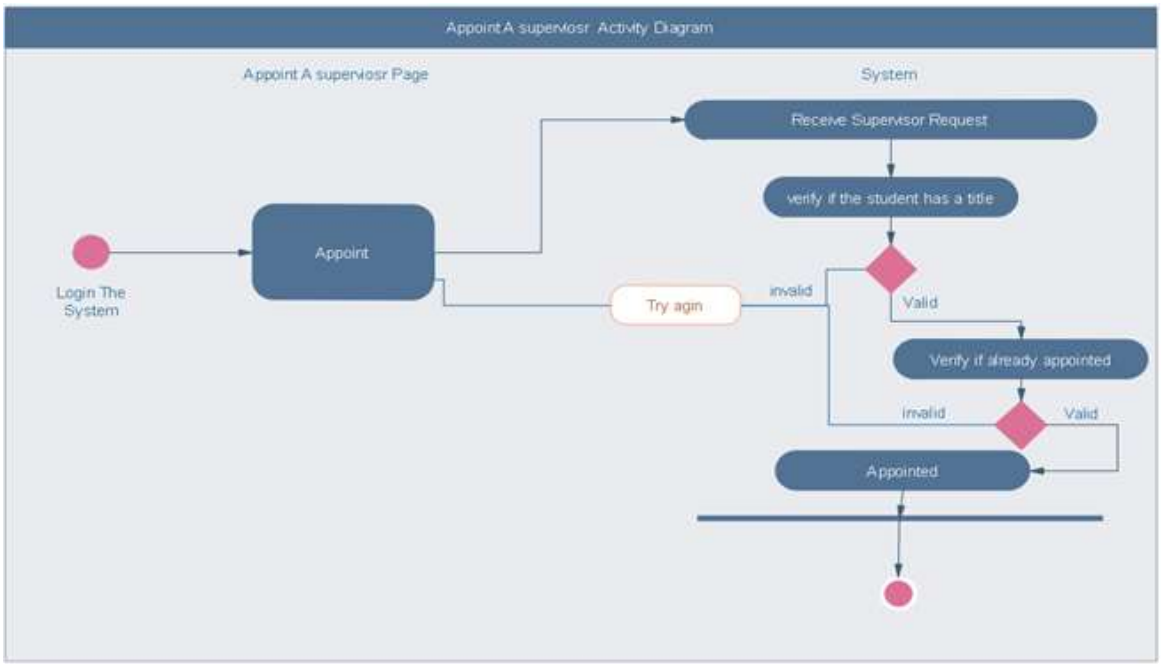


Figure 4.13: Supervisor appointment steps

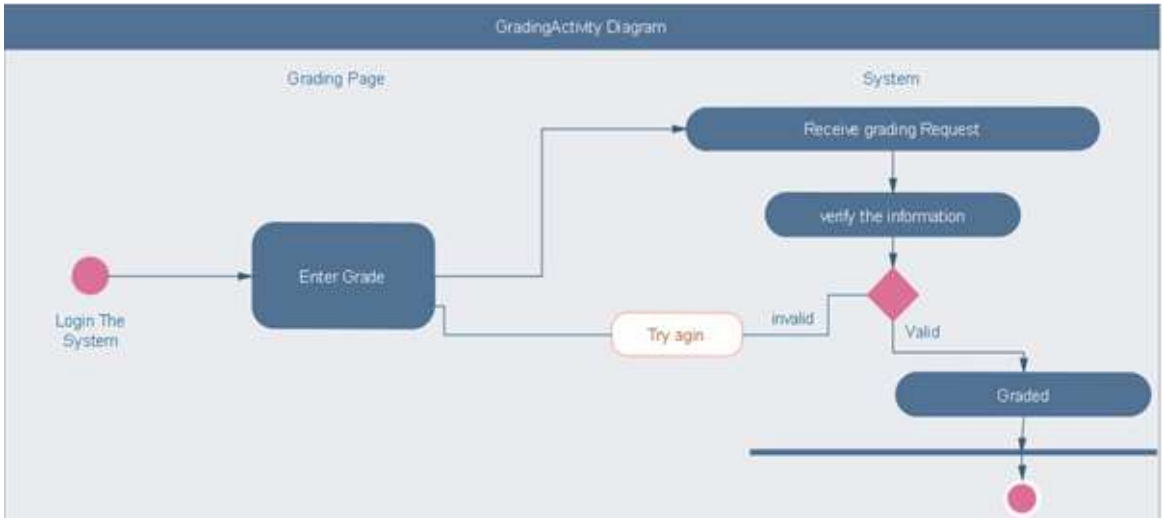


Figure 4.14: Grading steps

4.4. Database Structure and Design

After analyzing and defining the software specifications, software design is the first of the three methodological activities needed to develop and authenticate the software. Each activity transforms data in such a way that validated computer software eventually results in it.

The term database design can be used to describe various different key components of an overall relational database.

4.4.1. Tables

The database is described by the below schema:

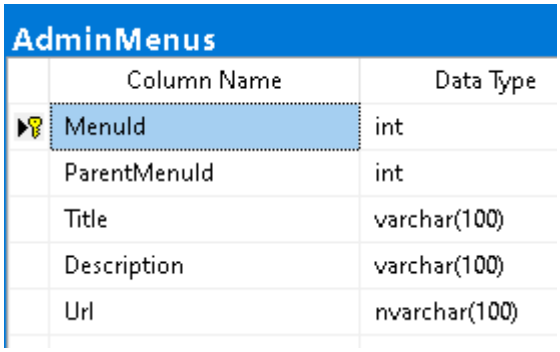


Figure 4.15: Administrator menus

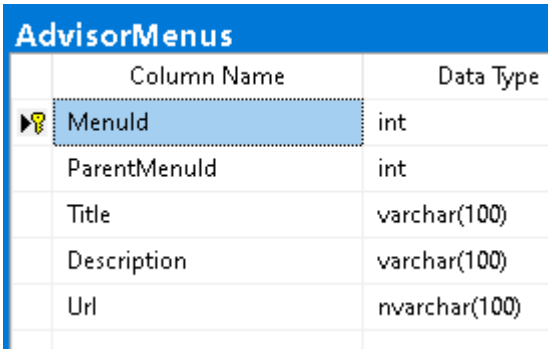


Figure 4.16: Advisor menus

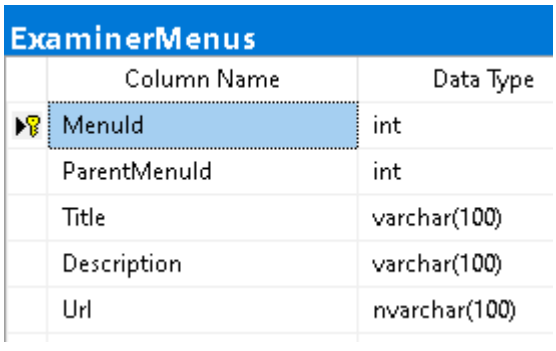


Figure 4.17: Examiner menus

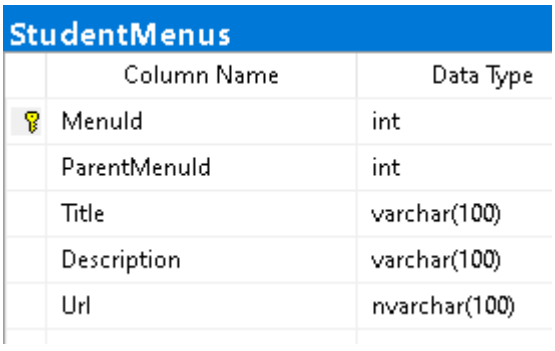


Figure 4.18: Student menus

SpervisorMenus

Column Name	Data Type
MenuId	int
ParentMenuId	int
Title	varchar(100)
Description	varchar(100)
Url	nvarchar(100)

Figure 4.19: Supervisor menus

logfile

Column Name	Data Type
LogID	int
loguser	varchar(50)
Description	varchar(100)
logdate	date

Figure 4.20: Logfile

Advisor

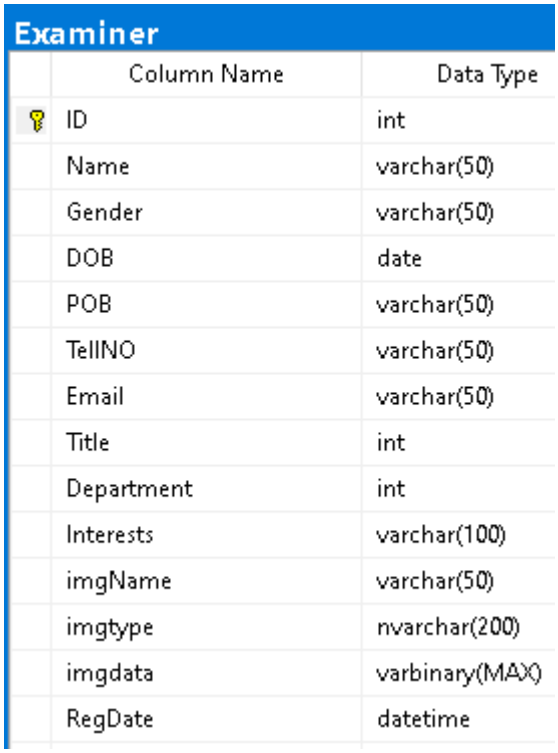
Column Name	Data Type
ID	int
Name	varchar(50)
Gender	varchar(50)
DOB	date
POB	varchar(50)
TelINO	varchar(50)
Email	varchar(50)
Title	int
Department	int
Interests	varchar(100)
imgName	varchar(50)
imgtype	nvarchar(200)
imgdata	varbinary(MAX)
RegDate	datetime

Figure 4.21: Advisors

Supervisor

Column Name	Data Type
ID	int
Name	varchar(50)
Gender	varchar(50)
DOB	date
POB	varchar(50)
TelINO	varchar(50)
Email	varchar(50)
Title	int
Department	int
Interests	varchar(100)
imgName	varchar(50)
imgtype	nvarchar(200)
imgdata	varbinary(MAX)
RegDate	datetime

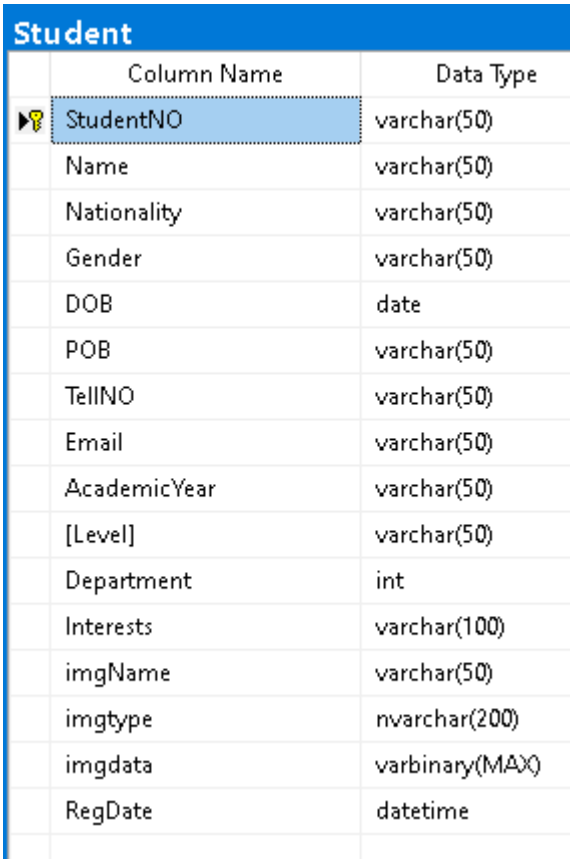
Figure 4.22: Supervisors



Examiner

Column Name	Data Type
ID	int
Name	varchar(50)
Gender	varchar(50)
DOB	date
POB	varchar(50)
TelINO	varchar(50)
Email	varchar(50)
Title	int
Department	int
Interests	varchar(100)
imgName	varchar(50)
imgtype	nvarchar(200)
imgdata	varbinary(MAX)
RegDate	datetime

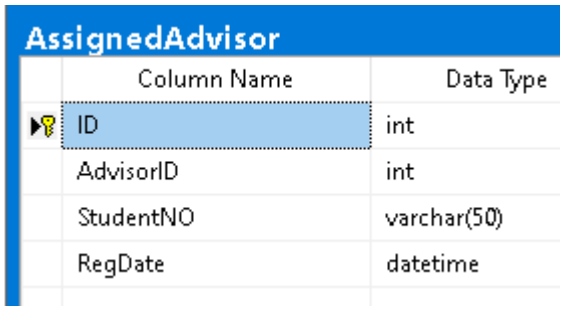
Figure 4.23: Examiners



Student

Column Name	Data Type
StudentNO	varchar(50)
Name	varchar(50)
Nationality	varchar(50)
Gender	varchar(50)
DOB	date
POB	varchar(50)
TelINO	varchar(50)
Email	varchar(50)
AcademicYear	varchar(50)
[Level]	varchar(50)
Department	int
Interests	varchar(100)
imgName	varchar(50)
imgtype	nvarchar(200)
imgdata	varbinary(MAX)
RegDate	datetime

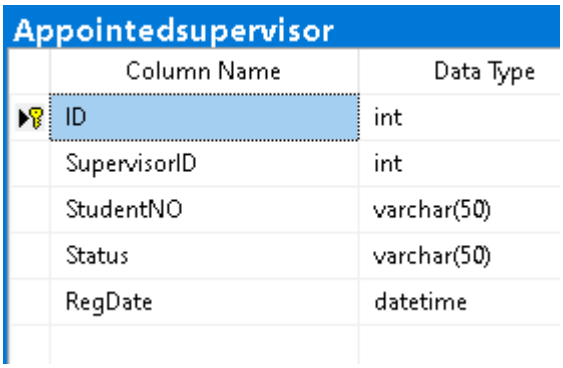
Figure 4.24: Students



AssignedAdvisor

Column Name	Data Type
ID	int
AdvisorID	int
StudentNO	varchar(50)
RegDate	datetime

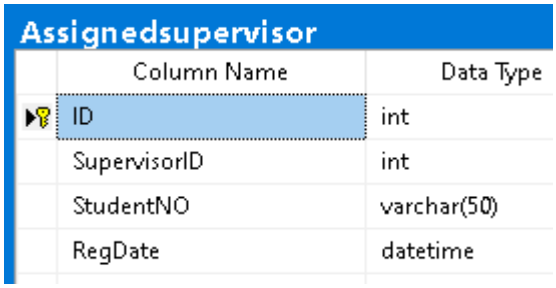
Figure 4.25: Assigned advisors



Appointedsupervisor

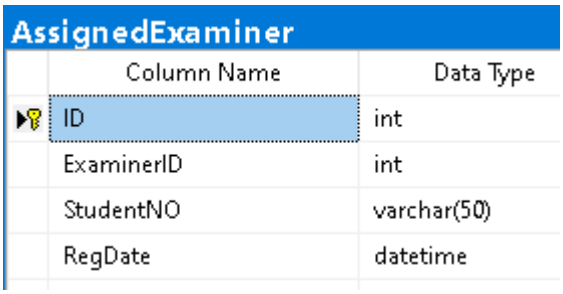
Column Name	Data Type
ID	int
SupervisorID	int
StudentNO	varchar(50)
Status	varchar(50)
RegDate	datetime

Figure 4.26: appointed supervisors



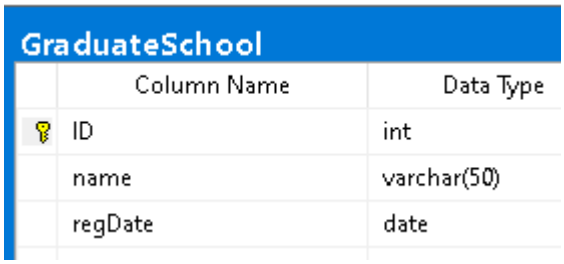
Assignedsupervisor	
Column Name	Data Type
ID	int
SupervisorID	int
StudentNO	varchar(50)
RegDate	datetime

Figure 4.27: Assigned supervisors



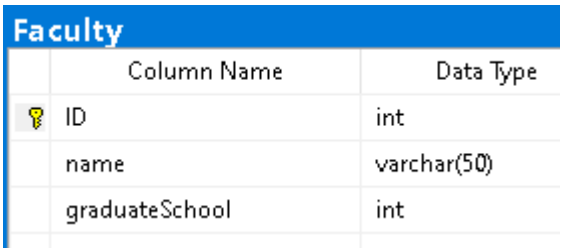
AssignedExaminer	
Column Name	Data Type
ID	int
ExaminerID	int
StudentNO	varchar(50)
RegDate	datetime

Figure 4.28: Assigned Examiners



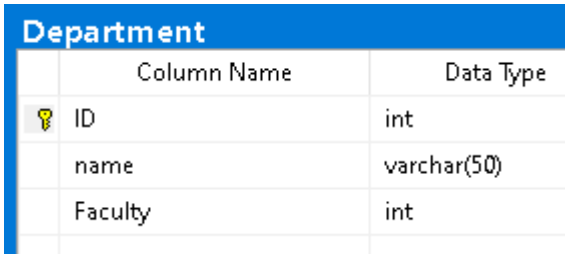
GraduateSchool	
Column Name	Data Type
ID	int
name	varchar(50)
regDate	date

Figure 4.29: Graduate school



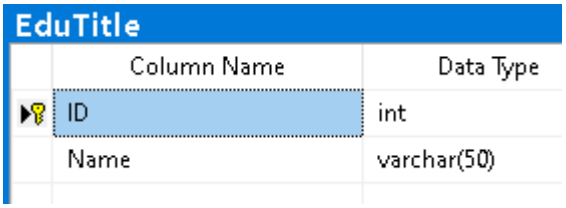
Faculty	
Column Name	Data Type
ID	int
name	varchar(50)
graduateSchool	int

Figure 4.30: Faculty



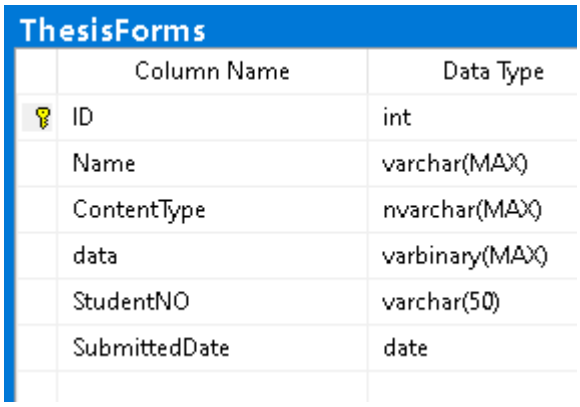
Department	
Column Name	Data Type
ID	int
name	varchar(50)
Faculty	int

Figure 4.31: Departments



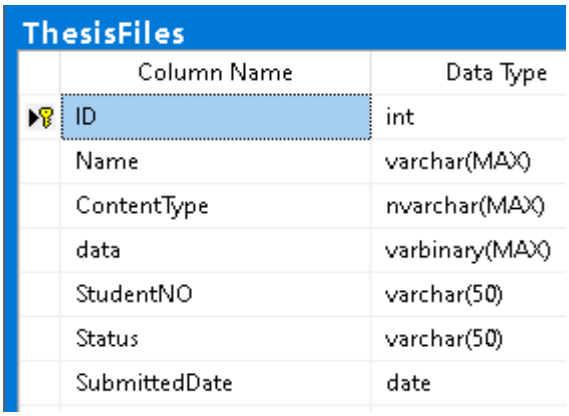
EduTitle	
Column Name	Data Type
ID	int
Name	varchar(50)

Figure 4.32: Academic titles.



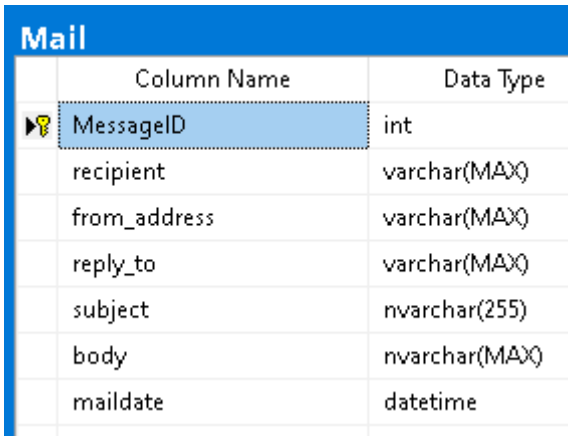
ThesisForms	
Column Name	Data Type
ID	int
Name	varchar(MAX)
ContentType	nvarchar(MAX)
data	varbinary(MAX)
StudentNO	varchar(50)
SubmittedDate	date

Figure 4.33: Thesis forms



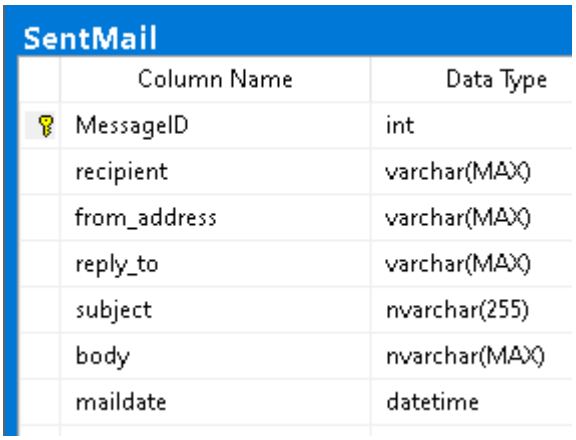
ThesisFiles	
Column Name	Data Type
ID	int
Name	varchar(MAX)
ContentType	nvarchar(MAX)
data	varbinary(MAX)
StudentNO	varchar(50)
Status	varchar(50)
SubmittedDate	date

Figure 4.34: Thesis files.



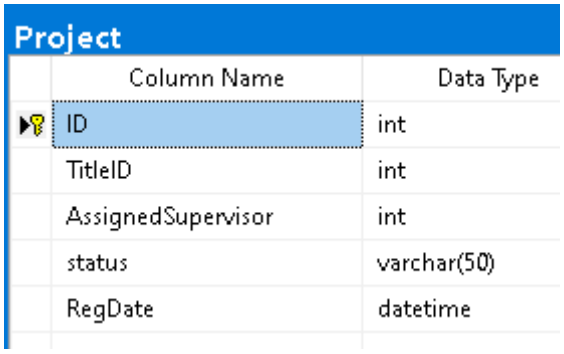
Mail	
Column Name	Data Type
MessageID	int
recipient	varchar(MAX)
from_address	varchar(MAX)
reply_to	varchar(MAX)
subject	nvarchar(255)
body	nvarchar(MAX)
maildate	datetime

Figure 4.35: Mail or Inbox table



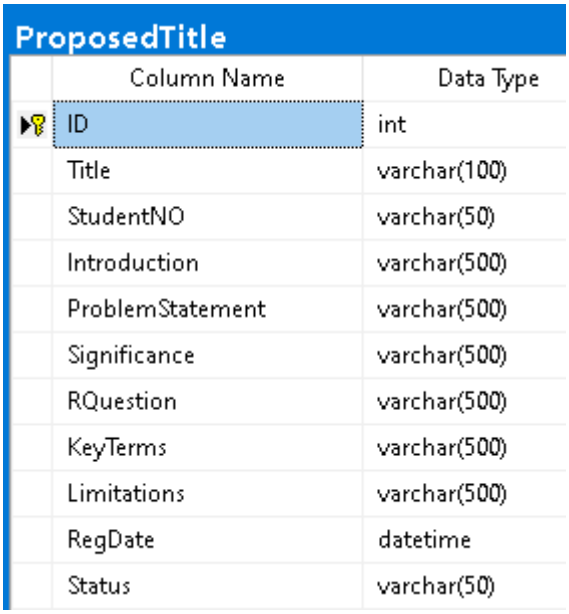
SentMail	
Column Name	Data Type
MessageID	int
recipient	varchar(MAX)
from_address	varchar(MAX)
reply_to	varchar(MAX)
subject	nvarchar(255)
body	nvarchar(MAX)
maildate	datetime

Figure 4.36: sent mail or outbox table



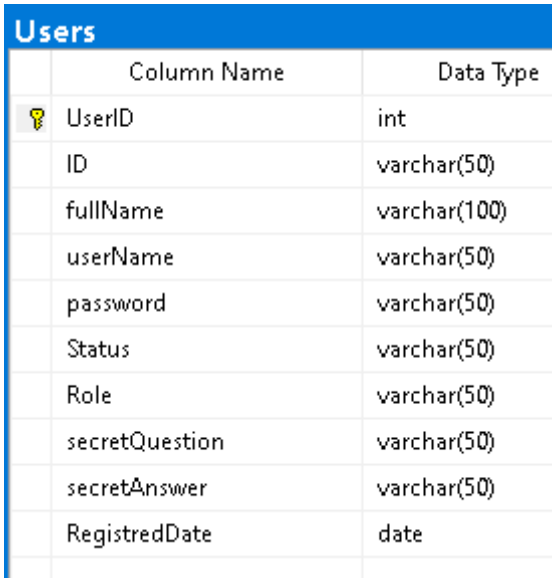
Project	
Column Name	Data Type
ID	int
TitleID	int
AssignedSupervisor	int
status	varchar(50)
RegDate	datetime

Figure 4.37: Projects



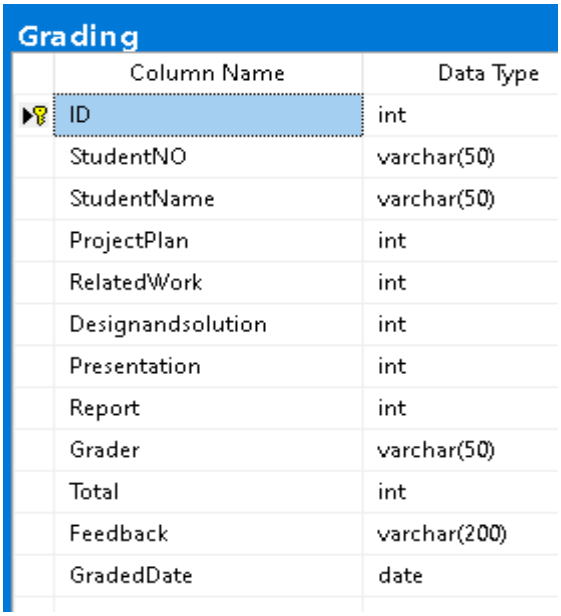
ProposedTitle	
Column Name	Data Type
ID	int
Title	varchar(100)
StudentNO	varchar(50)
Introduction	varchar(500)
ProblemStatement	varchar(500)
Significance	varchar(500)
RQuestion	varchar(500)
KeyTerms	varchar(500)
Limitations	varchar(500)
RegDate	datetime
Status	varchar(50)

Figure 4.38: Proposed titles.



Users	
Column Name	Data Type
UserID	int
ID	varchar(50)
fullName	varchar(100)
userName	varchar(50)
password	varchar(50)
Status	varchar(50)
Role	varchar(50)
secretQuestion	varchar(50)
secretAnswer	varchar(50)
RegistredDate	date

Figure 4.39: Users table



Grading	
Column Name	Data Type
ID	int
StudentNO	varchar(50)
StudentName	varchar(50)
ProjectPlan	int
RelatedWork	int
Designandsolution	int
Presentation	int
Report	int
Grader	varchar(50)
Total	int
Feedback	varchar(200)
GradedDate	date

Figure 4.40: Grading table

4.4.2. Database diagram

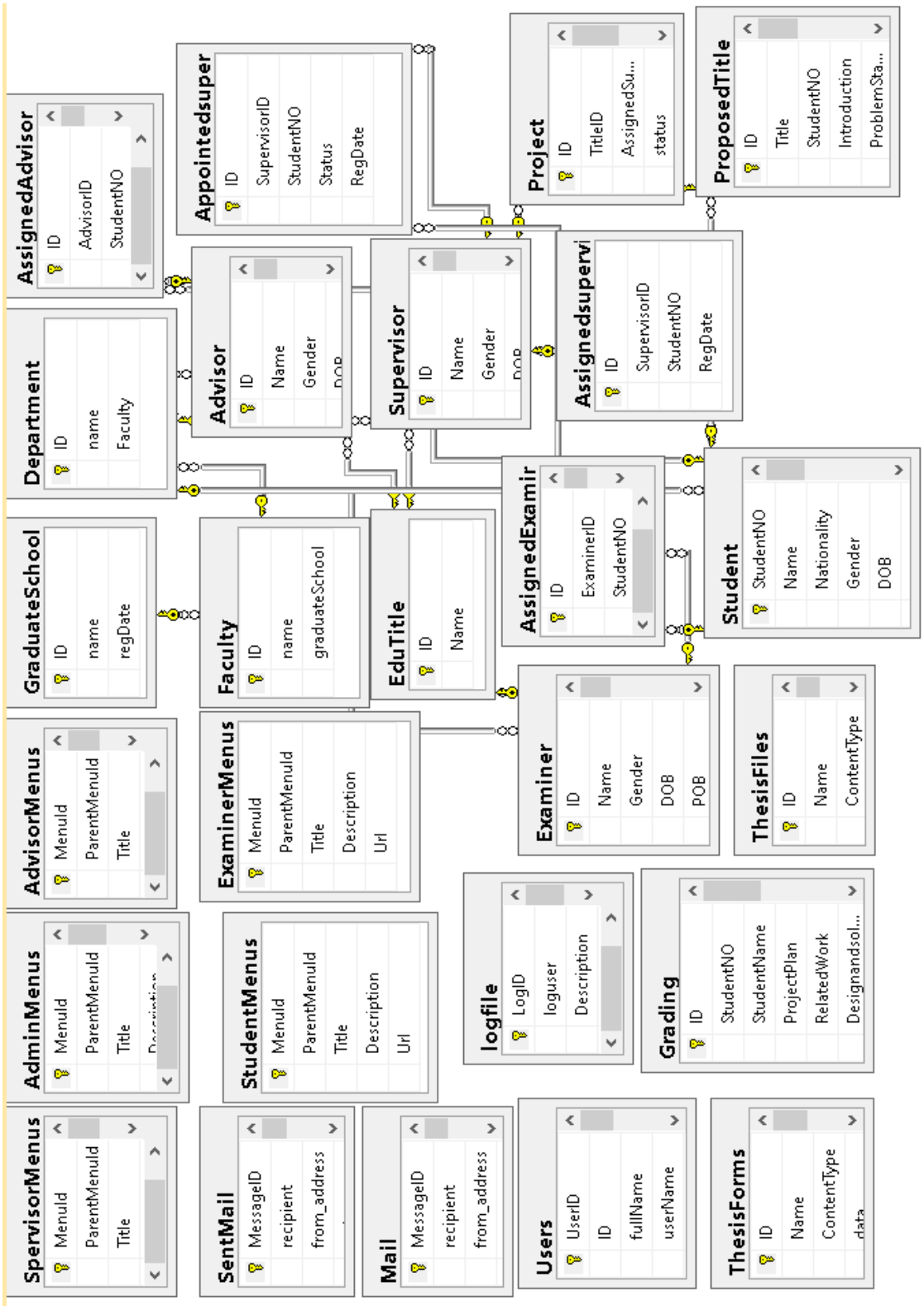


Figure 4.41: Database diagram.

CHAPTER 5

IMPLEMENTATION AND TESTING

5.1. Lab Environment

The implementation and testing of web-based repository system for graduation projects is performed in the lab environment shown in Figure 5.1. The test lab is set up with the hardware and software mentioned in Chapter 3.

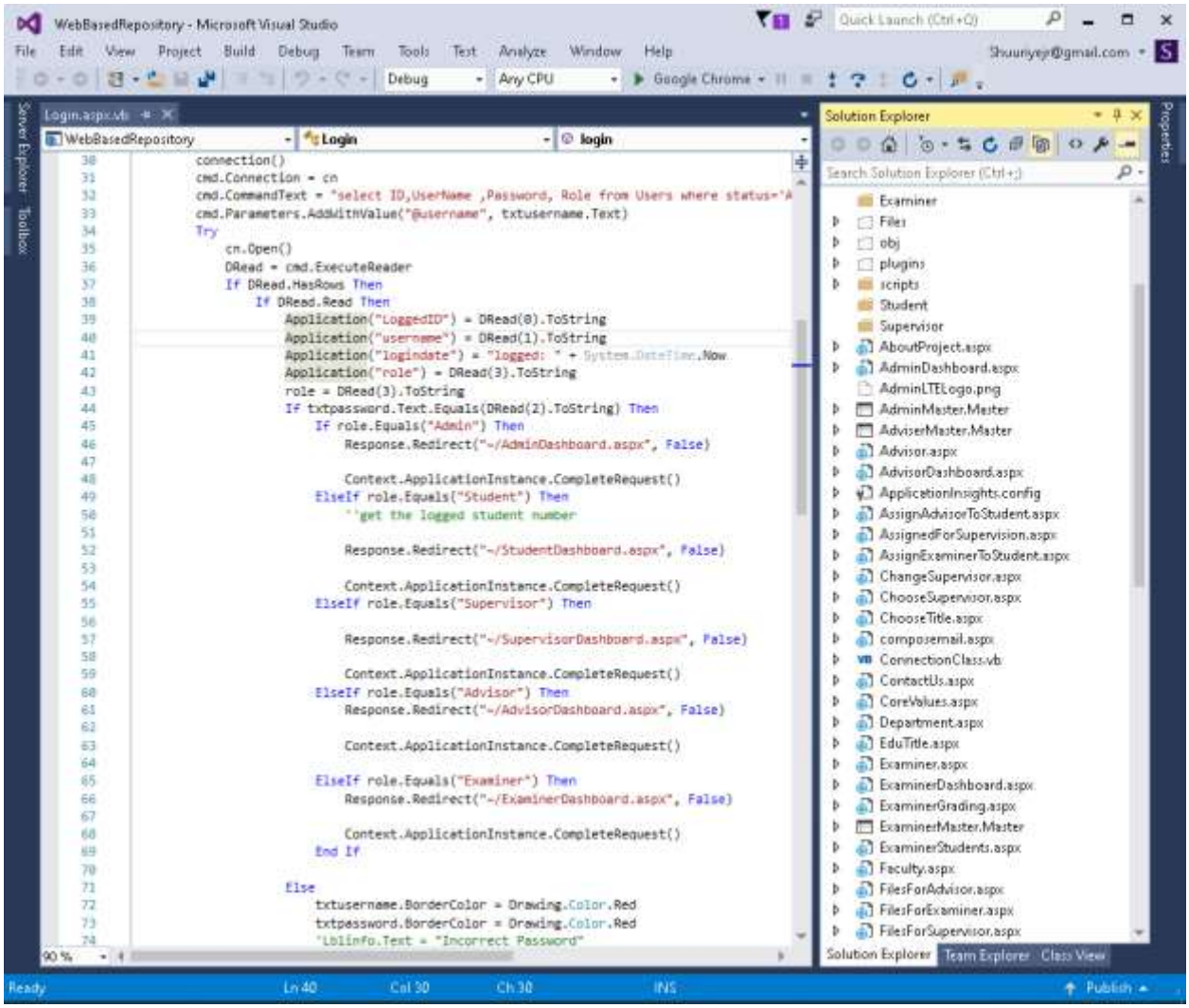


Figure 5.1: Visual Studio 2015 IDE.

5.2. Start Page

The first page that welcomes you when you visit our website is the star page as shown in Figure 5.2.

Figure 5.2: Start Page

5.3. Thesis list page

You can search projects by department or by academic year or both of them, no matter whether you have a username and password or not.

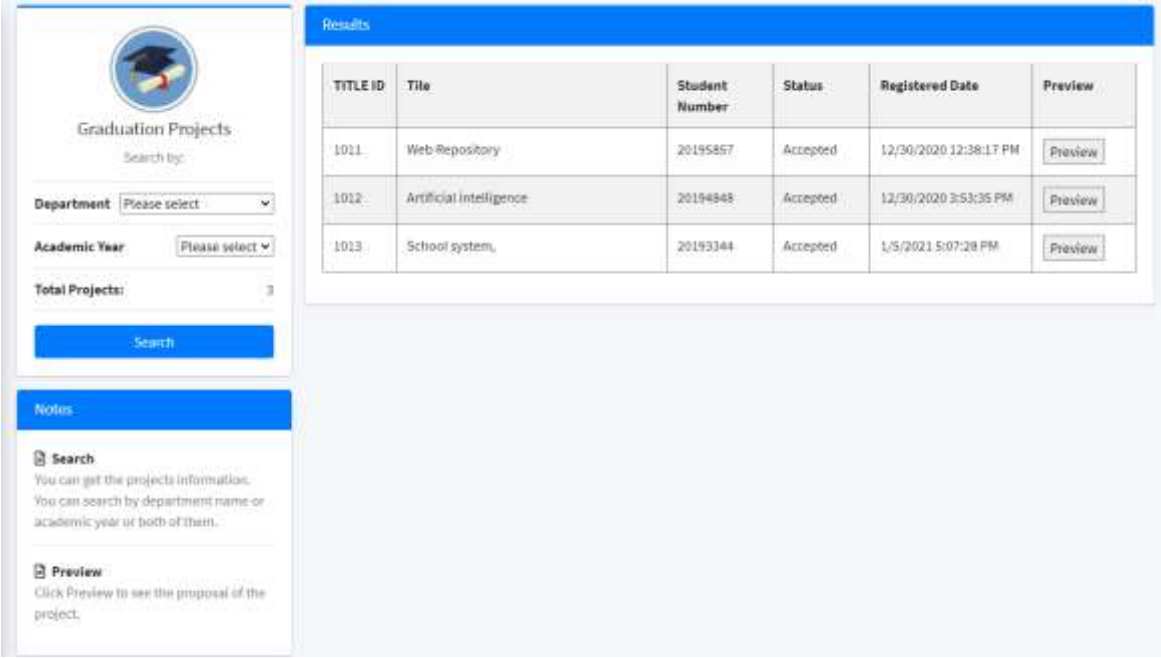


Figure 5.3: Thesis List Page

You can view the details of the title that attracts you as shown in Figure 5.4.

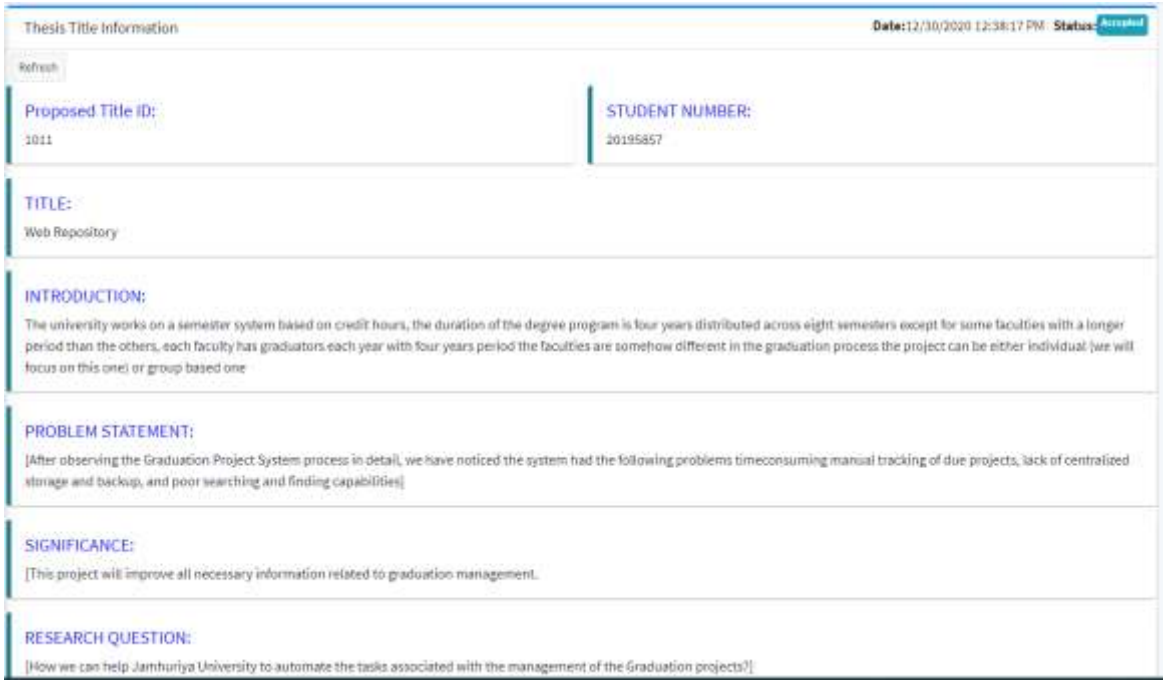


Figure 5.4: Thesis Details Page or proposal.

5.4. Login page

This is the page that allows you to login the system if you are registered user with username and password.

Figure 5.5: Login Page.

Every user in this system has a role, he/she will be redirected to his/her appropriate home page.

If the user forgets his/her password, he/she will recover easily using forgot page.

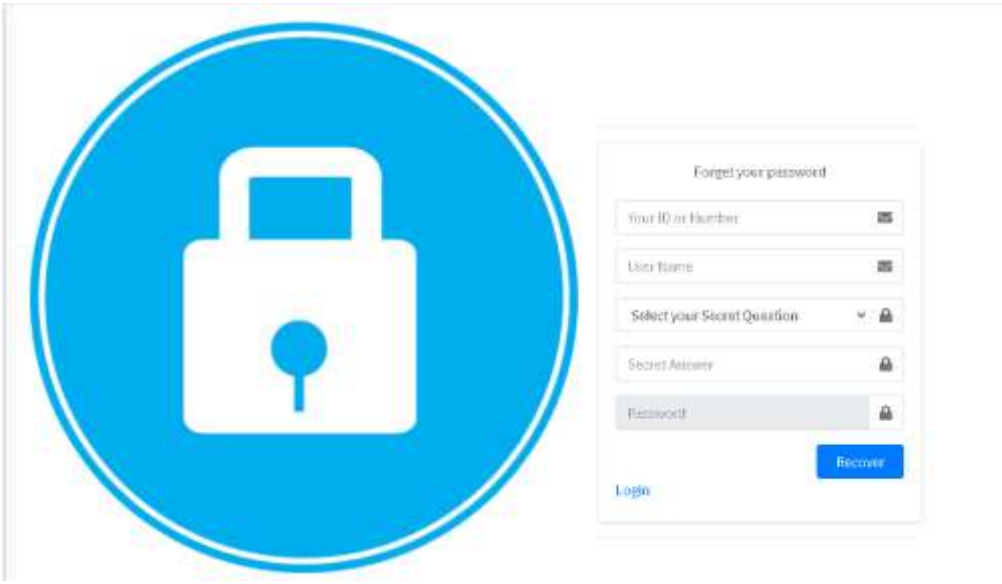


Figure 5.6: Forgot password Page.

5.5. Admin Pages

The admin is the one who maintains and manages the website in order to operate.

5.5.1. Admin dashboard

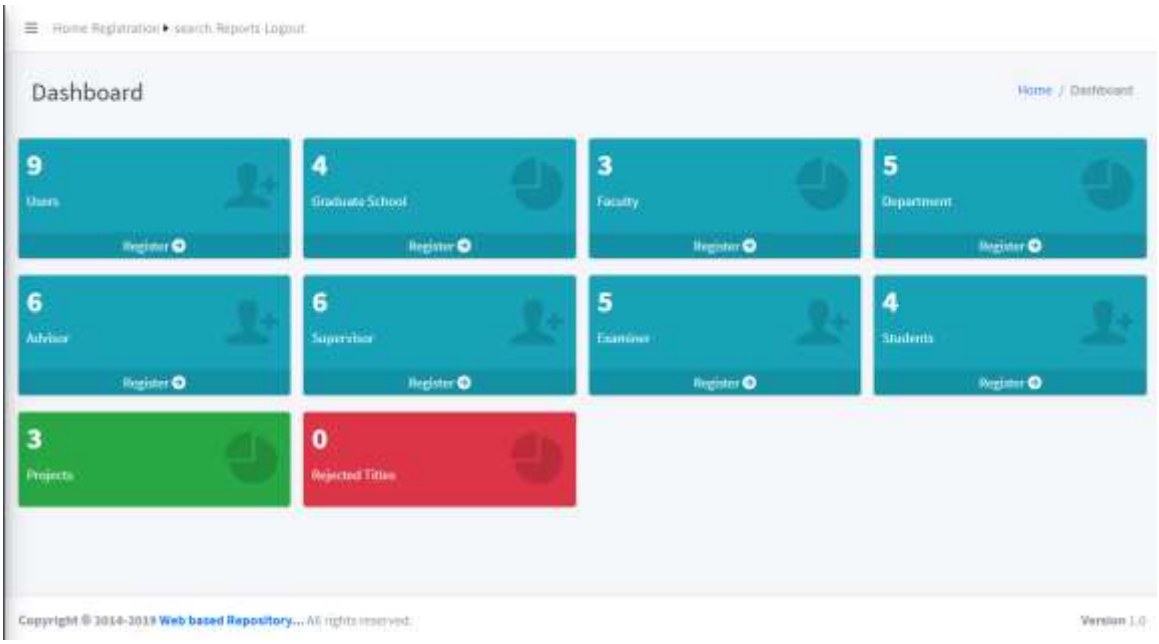


Figure 5.7: Admin Dashboard.

5.5.2. Registration

The admin will register all these members and give those users to access the system.

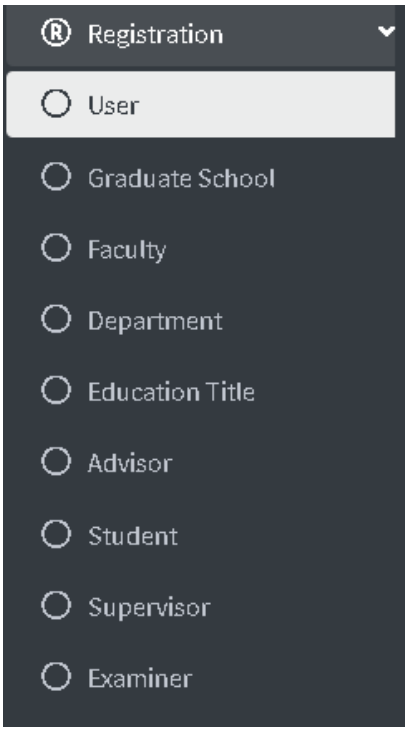


Figure 5.8: Registration.

5.5.3. Assigning examiners or an advisor to a student

The admin will assign one advisor to a student, and three examiners per student.

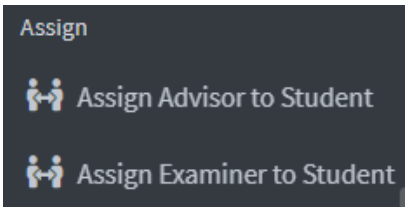


Figure 5.9: Assigning an advisor or examiner to a student.

If the admin tries to assign more than one advisor this error will appear

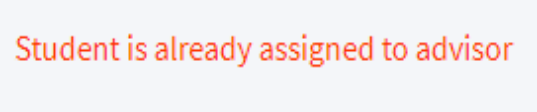


Figure 5.10: Error student already assigned to an advisor

The same happens when the admin tries to assign more than three examiners for the student

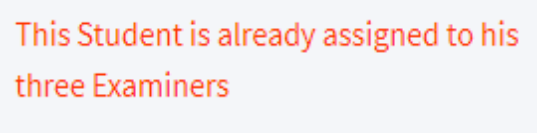


Figure 5.11: Error student already assigned to his/her three examiners.

5.6. Student pages

5.6.1. Student homepage

If the student log in the system he/she will be redirected to the student homepage

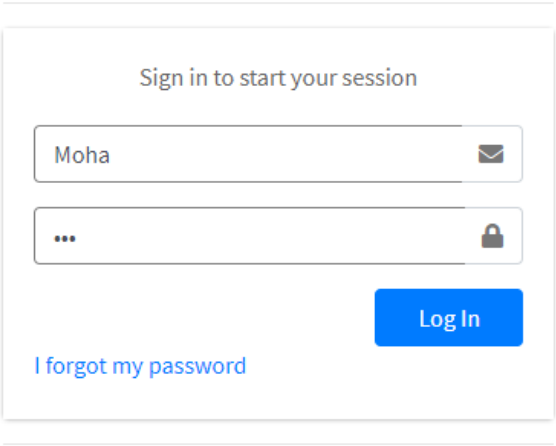


Figure 5.12: Student Login

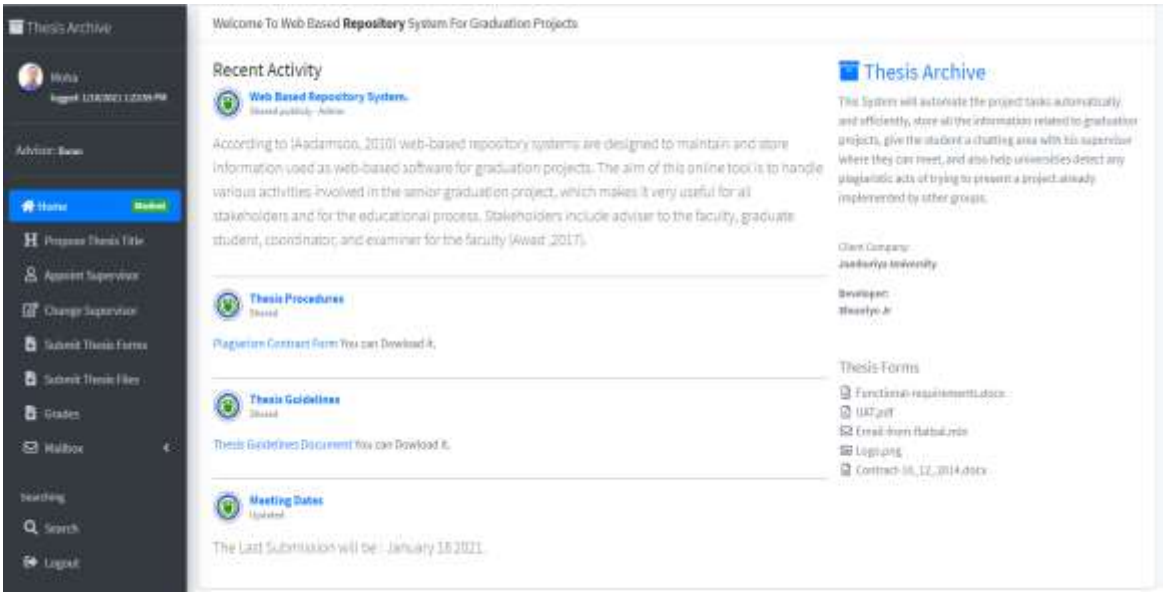


Figure 5.13: Student dashboard

5.6.2. Propose a title

The student will propose a title after logged in successfully. he/she will provide the information and click register. Once the information was registered, he/she can update only.

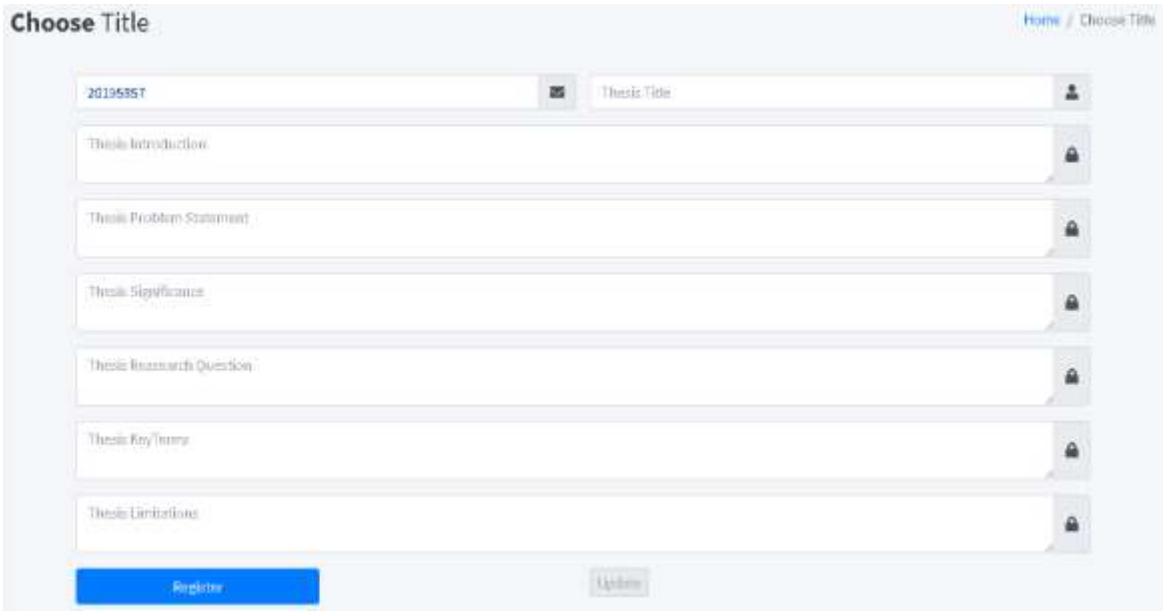


Figure 5.14: Propose a thesis title.

5.6.3. Appoint supervisor

The next step of the student after choosing a title is to appoint a supervisor. If the student already appointed a supervisor an error will display.

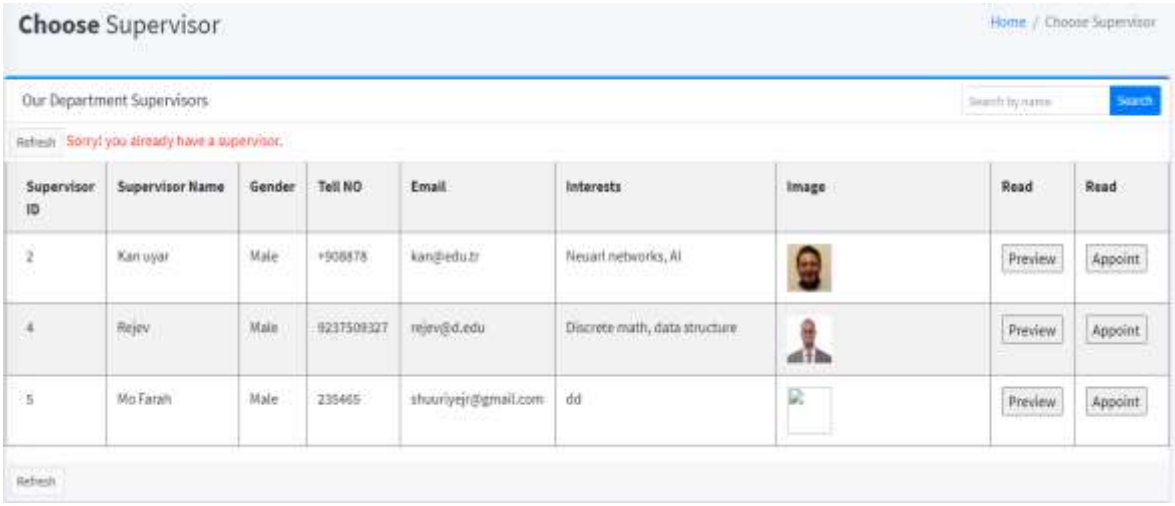


Figure 5.15: Appoint a supervisor.

The student can view the supervisor details by clicking the preview button.

Figure 5.16: Preview supervisor information.

5.6.4. Change supervisor

The student can change his/her appointed supervisor.

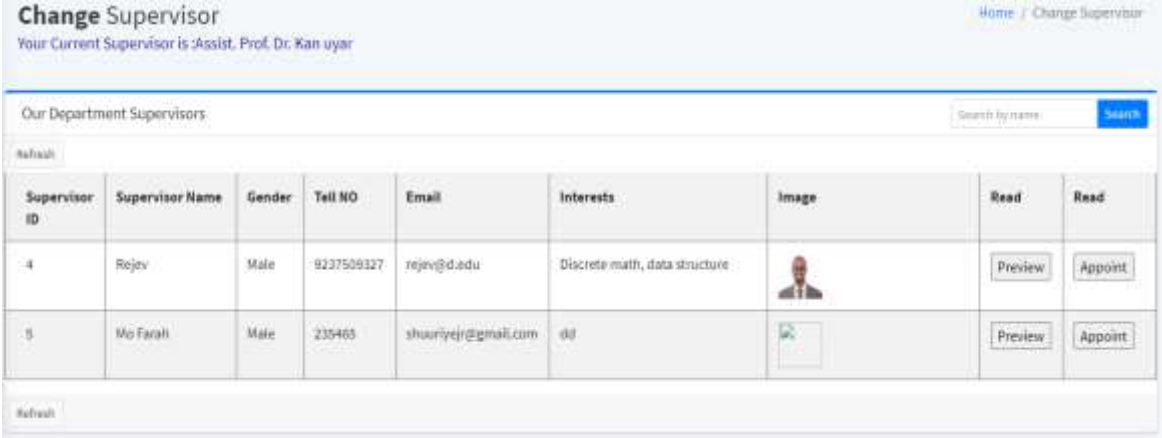


Figure 5.17: Change Supervisor.

The student can also preview supervisor details if he/she needs to know more about this supervisor.

Figure 5.18: Preview supervisor information.

5.6.5. Submit forms

If the supervisor accepted to be a supervisor for this student, he/she will begin to submit forms.

Figure 5.19: Submit thesis Forms.

5.6.6. Submit files

The student should submit thesis files.

Figure 5.20: Submit thesis Files.

5.6.7. View my grades

The student can view his/her grades and print it.

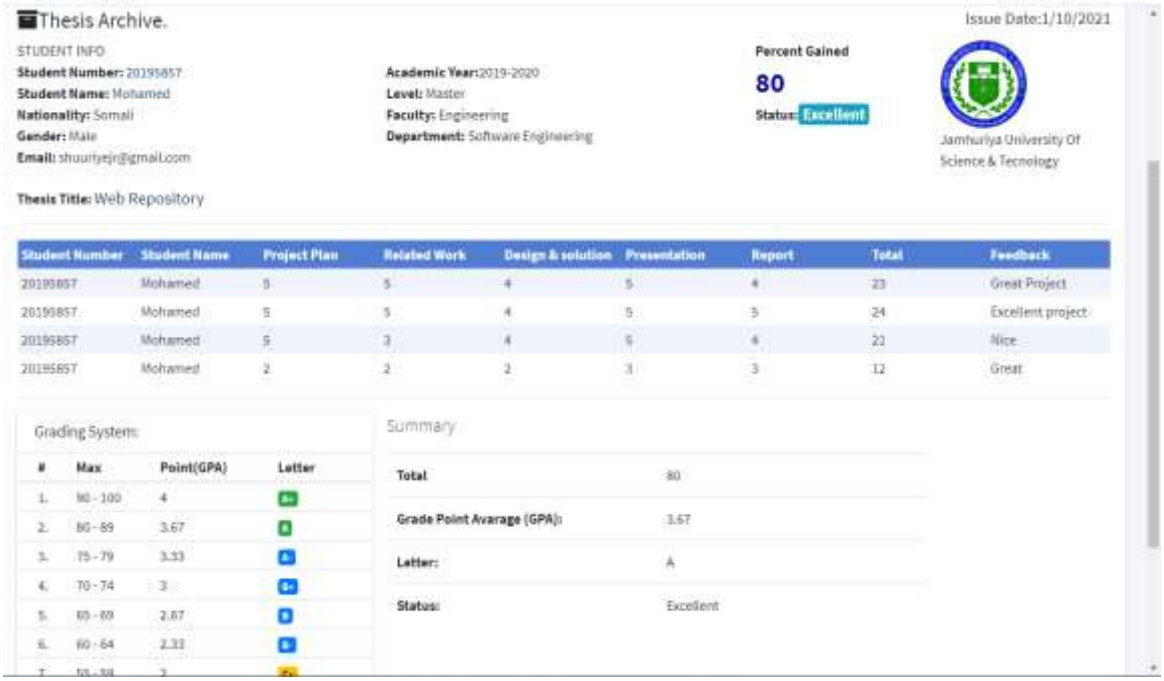


Figure 5.21: student grades.

When the student click button print this Print preview page will display:

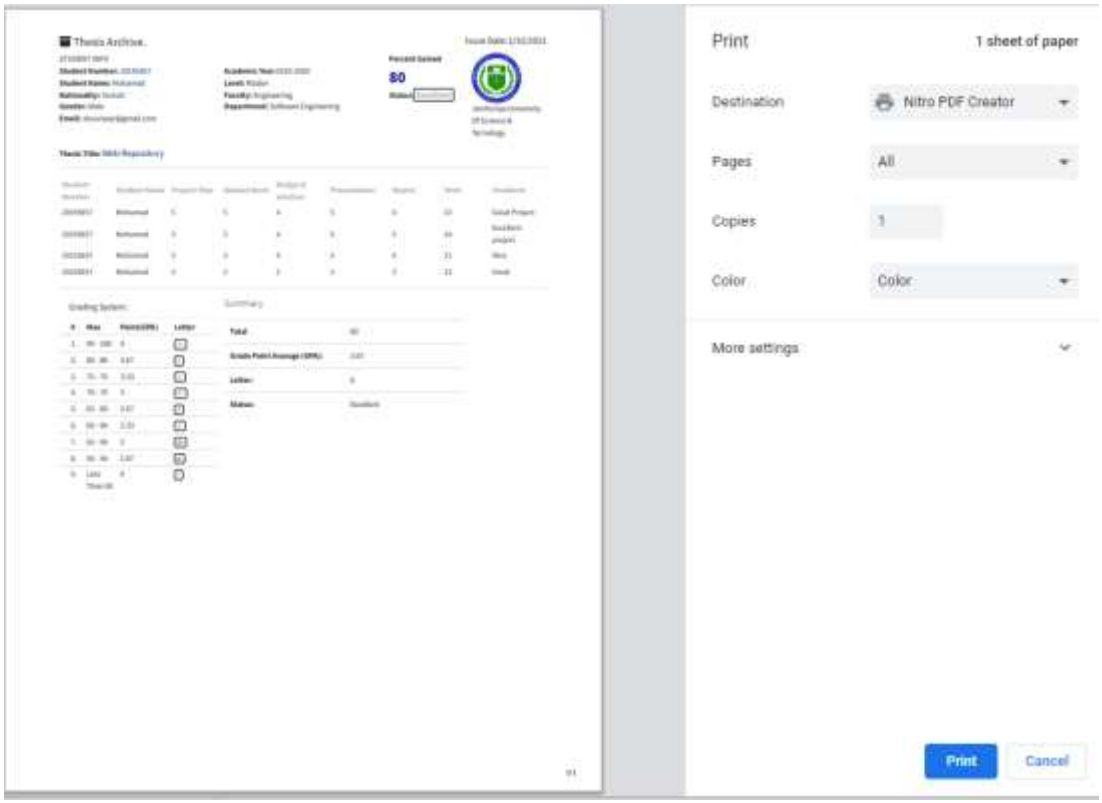


Figure 5.22: student grades print preview.

5.7. Supervisor Pages

If the supervisor login the system he/she will be redirected to supervisor home page.

5.7.1. Supervisor dashboard

Supervisor dashboard gives the supervisor a summary about how many students he/she supervised until now, the titles of the graduates, the pending list for supervisor request and the proposed titles.

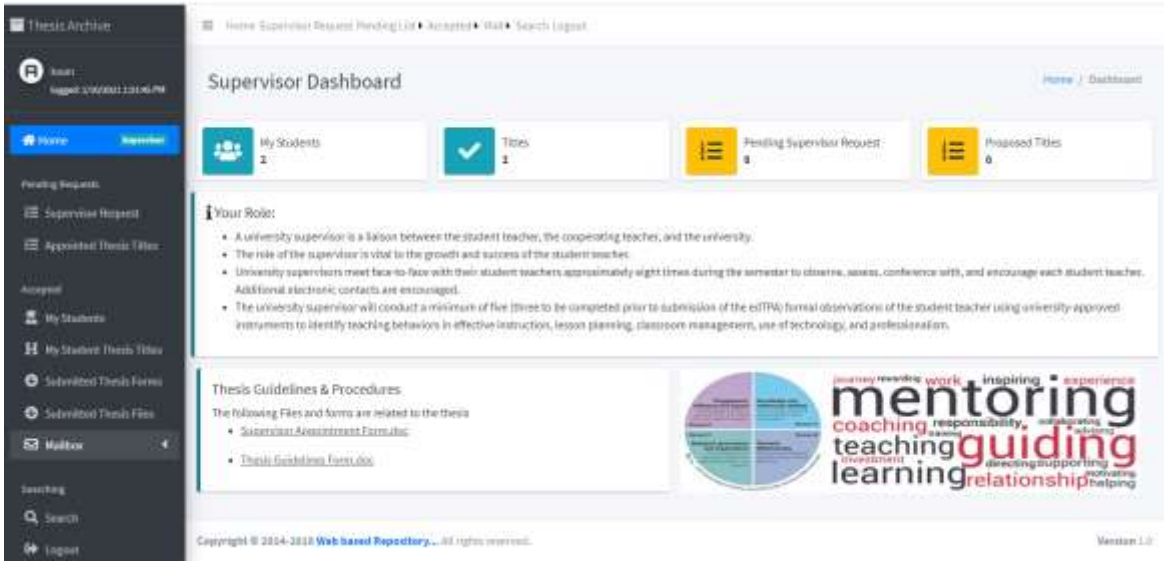


Figure 5.23: Supervisor Dashboard.

5.7.2. Pending supervisor request

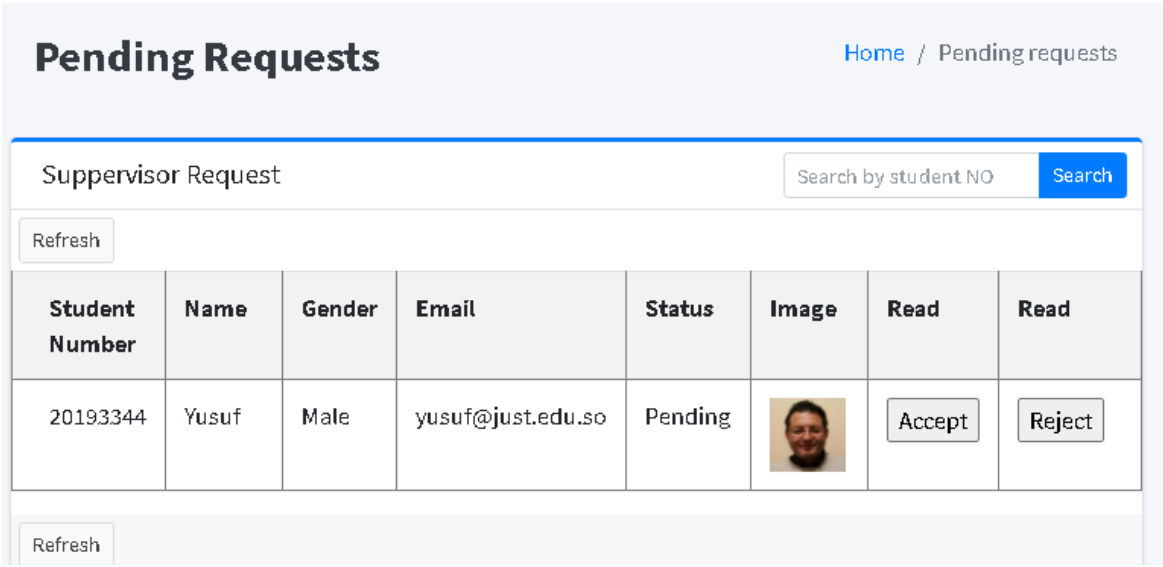


Figure 5.24: Pending Supervisor Request.

5.7.3. Proposed titles

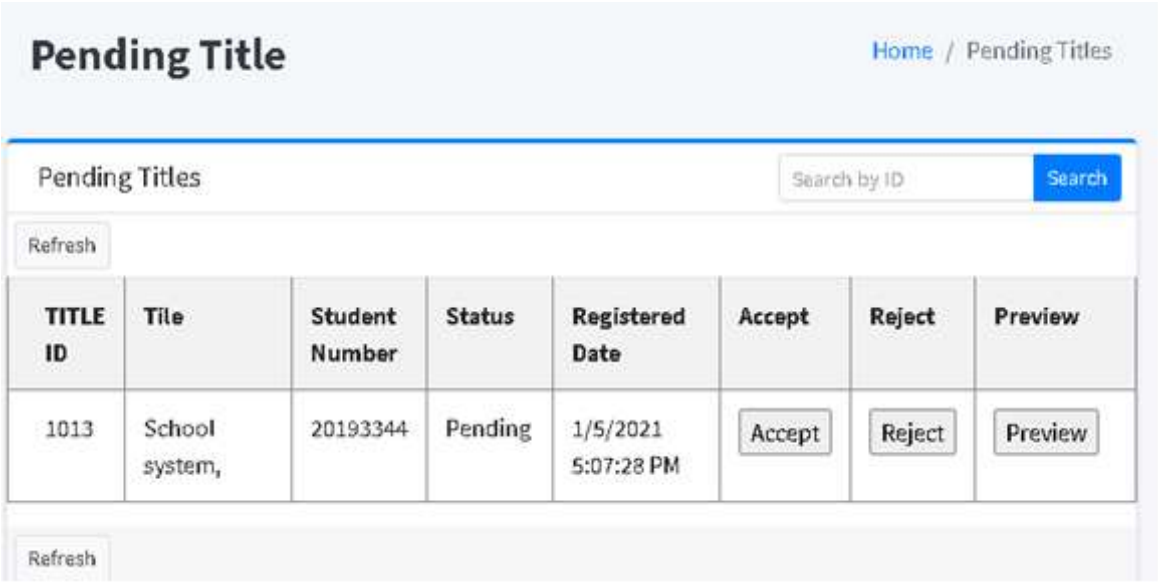


Figure 5.25: Pending Proposed Thesis titles.

5.7.4. Submitted thesis files

When the supervisor receives the thesis files, he/she has to accept or reject, and also, he/she can download it.

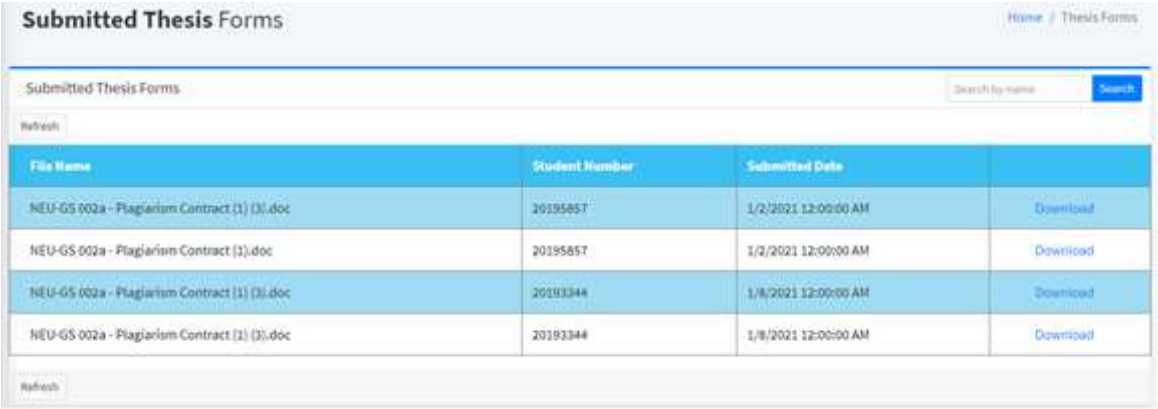


Figure 5.26: Submitted thesis files.

5.7.5. Supervisor accepted students

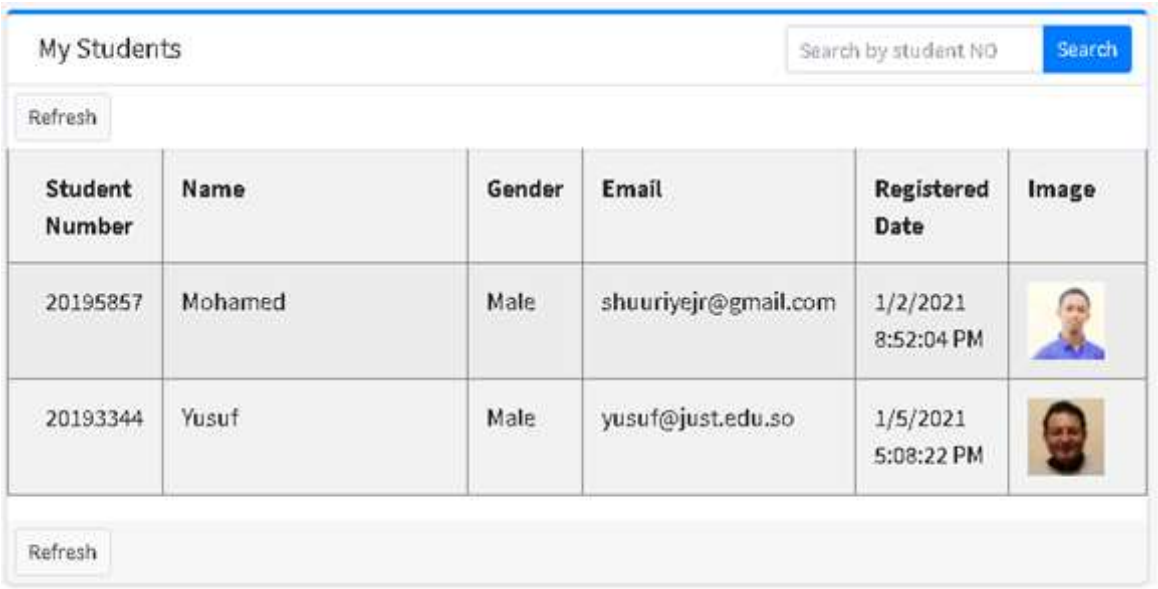


Figure 5.27: Supervisor Accepted Students.

5.8. Advisor Pages

The advisor can login the system to contact his/her students and give grades.

5.8.1. Advisor dashboard

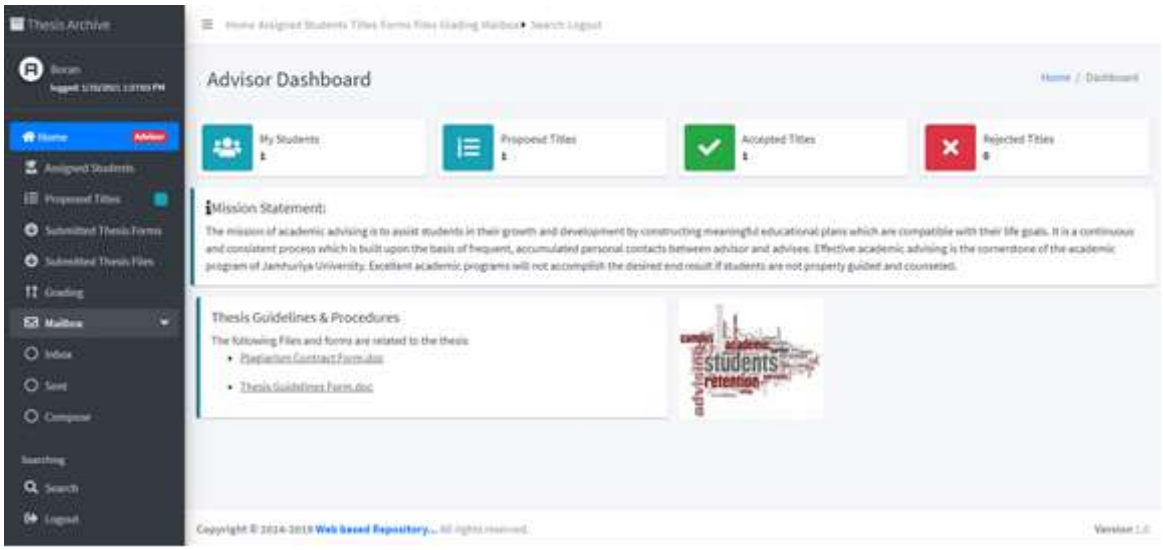


Figure 5.28: Advisor Dashboard

5.8.2. Advisor assigned students

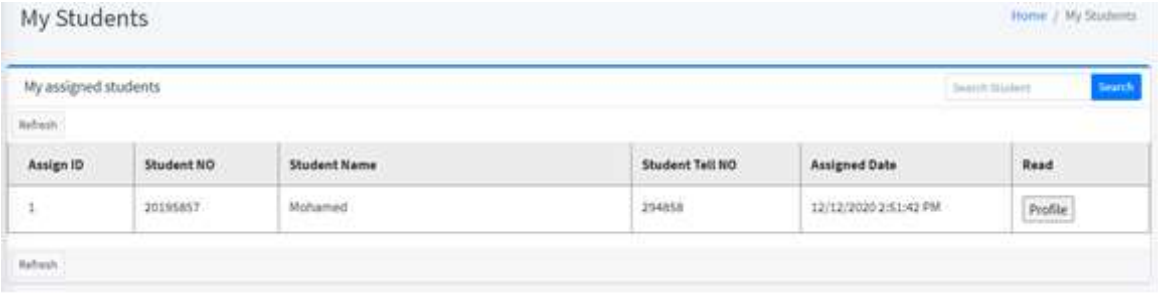


Figure 5.29: Advisor assigned Students.

5.8.3. Proposed titles

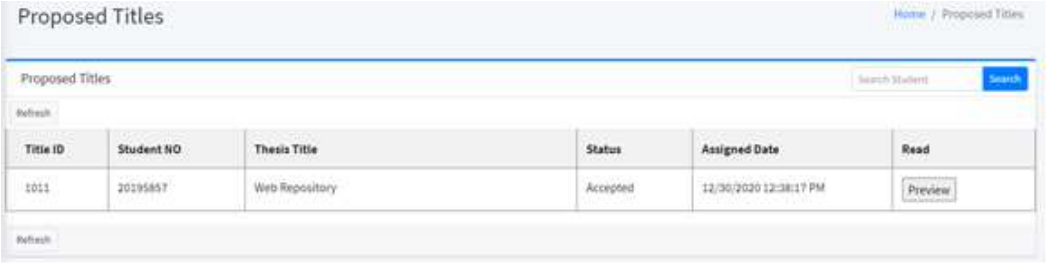


Figure 5.30: Students Proposed Titles

5.8.4. Submitted forms

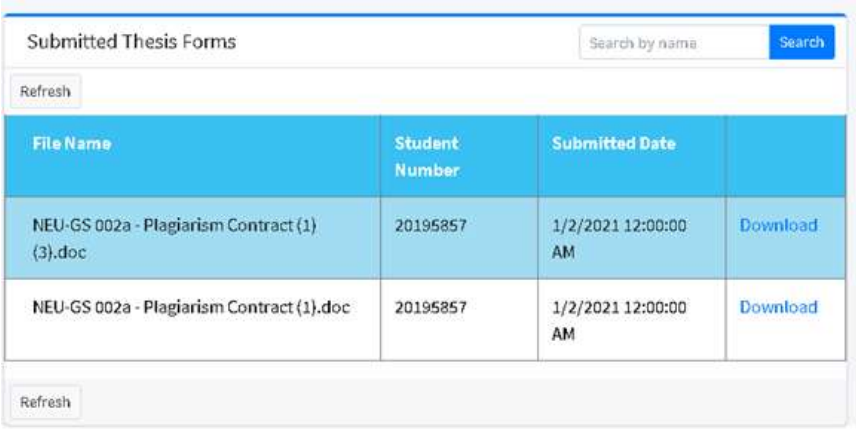


Figure 5.31: Students Submitted Thesis forms

5.8.5. Submitted files

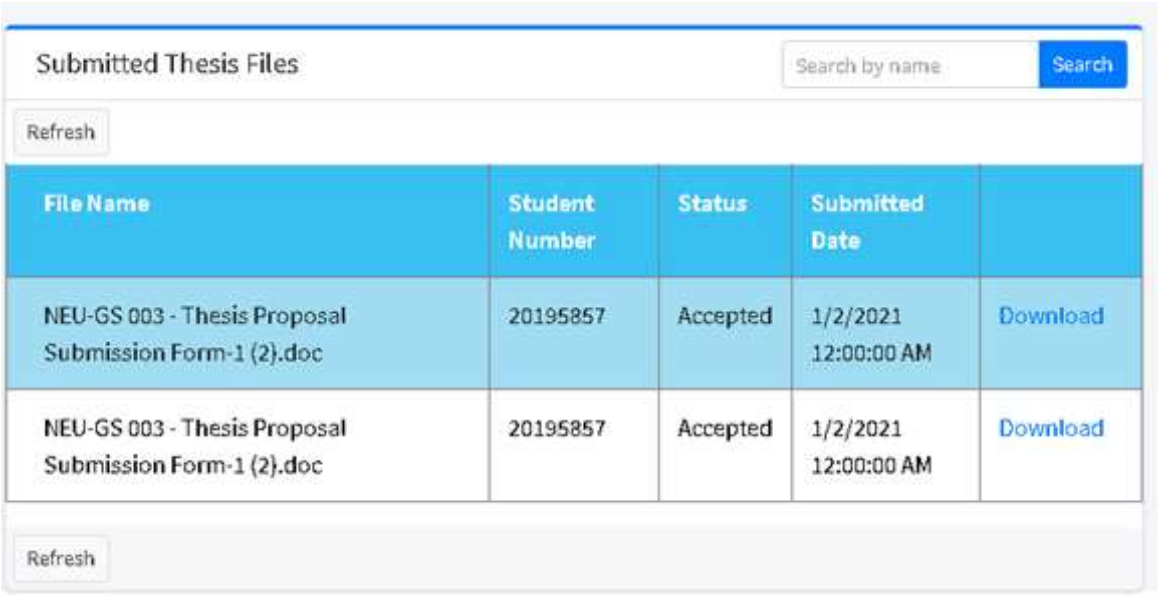


Figure 5.32: Students Submitted Thesis Files.

5.8.6. Advisor grading

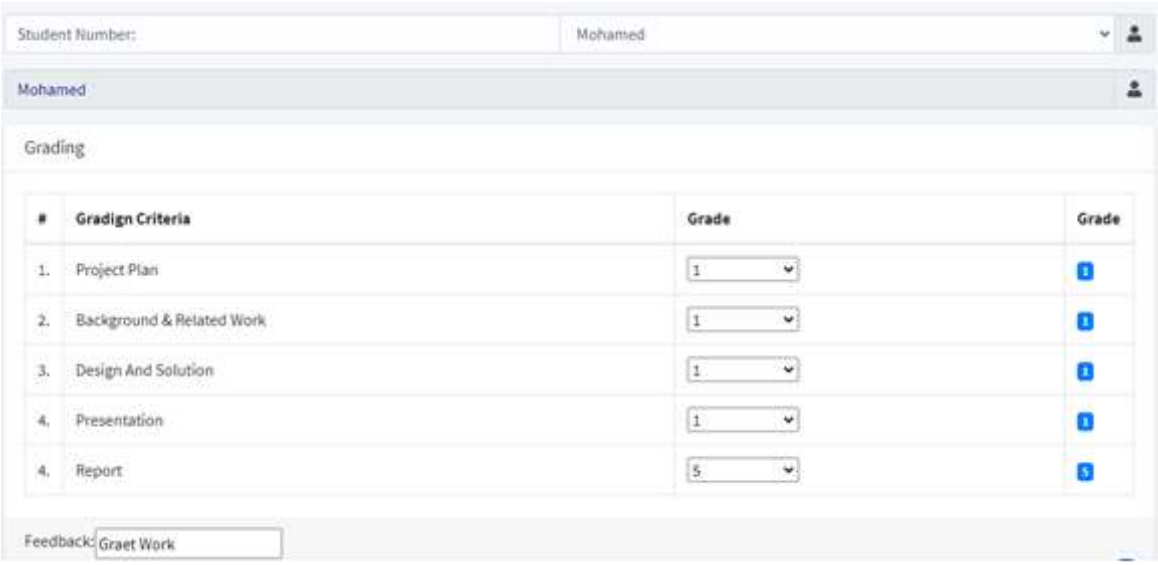


Figure 5.33: Advisor Grading.

5.9. Examiner Pages

The examiners are the graduation committee they give grades to students. Every student has three examiners and one adviser.

5.9.1. Examiner dashboard

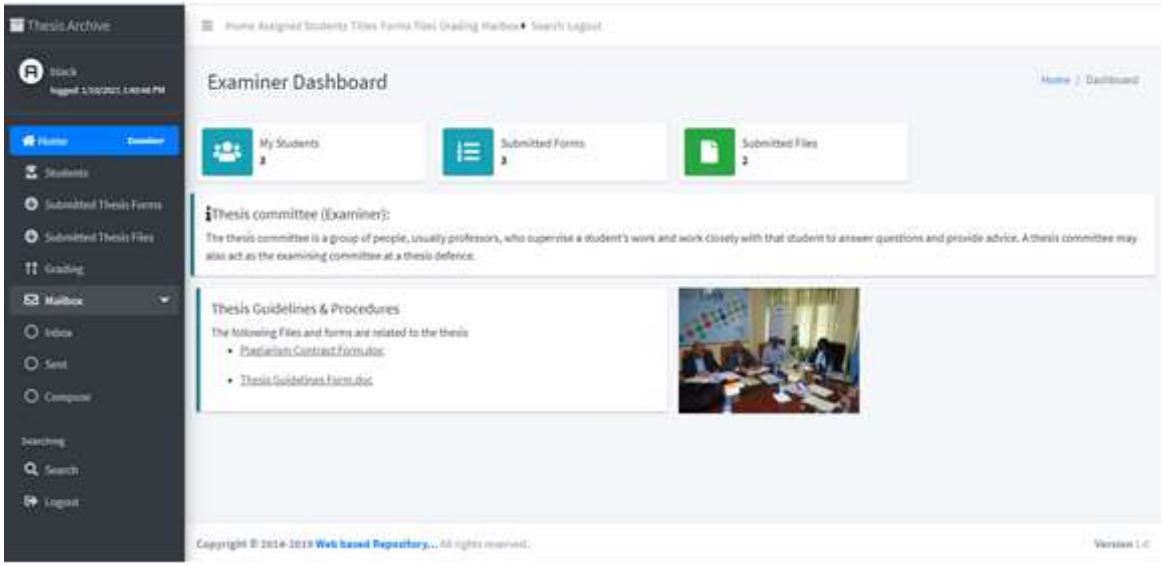


Figure 5.34: Examiner Dashboard.

5.9.2. Examiner assigned students

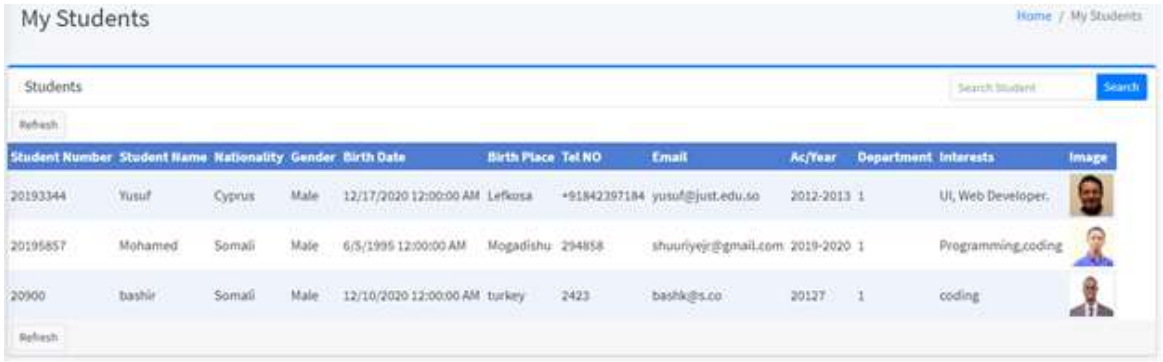


Figure 5.35: Examiner Assigned Students.

5.9.3. Submitted forms

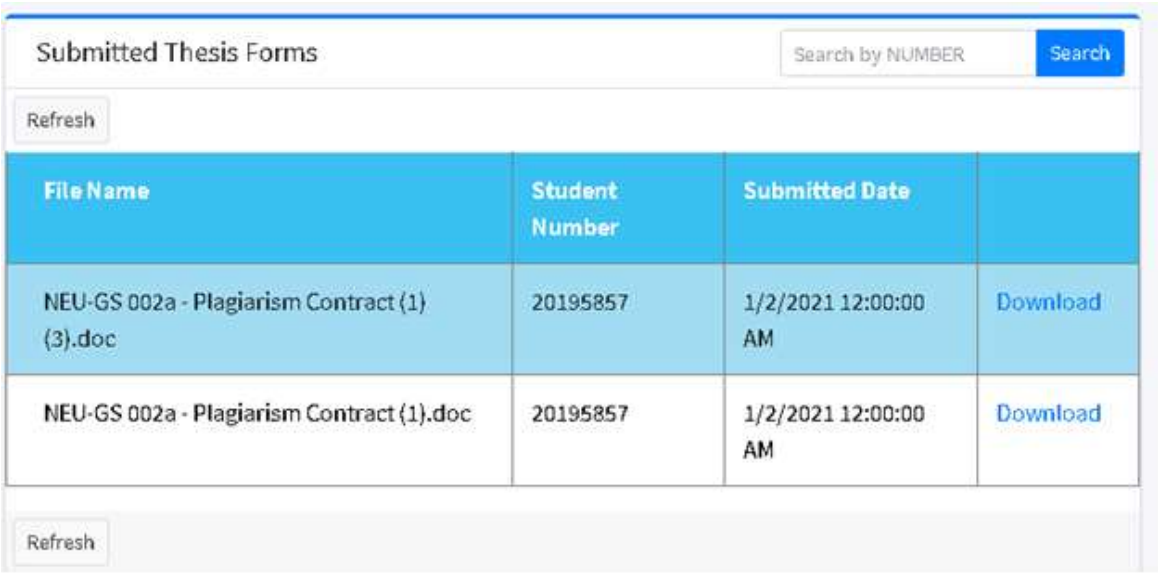


Figure 5.36: Submitted thesis forms.

5.9.4. Submitted files

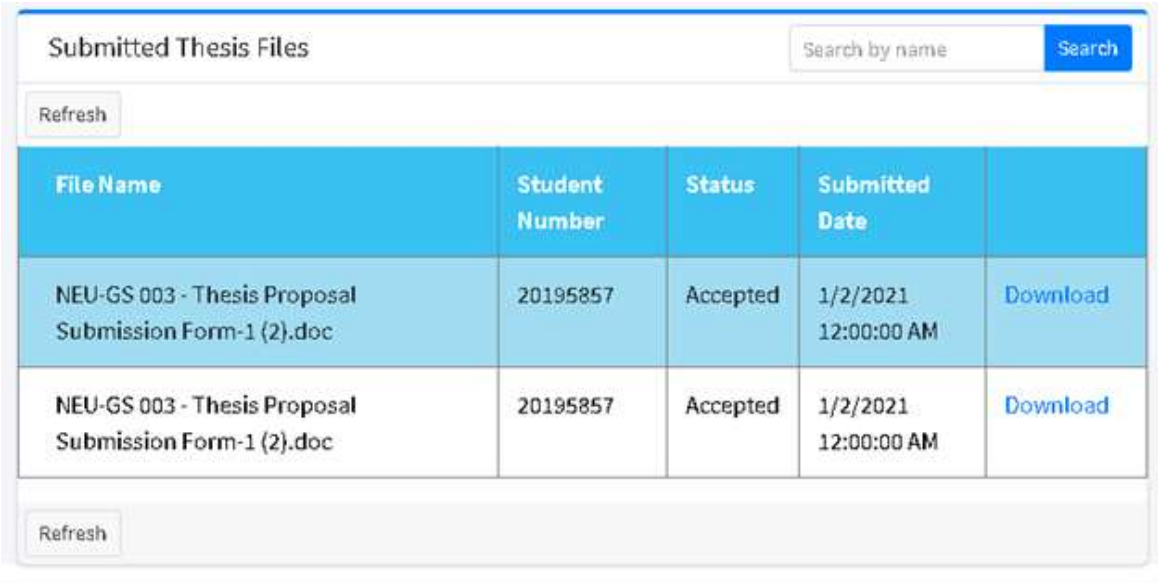


Figure 5.37: Submitted thesis files.

5.9.5. Examiner grading

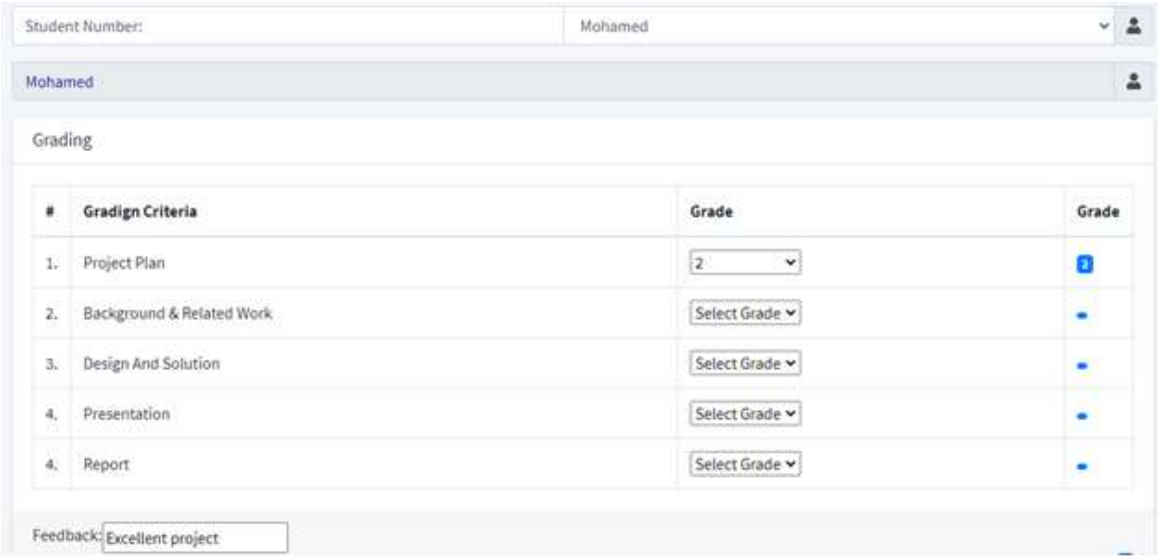


Figure 5.38: Examiner Grading

5.10. Mail System

Mailing system is developed inside our website. Everyone can send and receive messages.

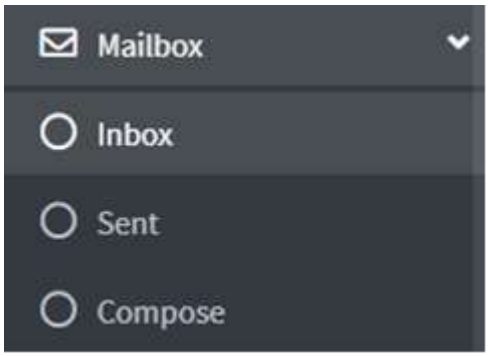


Figure 5.39: Mailing components

5.10.1. Inbox

The received messages will be listed here.

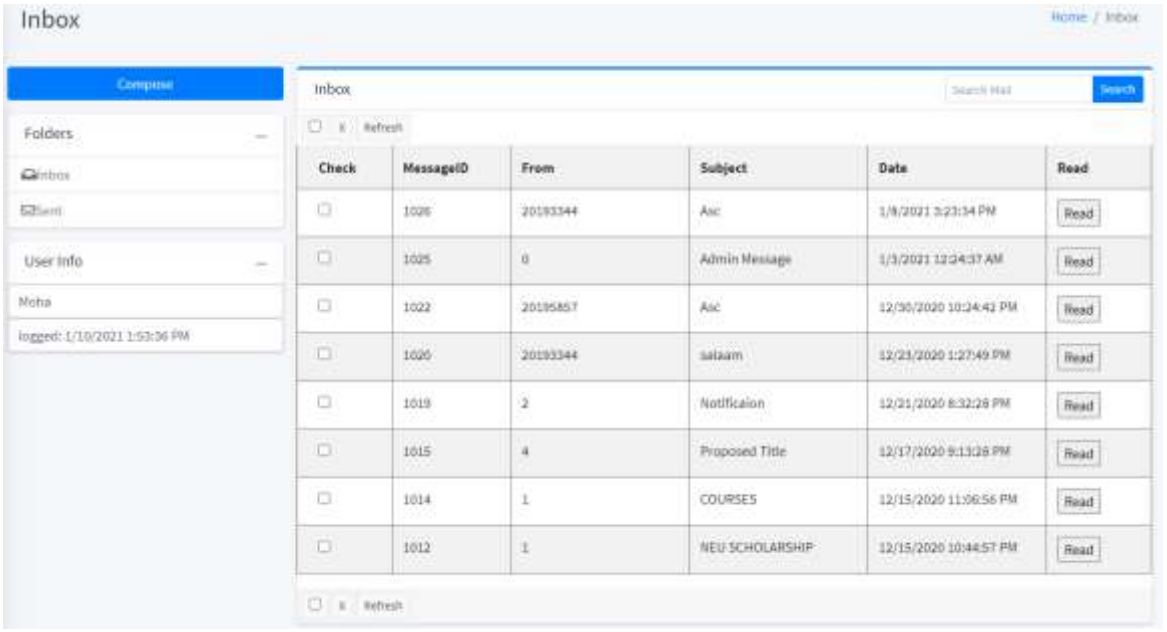


Figure 5.40: Inbox

You can read your inbox messages

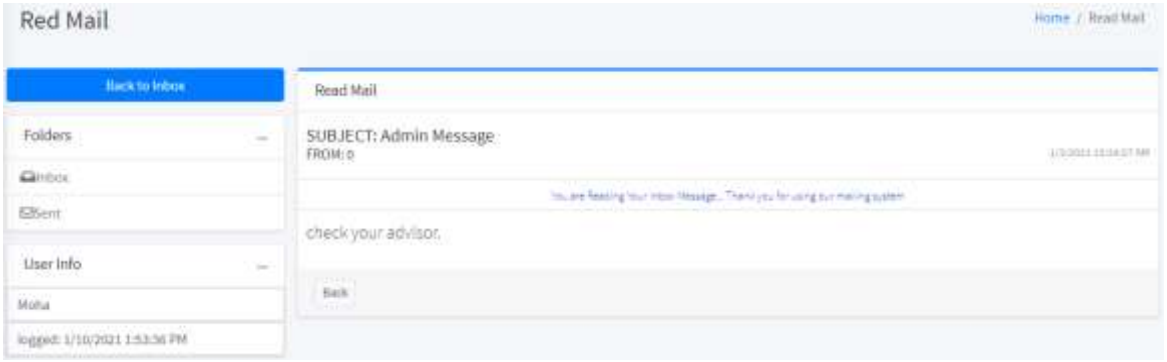


Figure 5.41: Read Inbox Messages.

5.10.2. Sent

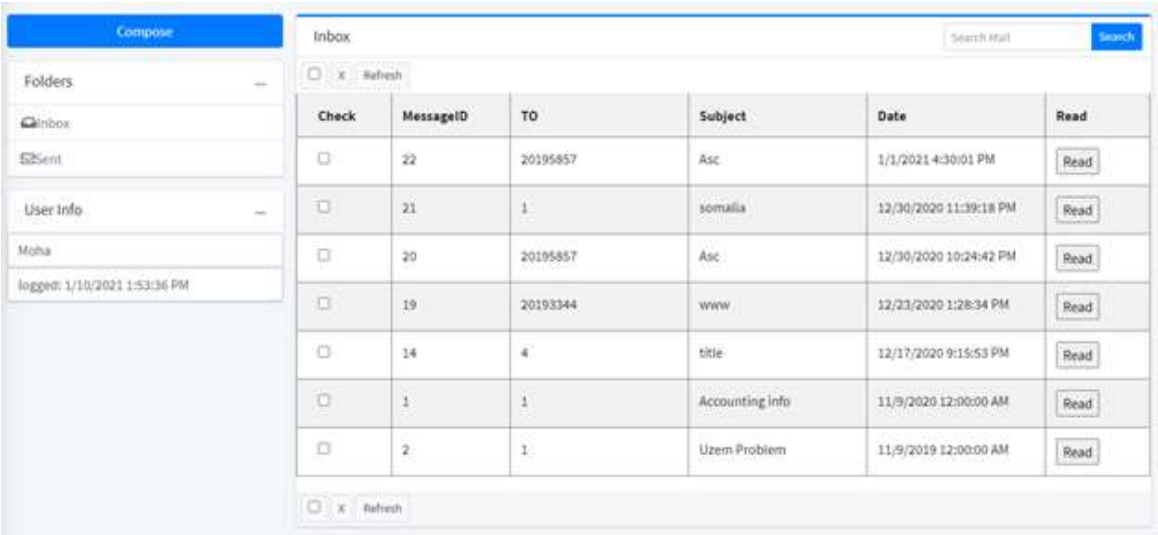


Figure 5.42: Sent Messages

You can also read your sent messages in the list

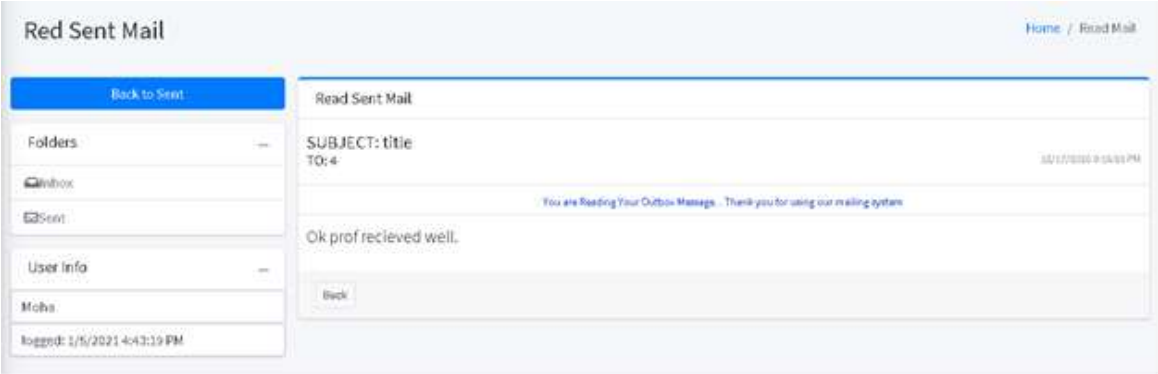


Figure 5.43: Read Sent Messages

CHAPTER 6

CONCLUSION & RECOMMENDATIONS

6.1. Conclusion

The graduation project tasks including storing and retrieving were originally recorded in a spreadsheet and also have shortage of security, data sometimes may happen to lose, and data attenuation, but security, availability and the confidentiality of graduation projects information were not dealt with in early implementations of these services.

The graduation projects information security concern is raised especially while it is stored in a spreadsheet. Another concern was raised if the user loses his/her computer, where the information already stored are at risk especially if the computer storage contains sensitive information about these projects.

The main objective of this work was to moderate security threats of graduation projects information by protecting the information confidentiality, integrity, authentication and non-repudiation. Another motive of this research was to provide a means for protecting saved information.

a) Accomplishment of objectives

In this section we restate the research objectives and we describe how each of them was achieved.

Objective 1

To develop online graduation projects management system that will allow to automate the projects related tasks automatically and efficient.

- We analyzed the tasks related to graduation projects information and demonstrated how to develop online web-based repository system which have the security levels specifically confidentiality, integrity, and authentication.

- The system has also some Constraints that improves the security such as: all users must have accessed the system with valid usernames and passwords, and all data accessed by the system must be stored in a database.

Objective 2

To develop online repository for graduation projects.

- We developed online web-based repository system to facilitate the management of graduation projects, and also to solve the security issue. The project will also be useful for the students for getting easily more information and accurate about graduation titles.

Objective 3

To help universities detect any plagiaristic acts of trying to present a project already implemented by other groups.

- Although the system can store information about projects already implemented by groups, other groups cannot be assigned to that project.

- Each project can be assigned only to one student.

6.2. Recommendations

Some enhancements can be made to this online web-based repository system for graduation projects in the future in terms of adding new features or improving features it already implemented.

Although online web-based repository system does not have originality checking of the submitted files adding this feature is recommended as future enhancement. To expand this work a prevention technique may be added from SQL-injection attacks and an offline framework can be implementing to use when the internet went off.

REFERENCES

Aadamsoo, A.-M. (2010). WEB BASED PROJECT MANAGEMENTSYSTEM. Technology and Communication.

Amy Affleck, Aneesh Krishna, Narasimaha R. Achuthan, Non-Functional Requirements Framework: A Mathematical Programming Approach, The Computer Journal, Volume 58, Issue 5, May 2015, Pages 1122–1139, https://doi.org/10.1093/comjnl/bxu027

Andrew, Th. (2004). Theses Alive. An E-theses management system for the UK. University of Edinburgh

Awad, M., (2017) GPMS: An educational supportive graduation project management system. Comput Appl Eng Educ. 2017; 25: 881– 894. https://doi.org/10.1002/cae.21841

Blanchard, C. (2013). *Avoiding senioritis: Student perceptions of engagement and efficacy during senior project* (Order No. 3556910). Available from ProQuest Dissertations & Theses Global. (1330391092). Retrieved from https://search.proquest.com/dissertations-theses/avoiding-senioritis-student-perceptions/docview/1330391092/se-2?accountid=15309

Brandon Gaille. (2019). 15 Desktop vs Web Application Pros and Cons. Retrieved November 10, 2020, from https://brandongaille.com/15-desktop-vs-web-application-pros-and-cons/

C.laudon, K. (2014). E-commerce 2014 business. Technology. Society. Pearson Education.

Donald Firesmith, Modern Requirements Specification, Journal of Object Technology, Vol. 2, No. 1, March-April 2003.Google Scholar

Easttom, C. (2012). Computer Security Fundamentals. Second Edition: Pearson Education.

Evjen, B. (2010). Professional ASP.neT 4 in C# and VB. Wiley Publishing, Inc

Exoft (2017). Desktop or web application: what to develop. Retrieved November 19,2020 from https://exoft.net/desktop-or-web-application-what-to-develop/

Geeksforgeeks (2019). Types of Information Syste. Retrieved November 10,2020, from https://www.geeksforgeeks.org/types-of-information-system/

Golden Grid System, (2020). Web Development Processes and Technical Environments.

Retrieved from https://goldengridsystem.com/web-development-processes-and-technical-environments/

Graduation Project. (2020). Retrieved November 10, 2020, from https://www.oxfordasd.org/Page/609 .

International Data Corporation. (2001, April). ETRENDS, eLearning is burgeoning. Retrieved from the World Wide Web: www.idc.com.

Jäger D., Schleicher A., Westfechtel B. (1999). Using UML for Software Process Modeling. In: Nierstrasz O., Lemoine M. (eds) Software Engineering — ESEC/FSE '99. ESEC 1999, SIGSOFT FSE 1999. Lecture Notes in Computer Science, vol 1687. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-48166-4_7

Jinfonet, (2020). What is a 3-Tier Architecture? Retrieved from https://www.jinfonet.com/resources/bi-defined/3-tier-architecture-complete-overview/

Khanore, S., & Patil, R., & Dand, H. (2011). Management information system, Institute of Distance and Open Learning, University of Mumbai.

KHELIFI, A., AL-RAJAB, M., KAREM, S., SUBHI, L. (2012). Graduation Project Online Management System. Recent Researches in Software Engineering, Parallel and Distributed Systems.

Komka, J., (2011). Information System of University: Goals and Problems. Daunoravicius Vilnius Gediminas Technical University.

Laudon, K. & Laudon, J. (2006) Management Information Systems: Managing the Digital Firm, 9th ed. Prentice Hall.

Laudon, K. (2012). Management Information Systems. Pearson Education.

Laudon, K. C., Laudon, J. P. (2014). Management Information Systems Managing the Digital Firm. 13th EDITION.

Laudon,K.C, & Laudon,K. J. (2012). Management Information Systems. Pearson Education.

Middle Georgia State University, (2020). Web Development Environment. Retrieved from http://itwebtutorials.mga.edu/php/chp1/web-development-environments.aspx

Nowduri1, S., & Al-Dossary, S. (2012). Management Information Systems and Its Support

to Sustainable Small and Medium Enterprises International Journal of Business and Management; Vol. 7, No. 19, pp. 125–131.

O'Brien, J.A., & Marakas, G.M. (2007). Management information systems -10th ed., by McGraw-Hill/Irwin, a business unit of The McGrawHill Companies.

OMG, (2017). Unified Modeling Language, Retrieved November 19, 2020, from http://www.uml.org/

Patterson, A. (2005) Information Systems – Using Information, Learning and Teaching Scotland.

Sommerville, (2016). Software engineering book. Pearson education.

Teach ICT - GCSE ICT Revision, Theory Types of computers. (n.d.). Retrieved November 19, 2020, from https://www.teach-ict.com/ks3/year7/data_handling/miniweb/pg3.htm .

Tutorials point. (2014). Management information system, Tutorials Point (I) Pvt. Ltd.

UML Use Case Diagrams, (2014). Retrieved November 19, 2020 from Use case diagrams are UML diagrams describing units of useful functionality (use cases) performed by a system in collaboration with external users (actors). (uml-diagrams.org)

APPENDICES

APPENDIX 1
USER GUIDE

Web based repository system is a website developed by the researcher. This is website is developed for automating the graduation project tasks with minimum effort and less time and effort. It allows the users to interact and communicate each other to handle the tasks probably.

You are required to have a working web browser in your device (Laptop, mobile or tablet). The website is compatible and responsive for all screen sizes. Furthermore, you must have an active internet connection as we are trying to publish this website if it is possible.

In terms of an access, the user must have a valid username and a password to login the system. So, if the user has no username, he/she can visit the website as a guest, search titles based on their departments, academic year and both of them, and also, he/she can preview the proposal or the details of each title.

The grading policy were designed by the university, the examiners will give maximum 5 marks per section. The total grades of the three examiners will be 75 (25 per one), and 25 of the advisor grades, which makes a total of 100 marks.

APPENDIX 2

SOURCE CODES

Login page Code

```
Imports System.Collections.Generic
Imports System.Linq
Imports System.Web
Imports System.Web.UI
Imports System.Web.UI.WebControls
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Imports System.IO
Public Class Login
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim dr As SqlDataReader
Dim strings As New ConnectionClass
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Dim role As String = ""
Sub login()
connection()
cmd.Connection = cn
cmd.CommandText = "select ID,UserName ,Password, Role from Users where status='Active' and UserName=@username"
cmd.Parameters.AddWithValue("@username", txtusername.Text)
Try
cn.Open()
DRead = cmd.ExecuteReader
If DRead.HasRows Then
If DRead.Read Then
Application("LoggedID") = DRead(0).ToString
Application("username") = DRead(1).ToString
Application("logindate") = "logged: " + System.DateTime.Now
Application("role") = DRead(3).ToString
```

```
role = DRead(3).ToString
If txtpassword.Text.Equals(DRead(2).ToString) Then
If role.Equals("Admin") Then
Response.Redirect("~/AdminDashboard.aspx", False)
Context.ApplicationInstance.CompleteRequest()
ElseIf role.Equals("Student") Then
"get the logged student number
Response.Redirect("~/StudentDashboard.aspx", False)
Context.ApplicationInstance.CompleteRequest()
ElseIf role.Equals("Supervisor") Then
Response.Redirect("~/SupervisorDashboard.aspx", False)
Context.ApplicationInstance.CompleteRequest()
ElseIf role.Equals("Advisor") Then
Response.Redirect("~/AdvisorDashboard.aspx", False)
Context.ApplicationInstance.CompleteRequest()
ElseIf role.Equals("Examiner") Then
Response.Redirect("~/ExaminerDashboard.aspx", False)
Context.ApplicationInstance.CompleteRequest()
End If
'ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
'  position 'top-end',
'  icon: 'success',
'title: 'You logged in successfully',
'showConfirmButton: false,
'  timer: 1500
'}));", True)
"insert logfile
Logfile()
Else
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Username or password is wrong!'
});", True)
txtusername.BorderColor = Drawing.Color.Red
txtpassword.BorderColor = Drawing.Color.Red
'Lblinfo.Text = "Incorrect Password"
End If
End If
Else
'ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
'  icon: 'error',
'  title: 'Oops...',
'  text: 'Plz fill the Username and Password Fields!'
'});", True)
End If
DRead.Close()
```

```
cn.Close()
Catch generatedExceptionName As Exception
MsgBox(generatedExceptionName.Message)
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not IsPostBack Then
txtusername.BorderColor = Drawing.Color.Gray
txtpassword.BorderColor = Drawing.Color.Gray
txtusername.Text = ""
txtpassword.Text = ""
txtusername.Focus()
End If
End Sub
Protected Sub btnSubmit_Click(sender As Object, e As EventArgs) Handles btnSubmit.Click
If txtusername.Text <> "" Or txtpassword.Text <> "" Then
login()
Else
txtusername.BorderColor = Drawing.Color.Red
txtpassword.BorderColor = Drawing.Color.Red
End If
End Sub
Protected Sub txtpassword_TextChanged(sender As Object, e As EventArgs) Handles txtpassword.TextChanged
txtusername.BorderColor = Drawing.Color.Gray
txtpassword.BorderColor = Drawing.Color.Gray
End Sub
Protected Sub txtusername_TextChanged(sender As Object, e As EventArgs) Handles txtusername.TextChanged
txtusername.BorderColor = Drawing.Color.Gray
txtpassword.BorderColor = Drawing.Color.Gray
End Sub
Sub Logfile()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", Application("LoggedID"))
cmd.Parameters.AddWithValue("@description", "Logged in the system")
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
```

```
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Admin Dashoard Code

```
Imports System.IO
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class AdminDashboard
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not Me.IsPostBack Then
getAdvisior()
getFaculty()
getStudents()
getDepartment()
getSupervisor()
getUsers()
getGraduateschool()
getExaminer()
getProjects()
getRejectedProjects()
End If
End Sub
"get students count
Sub getStudents()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "  select count(*) from Student"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.Read Then
lblStudents.Text = DRead(0)
End If
conn.Close()
```

```
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
"get advisor count
Sub getAdvisior()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "  select count(*) from Advisor"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.Read Then
lblAdvisor.Text = DRead(0)
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
"Get sUPERVISOR
Sub getSupervisor()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "  select count(*) from Supervisor"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.Read Then
lblSupervisor.Text = DRead(0)
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
```

```
"get faculty count
Sub getFaculty()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "  select count(*) from Faculty"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.Read Then
lblFaculty.Text = DRead(0)
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
"get departments
Sub getDepartment()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "  select count(*) from Department"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.Read Then
lblDepartment.Text = DRead(0)
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
"get users
Sub getUsers()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "  select count(*) from Users"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.Read Then
```

```
lblusers.Text = DRead(0)
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
"get Graduate school
Sub getGraduateschool()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "  select count(*) from GraduateSchool"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.Read Then
lblGSchool.Text = DRead(0)
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
"get Examiner
Sub getExaminer()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "  select count(*) from Examiner"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.Read Then
lblExaminer.Text = DRead(0)
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
```

```
End Sub
"get projects
Sub getProjects()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "  select count(*) from project"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.Read Then
lblProjects.Text = DRead(0)
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
"get rejected projects
Sub getRejectedProjects()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "  select count(*) from ProposedTitle where status='Rejected'"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.Read Then
lblRejected.Text = DRead(0)
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Advisor Registration code

```
Imports System.IO
Imports System.Data
Imports System.Configuration
Imports System.Data.SqlClient
Public Class Advisor
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not Me.IsPostBack Then
BindDropDownList()
BindDDLTitle()
BindGView()
Logfile(Application("LoggedID"), "Visited Advisor page")
End If
End Sub
"bind to gridview
Sub BindGView()
Using conn As SqlConnection = New SqlConnection(cnstring)
Using sda As SqlDataAdapter = New SqlDataAdapter("SELECT ID,Name,Gender,DOB,POB,TellNO,Email,Title,Department,Interests,imgdata from Advisor", conn)
Dim dt As DataTable = New DataTable()
sda.Fill(dt)
GVAdvisor.DataSource = dt
GVAdvisor.DataBind()
End Using
End Using
End Sub
"bind Departmen dropdown list items from sql
Sub BindDropDownList()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
```

```
sqlCmd.CommandText = “SELECT ID,name FROM Department”
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLDepartment.DataSource = dt
DDLDepartment.DataValueField = “ID”
DDLDepartment.DataTextField = “name”
DDLDepartment.DataBind()
sqlConn.Close()
'Adding “Please select” option in dropdownlist for validation
DDLDepartment.Items.Insert(0, New ListItem(“Please select”, “0”))
End Using
End Using
Catch
End Try
End Sub
"bind title dropdown list items from sql
Sub BindDDLTitle()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = “SELECT ID,name FROM EduTitle”
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLTitle.DataSource = dt
DDLTitle.DataValueField = “ID”
DDLTitle.DataTextField = “name”
DDLTitle.DataBind()
sqlConn.Close()
'Adding “Please select” option in dropdownlist for validation
DDLTitle.Items.Insert(0, New ListItem(“Please select”, “0”))
End Using
End Using
Catch
End Try
End Sub
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles
btnsubmit.Click
If txtDOB.Text <> "" OrElse txtEmail.Text <> "" OrElse txtinterests.Text <> "" OrElse
txtName.Text <> "" OrElse txtTelNO.Text <> "" OrElse txtPOB.Text <> "" OrElse
DDLDepartment.SelectedIndex <> 0 OrElse DDLGender.SelectedIndex <> 0 OrElse
DDLTitle.SelectedIndex <> 0 Then
```

```
Dim bytes As Byte()
Using br As BinaryReader = New BinaryReader(FileUpload1.PostedFile.InputStream)
bytes = br.ReadBytes(FileUpload1.PostedFile.ContentLength)
End Using
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "INSERT INTO Advisor VALUES( @Name, @Gender, @DOB, @POB, @TellNO, @Email, @Title, @Department, @Interests,@imgName,@imgType, @imgdata,@regdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@Name", txtName.Text)
cmd.Parameters.AddWithValue("@Gender", DDLGender.Text)
cmd.Parameters.AddWithValue("@DOB", CDate(txtDOB.Text))
cmd.Parameters.AddWithValue("@POB", txtPOB.Text)
cmd.Parameters.AddWithValue("@TellNO", txtTelNO.Text)
cmd.Parameters.AddWithValue("@Email", txtEmail.Text)
cmd.Parameters.AddWithValue("@Title", DDLTitle.SelectedValue)
cmd.Parameters.AddWithValue("@Department", DDLDepartment.SelectedValue)
cmd.Parameters.AddWithValue("@Interests", txtinterests.Text)
cmd.Parameters.AddWithValue("@imgName",
Path.GetFileName(FileUpload1.PostedFile.FileName))
cmd.Parameters.AddWithValue("@imgType", FileUpload1.PostedFile.ContentType)
cmd.Parameters.AddWithValue("@imgData", bytes)
cmd.Parameters.AddWithValue("@regdate", Date.Now)
Try
conn.Open()
cmd.ExecuteNonQuery()
clear()
conn.Close()
Logfile(Application("LoggedID"), "Insereted Advisor")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'Advisor has been saved', showConfirmButton: false, timer: 1500});", True)
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End If
End Sub
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles btnupdate.Click
If txtEmail.Text <> "" OrElse txtinterests.Text <> "" OrElse txtName.Text <> "" OrElse txtTelNO.Text <> "" OrElse txtPOB.Text <> "" OrElse DDLDepartment.SelectedIndex <> 0 OrElse DDLGender.SelectedIndex <> 0 OrElse DDLTitle.SelectedIndex <> 0 Then
If FileUpload1.HasFile = True Then
```

```
Dim bytes As Byte()
Using br As BinaryReader = New BinaryReader(FileUpload1.PostedFile.InputStream)
bytes = br.ReadBytes(FileUpload1.PostedFile.ContentLength)
End Using
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Update Advisor set Name= @uName,Gender= @uGender,
DOB=@uDOB, POB=@uPOB, TellNO=@uTellNO, Email=@uEmail,Title= @uTitle,
Department=@uDepartment,Interests=
@uInterests,imgName=@uimgName,imgType=@uimgType, imgdata=@uimgdata" &
" where ID=@AID"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@AID", Session("AdvisorID"))
cmd.Parameters.AddWithValue("@uName", txtName.Text)
cmd.Parameters.AddWithValue("@uGender", DDLGender.Text)
If txtDOB.Text = "" Then
cmd.Parameters.AddWithValue("@uDOB", Session("ADOB"))
Else
cmd.Parameters.AddWithValue("@uDOB", CDate(txtDOB.Text))
End If
cmd.Parameters.AddWithValue("@uPOB", txtPOB.Text)
cmd.Parameters.AddWithValue("@uTellNO", txtTelNO.Text)
cmd.Parameters.AddWithValue("@uEmail", txtEmail.Text)
cmd.Parameters.AddWithValue("@uTitle", DDLTitle.SelectedValue)
cmd.Parameters.AddWithValue("@uDepartment", DDLDepartment.SelectedValue)
cmd.Parameters.AddWithValue("@uInterests", txtinterests.Text)
cmd.Parameters.AddWithValue("@uimgName",
Path.GetFileName(FileUpload1.PostedFile.FileName))
cmd.Parameters.AddWithValue("@uimgType", FileUpload1.PostedFile.ContentType)
cmd.Parameters.AddWithValue("@uimgData", bytes)
Try
conn.Open()
cmd.ExecuteNonQuery()
clear()
conn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Advisor Details Updated Successfully.'
});", True)
Logfile(Application("LoggedID"), "Updated Advisor detials")
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
Else
```

```
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Update Advisor set Name= @uName,Gender= @uGender, DOB=@uDOB, POB=@uPOB, TellNO=@uTellNO, Email=@uEmail,Title= @uTitle, Department=@uDepartment,Interests= @uInterests " &
" where ID=@AID"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@AID", Session("AdvisorID"))
cmd.Parameters.AddWithValue("@uName", txtName.Text)
cmd.Parameters.AddWithValue("@uGender", DDLGender.Text)
If txtDOB.Text = "" Then
cmd.Parameters.AddWithValue("@uDOB", Session("ADOB"))
Else
cmd.Parameters.AddWithValue("@uDOB", CDate(txtDOB.Text))
End If
cmd.Parameters.AddWithValue("@uPOB", txtPOB.Text)
cmd.Parameters.AddWithValue("@uTellNO", txtTelNO.Text)
cmd.Parameters.AddWithValue("@uEmail", txtEmail.Text)
cmd.Parameters.AddWithValue("@uTitle", DDLTitle.SelectedValue)
cmd.Parameters.AddWithValue("@uDepartment", DDLDepartment.SelectedValue)
cmd.Parameters.AddWithValue("@uInterests", txtinterests.Text)
Try
conn.Open()
cmd.ExecuteNonQuery()
clear()
conn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Advisor Details Updated Successfully.'
});", True)
Logfile(Application("LoggedID"), "Updated Advisor detials")
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End If
End If
End Sub
Protected Sub btndelete_Click(sender As Object, e As EventArgs) Handles btndelete.Click
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "delete Advisor where ID =@AdvisorID"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@AdvisorID", Session("AdvisorID"))
Try
conn.Open()
```

```
cmd.ExecuteNonQuery()
clear()
conn.Close()
Logfile(Application("LoggedID"), "Deleted Advisor detials")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Advisor Info Deleted Successfully.'
});", True)
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
Protected Sub GVAdvisor_SelectedIndexChanged(sender As Object, e As EventArgs) Handles GVAdvisor.SelectedIndexChanged
Session("AdvisorID") = GVAdvisor.SelectedRow.Cells(1).Text.ToString
"bind data ffrom gridview to controls
getdatatocontrols()
btnsubmit.Enabled = False
btndelete.Enabled = True
btnupdate.Enabled = True
End Sub
"Bind Data into the controls
Sub getdatatocontrols()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select * from  Advisor where ID=@NO"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@NO", Session("AdvisorID"))
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
Session("AdvisorID") = DRead(0)
txtName.Text = DRead(1)
DDLGender.Text = DRead(2)
Session("ADOB") = CDate(DRead(3))
txtDOB.Text = CDate(DRead(3))
txtPOB.Text = DRead(4)
txtTelNO.Text = DRead(5)
txtEmail.Text = DRead(6)
DDLTitle.Text = DRead(7)
DDLDepartment.Text = DRead(8)
txtinterests.Text = DRead(9)
```

```
Session("imgName") = DRead(10)
Session("imgType") = DRead(11)
Session("imgData") = DRead(12)
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
Protected Sub GVAdvisor_RowDataBound(sender As Object, e As GridViewRowEventArgs) Handles GVAdvisor.RowDataBound
If e.Row.RowType = DataControlRowType.DataRow Then
Dim dr As DataRowView = CType(e.Row.DataItem, DataRowView)
Dim imageUrl As String = "" & Convert.ToBase64String(CType(dr("imgdata"), Byte()))
CType(e.Row.FindControl("Image1"), Image).ImageUrl = imageUrl
End If
End Sub
"clear function
Sub clear()
txtName.Text = ""
DDLGender.SelectedIndex = 0
BindDropDownList()
BindDDLTitle()
BindGView()
txtDOB.Text = ""
txtPOB.Text = ""
txtTelNO.Text = ""
txtEmail.Text = ""
DDLTitle.Text = ""
txtinterests.Text = ""
DDLDepartment.SelectedIndex = 0
txtinterests.Text = ""
Session("imgName") = ""
Session("imgType") = ""
Session("imgData") = ""
txtName.BorderColor = Drawing.Color.Black
Session("StudentNO") = ""
txtName.Focus()
btnupdate.Enabled = False
btndelete.Enabled = False
btnsubmit.Enabled = True
End Sub
```

```
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Assign a student to an adviser

```
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class AssignAdvisorToStudent
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public userID As String = ""
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not IsPostBack Then
BindGrid()
BindDropDownListDepartment()
btnsubmit.Enabled = True
End If
End Sub
"bind dropdown list Department from sql
Sub BindDropDownListDepartment()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = "SELECT ID,name FROM Department"
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
ddlDepartment.DataSource = dt
ddlDepartment.DataValueField = "ID"
ddlDepartment.DataTextField = "name"
ddlDepartment.DataBind()
sqlConn.Close()
'Adding "Please select" option in dropdownlist for validation
ddlDepartment.Items.Insert(0, New ListItem("Please select", "0"))
End Using
```

```
End Using
Catch
End Try
End Sub
"bind dropdown list Student from sql
Sub BindDropDownListStudent()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = "SELECT StudentNO,name FROM Student Where
Department=@DP"
sqlCmd.Parameters.AddWithValue("@DP", ddlDepartment.SelectedValue)
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLStudentNO.DataSource = dt
DDLStudentNO.DataValueField = "StudentNO"
DDLStudentNO.DataTextField = "name"
DDLStudentNO.DataBind()
sqlConn.Close()
'Adding "Please select" option in dropdownlist for validation
DDLStudentNO.Items.Insert(0, New ListItem("Please select", "0"))
End Using
End Using
Catch
End Try
End Sub
"bind dropdown list Advisor from sql
Sub BindDropDownListAdvisor()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = "SELECT ID,name FROM Advisor where Department =@D"
sqlCmd.Parameters.AddWithValue("@D", ddlDepartment.SelectedValue)
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
ddlAdvisor.DataSource = dt
ddlAdvisor.DataValueField = "ID"
ddlAdvisor.DataTextField = "name"
ddlAdvisor.DataBind()
sqlConn.Close()
```

```
'Adding "Please select" option in dropdownlist for validation
ddlAdvisor.Items.Insert(0, New ListItem("Please select", "0"))
End Using
End Using
Catch
End Try
End Sub
"bind gridview
Private Sub BindGrid()
Using con As New SqlConnection(cnstring)
Using cmd As New SqlCommand("SELECT * FROM AssignedAdvisor")
Using sda As New SqlDataAdapter()
cmd.Connection = con
sda.SelectCommand = cmd
Using dt As New DataTable()
sda.Fill(dt)
GVATS.DataSource = dt
GVATS.DataBind()
End Using
End Using
End Using
End Using
End Sub
Protected Sub ddlDepartment_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles ddlDepartment.SelectedIndexChanged
If ddlDepartment.SelectedIndex <> 0 Then
ddlAdvisor.Items.Clear()
DDLStudentNO.Items.Clear()
BindDropDownListAdvisor()
BindDropDownListStudent()
End If
End Sub
Protected Sub GVATS_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles GVATS.SelectedIndexChanged
Session("AssignedID") = GVATS.SelectedRow.Cells(1).Text.ToString
Session("AdvisorID") = GVATS.SelectedRow.Cells(2).Text.ToString
Session("StudentNO") = GVATS.SelectedRow.Cells(3).Text.ToString
ddlDepartment.Items.Clear()
BindDropDownListDepartment()
DDLStudentNO.Text = GVATS.SelectedRow.Cells(3).Text.ToString
ddlAdvisor.Text = GVATS.SelectedRow.Cells(2).Text.ToString
btnsubmit.Enabled = False
btndelete.Enabled = True
btnupdate.Enabled = True
End Sub
Sub clear()
ddlDepartment.Items.Clear()
```

```
ddlAdvisor.Items.Clear()
DDLStudentNO.Items.Clear()
BindDropDownListDepartment()
ddlDepartment.Focus()
lblError.Text = ""
btnsubmit.Enabled = True
btndelete.Enabled = False
btnupdate.Enabled = False
End Sub
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles
btnsubmit.Click
Check()
If checkAssign = 0 Then
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"INSERT INTO AssignedAdvisor " &
" VALUES (@AdvisorID,@StudentNO, @regdate) " &
"COMMIT"
cmd.Parameters.AddWithValue("@AdvisorID", ddlAdvisor.SelectedValue)
cmd.Parameters.AddWithValue("@StudentNO", DDLStudentNO.SelectedValue)
cmd.Parameters.AddWithValue("@regdate", DateTime.Now)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("Saved...", "insertion", MessageBoxButtons.OK,
MessageBoxIcon.Information)
clear()
cn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-
end', icon: 'success', title: 'Assigned advisor to student ', showConfirmButton: false,
timer: 1500});", True)
Logfile(Application("LoggedID"), "Assigned advisor to student")
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
BindGrid()
Else
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'This Student is already assigned to advisor!'
});", True)
```

```
lblError.Text = "This Student is already assigned to advisor"
End If
End Sub
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles btnupdate.Click
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Update  AssignedAdvisor " &
" set  AdvisorID=@AdvisID,StudentNO=@StdNO " &
" where ID=@ASID " &
"COMMIT"
cmd.Parameters.AddWithValue("@ASID", CInt(Session("AssignedID")))
If ddlAdvisor.SelectedIndex = 0 Then
cmd.Parameters.AddWithValue("@AdvisID", CInt(Session("AdvisorID")))
Else
cmd.Parameters.AddWithValue("@AdvisID", ddlAdvisor.SelectedValue)
End If
If DDLStudentNO.SelectedIndex = 0 Then
cmd.Parameters.AddWithValue("@StdNO", Session("StudentNO"))
Else
cmd.Parameters.AddWithValue("@StdNO", DDLStudentNO.SelectedValue)
End If
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("Saved...", "insertion", MessageBoxButtons.OK, MessageBoxIcon.Information)
clear()
cn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'changed assigned advisor to student.'
});", True)
Logfile(Application("LoggedID"), "changed assigned advisor to student")
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
BindGrid()
End Sub
Protected Sub btndelete_Click(sender As Object, e As EventArgs) Handles btndelete.Click
cmd = New SqlCommand()
```

```
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Delete AssignedAdvisor " &
" where ID=@AID " &
"COMMIT"
cmd.Parameters.AddWithValue("@AID", CInt(Session("AssignedID")))
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("Saved...", "insertion", MessageBoxButtons.OK,
MessageBoxIcon.Information)
clear()
cn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Deleted assigned advisor to student.'
});", True)
Logfile(Application("LoggedID"), "Deleted assigned advisor to student")
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
BindGrid()
End Sub
"check if the Student is already assigned to an advisor
Public checkAssign As Integer = 0
Sub Check()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select * from AssignedAdvisor " &
" where StudentNO= @SID " &
"COMMIT"
cmd.Parameters.AddWithValue("@SID", DDLStudentNO.SelectedValue)
Try
connection()
cn.Open()
checkAssign = CInt(cmd.ExecuteScalar())
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
```

```
connection()
cmd.Parameters.Clear()
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Assign a student to an examiner

```
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class AssignExaminerToStudent
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public userID As String = ""
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not IsPostBack Then
BindGrid()
BindDropDownListDepartment()
btnsubmit.Enabled = True
End If
End Sub
"bind dropdown list Department from sql
Sub BindDropDownListDepartment()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = "SELECT ID,name FROM Department"
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
ddlDepartment.DataSource = dt
ddlDepartment.DataValueField = "ID"
ddlDepartment.DataTextField = "name"
ddlDepartment.DataBind()
sqlConn.Close()
'Adding "Please select" option in dropdownlist for validation
ddlDepartment.Items.Insert(0, New ListItem("Select Department", "0"))
End Using
```

```
End Using
Catch
End Try
End Sub
"bind dropdown list Student from sql
Sub BindDropDownListStudent()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = "SELECT StudentNO,name FROM Student Where
Department=@DP"
sqlCmd.Parameters.AddWithValue("@DP", ddlDepartment.SelectedValue)
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLStudentNO.DataSource = dt
DDLStudentNO.DataValueField = "StudentNO"
DDLStudentNO.DataTextField = "name"
DDLStudentNO.DataBind()
sqlConn.Close()
'Adding "Please select" option in dropdownlist for validation
DDLStudentNO.Items.Insert(0, New ListItem("Select Student", "0"))
End Using
End Using
Catch
End Try
End Sub
"bind dropdown list examiner from sql
Sub BindDropDownListExainer()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = "SELECT ID,name FROM Examiner where Department =@D"
sqlCmd.Parameters.AddWithValue("@D", ddlDepartment.SelectedValue)
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
ddlExaminer.DataSource = dt
ddlExaminer.DataValueField = "ID"
ddlExaminer.DataTextField = "name"
ddlExaminer.DataBind()
sqlConn.Close()
```

```
'Adding "Please select" option in dropdownlist for validation
ddlExaminer.Items.Insert(0, New ListItem("Select Examiner", "0"))
End Using
End Using
Catch
End Try
End Sub
"bind gridview
Private Sub BindGrid()
Using con As New SqlConnection(cnstring)
Using cmd As New SqlCommand("SELECT * FROM AssignedExaminer")
Using sda As New SqlDataAdapter()
cmd.Connection = con
sda.SelectCommand = cmd
Using dt As New DataTable()
sda.Fill(dt)
GVETS.DataSource = dt
GVETS.DataBind()
End Using
End Using
End Using
End Using
End Sub
Protected Sub ddlDepartment_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles ddlDepartment.SelectedIndexChanged
If ddlDepartment.SelectedIndex <> 0 Then
ddlExaminer.Items.Clear()
DDLStudentNO.Items.Clear()
BindDropDownListExainer()
BindDropDownListStudent()
End If
End Sub
Protected Sub GVETS_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles GVETS.SelectedIndexChanged
Session("AssignedID") = GVETS.SelectedRow.Cells(1).Text.ToString
Session("ExaminerID") = GVETS.SelectedRow.Cells(2).Text.ToString
Session("StudentNO") = GVETS.SelectedRow.Cells(3).Text.ToString
ddlDepartment.Items.Clear()
BindDropDownListDepartment()
DDLStudentNO.Text = GVETS.SelectedRow.Cells(3).Text.ToString
ddlExaminer.Text = GVETS.SelectedRow.Cells(2).Text.ToString
btnsubmit.Enabled = False
btndelete.Enabled = True
btnupdate.Enabled = True
End Sub
Sub clear()
```

```
ddlDepartment.Items.Clear()
ddlExaminer.Items.Clear()
DDLStudentNO.Items.Clear()
BindDropDownListDepartment()
ddlDepartment.Focus()
lblError.Text = ""
btnsubmit.Enabled = True
btndelete.Enabled = False
btnupdate.Enabled = False
End Sub
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles btnsubmit.Click
CheckMax()
Check()
If MaxAssign <> 3 Then
If checkAssign = 0 Then
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"INSERT INTO AssignedExaminer " &
" VALUES (@ExaminerID,@StudentNO, @regdate) " &
"COMMIT"
cmd.Parameters.AddWithValue("@ExaminerID", ddlExaminer.SelectedValue)
cmd.Parameters.AddWithValue("@StudentNO", DDLStudentNO.SelectedValue)
cmd.Parameters.AddWithValue("@regdate", DateTime.Now.ToShortDateString)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("Saved...", "insertion", MessageBoxButtons.OK, MessageBoxIcon.Information)
clear()
cn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'You Assigned Examiner to a student', showConfirmButton: false, timer: 1500});", True)
Logfile(Application("LoggedID"), "Assigned Examiner to a student")
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
BindGrid()
Else
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
```

```
title: 'Oops...',
text: 'This Student is already assigned to this Examiner!'
});", True)
lblError.Text = "This Student is already assigned to this Examiner"
End If
Else
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'This Student is already assigned to his/her 3 Examiners!'
});", True)
lblError.Text = "This Student is already assigned to his/her three Examiners"
End If
End Sub
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles btnupdate.Click
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Update  AssignedExaminer " &
" set ExaminerID=@EXID,StudentNO=@StdNO " &
" where ID=@ASID " &
"COMMIT"
cmd.Parameters.AddWithValue("@ASID", CInt(Session("AssignedID")))
If ddlExaminer.SelectedIndex = 0 Then
cmd.Parameters.AddWithValue("@EXID", CInt(Session("ExaminerID")))
Else
cmd.Parameters.AddWithValue("@EXID", ddlExaminer.SelectedValue)
End If
If DDLStudentNO.SelectedIndex = 0 Then
cmd.Parameters.AddWithValue("@StdNO", Session("StudentNO"))
Else
cmd.Parameters.AddWithValue("@StdNO", DDLStudentNO.SelectedValue)
End If
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("Saved...", "insertion", MessageBoxButtons.OK, MessageBoxIcon.Information)
clear()
cn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Updated Assigned Examiner to a student.'
```

```
});", True)
Logfile(Application("LoggedID"), "Updated Assigned Examiner to a student")
lblError.Text = "Updated."
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
BindGrid()
End Sub
Protected Sub btndelete_Click(sender As Object, e As EventArgs) Handles btndelete.Click
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Delete AssignedExaminer " &
" where ID=@AID " &
"COMMIT"
cmd.Parameters.AddWithValue("@AID", CInt(Session("AssignedID")))
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("Saved...", "insertion", MessageBoxButtons.OK,
MessageBoxIcon.Information)
clear()
lblError.Text = "Deleted."
cn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Deleted Assigned Examiner to a student.'
});", True)
Logfile(Application("LoggedID"), "Deleted Assigned Examiner to a student")
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
BindGrid()
End Sub
"check if the Student is already assigned to this examiner
Public checkAssign As Integer = 0
Sub Check()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
```

```
" BEGIN TRANSACTION " &
"Select * from AssignedExaminer " &
" where StudentNO= @SID and ExaminerID=@EID " &
"COMMIT"
cmd.Parameters.AddWithValue("@SID", DDLStudentNO.SelectedValue)
cmd.Parameters.AddWithValue("@EID", ddlExaminer.SelectedValue)
Try
connection()
cn.Open()
checkAssign = CInt(cmd.ExecuteScalar())
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
"check if the Student is already assigned to his/her 3 examiners
Public MaxAssign As Integer = 0
Sub CheckMax()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select count(*) from AssignedExaminer " &
" where StudentNO= @SID " &
"COMMIT"
cmd.Parameters.AddWithValue("@SID", DDLStudentNO.SelectedValue)
Try
connection()
cn.Open()
MaxAssign = CInt(cmd.ExecuteScalar())
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
```

```
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Department Registration

```
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class Department
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public userID As String = ""
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not IsPostBack Then
BindGrid()
BindDropDownList()
btnsubmit.Enabled = True
Session("DeptID") = ""
End If
End Sub
"bind gridview
Private Sub BindGrid()
Using con As New SqlConnection(cnstring)
Using cmd As New SqlCommand("SELECT * FROM Department")
Using sda As New SqlDataAdapter()
cmd.Connection = con
sda.SelectCommand = cmd
Using dt As New DataTable()
sda.Fill(dt)
GVDepartment.DataSource = dt
GVDepartment.DataBind()
End Using
End Using
End Using
End Using
End Sub
```

```
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles btnsubmit.Click
If txtName.Text = "" OrElse DDLFaculty.SelectedIndex = 0 Then
Return
Exit Sub
End If
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"INSERT INTO Department " &
" VALUES (@name,@Faculty) " &
"COMMIT"
cmd.Parameters.AddWithValue("@name", txtName.Text)
cmd.Parameters.AddWithValue("@Faculty", DDLFaculty.SelectedValue)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("Saved...", "insertion", MessageBoxButtons.OK, MessageBoxIcon.Information)
clear()
cn.Close()
BindGrid()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'Department has been saved', showConfirmButton: false, timer: 1500});", True)
Logfile(Application("LoggedID"), "Insrted department")
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles btnupdate.Click
If txtName.Text = "" OrElse DDLFaculty.SelectedIndex = 0 Then
Return
Exit Sub
End If
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
" Update Department " &
" set name=@Name,Faculty=@F" &
" where ID=@DID " &
```

```
" COMMIT"
cmd.Parameters.AddWithValue("@Name", txtName.Text)
cmd.Parameters.AddWithValue("@F", DDLFaculty.SelectedValue)
cmd.Parameters.AddWithValue("@DID", Session("DeptID"))
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("updated...", "insertion", MessageBoxButtons.OK,
MessageBoxIcon.Information)
clear()
cn.Close()
BindGrid()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Department info Updated Successfully.'
});", True)
Logfile(Application("LoggedID"), "Uppdate department info")
Catch ex As Exception
'MsgBox.Show("updating Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub btndelete_Click(sender As Object, e As EventArgs) Handles btndelete.Click
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
" Delete Department " &
" where ID= @DprtmntID " &
" COMMIT"
cmd.Parameters.AddWithValue("@DprtmntID", Session("DeptID"))
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("deleted...", "insertion", MessageBoxButtons.OK,
MessageBoxIcon.Information)
clear()
cn.Close()
BindGrid()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Department Deleted Successfully.'
```

```
});", True)
Logfile(Application("LoggedID"), "Deleted department info")
Catch ex As Exception
'MsgBox.Show("deleting Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
"clear function
Sub clear()
txtName.Text = ""
BindDropDownList()
DDLFaculty.SelectedIndex = 0
txtName.BorderColor = Drawing.Color.Black
DDLFaculty.BorderColor = Drawing.Color.Black
txtName.Focus()
Session("DeptID") = ""
btnupdate.Enabled = False
btndelete.Enabled = False
btnsubmit.Enabled = True
End Sub
Protected Sub GVDepartment_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles GVDepartment.SelectedIndexChanged
Session("DeptID") = GVDepartment.SelectedRow.Cells(1).Text.ToString
txtName.Text = GVDepartment.SelectedRow.Cells(2).Text
DDLFaculty.Text = GVDepartment.SelectedRow.Cells(3).Text
btnsubmit.Enabled = False
btndelete.Enabled = True
btnupdate.Enabled = True
End Sub
Protected Sub OnPageIndexChanging(sender As Object, e As GridViewPageEventArgs)
GVDepartment.PageIndex = e.NewPageIndex
Me.BindGrid()
End Sub
"bind dropdown list items from sql
Sub BindDropDownList()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = “SELECT ID,name FROM Faculty”
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLFaculty.DataSource = dt
DDLFaculty.DataValueField = “ID”
```

```
DDLFaculty.DataTextField = “name”
DDLFaculty.DataBind()
sqlConn.Close()
'Adding “Please select” option in dropdownlist for validation
DDLFaculty.Items.Insert(0, New ListItem(“Please select”, “0”))
End Using
End Using
Catch
End Try
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Education title registration

```
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class EduTitle
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public userID As String = ""
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not IsPostBack Then
BindGrid()
btnsubmit.Enabled = True
Session("EduID") = ""
End If
End Sub
"bind gridview edu titles
Private Sub BindGrid()
Using con As New SqlConnection(cnstring)
Using cmd As New SqlCommand("SELECT * FROM EduTitle")
Using sda As New SqlDataAdapter()
cmd.Connection = con
sda.SelectCommand = cmd
Using dt As New DataTable()
sda.Fill(dt)
GVEduTitle.DataSource = dt
GVEduTitle.DataBind()
End Using
End Using
End Using
End Using
End Sub
Protected Sub OnPageIndexChanging(sender As Object, e As GridViewPageEventArgs)
GVEduTitle.PageIndex = e.NewPageIndex
Me.BindGrid()
```

```
End Sub
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles btnupdate.Click
If txtName.Text = "" Then
Return
Exit Sub
End If
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
" Update EduTitle " &
" set name=@Nme " &
" where ID=@EDID " &
" COMMIT"
cmd.Parameters.AddWithValue("@Nme", txtName.Text)
cmd.Parameters.AddWithValue("@EDID", Session("EduID"))
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("updated...", "insertion", MessageBoxButtons.OK, MessageBoxIcon.Information)
clear()
cn.Close()
BindGrid()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Education Title Updated Successfully.'
});", True)
Logfile(Application("LoggedID"), "Update Education title")
Catch ex As Exception
'MsgBox.Show("updating Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub btndelete_Click(sender As Object, e As EventArgs) Handles btndelete.Click
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
" Delete EduTitle " &
" where ID= @eduID " &
" COMMIT"
cmd.Parameters.AddWithValue("@eduID", Session("EduID"))
```

```
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("deleted...", "insertion", MessageBoxButtons.OK,
MessageBoxIcon.Information)
clear()
cn.Close()
BindGrid()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Education Title Deleted Successfully.'
});", True)
Logfile(Application("LoggedID"), "Deleted Education title info")
Catch ex As Exception
'MsgBox.Show("deleting Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub GVEduTitle_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles GVEduTitle.SelectedIndexChanged
Session("EduID") = GVEduTitle.SelectedRow.Cells(1).Text.ToString
txtName.Text = GVEduTitle.SelectedRow.Cells(2).Text
btnsubmit.Enabled = False
btndelete.Enabled = True
btnupdate.Enabled = True
End Sub
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles
btnsubmit.Click
If txtName.Text = "" Then
txtName.BorderColor = Drawing.Color.Red
Return
Exit Sub
End If
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"INSERT INTO EduTitle " &
" VALUES (@name) " &
"COMMIT"
cmd.Parameters.AddWithValue("@name", txtName.Text)
Try
connection()
cn.Open()
```

```
cmd.ExecuteNonQuery()
'MessageBox.Show("Saved...", "insertion", MessageBoxButtons.OK, MessageBoxIcon.Information)
clear()
cn.Close()
BindGrid()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'Your Education title has been saved', showConfirmButton: false, timer: 1500});", True)
Logfile(Application("LoggedID"), "Insered Education title info")
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
"clear function
Sub clear()
txtName.Text = ""
txtName.BorderColor = Drawing.Color.Black
txtName.Focus()
Session("EduID") = ""
btnupdate.Enabled = False
btndelete.Enabled = False
btnsubmit.Enabled = True
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Examiner Registration

```
Imports System.IO
Imports System.Data
Imports System.Configuration
Imports System.Data.SqlClient
Public Class Examiner
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not Me.IsPostBack Then
BindDropDownList()
BindDDLTitle()
BindGView()
End If
End Sub
"bind to gridview
Sub BindGView()
Using conn As SqlConnection = New SqlConnection(cnstring)
Using sda As SqlDataAdapter = New SqlDataAdapter("SELECT ID,Name,Gender,DOB,POB,TellNO,Email,Title,Department,Interests,imgdata from Examiner", conn)
Dim dt As DataTable = New DataTable()
Try
sda.Fill(dt)
GVExaminor.DataSource = dt
GVExaminor.DataBind()
Catch ex As Exception
End Try
End Using
End Using
End Sub
"bind Departmen dropdown list items from sql
Sub BindDropDownList()
```

```
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = “SELECT ID,name FROM Department”
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLDepartment.DataSource = dt
DDLDepartment.DataValueField = “ID”
DDLDepartment.DataTextField = “name”
DDLDepartment.DataBind()
sqlConn.Close()
'Adding “Please select” option in dropdownlist for validation
DDLDepartment.Items.Insert(0, New ListItem(“Please select”, “0”))
End Using
End Using
Catch
End Try
End Sub
"bind title dropdown list items from sql
Sub BindDDLTitle()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = “SELECT ID,name FROM EduTitle”
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLTitle.DataSource = dt
DDLTitle.DataValueField = “ID”
DDLTitle.DataTextField = “name”
DDLTitle.DataBind()
sqlConn.Close()
'Adding “Please select” option in dropdownlist for validation
DDLTitle.Items.Insert(0, New ListItem(“Please select”, “0”))
End Using
End Using
Catch
End Try
End Sub
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles
btnsubmit.Click
```

```
If txtDOB.Text <> "" OrElse txtEmail.Text <> "" OrElse txtinterests.Text <> "" OrElse txtName.Text <> "" OrElse txtTelNO.Text <> "" OrElse txtPOB.Text <> "" OrElse DDLDepartment.SelectedIndex <> 0 OrElse DDLGender.SelectedIndex <> 0 OrElse DDLTitle.SelectedIndex <> 0 Then
Dim bytes As Byte()
Using br As BinaryReader = New BinaryReader(FileUpload1.PostedFile.InputStream)
bytes = br.ReadBytes(FileUpload1.PostedFile.ContentLength)
End Using
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "INSERT INTO Examiner VALUES( @Name, @Gender, @DOB, @POB, @TellNO, @Email, @Title, @Department, @Interests,@imgName,@imgType, @imgdata,@regdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@Name", txtName.Text)
cmd.Parameters.AddWithValue("@Gender", DDLGender.Text)
cmd.Parameters.AddWithValue("@DOB", CDate(txtDOB.Text))
cmd.Parameters.AddWithValue("@POB", txtPOB.Text)
cmd.Parameters.AddWithValue("@TellNO", txtTelNO.Text)
cmd.Parameters.AddWithValue("@Email", txtEmail.Text)
cmd.Parameters.AddWithValue("@Title", DDLTitle.SelectedValue)
cmd.Parameters.AddWithValue("@Department", DDLDepartment.SelectedValue)
cmd.Parameters.AddWithValue("@Interests", txtinterests.Text)
cmd.Parameters.AddWithValue("@imgName",
Path.GetFileName(FileUpload1.PostedFile.FileName))
cmd.Parameters.AddWithValue("@imgType", FileUpload1.PostedFile.ContentType)
cmd.Parameters.AddWithValue("@imgData", bytes)
cmd.Parameters.AddWithValue("@regdate", Date.Now)
Try
conn.Open()
cmd.ExecuteNonQuery()
clear()
conn.Close()
Logfile(Application("LoggedID"), "Inserted Examiner Info")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'Examiner has been Registred', showConfirmButton: false, timer: 1500});", True)
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End If
End Sub
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles btnupdate.Click
```

```
If txtEmail.Text <> "" OrElse txtinterests.Text <> "" OrElse txtName.Text <> "" OrElse txtTelNO.Text <> "" OrElse txtPOB.Text <> "" OrElse DDLDepartment.SelectedIndex <> 0 OrElse DDLGender.SelectedIndex <> 0 OrElse DDLTitle.SelectedIndex <> 0 Then
If FileUpload1.HasFile = True Then
Dim bytes As Byte()
Using br As BinaryReader = New BinaryReader(FileUpload1.PostedFile.InputStream)
bytes = br.ReadBytes(FileUpload1.PostedFile.ContentLength)
End Using
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Update Examiner set Name= @uName,Gender= @uGender, DOB=@uDOB, POB=@uPOB, TellNO=@uTellNO, Email=@uEmail,Title= @uTitle, Department=@uDepartment,Interests=
@uInterests,imgName=@uimgName,imgType=@uimgType, imgdata=@uimgdata" &
" where ID=@SUperID"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@SUperID", Session("ExaminerID"))
cmd.Parameters.AddWithValue("@uName", txtName.Text)
cmd.Parameters.AddWithValue("@uGender", DDLGender.Text)
If txtDOB.Text = "" Then
cmd.Parameters.AddWithValue("@uDOB", Session("EXDOB"))
Else
cmd.Parameters.AddWithValue("@uDOB", CDate(txtDOB.Text))
End If
cmd.Parameters.AddWithValue("@uPOB", txtPOB.Text)
cmd.Parameters.AddWithValue("@uTellNO", txtTelNO.Text)
cmd.Parameters.AddWithValue("@uEmail", txtEmail.Text)
cmd.Parameters.AddWithValue("@uTitle", DDLTitle.SelectedValue)
cmd.Parameters.AddWithValue("@uDepartment", DDLDepartment.SelectedValue)
cmd.Parameters.AddWithValue("@uInterests", txtinterests.Text)
cmd.Parameters.AddWithValue("@uimgName",
Path.GetFileName(FileUpload1.PostedFile.FileName))
cmd.Parameters.AddWithValue("@uimgType", FileUpload1.PostedFile.ContentType)
cmd.Parameters.AddWithValue("@uimgData", bytes)
Try
conn.Open()
cmd.ExecuteNonQuery()
clear()
conn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Examiner info Updated Successfully with new Image.'
});", True)
Logfile(Application("LoggedID"), "updated Examiner Info")
Catch ex As Exception
conn.Close()
End Try
```

```
cmd.Parameters.Clear()
End Using
End Using
Else
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Update Examiner set Name= @uName,Gender= @uGender, DOB=@uDOB, POB=@uPOB, TellNO=@uTellNO, Email=@uEmail,Title= @uTitle, Department=@uDepartment,Interests= @uInterests " &
" where ID=@AID"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@AID", Session("ExaminerID"))
cmd.Parameters.AddWithValue("@uName", txtName.Text)
cmd.Parameters.AddWithValue("@uGender", DDLGender.Text)
If txtDOB.Text = "" Then
cmd.Parameters.AddWithValue("@uDOB", Session("EXDOB"))
Else
cmd.Parameters.AddWithValue("@uDOB", CDate(txtDOB.Text))
End If
cmd.Parameters.AddWithValue("@uPOB", txtPOB.Text)
cmd.Parameters.AddWithValue("@uTellNO", txtTelNO.Text)
cmd.Parameters.AddWithValue("@uEmail", txtEmail.Text)
cmd.Parameters.AddWithValue("@uTitle", DDLTitle.SelectedValue)
cmd.Parameters.AddWithValue("@uDepartment", DDLDepartment.SelectedValue)
cmd.Parameters.AddWithValue("@uInterests", txtinterests.Text)
Try
conn.Open()
cmd.ExecuteNonQuery()
clear()
conn.Close()
Logfile(Application("LoggedID"), "updated Examiner Info")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Examiner info Updated Successfully...'
});", True)
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End If
End If
End Sub
Protected Sub btndelete_Click(sender As Object, e As EventArgs) Handles btndelete.Click
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "delete Examiner where ID =@ID"
```

```
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@ID", Session("ExaminerID"))
Try
conn.Open()
cmd.ExecuteNonQuery()
clear()
conn.Close()
Logfile(Application("LoggedID"), "Deleted Examiner Info")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Examiner Deleted Successfully.'
});", True)
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
Protected Sub GVExaminor_SelectedIndexChanged(sender As Object, e As EventArgs) Handles GVExaminor.SelectedIndexChanged
Session("ExaminerID") = GVExaminor.SelectedRow.Cells(1).Text.ToString
"bind data ffrom gridview to controls
getdatatocontrols()
btnsubmit.Enabled = False
btndelete.Enabled = True
btnupdate.Enabled = True
End Sub
"Bind Data into the controls
Sub getdatatocontrols()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select * from  Examiner where ID=@ENO"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@ENO", Session("ExaminerID"))
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
Session("ExaminerID") = DRead(0)
txtName.Text = DRead(1)
DDLGender.Text = DRead(2)
Session("EXDOB") = CDate(DRead(3))
txtDOB.Text = CDate(DRead(3))
txtPOB.Text = DRead(4)
txtTelNO.Text = DRead(5)
```

```
txtEmail.Text = DRead(6)
DDLTitle.Text = DRead(7)
DDLDepartment.Text = DRead(8)
txtinterests.Text = DRead(9)
Session("imgName") = DRead(10)
Session("imgType") = DRead(11)
Session("imgData") = DRead(12)
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
Protected Sub GVExaminor_RowDataBound(sender As Object, e As GridViewRowEventArgs) Handles GVExaminor.RowDataBound
If e.Row.RowType = DataControlRowType.DataRow Then
Dim dr As DataRowView = CType(e.Row.DataItem, DataRowView)
Dim imageUrl As String = "" & Convert.ToBase64String(CType(dr("imgdata"), Byte()))
CType(e.Row.FindControl("Image1"), Image).ImageUrl = imageUrl
End If
End Sub
"clear function
Sub clear()
txtName.Text = ""
DDLGender.SelectedIndex = 0
BindDropDownList()
BindDDLTitle()
BindGView()
txtDOB.Text = ""
txtPOB.Text = ""
txtTelNO.Text = ""
txtEmail.Text = ""
txtinterests.Text = ""
DDLTitle.Text = ""
DDLDepartment.SelectedIndex = 0
txtinterests.Text = ""
Session("imgName") = ""
Session("imgType") = ""
Session("imgData") = ""
txtName.BorderColor = Drawing.Color.Black
Session("ExaminerID") = ""
txtName.Focus()
```

```
btnupdate.Enabled = False
btndelete.Enabled = False
btnsubmit.Enabled = True
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Faculty Registration code

```
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class Faculty
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public userID As String = ""
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not IsPostBack Then
BindGrid()
BindDropDownList()
btnsubmit.Enabled = True
Session("FacultyID") = ""
End If
End Sub
Private Sub BindGrid()
Using con As New SqlConnection(cnstring)
Using cmd As New SqlCommand("SELECT * FROM Faculty")
Using sda As New SqlDataAdapter()
cmd.Connection = con
sda.SelectCommand = cmd
Using dt As New DataTable()
sda.Fill(dt)
GVFaculty.DataSource = dt
GVFaculty.DataBind()
End Using
End Using
End Using
End Using
End Sub
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles btnsubmit.Click
If txtName.Text = "" OrElse DDLGschool.SelectedIndex = 0 Then
```

```
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Plz fill the blanks and try again!'
});", True)
Return
Exit Sub
End If
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"INSERT INTO Faculty " &
" VALUES (@name,@GS) " &
"COMMIT"
cmd.Parameters.AddWithValue("@name", txtName.Text)
cmd.Parameters.AddWithValue("@Gs", DDLGschool.SelectedValue)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
clear()
cn.Close()
BindGrid()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'Faculty has been Registred', showConfirmButton: false, timer: 1500});", True)
Logfile(Application("LoggedID"), "Inserted Faculty info")
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles btnupdate.Click
If txtName.Text = "" OrElse DDLGschool.SelectedIndex = 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Plz fill the blanks and try again!'
});", True)
Return
Exit Sub
End If
cmd = New SqlCommand()
cmd.Connection = cn
```

```
cmd.CommandText =
" BEGIN TRANSACTION " &
" Update Faculty " &
" set name=@Name,graduateSchool=@G " &
" where ID=@FID " &
" COMMIT"
cmd.Parameters.AddWithValue("@Name", txtName.Text)
cmd.Parameters.AddWithValue("@G", DDLGschool.SelectedValue)
cmd.Parameters.AddWithValue("@FID", Session("FacultyID"))
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
clear()
cn.Close()
BindGrid()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Faculty Details Updated Successfully.'
});", True)
Logfile(Application("LoggedID"), "Updated Faculty info")
Catch ex As Exception
'MsgBox.Show("updating Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub btndelete_Click(sender As Object, e As EventArgs) Handles btndelete.Click
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
" Delete Faculty " &
" where ID= @FacultyID " &
" COMMIT"
cmd.Parameters.AddWithValue("@FacultyID", Session("FacultyID"))
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
clear()
cn.Close()
BindGrid()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
```

```
text: 'Faculty Deleted Successfully.'
});", True)
Logfile(Application("LoggedID"), "Deleted Faculty info")
Catch ex As Exception
'MsgBox.Show("deleting Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
"clear function
Sub clear()
txtName.Text = ""
BindDropDownList()
DDLGschool.SelectedIndex = 0
txtName.BorderColor = Drawing.Color.Black
DDLGschool.BorderColor = Drawing.Color.Black
txtName.Focus()
Session("FacultyID") = ""
btnupdate.Enabled = False
btndelete.Enabled = False
btnsubmit.Enabled = True
End Sub
Protected Sub GVFaculty_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles GVFaculty.SelectedIndexChanged
Session("FacultyID") = GVFaculty.SelectedRow.Cells(1).Text.ToString
txtName.Text = GVFaculty.SelectedRow.Cells(2).Text
DDLGschool.Text = GVFaculty.SelectedRow.Cells(3).Text
btnsubmit.Enabled = False
btndelete.Enabled = True
btnupdate.Enabled = True
End Sub
Protected Sub OnPageIndexChanging(sender As Object, e As GridViewPageEventArgs)
GVFaculty.PageIndex = e.NewPageIndex
Me.BindGrid()
End Sub
"bind dropdown list items from sql
Sub BindDropDownList()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = “SELECT ID,name FROM GraduateSchool”
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLGschool.DataSource = dt
```

```
DDLGschool.DataValueField = “ID”
DDLGschool.DataTextField = “name”
DDLGschool.DataBind()
sqlConn.Close()
'Adding “Please select” option in dropdownlist for validation
DDLGschool.Items.Insert(0, New ListItem(“Please select”, “0”))
End Using
End Using
Catch
End Try
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Graduate School Code

```
Imports System.IO
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class GraduateSchool
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not IsPostBack Then
BindGrid()
btnsubmit.Enabled = True
Session("GSID") = ""
End If
End Sub
Private Sub BindGrid()
Using con As New SqlConnection(cnstring)
Using cmd As New SqlCommand("SELECT * FROM GraduateSchool")
Using sda As New SqlDataAdapter()
cmd.Connection = con
sda.SelectCommand = cmd
Using dt As New DataTable()
sda.Fill(dt)
GVGschool.DataSource = dt
GVGSchool.DataBind()
End Using
End Using
End Using
End Using
End Sub
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles btnsubmit.Click
If txtName.Text = "" Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
```

```
icon: 'error',
title: 'Oops...',
text: 'Plz fill the blanks!'
});", True)
Return
Exit Sub
End If
If Session("Checkresult") <> 0 Then
lblexist.Visible = True
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'This Graduate school Already exists!'
});", True)
lblexist.Text = "This Graduate school Already exists.."
Return
Exit Sub
End If
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"INSERT INTO GraduateSchool " &
" VALUES (@Name,@regDATE) " &
"COMMIT"
cmd.Parameters.AddWithValue("@Name", txtName.Text)
cmd.Parameters.AddWithValue("@regDATE", DateTime.Now)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("Saved...", "insertion", MessageBoxButtons.OK,
MessageBoxIcon.Information)
clear()
cn.Close()
BindGrid()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-
end', icon: 'success', title: 'Graduate school has been saved', showConfirmButton: false,
timer: 1500});", True)
Logfile(Application("LoggedID"), "Graduate school inserted")
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
```

```
"clear function
Sub clear()
txtName.Text = ""
lblexist.Text = ""
txtName.BorderColor = Drawing.Color.Black
lblexist.Visible = False
txtName.Focus()
Session("GSID") = ""
btnupdate.Enabled = False
btndelete.Enabled = False
btnsubmit.Enabled = True
End Sub
"check if the graduate school is already registred
'Dim Checkresult As Integer = 0
Sub check()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select * from GraduateSchool " &
" where Name= @nme " &
"COMMIT"
cmd.Parameters.AddWithValue("@nme", txtName.Text)
Try
connection()
cn.Open()
Session("Checkresult") = CInt(cmd.ExecuteScalar())
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Protected Sub GVGschool_SelectedIndexChanged(sender As Object, e As EventArgs) Handles GVGschool.SelectedIndexChanged
Session("GSID") = GVGschool.SelectedRow.Cells(1).Text.ToString
txtName.Text = GVGschool.SelectedRow.Cells(2).Text
btnsubmit.Enabled = False
btndelete.Enabled = True
btnupdate.Enabled = True
End Sub
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles btnupdate.Click
If txtName.Text = "" Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
```

```
title: 'Deleted...',
text: 'Plz fill the blanks!'
});", True)
Return
Exit Sub
End If
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
" Update GraduateSchool " &
" set name=@Name " &
" where ID=@GID " &
" COMMIT"
cmd.Parameters.AddWithValue("@Name", txtName.Text)
cmd.Parameters.AddWithValue("@GID", Session("GSID"))
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("updated...", "insertion", MessageBoxButtons.OK,
MessageBoxIcon.Information)
clear()
cn.Close()
BindGrid()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Graduate school Details Updated Successfully.'
});", True)
Logfile(Application("LoggedID"), "Graduate school Updated")
Catch ex As Exception
'MsgBox.Show("updating Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub btndelete_Click(sender As Object, e As EventArgs) Handles btndelete.Click
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
" Delete GraduateSchool " &
" where ID= @GSCID " &
" COMMIT"
cmd.Parameters.AddWithValue("@GSCID", Session("GSID"))
```

```
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
'MessageBox.Show("deleted...", "insertion", MessageBoxButtons.OK,
MessageBoxIcon.Information)
clear()
cn.Close()
BindGrid()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Graduate school Deleted Successfully.'
});", True)
Logfile(Application("LoggedID"), "Graduate school deleted")
Catch ex As Exception
'MsgBox.Show("deleting Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Student Registration code

```
Imports System.IO
Imports System.Data
Imports System.Configuration
Imports System.Data.SqlClient
Public Class Student
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not Me.IsPostBack Then
BindDropDownList()
BindGView()
End If
End Sub
"bind to gridview
Sub BindGView()
Using conn As SqlConnection = New SqlConnection(cnstring)
Using sda As SqlDataAdapter = New SqlDataAdapter("SELECT StudentNO,Name,Nationality,Gender,DOB,POB,TellNO,Email,AcademicYear,Level,Department,Interests,imgdata from student", conn)
Dim dt As DataTable = New DataTable()
sda.Fill(dt)
GVStudent.DataSource = dt
GVStudent.DataBind()
End Using
End Using
End Sub
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles btnsubmit.Click
If txtstudentNO.Text <> "" OrElse txtName.Text <> "" OrElse DDLNationality.SelectedIndex <> 0 OrElse DDLGender.SelectedIndex <> 0 OrElse txtDOB.Text <> "" OrElse txtPOB.Text <> "" OrElse txtTelNO.Text <> "" OrElse txtEmail.Text <> "" OrElse
```

```
txtAcademicyear.Text <> "" OrElse DDLlevel.SelectedIndex <> 0 OrElse
DDLDepartment.SelectedIndex <> 0 OrElse
txtinterests.Text <> "" Then
Dim bytes As Byte()
Using br As BinaryReader = New BinaryReader(FileUpload1.PostedFile.InputStream)
bytes = br.ReadBytes(FileUpload1.PostedFile.ContentLength)
End Using
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "INSERT INTO student VALUES(@StudentNO, @Name,
@Nationality, @Gender, @DOB, @POB, @TellNO, @Email, @AcademicYear, @Level,
@Department, @Interests,@imgName,@imgType, @imgdata,@regdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@StudentNO", txtstudentNO.Text)
cmd.Parameters.AddWithValue("@Name", txtName.Text)
cmd.Parameters.AddWithValue("@Nationality", DDLNationality.Text)
cmd.Parameters.AddWithValue("@Gender", DDLGender.Text)
cmd.Parameters.AddWithValue("@DOB", CDate(txtDOB.Text))
cmd.Parameters.AddWithValue("@POB", txtPOB.Text)
cmd.Parameters.AddWithValue("@TellNO", txtTelNO.Text)
cmd.Parameters.AddWithValue("@Email", txtEmail.Text)
cmd.Parameters.AddWithValue("@AcademicYear", txtAcademicyear.Text)
cmd.Parameters.AddWithValue("@Level", DDLlevel.Text)
cmd.Parameters.AddWithValue("@Department", DDLDepartment.SelectedValue)
cmd.Parameters.AddWithValue("@Interests", txtinterests.Text)
cmd.Parameters.AddWithValue("@imgName",
Path.GetFileName(FileUpload1.PostedFile.FileName))
cmd.Parameters.AddWithValue("@imgType", FileUpload1.PostedFile.ContentType)
cmd.Parameters.AddWithValue("@imgdata", bytes)
cmd.Parameters.AddWithValue("@regdate", Date.Now)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-
end', icon: 'success', title: 'Student Registred Successfully', showConfirmButton: false,
timer: 1500});", True)
Logfile(Application("LoggedID"), "Student info Inserted")
BindGView()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
Else
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
```

```
title: 'Oops...',
text: 'Plz fill the blanks!'
});", True)
End If
End Sub
"bind dropdown list items from sql
Sub BindDropDownList()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = "SELECT ID,name FROM Department"
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLDepartment.DataSource = dt
DDLDepartment.DataValueField = "ID"
DDLDepartment.DataTextField = "name"
DDLDepartment.DataBind()
sqlConn.Close()
'Adding "Please select" option in dropdownlist for validation
DDLDepartment.Items.Insert(0, New ListItem("Please select", "0"))
End Using
End Using
Catch
End Try
End Sub
Protected Sub GVStudent_RowDataBound(sender As Object, e As
GridViewRowEventArgs) Handles GVStudent.RowDataBound
If e.Row.RowType = DataControlRowType.DataRow Then
Dim dr As DataRowView = CType(e.Row.DataItem, DataRowView)
Dim imageUrl As String = "" &
Convert.ToBase64String(CType(dr("imgdata"), Byte()))
CType(e.Row.FindControl("Image1"), Image).ImageUrl = imageUrl
End If
End Sub
Protected Sub GVStudent_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles GVStudent.SelectedIndexChanged
Session("StudentNO") = GVStudent.SelectedRow.Cells(1).Text.ToString
"bind data ffrom gridview to controls
getdatatocontrols()
btnsubmit.Enabled = False
btndelete.Enabled = True
btnupdate.Enabled = True
End Sub
```

```
"Bind Data into the controls
Sub getdatatocontrols()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select * from  student where StudentNO=@StdNO"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@StdNO", Session("StudentNO"))
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
txtstudentNO.Text = DRead(0)
txtName.Text = DRead(1)
DDLNationality.Text = DRead(2)
DDLGender.Text = DRead(3)
txtDOB.Text = CDate(DRead(4))
Session("DOB") = CDate(DRead(4))
txtPOB.Text = DRead(5)
txtTelNO.Text = DRead(6)
txtEmail.Text = DRead(7)
txtAcademicyear.Text = DRead(8)
DDLlevel.Text = DRead(9)
DDLDepartment.Text = DRead(10)
txtinterests.Text = DRead(11)
Session("imgName") = DRead(12)
Session("imgType") = DRead(13)
Session("imgData") = DRead(14)
End While
End If
conn.Close()
BindGView()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles
btnupdate.Click
If txtstudentNO.Text <> "" OrElse txtName.Text <> "" OrElse
DDLNationality.SelectedIndex <> 0 OrElse DDLGender.SelectedIndex <> 0 OrElse
txtDOB.Text <> "" OrElse txtTelNO.Text <> "" OrElse txtEmail.Text <> "" OrElse
txtAcademicyear.Text <> "" OrElse DDLlevel.SelectedIndex <> 0 OrElse
DDLDepartment.SelectedIndex <> 0 OrElse
txtinterests.Text <> "" Then
If FileUpload1.HasFile = True Then
```

```
Dim bytes As Byte()
Using br As BinaryReader = New BinaryReader(FileUpload1.PostedFile.InputStream)
bytes = br.ReadBytes(FileUpload1.PostedFile.ContentLength)
End Using
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Update student set StudentNO=@SNO, Name=@sNme,
Nationality=@SNationality, Gender=@SGender,DOB= @SDOB, POB=@SPOB,
TellNO=@STellNO, Email=@SEmail, AcademicYear=@SAcademicYear,Level=
@SLevel,Department= @SDepartment,
Interests=@SInterests,imgName=@SimgName,imgType=@SimgType,
imgdata=@Simgdata,regdate=@Sregdate " &
" WHERE StudentNO=@snumber"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@snumber", Session("StudentNO"))
cmd.Parameters.AddWithValue("@SNO", txtstudentNO.Text)
cmd.Parameters.AddWithValue("@SNme", txtName.Text)
cmd.Parameters.AddWithValue("@SNationality", DDLNationality.Text)
cmd.Parameters.AddWithValue("@SGender", DDLGender.Text)
If txtDOB.Text = "" Then
cmd.Parameters.AddWithValue("@SDOB", Session("DOB"))
Else
cmd.Parameters.AddWithValue("@SDOB", CDate(txtDOB.Text))
End If
cmd.Parameters.AddWithValue("@SPOB", txtPOB.Text)
cmd.Parameters.AddWithValue("@STellNO", txtTelNO.Text)
cmd.Parameters.AddWithValue("@SEmail", txtEmail.Text)
cmd.Parameters.AddWithValue("@SAcademicYear", txtAcademicyear.Text)
cmd.Parameters.AddWithValue("@SLevel", DDLlevel.Text)
cmd.Parameters.AddWithValue("@SDepartment", DDLDepartment.SelectedValue)
cmd.Parameters.AddWithValue("@SInterests", txtinterests.Text)
cmd.Parameters.AddWithValue("@SimgName",
Path.GetFileName(FileUpload1.PostedFile.FileName))
cmd.Parameters.AddWithValue("@SimgType", FileUpload1.PostedFile.ContentType)
cmd.Parameters.AddWithValue("@SimgData", bytes)
cmd.Parameters.AddWithValue("@Sregdate", Date.Now)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
clear()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Student Details Updated Successfully.'
});", True)
Logfile(Application("LoggedID"), "Student info updated")
BindGView()
```

```
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
Else
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Update student set StudentNO=@StdNO, Name=@stdNme, Nationality=@StdNationality, Gender=@StdGender,DOB= @StdDOB, POB=@StdPOB, TellNO=@StdTellNO, Email=@StdEmail, AcademicYear=@stdAcademicYear,Level= @StdLevel,Department= @StdDepartment, Interests=@StdInterests " &
" WHERE StudentNO=@stdnumber"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@stdnumber", Session("StudentNO"))
cmd.Parameters.AddWithValue("@StdNO", txtstudentNO.Text)
cmd.Parameters.AddWithValue("@stdNme", txtName.Text)
cmd.Parameters.AddWithValue("@stdNationality", DDLNationality.Text)
cmd.Parameters.AddWithValue("@stdGender", DDLGender.Text)
If txtDOB.Text = "" Then
cmd.Parameters.AddWithValue("@SDOB", Session("DOB"))
Else
cmd.Parameters.AddWithValue("@SDOB", CDate(txtDOB.Text))
End If
cmd.Parameters.AddWithValue("@stdPOB", txtPOB.Text)
cmd.Parameters.AddWithValue("@stdTellNO", txtTelNO.Text)
cmd.Parameters.AddWithValue("@stdEmail", txtEmail.Text)
cmd.Parameters.AddWithValue("@stdAcademicYear", txtAcademicyear.Text)
cmd.Parameters.AddWithValue("@stdLevel", DDLlevel.Text)
cmd.Parameters.AddWithValue("@stdDepartment", DDLDepartment.SelectedValue)
cmd.Parameters.AddWithValue("@stdInterests", txtinterests.Text)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Logfile(Application("LoggedID"), "Student info updated")
clear()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Student Details Updated Successfully.'
});", True)
BindGView()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
```

```
End Using
End Using
End If
End If
End Sub
Protected Sub btndelete_Click(sender As Object, e As EventArgs) Handles btndelete.Click
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Delete student WHERE StudentNO=@studentnumber"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@studentnumber", Session("StudentNO"))
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
BindGView()
Logfile(Application("LoggedID"), "Student info deleted")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Student Details Deleted Successfully.'
});", True)
clear()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
"clear function
Sub clear()
txtstudentNO.Text = ""
txtName.Text = ""
DDLNationality.SelectedIndex = 0
DDLGender.SelectedIndex = 0
BindDropDownList()
txtDOB.Text = ""
txtPOB.Text = ""
txtTelNO.Text = ""
txtEmail.Text = ""
txtAcademicyear.Text = ""
DDLlevel.SelectedIndex = 0
DDLDepartment.SelectedIndex = 0
txtinterests.Text = ""
txtName.BorderColor = Drawing.Color.Black
```

```
txtstudentNO.Focus()
btnupdate.Enabled = False
btndelete.Enabled = False
btnsubmit.Enabled = True
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
Imports System.IO
Imports System.Data
Imports System.Configuration
Imports System.Data.SqlClient
Public Class Supervisor
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V;
database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
If Not Me.IsPostBack Then
BindDropDownList()
BindDDLTitle()
```

```
BindGView()
End If
End Sub
"bind to gridview
Sub BindGView()
Using conn As SqlConnection = New SqlConnection(cnstring)
Using sda As SqlDataAdapter = New SqlDataAdapter("SELECT ID,Name,Gender,DOB,POB,TellNO,Email,Title,Department,Interests,imgdata from Supervisor", conn)
Dim dt As DataTable = New DataTable()
Try
sda.Fill(dt)
GVSupervisor.DataSource = dt
GVSupervisor.DataBind()
Catch ex As Exception
End Try
End Using
End Using
End Sub
"bind Departmen dropdown list items from sql
Sub BindDropDownList()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = “SELECT ID,name FROM Department”
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLDepartment.DataSource = dt
DDLDepartment.DataValueField = “ID”
DDLDepartment.DataTextField = “name”
DDLDepartment.DataBind()
sqlConn.Close()
'Adding “Please select” option in dropdownlist for validation
DDLDepartment.Items.Insert(0, New ListItem(“Please select”, “0”))
End Using
End Using
Catch
End Try
End Sub
"bind title dropdown list items from sql
Sub BindDDLTitle()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
```

```
sqlCmd.CommandText = “SELECT ID,name FROM EduTitle”
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLTitle.DataSource = dt
DDLTitle.DataValueField = “ID”
DDLTitle.DataTextField = “name”
DDLTitle.DataBind()
sqlConn.Close()
'Adding “Please select” option in dropdownlist for validation
DDLTitle.Items.Insert(0, New ListItem(“Please select”, “0”))
End Using
End Using
Catch
End Try
End Sub
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles
btnsubmit.Click
If txtDOB.Text <> "" OrElse txtEmail.Text <> "" OrElse txtinterests.Text <> "" OrElse
txtName.Text <> "" OrElse txtTelNO.Text <> "" OrElse txtPOB.Text <> "" OrElse
DDLDepartment.SelectedIndex <> 0 OrElse DDLGender.SelectedIndex <> 0 OrElse
DDLTitle.SelectedIndex <> 0 Then
Dim bytes As Byte()
Using br As BinaryReader = New BinaryReader(FileUpload1.PostedFile.InputStream)
bytes = br.ReadBytes(FileUpload1.PostedFile.ContentLength)
End Using
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "INSERT INTO Supervisor VALUES( @Name, @Gender, @DOB,
@POB, @TellNO, @Email, @Title, @Department, @Interests,@imgName,@imgType,
@imgdata,@regdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@Name", txtName.Text)
cmd.Parameters.AddWithValue("@Gender", DDLGender.Text)
cmd.Parameters.AddWithValue("@DOB", CDate(txtDOB.Text))
cmd.Parameters.AddWithValue("@POB", txtPOB.Text)
cmd.Parameters.AddWithValue("@TellNO", txtTelNO.Text)
cmd.Parameters.AddWithValue("@Email", txtEmail.Text)
cmd.Parameters.AddWithValue("@Title", DDLTitle.SelectedValue)
cmd.Parameters.AddWithValue("@Department", DDLDepartment.SelectedValue)
cmd.Parameters.AddWithValue("@Interests", txtinterests.Text)
cmd.Parameters.AddWithValue("@imgName",
Path.GetFileName(FileUpload1.PostedFile.FileName))
cmd.Parameters.AddWithValue("@imgType", FileUpload1.PostedFile.ContentType)
cmd.Parameters.AddWithValue("@imgData", bytes)
cmd.Parameters.AddWithValue("@regdate", Date.Now)
```

```
Try
conn.Open()
cmd.ExecuteNonQuery()
clear()
conn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'Supervisor has been Registred', showConfirmButton: false, timer: 1500});", True)
Logfile(Application("LoggedID"), "inserted new student")
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
Else
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'plz fill the blanks!'
});", True)
End If
End Sub
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles btnupdate.Click
If txtEmail.Text <> "" OrElse txtinterests.Text <> "" OrElse txtName.Text <> "" OrElse txtTelNO.Text <> "" OrElse txtPOB.Text <> "" OrElse DDLDepartment.SelectedIndex <> 0 OrElse DDLGender.SelectedIndex <> 0 OrElse DDLTitle.SelectedIndex <> 0 Then
If FileUpload1.HasFile = True Then
Dim bytes As Byte()
Using br As BinaryReader = New BinaryReader(FileUpload1.PostedFile.InputStream)
bytes = br.ReadBytes(FileUpload1.PostedFile.ContentLength)
End Using
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Update Supervisor set Name= @uName,Gender= @uGender, DOB=@uDOB, POB=@uPOB, TellNO=@uTellNO, Email=@uEmail,Title= @uTitle, Department=@uDepartment,Interests=
@uInterests,imgName=@uimgName,imgType=@uimgType, imgdata=@uimgdata" &
" where ID=@SUperID"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@SUperID", Session("SupervisorID"))
cmd.Parameters.AddWithValue("@uName", txtName.Text)
cmd.Parameters.AddWithValue("@uGender", DDLGender.Text)
If txtDOB.Text = "" Then
cmd.Parameters.AddWithValue("@uDOB", Session("SUDOB"))
Else
```

```
cmd.Parameters.AddWithValue("@uDOB", CDate(txtDOB.Text))
End If
cmd.Parameters.AddWithValue("@uPOB", txtPOB.Text)
cmd.Parameters.AddWithValue("@uTellNO", txtTelNO.Text)
cmd.Parameters.AddWithValue("@uEmail", txtEmail.Text)
cmd.Parameters.AddWithValue("@uTitle", DDLTitle.SelectedValue)
cmd.Parameters.AddWithValue("@uDepartment", DDLDepartment.SelectedValue)
cmd.Parameters.AddWithValue("@uInterests", txtinterests.Text)
cmd.Parameters.AddWithValue("@uimgName",
Path.GetFileName(FileUpload1.PostedFile.FileName))
cmd.Parameters.AddWithValue("@uimgType", FileUpload1.PostedFile.ContentType)
cmd.Parameters.AddWithValue("@uimgData", bytes)
Try
conn.Open()
cmd.ExecuteNonQuery()
clear()
conn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Student information Updated Successfully.'
});", True)
Logfile(Application("LoggedID"), "Updated student details")
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
Else
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Update Supervisor set Name= @uName,Gender= @uGender,
DOB=@uDOB, POB=@uPOB, TellNO=@uTellNO, Email=@uEmail,Title= @uTitle,
Department=@uDepartment,Interests= @uInterests " &
" where ID=@AID"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@AID", Session("SupervisorID"))
cmd.Parameters.AddWithValue("@uName", txtName.Text)
cmd.Parameters.AddWithValue("@uGender", DDLGender.Text)
If txtDOB.Text = "" Then
cmd.Parameters.AddWithValue("@uDOB", Session("SUDOB"))
Else
cmd.Parameters.AddWithValue("@uDOB", CDate(txtDOB.Text))
End If
cmd.Parameters.AddWithValue("@uPOB", txtPOB.Text)
cmd.Parameters.AddWithValue("@uTellNO", txtTelNO.Text)
cmd.Parameters.AddWithValue("@uEmail", txtEmail.Text)
```

```
cmd.Parameters.AddWithValue("@uTitle", DDLTitle.SelectedValue)
cmd.Parameters.AddWithValue("@uDepartment", DDLDepartment.SelectedValue)
cmd.Parameters.AddWithValue("@uInterests", txtinterests.Text)
Try
conn.Open()
cmd.ExecuteNonQuery()
clear()
conn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Student information Updated Successfully.'
});", True)
Logfile(Application("LoggedID"), "Updated student details")
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End If
End If
End Sub
Protected Sub btndelete_Click(sender As Object, e As EventArgs) Handles btndelete.Click
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "delete Supervisor where ID =@SUID"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@SUID", Session("SupervisorID"))
Try
conn.Open()
cmd.ExecuteNonQuery()
clear()
conn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Student Information Deleted Successfully.'
});", True)
Logfile(Application("LoggedID"), "student details deleted")
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
```

```
Protected Sub GVSupervisor_SelectedIndexChanged(sender As Object, e As EventArgs) Handles GVSupervisor.SelectedIndexChanged
Session("SupervisorID") = GVSupervisor.SelectedRow.Cells(1).Text.ToString
"bind data ffrom gridview to controls
getdatatocontrols()
btnsubmit.Enabled = False
btndelete.Enabled = True
btnupdate.Enabled = True
End Sub
"Bind Data into the controls
Sub getdatatocontrols()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select * from  Supervisor where ID=@SNO"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@SNO", Session("SupervisorID"))
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
Session("SupervisorID") = DRead(0)
txtName.Text = DRead(1)
DDLGender.Text = DRead(2)
Session("SUDOB") = CDate(DRead(3))
txtDOB.Text = CDate(DRead(3))
txtPOB.Text = DRead(4)
txtTelNO.Text = DRead(5)
txtEmail.Text = DRead(6)
DDLTitle.Text = DRead(7)
DDLDepartment.Text = DRead(8)
txtinterests.Text = DRead(9)
Session("imgName") = DRead(10)
Session("imgType") = DRead(11)
Session("imgData") = DRead(12)
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
Protected Sub GVSupervisor_RowDataBound(sender As Object, e As GridViewRowEventArgs) Handles GVSupervisor.RowDataBound
If e.Row.RowType = DataControlRowType.DataRow Then
```

```
Dim dr As DataRowView = CType(e.Row.DataItem, DataRowView)
Dim imageUrl As String = "" &
Convert.ToBase64String(CType(dr("imgdata"), Byte()))
CType(e.Row.FindControl("Image1"), Image).ImageUrl = imageUrl
End If
End Sub
"clear function
Sub clear()
txtName.Text = ""
DDLGender.SelectedIndex = 0
BindDropDownList()
BindDDLTitle()
BindGView()
txtDOB.Text = ""
txtPOB.Text = ""
txtTelNO.Text = ""
txtEmail.Text = ""
txtinterests.Text = ""
DDLTitle.Text = ""
DDLDepartment.SelectedIndex = 0
txtinterests.Text = ""
Session("imgName") = ""
Session("imgType") = ""
Session("imgData") = ""
txtName.BorderColor = Drawing.Color.Black
Session("SupervisorID") = ""
txtName.Focus()
btnupdate.Enabled = False
btndelete.Enabled = False
btnsubmit.Enabled = True
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
```

```
End Using
End Sub
End Class
```

User Registration code

```
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class Users
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public userID As String = ""
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not IsPostBack Then
BindGrid()
btnsubmit.Enabled = True
Session("ID") = ""
End If
End Sub
Private Sub BindGrid()
Using con As New SqlConnection(cnstring)
Using cmd As New SqlCommand("SELECT * FROM Users")
Using sda As New SqlDataAdapter()
cmd.Connection = con
sda.SelectCommand = cmd
Using dt As New DataTable()
sda.Fill(dt)
GVusers.DataSource = dt
GVusers.DataBind()
End Using
End Using
End Using
End Using
End Sub
Protected Sub OnPageIndexChanging(sender As Object, e As GridViewPageEventArgs)
GVusers.PageIndex = e.NewPageIndex
Me.BindGrid()
End Sub
```

```
"Button submit code
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles btnsubmit.Click
If txtName.Text = "" OrElse txtUserName.Text = "" OrElse txtpassword.Text = "" OrElse txtRetypePassword.Text = "" OrElse DDLRole.SelectedIndex = 0 OrElse DDLstatus.SelectedIndex = 0 OrElse txtAnswer.Text = "" OrElse ddlquestion.SelectedIndex = 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'plz fill the blank!'
});", True)
lblinfo.Text = "plz fill the blank..."
Return
Exit Sub
End If
checkUsername()
CheckUserID()
If checkusernme <> 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'sorry! this Username is already taken!'
});", True)
lblinfo.Text = "Username is already taken"
Return
Exit Sub
End If
If checkUsrID <> 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'sorry! this user is already exist!'
});", True)
lblinfo.Text = "this user is already exist"
Return
Exit Sub
End If
" to get the id of the username if he/she is an admin
checkID()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"INSERT INTO users " &
```

```
" VALUES (@ID,@fullName,@username, @password, @status, @Role,@Question,@Answer, @regdate) " &
"COMMIT"
If DDLRole.SelectedIndex = 1 Then
AdminCount += 1
cmd.Parameters.AddWithValue("@ID", AdminCount)
Else
cmd.Parameters.AddWithValue("@ID", DDLID.Text)
End If
cmd.Parameters.AddWithValue("@Question", ddlquestion.Text)
cmd.Parameters.AddWithValue("@Answer", txtAnswer.Text)
cmd.Parameters.AddWithValue("@fullName", txtName.Text)
cmd.Parameters.AddWithValue("@username", txtUserName.Text)
cmd.Parameters.AddWithValue("@password", txtpassword.Text)
cmd.Parameters.AddWithValue("@status", DDLstatus.Text)
cmd.Parameters.AddWithValue("@Role", DDLRole.Text)
cmd.Parameters.AddWithValue("@regdate", DateTime.Now)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
clear()
cn.Close()
Logfile(Application("LoggedID"), "Registred a new user")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'User has been saved', showConfirmButton: false, timer: 1500});", True)
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
BindGrid()
End Sub
" retype password text changed to check if it matches the password
Protected Sub txtRetypePassword_TextChanged(sender As Object, e As EventArgs) Handles txtRetypePassword.TextChanged
If txtRetypePassword.Text <> txtpassword.Text Then
txtRetypePassword.Text = ""
txtRetypePassword.BorderColor = Drawing.Color.Red
txtRetypePassword.Focus()
End If
End Sub
"clear function
Sub clear()
lblinfo.Text = ""
txtName.Text = ""
```

```
txtpassword.Text = ""
txtRetypePassword.Text = ""
txtUserName.Text = ""
DDLRole.SelectedIndex = 0
DDLstatus.SelectedIndex = 0
ddlquestion.SelectedIndex = 0
txtAnswer.Text = ""
DDLID.SelectedIndex = 0
txtName.BorderColor = Drawing.Color.Black
txtpassword.BorderColor = Drawing.Color.Black
txtRetypePassword.BorderColor = Drawing.Color.Black
txtUserName.BorderColor = Drawing.Color.Black
DDLRole.BorderColor = Drawing.Color.Black
DDLstatus.BorderColor = Drawing.Color.Black
txtName.Focus()
Session("ID") = ""
Session("USID") = ""
btnupdate.Enabled = False
btndelete.Enabled = False
btnsubmit.Enabled = True
End Sub
"Gridview selected index changed
Protected Sub GVusers_SelectedIndexChanged(sender As Object, e As EventArgs) Handles GVusers.SelectedIndexChanged
Session("ID") = GVusers.SelectedRow.Cells(1).Text.ToString
userID = GVusers.SelectedRow.Cells(1).Text.ToString
Session("USID") = GVusers.SelectedRow.Cells(2).Text.ToString
txtName.Text = GVusers.SelectedRow.Cells(3).Text
txtUserName.Text = GVusers.SelectedRow.Cells(4).Text
txtpassword.Text = GVusers.SelectedRow.Cells(5).Text
txtRetypePassword.Text = GVusers.SelectedRow.Cells(5).Text
Dim status As String = GVusers.SelectedRow.Cells(6).Text
Dim role As String = GVusers.SelectedRow.Cells(7).Text
If status.Equals("Active") Then
DDLstatus.SelectedIndex = 1
Else
DDLstatus.SelectedIndex = 2
End If
If role.Equals("Admin") Then
DDLRole.SelectedIndex = 1
ElseIf role.Equals("Student") Then
DDLRole.SelectedIndex = 2
ElseIf role.Equals("Advisor") Then
DDLRole.SelectedIndex = 3
ElseIf role.Equals("Suoervisor") Then
DDLRole.SelectedIndex = 4
ElseIf role.Equals("Examiner") Then
```

```
DDLRole.SelectedIndex = 5
End If
btnsubmit.Enabled = False
btndelete.Enabled = True
btnupdate.Enabled = True
End Sub
"Button uPdate code
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles btnupdate.Click
If txtName.Text = "" OrElse txtUserName.Text = "" OrElse txtpassword.Text = "" OrElse txtRetypePassword.Text = "" OrElse DDLRole.SelectedIndex = 0 OrElse DDLstatus.SelectedIndex = 0 OrElse txtAnswer.Text = "" OrElse ddlquestion.SelectedIndex = 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'plz fill the blank!'
});", True)
lblinfo.Text = "plz fill the blank..."
Return
Exit Sub
End If
checkUsername()
CheckUserID()
If checkusernme <> 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Sorry! Username is already taken!'
});", True)
lblinfo.Text = "Username is already taken"
Return
Exit Sub
End If
If checkUsrID <> 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Sorry! this User is already exist!'
});", True)
lblinfo.Text = "this user is already exist"
Return
Exit Sub
End If
checkID()
cmd = New SqlCommand()
cmd.Connection = cn
```

```
cmd.CommandText =
" BEGIN TRANSACTION " &
" Update users " &
"  set  ID=@ID,fullName=@Name,[userName]=@uname,  password=@psswrd,Status=
@stts,  Role=@Rle,secretQuestion=@Question,secretAnswer=@Answer,[RegistredDate]=
@rgdate " &
" where userID=@UID " &
" COMMIT"
cmd.Parameters.AddWithValue("@Name", txtName.Text)
If DDLID.Text = "" Then
cmd.Parameters.AddWithValue("@ID", Session("USID"))
Else
cmd.Parameters.AddWithValue("@ID", DDLID.SelectedValue)
End If
cmd.Parameters.AddWithValue("@Question", ddlquestion.Text)
cmd.Parameters.AddWithValue("@Answer", txtAnswer.Text)
cmd.Parameters.AddWithValue("@uname", txtUserName.Text)
cmd.Parameters.AddWithValue("@psswrd", txtpassword.Text)
cmd.Parameters.AddWithValue("@stts", DDLstatus.Text)
cmd.Parameters.AddWithValue("@Rle", DDLRole.Text)
cmd.Parameters.AddWithValue("@rgdate", DateTime.Now)
cmd.Parameters.AddWithValue("@UID", Session("ID"))
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
clear()
cn.Close()
Logfile(Application("LoggedID"), "Updated user details")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'User Details Updated Successfully.'
});", True)
Catch ex As Exception
'MsgBox.Show("updating Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
BindGrid()
End Sub
"Button delete code
Protected Sub btndelete_Click(sender As Object, e As EventArgs) Handles btndelete.Click
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
```

```
" Delete users " &
" where UserID= @USRID " &
" COMMIT"
cmd.Parameters.AddWithValue("@USRID", Session("ID"))
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
clear()
cn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'User Deleted Successfully.'
});", True)
Logfile(Application("LoggedID"), "Deleted a user")
Catch ex As Exception
'MsgBox.Show("deleting Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
BindGrid()
End Sub
Protected Sub DDLRole_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles DDLRole.SelectedIndexChanged
DDLID.Items.Clear()
If DDLRole.SelectedIndex = 1 Then
txtName.Text = "Admin"
checkID()
ElseIf DDLRole.SelectedIndex = 2 Then
BindDDLID("Student")
ElseIf DDLRole.SelectedIndex = 3 Then
BindDDLID("Advisor")
ElseIf DDLRole.SelectedIndex = 4 Then
BindDDLID("Supervisor")
ElseIf DDLRole.SelectedIndex = 5 Then
BindDDLID("Examiner")
End If
End Sub
"bind dropdown list ID items from sql
Sub BindDDLID(table As String)
If table = "Advisor" Then
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = "SELECT ID from Advisor"
sqlCmd.Connection = sqlConn
```

```
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLID.DataSource = dt
DDLID.DataValueField = "ID"
DDLID.DataTextField = "ID"
DDLID.DataBind()
sqlConn.Close()
'Adding "Please select" option in dropdownlist for validation
DDLID.Items.Insert(0, New ListItem("Please select", "0"))
End Using
End Using
Catch
End Try
ElseIf table = "Supervisor" Then
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = "SELECT ID from Supervisor"
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLID.DataSource = dt
DDLID.DataValueField = "ID"
DDLID.DataTextField = "ID"
DDLID.DataBind()
sqlConn.Close()
'Adding "Please select" option in dropdownlist for validation
DDLID.Items.Insert(0, New ListItem("Please select", "0"))
End Using
End Using
Catch
End Try
ElseIf table = "Examiner" Then
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = "SELECT ID from Examiner"
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLID.DataSource = dt
```

```
DDLID.DataValueField = "ID"
DDLID.DataTextField = "ID"
DDLID.DataBind()
sqlConn.Close()
'Adding "Please select" option in dropdownlist for validation
DDLID.Items.Insert(0, New ListItem("Please select", "0"))
End Using
End Using
Catch
End Try
ElseIf table = "Student" Then
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = "SELECT StudentNO from student"
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
DDLID.DataSource = dt
DDLID.DataValueField = "StudentNO"
DDLID.DataTextField = "StudentNO"
DDLID.DataBind()
sqlConn.Close()
'Adding "Please select" option in dropdownlist for validation
DDLID.Items.Insert(0, New ListItem("Please select", "0"))
End Using
End Using
Catch
End Try
End If
End Sub
Public AdminCount As Integer = 0
Sub checkID()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select count(*) from Users " &
" where Role= @nme " &
"COMMIT"
cmd.Parameters.AddWithValue("@nme", "Admin")
Try
connection()
cn.Open()
AdminCount = CInt(cmd.ExecuteScalar())
```

```
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Protected Sub DDLID_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
DDLID.SelectedIndexChanged
Dim conditional As String = DDLID.SelectedValue.ToString
getname(conditional)
End Sub
"check if the usernAME is already exist
Public checkusernme As Integer = 0
Sub checkUsername()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select * from Users " &
" where userName= @user " &
"COMMIT"
cmd.Parameters.AddWithValue("@user", txtUserName.Text)
Try
connection()
cn.Open()
checkusernme = CInt(cmd.ExecuteScalar())
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
"check if the userID is already exist
Public checkUsrID As Integer = 0
Sub CheckUserID()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select * from Users " &
" where ID= @ID and fullName=@name " &
"COMMIT"
```

```
cmd.Parameters.AddWithValue("@ID", DDLID.SelectedValue)
cmd.Parameters.AddWithValue("@name", txtName.Text)
Try
connection()
cn.Open()
checkUsrID = CInt(cmd.ExecuteScalar())
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
"read the name from the the database
Sub getname(ID As String)
If DDLRole.SelectedValue = "Advisor" Then
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select Name from  Advisor where ID=@AID"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@AID", ID)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
txtName.Text = DRead(0)
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
ElseIf DDLRole.SelectedValue = "Supervisor" Then
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select Name from  Supervisor  where ID=@SID"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@SID", ID)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
txtName.Text = DRead(0)
```

```
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
ElseIf DDLRole.SelectedValue = "Examiner" Then
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select Name from  Examiner where ID=@EID"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@EID", ID)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
txtName.Text = DRead(0)
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
ElseIf DDLRole.SelectedValue = "Student" Then
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select Name from  Student where StudentNO=@STID"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@STID", ID)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
txtName.Text = DRead(0)
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
```

```
End Using
End Using
End If
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Choose title code

```
Imports System.IO
Imports System.Data
Imports System.Configuration
Imports System.Data.SqlClient
Public Class ChooseTitle
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not Me.IsPostBack Then
BindGView()
lblStudentNO.Text = Application("LoggedID")
End If
End Sub
"bind to gridview
Sub BindGView()
Using conn As SqlConnection = New SqlConnection(cnstring)
Using sda As SqlDataAdapter = New SqlDataAdapter("SELECT ID,Title from ProposedTitle where StudentNO='" & Application("LoggedID") & "' ", conn)
Dim dt As DataTable = New DataTable()
sda.Fill(dt)
GVTitle.DataSource = dt
GVTitle.DataBind()
End Using
End Using
End Sub
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles btnsubmit.Click
If txtTitle.Text = "" OrElse txtSignificance.Text = "" OrElse txtRQuestion.Text = "" OrElse txtProblemStatement.Text = "" OrElse
txtLimitations.Text = "" OrElse txtKeyTerms.Text = "" OrElse txtIntroduction.Text = "" Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
```

```
title: 'Oops...',
text: 'Plz Fill the blanks!'
});", True)
lblerror.Text = "Plz fill the blanks..."
Return
Exit Sub
End If
checkTitile()
checkstudentNumber()
If titlechecking <> 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Sorry...',
text: 'Sorry, This Title is already taken!'
});", True)
lblerror.Text = "Sorry, This Title is already taken!"
Return
Exit Sub
End If
If StudentNOChecking <> 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Sorry...',
text: 'Sorry! you already have a title , you can update only!'
});", True)
lblerror.Text = "Sorry! you already have a title , you can update only"
Return
Exit Sub
End If
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"INSERT                                                                    INTO
ProposedTitle(Title,StudentNO,Introduction,ProblemStatement,Significance,RQuestion,Ke
yTerms,Limitations,RegDate,Status) " &
"                                                                          VALUES
(@TTitle,@TSNO,@Titro,@Tproblem,@Tsignificance,@TRQ,@Tkeyterms,@TLimitatio
ns,@TRegdate,@TStatus) " &
" COMMIT"
cmd.Parameters.AddWithValue("@TTitle", txtTitle.Text)
cmd.Parameters.AddWithValue("@TSNO", Application("LoggedID"))
cmd.Parameters.AddWithValue("@Titro", txtIntroduction.Text)
cmd.Parameters.AddWithValue("@Tproblem", txtProblemStatement.Text)
cmd.Parameters.AddWithValue("@Tsignificance", txtSignificance.Text)
cmd.Parameters.AddWithValue("@TRQ", txtRQuestion.Text)
cmd.Parameters.AddWithValue("@Tkeyterms", txtKeyTerms.Text)
```

```
cmd.Parameters.AddWithValue("@TLimitations", txtLimitations.Text)
cmd.Parameters.AddWithValue("@TRegdate", System.DateTime.Now)
cmd.Parameters.AddWithValue("@TStatus", "Pending")
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
lblerror.Text = "Saved!"
lblerror.ForeColor = Drawing.Color.Green
'MessageBox.Show("Saved...", "insertion", MessageBoxButtons.OK,
MessageBoxIcon.Information)
clear()
cn.Close()
BindGView()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire('Thesis
Title!','You Proposed a title', 'success');", True)
Logfile(Application("LoggedID"), "Choosed a title")
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
"clear function
Sub clear()
txtTitle.Text = ""
txtIntroduction.Text = ""
txtProblemStatement.Text = ""
txtSignificance.Text = ""
txtRQuestion.Text = ""
txtKeyTerms.Text = ""
txtLimitations.Text = ""
lblerror.Text = ""
lblerror.ForeColor = Drawing.Color.Red
txtTitle.Focus()
btnupdate.Enabled = False
btnsubmit.Enabled = True
End Sub
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles
btnupdate.Click
Dim emptyTextBoxes = From txt In panelchoosetitle.Controls.OfType(Of TextBox)()
Where String.IsNullOrEmpty(txt.Text)
If Not emptyTextBoxes.Any() Then
checkbelongs()
If checkbelong <> 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
```

```
title: 'Oops...',
text: 'Sorry, This Title is already taken!'
});", True)
lblerror.Text = "Sorry, This Title is already taken!"
Return
Exit Sub
End If
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Update  ProposedTitle " &
"                                                                                                    set
Title=@PTitle,StudentNO=@PSNO,Introduction=@itrod,ProblemStatement=@problemS,
Significance=@Psignificance,RQuestion=@PRQ,KeyTerms=@Pkeyterms,Limitations=@
PLimitations " &
" WHERE ID =@UPID " &
"COMMIT"
cmd.Parameters.AddWithValue("@UPID", Application("ProposedID"))
cmd.Parameters.AddWithValue("@PTitle", txtTitle.Text)
cmd.Parameters.AddWithValue("@PSNO", Application("LoggedID"))
cmd.Parameters.AddWithValue("@itrod", txtIntroduction.Text)
cmd.Parameters.AddWithValue("@problemS", txtProblemStatement.Text)
cmd.Parameters.AddWithValue("@Psignificance", txtSignificance.Text)
cmd.Parameters.AddWithValue("@PRQ", txtRQuestion.Text)
cmd.Parameters.AddWithValue("@Pkeyterms", txtKeyTerms.Text)
cmd.Parameters.AddWithValue("@PLimitations", txtLimitations.Text)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
lblerror.Text = "Updated"
lblerror.ForeColor = Drawing.Color.Green
'MessageBox.Show("updated...",          "insertion",          MessageBoxButtons.OK,
MessageBoxIcon.Information)
clear()
cn.Close()
BindGView()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Thesis Title Details Updated Successfully.'
});", True)
Logfile(Application("LoggedID"), "Update a title")
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
```

```
End Try
cmd.Parameters.Clear()
Else
lblerror.Text = "Plz fill the blanks..."
Return
Exit Sub
End If
End Sub
Protected Sub GVTitle_SelectedIndexChanged(sender As Object, e As EventArgs) Handles GVTitle.SelectedIndexChanged
Application("ProposedID") = GVTitle.SelectedRow.Cells(1).Text.ToString
getdatatocontrols()
btnsubmit.Enabled = False
btnupdate.Enabled = True
End Sub
"Bind Data from gridview into the controls
Sub getdatatocontrols()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select * from  ProposedTitle where ID=@PID"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@PID", Application("ProposedID"))
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
'Application("ProposedTitle") = DRead(1)
txtTitle.Text = DRead(1)
lblStudentNO.Text = DRead(2)
txtIntroduction.Text = DRead(3)
txtProblemStatement.Text = DRead(4)
txtSignificance.Text = DRead(5)
txtRQuestion.Text = DRead(6)
txtKeyTerms.Text = DRead(7)
txtLimitations.Text = DRead(8)
End While
End If
conn.Close()
BindGView()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
Dim titlechecking As Integer = 0
```

```
"check if the title is already taken
Sub checkTitile()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select * from ProposedTitle " &
" WHERE Title =@CheckingTitle " &
"COMMIT"
cmd.Parameters.AddWithValue("@CheckingTitle", txtTitle.Text)
Try
connection()
cn.Open()
titlechecking = cmd.ExecuteScalar()
cn.Close()
Catch ex As Exception
'MsgBox.Show("checking Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
Dim StudentNOChecking As Integer = 0
"check if the student has already proposed a title so he/she can update only
Sub checkstudentNumber()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select * from ProposedTitle " &
" WHERE StudentNO =@CheckingSNO " &
"COMMIT"
cmd.Parameters.AddWithValue("@CheckingSNO", Application("LoggedID"))
Try
connection()
cn.Open()
StudentNOChecking = cmd.ExecuteScalar()
cn.Close()
Catch ex As Exception
'MsgBox.Show("checking Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
"check if the title was belongs to this student
Dim checkbelong As Integer = 0
Sub checkbelongs()
cmd = New SqlCommand()
```

```
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select * from ProposedTitle " &
" WHERE Title =@Ttle and not StudentNO=@Std " &
"COMMIT"
cmd.Parameters.AddWithValue("@Ttle", txtTitle.Text)
cmd.Parameters.AddWithValue("@Std", Application("LoggedID"))
Try
connection()
cn.Open()
checkbelong = cmd.ExecuteScalar()
cn.Close()
Catch ex As Exception
'MsgBox.Show("checking Error ")
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Choose Supervisor code

```
Imports System.IO
Imports System.Data
Imports System.Configuration
Imports System.Data.SqlClient
Public Class ChooseSupervisor
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not Me.IsPostBack Then
getdepartment()
BindGView()
End If
End Sub
Protected Sub GVAppointedSupervisor_RowDataBound(sender As Object, e As GridViewRowEventArgs) Handles GVAppointedSupervisor.RowDataBound
If e.Row.RowType = DataControlRowType.DataRow Then
Dim dr As DataRowView = CType(e.Row.DataItem, DataRowView)
Dim imageUrl As String = "" & Convert.ToBase64String(CType(dr("imgdata"), Byte()))
CType(e.Row.FindControl("Image1"), Image).ImageUrl = imageUrl
End If
End Sub
Dim DepartmentID As String = ""
Dim DepartmentName As String = ""
"get student department
Sub getdepartment()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select Department.ID, Department.name from Department " &
" inner join student " &
" on Department.ID=Student.Department " &
" where student.StudentNO='" & Application("LoggedID") & "' "
Using cmd As SqlCommand = New SqlCommand(sql, conn)
```

```
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
DepartmentID = DRead(0).ToString
DepartmentName = DRead(1).ToString
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
"bind to gridview
Sub BindGView()
Using conn As SqlConnection = New SqlConnection(cnstring)
Using sda As SqlDataAdapter = New SqlDataAdapter("select ID, Name,Gender,TellNO,Email,Interests,Imgdata from Supervisor where Department='" & DepartmentID & "'", conn)
Dim dt As DataTable = New DataTable()
Try
sda.Fill(dt)
GVAppointedSupervisor.DataSource = dt
GVAppointedSupervisor.DataBind()
Catch ex As Exception
End Try
End Using
End Using
End Sub
Protected Sub preview(ByVal sender As Object, ByVal e As EventArgs)
'Reference the Button.
Dim btnread As Button = CType(sender, Button)
'Reference the GridView Row.
Dim row As GridViewRow = CType(btnread.NamingContainer, GridViewRow)
'Save the GridView Row in Session.
Application("SupervisorRow") = row
'Redirect to other Page.
Response.Redirect("PreviewSuperviosr.aspx")
End Sub
Dim supervisorId As String = ""
Protected Sub Appoint(ByVal sender As Object, ByVal e As EventArgs)
'Reference the Button.
Dim btnAppoint As Button = CType(sender, Button)
```

```
'Reference the GridView Row.
Dim row As GridViewRow = CType(btnAppoint.NamingContainer, GridViewRow)
'Save the GridView Row in Session.
Application("SupervisorRow") = row
submit()
End Sub
Sub submit()
If (Not (Application("SupervisorRow")) Is Nothing) Then
'Fetch the GridView Row from Session.
Dim row As GridViewRow = CType(Application("SupervisorRow"), GridViewRow)
'Fetch and display the Cell values.
supervisorId = row.Cells(0).Text
checkIfAppointed()
If checkifappointedS <> 0 Then
"You alreday have a supervisor
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Sorry! you already have a supervisor!'
});", True)
lblerror.Text = "Sorry! you already have a supervisor."
Return
Exit Sub
End If
checkTopic()
If chechkStopic = 0 Then
"choose topic
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Plz choose a topic before you try to appoint asupervisor!!'
});", True)
lblerror.Text = "Plz choose a topic before you try to appoint asupervisor!!"
Return
Exit Sub
End If
appointsupervisor()
End If
End Sub
Dim checkifappointedS As Integer = 0
"this function checks if the student has already appointed to a supervisor and not rejected
Sub checkIfAppointed()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select * from Appointedsupervisor " &
```

```
" where StudentNO= @S and Not Status='rejected'" &
"COMMIT"
cmd.Parameters.AddWithValue("@S", Application("LoggedID"))
Try
connection()
cn.Open()
checkifappointedS = CInt(cmd.ExecuteScalar())
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Dim chechkStopic As Integer = 0
"this function checks if the student has already choose a topic
Sub checkTopic()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select * from ProposedTitle " &
" where StudentNO= @SN " &
"COMMIT"
cmd.Parameters.AddWithValue("@SN", Application("LoggedID"))
Try
connection()
cn.Open()
chechkStopic = CInt(cmd.ExecuteScalar())
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Sub appointsupervisor()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
" INsert into Appointedsupervisor values (@SupervisorID,@StudentNO,'Pending'
,@regdate)" &
" COMMIT"
cmd.Parameters.AddWithValue("@StudentNO", Application("LoggedID"))
cmd.Parameters.AddWithValue("@SupervisorID", supervisorId)
```

```
cmd.Parameters.AddWithValue("@regdate", System.DateTime.Now)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
Logfile(Application("LoggedID"), "Appointed your supervisor")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'You appointed your supervisor..', showConfirmButton: false, timer: 1500});", True)
lblerror.Text = "You appointed your supervisor..."
lblerror.ForeColor = Drawing.Color.Green
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Protected Sub btnsearch_Click(sender As Object, e As EventArgs) Handles btnsearch.Click
If txtsearchSupervisor.Text <> "" Then
Using conn As SqlConnection = New SqlConnection(cnstring)
Using sda As SqlDataAdapter = New SqlDataAdapter("select ID, Name,Gender,TellNO,Email,Interests,Imgdata from Supervisor where Name='" & txtsearchSupervisor.Text & "'", conn)
Dim dt As DataTable = New DataTable()
Try
sda.Fill(dt)
GVAppointedSupervisor.DataSource = dt
GVAppointedSupervisor.DataBind()
Catch ex As Exception
End Try
End Using
End Using
End If
End Sub
Protected Sub btnrefresh_Click(sender As Object, e As EventArgs) Handles btnrefresh.Click
getdepartment()
BindGView()
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
```

```
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Change Supervisor code

```
Imports System.IO
Imports System.Data
Imports System.Configuration
Imports System.Data.SqlClient
Public Class ChangeSupervisor
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not Me.IsPostBack Then
getdepartment()
getOldSipervisor()
PrintOldSupervisor()
BindGView()
End If
End Sub
Protected Sub GVchangeSupervisor_RowDataBound(sender As Object, e As GridViewRowEventArgs) Handles GVchangeSupervisor.RowDataBound
If e.Row.RowType = DataControlRowType.DataRow Then
Dim dr As DataRowView = CType(e.Row.DataItem, DataRowView)
Dim imageUrl As String = "" & Convert.ToBase64String(CType(dr("imgdata"), Byte()))
CType(e.Row.FindControl("Image1"), Image).ImageUrl = imageUrl
End If
End Sub
"Get Old Supoervisor name
Dim oldsupervisorname As String = ""
Sub PrintOldSupervisor()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = " Select Supervisor.name,EduTitle.Name from Supervisor inner join EduTitle on Supervisor.Title=EduTitle.ID where Supervisor.id='" & OldSupervisorId & "'
"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
```

```
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
oldsupervisorname = DRead(0).ToString
lbloldsupervisor.Text = "Your Current Supervisor is :" + DRead(1).ToString + " " +
oldsupervisorname
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
Dim DepartmentID As String = ""
Dim DepartmentName As String = ""
"get student department
Sub getdepartment()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select Department.ID, Department.name from Department " &
" inner join student " &
" on Department.ID=Student.Department " &
" where student.StudentNO='" & Application("LoggedID") & "' "
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
DepartmentID = DRead(0).ToString
DepartmentName = DRead(1).ToString
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
"get student previous supervisor ID
Dim OldSupervisorId As String = ""
```

```
Sub getOldSipervisor()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select SupervisorID from Assignedsupervisor where StudentNO='" &
Application("LoggedID") & "' "
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.Read Then
OldSupervisorId = DRead(0).ToString
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
"bind to gridview
Sub BindGView()
Using conn As SqlConnection = New SqlConnection(cnstring)
Using sda As SqlDataAdapter = New SqlDataAdapter("select ID,
Name,Gender,TellNO,Email,Interests,Imgdata from Supervisor where Department='" &
DepartmentID & "' and not ID='" & OldSupervisorId & "'", conn)
Dim dt As DataTable = New DataTable()
Try
sda.Fill(dt)
GVchangeSupervisor.DataSource = dt
GVchangeSupervisor.DataBind()
Catch ex As Exception
End Try
End Using
End Using
End Sub
Protected Sub preview(ByVal sender As Object, ByVal e As EventArgs)
'Reference the Button.
Dim btnread As Button = CType(sender, Button)
'Reference the GridView Row.
Dim row As GridViewRow = CType(btnread.NamingContainer, GridViewRow)
'Save the GridView Row in Session.
Application("SupervisorRow") = row
'Redirect to other Page.
Response.Redirect("PreviewSuperviosr.aspx")
End Sub
Dim newsupervisorid As String = ""
Protected Sub Appoint(ByVal sender As Object, ByVal e As EventArgs)
```

```
'Reference the Button.
Dim btnAppoint As Button = CType(sender, Button)
'Reference the GridView Row.
Dim row As GridViewRow = CType(btnAppoint.NamingContainer, GridViewRow)
'Save the GridView Row in Session.
Application("SupervisorRow") = row
submit()
End Sub
Sub submit()
If (Not (Application("SupervisorRow")) Is Nothing) Then
'Fetch the GridView Row from Session.
Dim row As GridViewRow = CType(Application("SupervisorRow"), GridViewRow)
'Fetch and display the Cell values.
newsupervisorid = row.Cells(0).Text
checkTopic()
If chechkStopic = 0 Then
"choose topic
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Plz choose a topic before you try to appoint a supervisor!!'
});", True)
lblerror.Text = "Plz choose a topic before you try to appoint a supervisor!!"
lblerror.ForeColor = Drawing.Color.Red
Return
Exit Sub
End If
checkIfAppointed()
If checkifappointedS = 0 Then
"You  have not a supervisor
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Plz You Haven't choosen a supervisor try to appoint a supervisor!!'
});", True)
lblerror.Text = "Plz You Haven't choosen a supervisor try to appoint a supervisor!!"
lblerror.ForeColor = Drawing.Color.Red
Return
Exit Sub
End If
checkduplicatesupervisor()
If checkduplicate <> 0 Then
"You alreday have this  supervisor
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Sorry!! You already appointed this supervisor!'
```

});", True)
lblerror.Text = "Sorry!! You already appointed this supervisor"
lblerror.ForeColor = Drawing.Color.Red
Return
Exit Sub
End If
appointsupervisor()
Logfile(Application("LoggedID"), "Changed Supervisor")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'You appointed your new supervisor your new supervisor', showConfirmButton: false, timer: 1500});", True)
lblerror.Text = "You appointed your new supervisor..."
lblerror.ForeColor = Drawing.Color.Green
End If
End Sub
Dim checkifappointedS As Integer = 0
"this function checks if the student has already appointed to a supervisor and not rejected
Sub checkIfAppointed()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select StudentNO from Appointedsupervisor " &
" where StudentNO= @S and not status='Rejected'" &
"COMMIT"
cmd.Parameters.AddWithValue("@S", Application("LoggedID"))
Try
connection()
cn.Open()
checkifappointedS = CInt(cmd.ExecuteScalar())
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Dim checkduplicate As Integer = 0
"this function checks if the student has already appointed to a supervisor and not rejected
Sub checkduplicatesupervisor()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select StudentNO from Appointedsupervisor " &
" where StudentNO= @S and status='Pending' and SupervisorID=@New " &
"COMMIT"

```
cmd.Parameters.AddWithValue("@S", Application("LoggedID"))
cmd.Parameters.AddWithValue("@New", newsupervisorid)
Try
connection()
cn.Open()
checkduplicate = CInt(cmd.ExecuteScalar())
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Dim chechkStopic As Integer = 0
"this function checks if the student has already choose a topic
Sub checkTopic()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select * from ProposedTitle " &
" where StudentNO= @SN " &
"COMMIT"
cmd.Parameters.AddWithValue("@SN", Application("LoggedID"))
Try
connection()
cn.Open()
chechkStopic = CInt(cmd.ExecuteScalar())
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Sub appointsupervisor()
getOldSipervisor()
DeleteAssignedSupervisor()
updateOldsSpervisorStatus()
insertNewAppointedsupervisor()
End Sub
Sub DeleteAssignedSupervisor()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText = "Delete from Assignedsupervisor where StudentNO=@StudentNO
and SupervisorID=@OldID "
```

```
cmd.Parameters.AddWithValue("@StudentNO", Application("LoggedID"))
cmd.Parameters.AddWithValue("@OldID", OldSupervisorId)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
cn.Close()
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Sub updateOldsSpervisorStatus()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText = "Update Appointedsupervisor set status='Changed' where StudentNO=@StudentNO and SupervisorID=@OldID and not status='rejected' "
cmd.Parameters.AddWithValue("@StudentNO", Application("LoggedID"))
cmd.Parameters.AddWithValue("@OldID", OldSupervisorId)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
cn.Close()
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Sub insertNewAppointedsupervisor()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText = "INsert into Appointedsupervisor values (@SupervisorID,@StudentNO,'Pending' ,@regdate) "
cmd.Parameters.AddWithValue("@StudentNO", Application("LoggedID"))
cmd.Parameters.AddWithValue("@SupervisorID", newsupervisorid)
cmd.Parameters.AddWithValue("@regdate", System.DateTime.Now)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
cn.Close()
```

```
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Submit thesis files code

```
Imports System.IO
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class SubmitThesisForms
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Upload(sender As Object, e As EventArgs) Handles btnUpload.Click
If FileUpload1.HasFile Then
checkAssignedSupervisor()
If checkSupervisor <> 0 Then
Dim filename As String = Path.GetFileName(FileUpload1.PostedFile.FileName)
Dim contentType As String = FileUpload1.PostedFile.ContentType
Using fs As Stream = FileUpload1.PostedFile.InputStream
Using br As New BinaryReader(fs)
Dim bytes As Byte() = br.ReadBytes(DirectCast(fs.Length, Long))
Dim constr As String = cnstring
Using con As New SqlConnection(constr)
Dim query As String = "insert into ThesisForms(Name,ContentType,data,StudentNO,SubmittedDate) values (@Name, @ContentType, @Data,@SNO,@SDATE)"
cmd.CommandText = query
cmd.CommandType = CommandType.Text
cmd.Connection = con
cmd.Parameters.Add("@Name", SqlDbType.VarChar).Value = filename
cmd.Parameters.Add("@ContentType", SqlDbType.NVarChar).Value = contentType
cmd.Parameters.Add("@Data", SqlDbType.Binary).Value = bytes
cmd.Parameters.Add("SNO", SqlDbType.VarChar).Value = Application("LoggedID")
cmd.Parameters.Add("SDATE", SqlDbType.Date).Value = System.DateTime.Now
con.Open()
cmd.ExecuteNonQuery()
con.Close()
Logfile(Application("LoggedID"), "Submitted thesis form")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'Submitted thesis form successfully', showConfirmButton: false, timer: 1500});", True)
```

```
End Using
End Using
End Using
cmd.Parameters.Clear()
Response.Redirect(Request.Url.AbsoluteUri)
Else
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Plz choose a supervisor before you try to submit files!'
});", True)
lblerror.Text = "Plz choose a supervisor before you try to submit files!!"
Return
Exit Sub
End If
End If
End Sub
Protected Sub Page_Load(sender As Object, e As EventArgs) Handles Me.Load
If Not IsPostBack Then
BindGrid()
End If
End Sub
Private Sub BindGrid()
Dim constr As String = cnstring
Using con As New SqlConnection(constr)
Using cmd As New SqlCommand()
cmd.CommandText = "select Id, Name, StudentNO, SubmittedDate from ThesisForms
where studentNO ='" & Application("LoggedID") & "'"
cmd.Connection = con
con.Open()
GridView1.DataSource = cmd.ExecuteReader()
GridView1.DataBind()
con.Close()
End Using
End Using
End Sub
Protected Sub DownloadFile(sender As Object, e As EventArgs)
Dim id As Integer = Integer.Parse(TryCast(sender, LinkButton).CommandArgument)
Dim bytes As Byte()
Dim fileName As String, contentType As String
Dim constr As String = cnstring
Using con As New SqlConnection(constr)
Using cmd As New SqlCommand()
cmd.CommandText = "select Name, Data, ContentType from ThesisForms where Id=@Id"
cmd.Parameters.AddWithValue("@Id", id)
cmd.Connection = con
```

```
con.Open()
Using sdr As SqlDataReader = cmd.ExecuteReader()
sdr.Read()
bytes = DirectCast(sdr("Data"), Byte())
contentType = sdr("ContentType").ToString()
fileName = sdr("Name").ToString()
End Using
con.Close()
End Using
End Using
Response.Clear()
Response.Buffer = True
Response.Charset = ""
Response.Cache.SetCacheability(HttpCacheability.NoCache)
Response.ContentType = contentType
Response.AppendHeader("Content-Disposition", "attachment; filename=" + fileName)
Response.BinaryWrite(bytes)
Response.Flush()
Response.End()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'Downloaded thesis form successfully', showConfirmButton: false, timer: 1500});", True)
End Sub
Protected Sub DeleteFile(sender As Object, e As EventArgs)
Dim id As Integer = Integer.Parse(TryCast(sender, LinkButton).CommandArgument)
Dim constr As String = cnstring
Using con As New SqlConnection(constr)
Using cmd As New SqlCommand()
cmd.CommandText = "delete ThesisForms where Id=@Id"
cmd.Parameters.AddWithValue("@Id", id)
cmd.Connection = con
con.Open()
cmd.ExecuteNonQuery()
con.Close()
Logfile(Application("LoggedID"), "Deleted thesis form")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Thesis form Deleted Successfully.'
});", True)
End Using
End Using
BindGrid()
End Sub
```

```
Dim checkSupervisor As Integer = 0
"this function checks if the student has already appointed to a supervisor and not rejected
Sub checkAssignedSupervisor()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select * from Assignedsupervisor " &
" where StudentNO= @S " &
"COMMIT"
cmd.Parameters.AddWithValue("@S", Application("LoggedID"))
Try
connection()
cn.Open()
checkSupervisor = CInt(cmd.ExecuteScalar())
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Submit thesis forms code

```
Imports System.IO
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class SubmitThesisForms
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Upload(sender As Object, e As EventArgs) Handles btnUpload.Click
If FileUpload1.HasFile Then
checkAssignedSupervisor()
If checkSupervisor <> 0 Then
Dim filename As String = Path.GetFileName(FileUpload1.PostedFile.FileName)
Dim contentType As String = FileUpload1.PostedFile.ContentType
Using fs As Stream = FileUpload1.PostedFile.InputStream
Using br As New BinaryReader(fs)
Dim bytes As Byte() = br.ReadBytes(DirectCast(fs.Length, Long))
Dim constr As String = cnstring
Using con As New SqlConnection(constr)
Dim query As String = "insert into ThesisForms(Name,ContentType,data,StudentNO,SubmittedDate) values (@Name, @ContentType, @Data,@SNO,@SDATE)"
cmd.CommandText = query
cmd.CommandType = CommandType.Text
cmd.Connection = con
cmd.Parameters.Add("@Name", SqlDbType.VarChar).Value = filename
cmd.Parameters.Add("@ContentType", SqlDbType.NVarChar).Value = contentType
cmd.Parameters.Add("@Data", SqlDbType.Binary).Value = bytes
cmd.Parameters.Add("SNO", SqlDbType.VarChar).Value = Application("LoggedID")
cmd.Parameters.Add("SDATE", SqlDbType.Date).Value = System.DateTime.Now
con.Open()
cmd.ExecuteNonQuery()
con.Close()
Logfile(Application("LoggedID"), "Submitted thesis form")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'Submitted thesis form successfully', showConfirmButton: false, timer: 1500});", True)
```

```
End Using
End Using
End Using
cmd.Parameters.Clear()
Response.Redirect(Request.Url.AbsoluteUri)
Else
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Plz choose a supervisor before you try to submit files!'
});", True)
lblerror.Text = "Plz choose a supervisor before you try to submit files!!"
Return
Exit Sub
End If
End If
End Sub
Protected Sub Page_Load(sender As Object, e As EventArgs) Handles Me.Load
If Not IsPostBack Then
BindGrid()
End If
End Sub
Private Sub BindGrid()
Dim constr As String = cnstring
Using con As New SqlConnection(constr)
Using cmd As New SqlCommand()
cmd.CommandText = "select Id, Name, StudentNO, SubmittedDate from ThesisForms
where studentNO ='" & Application("LoggedID") & "'"
cmd.Connection = con
con.Open()
GridView1.DataSource = cmd.ExecuteReader()
GridView1.DataBind()
con.Close()
End Using
End Using
End Sub
Protected Sub DownloadFile(sender As Object, e As EventArgs)
Dim id As Integer = Integer.Parse(TryCast(sender, LinkButton).CommandArgument)
Dim bytes As Byte()
Dim fileName As String, contentType As String
Dim constr As String = cnstring
Using con As New SqlConnection(constr)
Using cmd As New SqlCommand()
cmd.CommandText = "select Name, Data, ContentType from ThesisForms where Id=@Id"
cmd.Parameters.AddWithValue("@Id", id)
cmd.Connection = con
```

```
con.Open()
Using sdr As SqlDataReader = cmd.ExecuteReader()
sdr.Read()
bytes = DirectCast(sdr("Data"), Byte())
contentType = sdr("ContentType").ToString()
fileName = sdr("Name").ToString()
End Using
con.Close()
End Using
End Using
Response.Clear()
Response.Buffer = True
Response.Charset = ""
Response.Cache.SetCacheability(HttpCacheability.NoCache)
Response.ContentType = contentType
Response.AppendHeader("Content-Disposition", "attachment; filename=" + fileName)
Response.BinaryWrite(bytes)
Response.Flush()
Response.End()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'Downloaded thesis form successfully', showConfirmButton: false, timer: 1500});", True)
End Sub
Protected Sub DeleteFile(sender As Object, e As EventArgs)
Dim id As Integer = Integer.Parse(TryCast(sender, LinkButton).CommandArgument)
Dim constr As String = cnstring
Using con As New SqlConnection(constr)
Using cmd As New SqlCommand()
cmd.CommandText = "delete ThesisForms where Id=@Id"
cmd.Parameters.AddWithValue("@Id", id)
cmd.Connection = con
con.Open()
cmd.ExecuteNonQuery()
con.Close()
Logfile(Application("LoggedID"), "Deleted thesis form")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Thesis form Deleted Successfully.'
});", True)
End Using
End Using
BindGrid()
End Sub
Dim checkSupervisor As Integer = 0
"this function checks if the student has already appointed to a supervisor and not rejected
Sub checkAssignedSupervisor()
```

```
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
"Select * from Assignedsupervisor " &
" where StudentNO= @S " &
"COMMIT"
cmd.Parameters.AddWithValue("@S", Application("LoggedID"))
Try
connection()
cn.Open()
checkSupervisor = CInt(cmd.ExecuteScalar())
Catch ex As Exception
'MsgBox.Show("Saving Error ")
cn.Close()
End Try
connection()
cmd.Parameters.Clear()
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Advisor grading code

```
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class Grading
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public userID As String = ""
Public cnstring As String = "Server=DESKTOP-77ANH2V;
database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Dim plan As Integer
Dim background As Integer
Dim design As Integer
Dim presentation As Integer
Dim report As Integer
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
If Not IsPostBack Then
getdepartment()
BindDropDownStudent()
BindGrid()
plan = 0
background = 0
design = 0
presentation = 0
report = 0
End If
End Sub
Dim DepartmentID As String = ""
"Get department
Sub getdepartment()
connection()
cmd.Connection = cn
cmd.CommandText = "select distinct(advisor.Department)  from Advisor " &
" inner join AssignedAdvisor " &
" On Advisor.ID=AssignedAdvisor.AdvisorID " &
"  where Advisor.ID=@ID"
```

```
cmd.Parameters.AddWithValue("@ID", Application("LoggedID"))
Try
cn.Open()
DRead = cmd.ExecuteReader
If DRead.Read Then
DepartmentID = DRead(0)
End If
DRead.Close()
cn.Close()
Catch generatedExceptionName As Exception
MsgBox(generatedExceptionName.Message)
End Try
cmd.Parameters.Clear()
End Sub
"bind gridview
Private Sub BindGrid()
Using con As New SqlConnection(cnstring)
Using cmd As New SqlCommand("Select Id,StudentNO,StudentName,ProjectPlan,RelatedWork,Designandsolution,Presentation,Report,Total,Feedback from Grading where grader='" & Application("LoggedID") & "' order by GradedDate desc")
Using sda As New SqlDataAdapter()
cmd.Connection = con
sda.SelectCommand = cmd
Using dt As New DataTable()
sda.Fill(dt)
GVGrades.DataSource = dt
GVGrades.DataBind()
End Using
End Using
End Using
End Using
End Sub
Sub BindDropDownStudent()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = "select Student.StudentNO,Student.Name from student " &
" inner join AssignedAdvisor " &
" on Student.StudentNO=AssignedAdvisor.StudentNO " &
" where Student.Department=@DP and AssignedAdvisor.AdvisorID=@AID "
sqlCmd.Parameters.AddWithValue("@DP", DepartmentID)
sqlCmd.Parameters.AddWithValue("@AID", Application("LoggedID"))
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
```

```
da.Fill(dt)
ddlStudentNumber.DataSource = dt
ddlStudentNumber.DataValueField = "StudentNO"
ddlStudentNumber.DataTextField = "Name"
ddlStudentNumber.DataBind()
sqlConn.Close()
'Adding "Please select" option in dropdownlist for validation
ddlStudentNumber.Items.Insert(0, New ListItem("Please select", "0"))
End Using
End Using
Catch
End Try
End Sub
Protected Sub DDLPlan_SelectedIndexChanged(sender As Object, e As EventArgs) Handles DDLPlan.SelectedIndexChanged
If DDLPlan.SelectedIndex <> 0 Then
lblplan.Text = DDLPlan.SelectedItem.Text
plan = CInt(DDLPlan.SelectedItem.Text)
totalcounter()
End If
End Sub
Protected Sub DDLbackgroud_SelectedIndexChanged(sender As Object, e As EventArgs) Handles DDLbackgroud.SelectedIndexChanged
If DDLbackgroud.SelectedIndex <> 0 Then
lblbackground.Text = DDLbackgroud.SelectedItem.Text
background = CInt(DDLbackgroud.SelectedItem.Text)
totalcounter()
End If
End Sub
Protected Sub ddldesign_SelectedIndexChanged(sender As Object, e As EventArgs) Handles ddldesign.SelectedIndexChanged
If ddldesign.SelectedIndex <> 0 Then
lblDesign.Text = ddldesign.SelectedItem.Text
design = CInt(ddldesign.SelectedItem.Text)
totalcounter()
End If
End Sub
Protected Sub ddlpresentation_SelectedIndexChanged(sender As Object, e As EventArgs) Handles ddlpresentation.SelectedIndexChanged
If ddlpresentation.SelectedIndex <> 0 Then
lblpresentation.Text = ddlpresentation.SelectedItem.Text
presentation = CInt(ddlpresentation.SelectedItem.Text)
totalcounter()
End If
End Sub
Protected Sub ddlreport_SelectedIndexChanged(sender As Object, e As EventArgs) Handles ddlreport.SelectedIndexChanged
```

```
If ddlreport.SelectedIndex <> 0 Then
lblreport.Text = ddlreport.SelectedItem.Text
report = CInt(ddlreport.SelectedItem.Text)
totalcounter()
End If
End Sub
Sub totalcounter()
lblTotal.Text = ""
lblTotal.Text = plan + background + design + presentation + report
End Sub
Protected Sub ddlStudentNumber_SelectedIndexChanged(sender As Object, e As EventArgs) Handles ddlStudentNumber.SelectedIndexChanged
If ddlStudentNumber.SelectedIndex <> 0 Then
txtname.Text = ddlStudentNumber.SelectedItem.Text
Else
txtname.Text = ""
End If
End Sub
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles btnsubmit.Click
If ddlStudentNumber.SelectedIndex = 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Plz Select student Number!'
});", True)
ddlStudentNumber.BorderColor = Drawing.Color.Red
lblerror.Text = "Plz Select student Number!!!"
Return
Exit Sub
End If
If DDLPlan.SelectedIndex = 0 Or ddlpresentation.SelectedIndex = 0 Or DDLbackgroud.SelectedIndex = 0 Or ddldesign.SelectedIndex = 0 Or ddlreport.SelectedIndex = 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Plz Give All Grades!'
});", True)
lblerror.Text = "Plz Give All Grades!!!"
Return
Exit Sub
End If
Session("StudentNumber") = ddlStudentNumber.SelectedValue
Checkifgraded()
If graded <> 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
```

```
icon: 'error',
title: 'Oops...',
text: 'Sorry! You've already graded this student!'
});", True)
lblerror.Text = "Sorry! You've already graded this student..."
Return
Exit Sub
End If
Dim totalgrade As Integer = CInt(DDLPlan.SelectedItem.Text) + CInt(ddldesign.SelectedItem.Text) + CInt(DDLbackgroud.SelectedItem.Text) + CInt(ddlpresentation.SelectedItem.Text) + CInt(ddlreport.SelectedItem.Text)
connection()
cmd.Connection = cn
cmd.CommandText = " BEGIN TRANSACTION " &
" insert into Grading(StudentNO,StudentName,ProjectPlan,RelatedWork,Designandsolution,Presentation,Report,Grader,Total,Feedback,GradedDate)
values(@SNO,@Name,@Plan,@relatedwork,@design,@presentation,@report,@grader,@total,@Feedback,@gdate) " &
" COMMIT"
cmd.Parameters.AddWithValue("@SNO", ddlStudentNumber.SelectedValue)
cmd.Parameters.AddWithValue("@Feedback", txtFeedback.Text)
cmd.Parameters.AddWithValue("@Name", txtname.Text)
cmd.Parameters.AddWithValue("@Plan", CInt(lblplan.Text))
cmd.Parameters.AddWithValue("@relatedwork", CInt(lblbackground.Text))
cmd.Parameters.AddWithValue("@design", CInt(lblDesign.Text))
cmd.Parameters.AddWithValue("@presentation", CInt(lblpresentation.Text))
cmd.Parameters.AddWithValue("@report", CInt(lblreport.Text))
cmd.Parameters.AddWithValue("@grader", Application("LoggedID"))
cmd.Parameters.AddWithValue("@total", totalgrade)
cmd.Parameters.AddWithValue("@gdate", System.DateTime.Now)
Try
cn.Open()
cmd.ExecuteNonQuery()
cn.Close()
lblerror.Text = "Graded Successfully.."
Logfile(Application("LoggedID"), "Grade entered")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'Grading has been saved', showConfirmButton: false, timer: 1500});", True)
BindGrid()
clear()
Catch generatedExceptionName As Exception
MsgBox(generatedExceptionName.Message)
End Try
cmd.Parameters.Clear()
```

```
End Sub
"check if the student is already graded or not
Dim graded As Integer = 0
Sub Checkifgraded()
connection()
cmd.Connection = cn
cmd.CommandText = "Select * From grading Where Grader =@ID and StudentNO=@SNO "
cmd.Parameters.AddWithValue("@ID", Application("LoggedID"))
cmd.Parameters.AddWithValue("@SNO", Session("StudentNumber"))
Try
cn.Open()
DRead = cmd.ExecuteReader
If DRead.Read Then
graded = 1
End If
DRead.Close()
cn.Close()
Catch generatedExceptionName As Exception
MsgBox(generatedExceptionName.Message)
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub GVGrades_SelectedIndexChanged(sender As Object, e As EventArgs) Handles GVGrades.SelectedIndexChanged
Application("GradeID") = GVGrades.SelectedRow.Cells(1).Text.ToString
Session("StudentNumber") = GVGrades.SelectedRow.Cells(2).Text.ToString
txtname.Text = GVGrades.SelectedRow.Cells(3).Text.ToString
lblplan.Text = GVGrades.SelectedRow.Cells(4).Text.ToString
lblbackground.Text = GVGrades.SelectedRow.Cells(5).Text.ToString
lblDesign.Text = GVGrades.SelectedRow.Cells(6).Text.ToString
lblpresentation.Text = GVGrades.SelectedRow.Cells(7).Text.ToString
lblreport.Text = GVGrades.SelectedRow.Cells(8).Text.ToString
lblTotal.Text = GVGrades.SelectedRow.Cells(9).Text.ToString
txtFeedback.Text = GVGrades.SelectedRow.Cells(10).Text.ToString
btnsubmit.Enabled = False
btndelete.Enabled = True
btnupdate.Enabled = True
End Sub
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles btnupdate.Click
Checkifgraded()
If graded = 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Sorry! You haven't  graded this student!'
```

```
});", True)
lblerror.Text = "Sorry! You haven't  graded this student..."
Return
Exit Sub
End If
Dim totalgrade As Integer = CInt(lblplan.Text) + CInt(lblDesign.Text) + CInt(lblbackground.Text) + CInt(lblpresentation.Text) + CInt(lblreport.Text)
connection()
cmd.Connection = cn
cmd.CommandText = " BEGIN TRANSACTION " &
" Update Grading set StudentNO=@SNO,StudentName=@Name,ProjectPlan=@Plan,RelatedWork=@relatedwork,Designandsolution=@design,Presentation=@presentation,Report=@report,Grader=@grader,Total=@total,Feedback=@Feedback,GradedDate=@gdate  " &
" where grader=@grader and studentNO=@SNO" &
" COMMIT"
cmd.Parameters.AddWithValue("@SNO", ddlStudentNumber.SelectedValue)
cmd.Parameters.AddWithValue("@Feedback", txtFeedback.Text)
cmd.Parameters.AddWithValue("@Name", txtname.Text)
cmd.Parameters.AddWithValue("@Plan", CInt(lblplan.Text))
cmd.Parameters.AddWithValue("@relatedwork", CInt(lblbackground.Text))
cmd.Parameters.AddWithValue("@design", CInt(lblDesign.Text))
cmd.Parameters.AddWithValue("@presentation", CInt(lblpresentation.Text))
cmd.Parameters.AddWithValue("@report", CInt(lblreport.Text))
cmd.Parameters.AddWithValue("@grader", Application("LoggedID"))
cmd.Parameters.AddWithValue("@total", totalgrade)
cmd.Parameters.AddWithValue("@gdate", System.DateTime.Now)
Try
cn.Open()
cmd.ExecuteNonQuery()
cn.Close()
lblerror.Text = "Updated Successfully.."
Logfile(Application("LoggedID"), "Grade Updated")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Grade Updated Successfully.'
});", True)
BindGrid()
clear()
Catch generatedExceptionName As Exception
MsgBox(generatedExceptionName.Message)
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub btndelete_Click(sender As Object, e As EventArgs) Handles btndelete.Click
Checkifgraded()
```

```
If graded = 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Sorry! You haven't  graded this student!'
});", True)
lblerror.Text = "Sorry! You haven't  graded this student..."
Return
Exit Sub
End If
connection()
cmd.Connection = cn
cmd.CommandText = " BEGIN TRANSACTION " &
" delete grading where studentNO=@stno and grader=@gr" &
" COMMIT"
cmd.Parameters.AddWithValue("@stno", ddlStudentNumber.SelectedValue)
cmd.Parameters.AddWithValue("@gr", Application("LoggedID"))
Try
cn.Open()
cmd.ExecuteNonQuery()
cn.Close()
lblerror.Text = "Deleted Successfully.."
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Grade Deleted Successfully.'
});", True)
Logfile(Application("LoggedID"), "Grade deleted")
BindGrid()
clear()
Catch generatedExceptionName As Exception
MsgBox(generatedExceptionName.Message)
End Try
cmd.Parameters.Clear()
End Sub
Sub clear()
btnsubmit.Enabled = True
btnupdate.Enabled = False
btndelete.Enabled = False
txtFeedback.Text = ""
txtname.Text = ""
lblplan.Text = 0
lblbackground.Text = 0
lblDesign.Text = 0
lblerror.Text = ""
lblpresentation.Text = 0
lblreport.Text = 0
```

```
lblTotal.Text = 0
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Examiner grading code

```
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class ExaminerGrading
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public userID As String = ""
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Dim plan As Integer
Dim background As Integer
Dim design As Integer
Dim presentation As Integer
Dim report As Integer
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not IsPostBack Then
getdepartment()
BindDropDownStudent()
BindGrid()
plan = 0
background = 0
design = 0
presentation = 0
report = 0
End If
End Sub
Dim DepartmentID As String = ""
"Get department
Sub getdepartment()
connection()
cmd.Connection = cn
cmd.CommandText = "select distinct(Examiner.Department)  from Examiner " &
"  where Examiner.ID=@ID"
cmd.Parameters.AddWithValue("@ID", Application("LoggedID"))
Try
```

```
cn.Open()
DRead = cmd.ExecuteReader
If DRead.Read Then
DepartmentID = DRead(0)
End If
DRead.Close()
cn.Close()
Catch generatedExceptionName As Exception
MsgBox(generatedExceptionName.Message)
End Try
cmd.Parameters.Clear()
End Sub
"bind gridview
Private Sub BindGrid()
Using con As New SqlConnection(cnstring)
Using cmd As New SqlCommand("Select Id,StudentNO,StudentName,ProjectPlan,RelatedWork,Designandsolution,Presentation,Report,Total,Feedback from Grading where grader='" & Application("LoggedID") & "' order by GradedDate desc")
Using sda As New SqlDataAdapter()
cmd.Connection = con
sda.SelectCommand = cmd
Using dt As New DataTable()
sda.Fill(dt)
GVGrades.DataSource = dt
GVGrades.DataBind()
End Using
End Using
End Using
End Using
End Sub
Sub BindDropDownStudent()
Try
Using sqlConn As New SqlConnection(cnstring)
Using sqlCmd As New SqlCommand()
sqlCmd.CommandText = "select Student.StudentNO,Student.Name from student " &
" inner join AssignedExaminer " &
" On AssignedExaminer.StudentNO=Student.StudentNO " &
" where AssignedExaminer.ExaminerID=@DP "
sqlCmd.Parameters.AddWithValue("@DP", DepartmentID)
sqlCmd.Parameters.AddWithValue("@AID", Application("LoggedID"))
sqlCmd.Connection = sqlConn
sqlConn.Open()
Dim da As New SqlDataAdapter(sqlCmd)
Dim dt As New DataTable()
da.Fill(dt)
ddlStudentNumber.DataSource = dt
```

```
ddlStudentNumber.DataValueField = "StudentNO"
ddlStudentNumber.DataTextField = "Name"
ddlStudentNumber.DataBind()
sqlConn.Close()
'Adding "Please select" option in dropdownlist for validation
ddlStudentNumber.Items.Insert(0, New ListItem("Please select", "0"))
End Using
End Using
Catch
End Try
End Sub
Protected Sub DDLPlan_SelectedIndexChanged(sender As Object, e As EventArgs) Handles DDLPlan.SelectedIndexChanged
If DDLPlan.SelectedIndex <> 0 Then
lblplan.Text = DDLPlan.SelectedItem.Text
plan = CInt(DDLPlan.SelectedItem.Text)
totalcounter()
End If
End Sub
Protected Sub DDLbackgroud_SelectedIndexChanged(sender As Object, e As EventArgs) Handles DDLbackgroud.SelectedIndexChanged
If DDLbackgroud.SelectedIndex <> 0 Then
lblbackground.Text = DDLbackgroud.SelectedItem.Text
background = CInt(DDLbackgroud.SelectedItem.Text)
totalcounter()
End If
End Sub
Protected Sub ddldesign_SelectedIndexChanged(sender As Object, e As EventArgs) Handles ddldesign.SelectedIndexChanged
If ddldesign.SelectedIndex <> 0 Then
lblDesign.Text = ddldesign.SelectedItem.Text
design = CInt(ddldesign.SelectedItem.Text)
totalcounter()
End If
End Sub
Protected Sub ddlpresentation_SelectedIndexChanged(sender As Object, e As EventArgs) Handles ddlpresentation.SelectedIndexChanged
If ddlpresentation.SelectedIndex <> 0 Then
lblpresentation.Text = ddlpresentation.SelectedItem.Text
presentation = CInt(ddlpresentation.SelectedItem.Text)
totalcounter()
End If
End Sub
Protected Sub ddlreport_SelectedIndexChanged(sender As Object, e As EventArgs) Handles ddlreport.SelectedIndexChanged
If ddlreport.SelectedIndex <> 0 Then
lblreport.Text = ddlreport.SelectedItem.Text
```

```
report = CInt(ddlreport.SelectedItem.Text)
totalcounter()
End If
End Sub
Sub totalcounter()
lblTotal.Text = ""
lblTotal.Text = plan + background + design + presentation + report
End Sub
Protected Sub ddlStudentNumber_SelectedIndexChanged(sender As Object, e As EventArgs) Handles ddlStudentNumber.SelectedIndexChanged
If ddlStudentNumber.SelectedIndex <> 0 Then
txtname.Text = ddlStudentNumber.SelectedItem.Text
Else
txtname.Text = ""
End If
End Sub
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles btnsubmit.Click
If ddlStudentNumber.SelectedIndex = 0 Then
ddlStudentNumber.BorderColor = Drawing.Color.Red
lblerror.Text = "Plz Select student Number!!!"
Return
Exit Sub
End If
If DDLPlan.SelectedIndex = 0 Or ddlpresentation.SelectedIndex = 0 Or DDLbackgroud.SelectedIndex = 0 Or ddldesign.SelectedIndex = 0 Or ddlreport.SelectedIndex = 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Plz Give All Grades!'
});", True)
lblerror.Text = "Plz Give All Grades!!!"
Return
Exit Sub
End If
Session("StudentNumber") = ddlStudentNumber.SelectedValue
Checkifgraded()
If graded <> 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Sorry! You've already graded this student!'
});", True)
lblerror.Text = "Sorry! You've already graded this student..."
Return
Exit Sub
```

```
End If
Dim totalgrade As Integer = CInt(DDLPlan.SelectedItem.Text) + CInt(ddldesign.SelectedItem.Text) + CInt(DDLbackgroud.SelectedItem.Text) + CInt(ddlpresentation.SelectedItem.Text) + CInt(ddlreport.SelectedItem.Text)
connection()
cmd.Connection = cn
cmd.CommandText = " BEGIN TRANSACTION " &
" insert into Grading(StudentNO,StudentName,ProjectPlan,RelatedWork,Designandsolution,Presentation,Report,Grader,Total,Feedback,GradedDate)
values(@SNO,@Name,@Plan,@relatedwork,@design,@presentation,@report,@grader,@total,@Feedback,@gdate) " &
" COMMIT"
cmd.Parameters.AddWithValue("@SNO", ddlStudentNumber.SelectedValue)
cmd.Parameters.AddWithValue("@Name", txtname.Text)
cmd.Parameters.AddWithValue("@Feedback", txtFeedback.Text)
cmd.Parameters.AddWithValue("@Plan", CInt(lblplan.Text))
cmd.Parameters.AddWithValue("@relatedwork", CInt(lblbackground.Text))
cmd.Parameters.AddWithValue("@design", CInt(lblDesign.Text))
cmd.Parameters.AddWithValue("@presentation", CInt(lblpresentation.Text))
cmd.Parameters.AddWithValue("@report", CInt(lblreport.Text))
cmd.Parameters.AddWithValue("@grader", Application("LoggedID"))
cmd.Parameters.AddWithValue("@total", totalgrade)
cmd.Parameters.AddWithValue("@gdate", System.DateTime.Now)
Try
cn.Open()
cmd.ExecuteNonQuery()
cn.Close()
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({position: 'top-end', icon: 'success', title: 'Graded Successfully.', showConfirmButton: false, timer: 1500});", True)
lblerror.Text = "Graded Successfully.."
Logfile(Application("LoggedID"), "Inserted Grade")
BindGrid()
clear()
Catch generatedExceptionName As Exception
MsgBox(generatedExceptionName.Message)
End Try
cmd.Parameters.Clear()
End Sub
"check if the student is already graded or not
Dim graded As Integer = 0
Sub Checkifgraded()
connection()
cmd.Connection = cn
cmd.CommandText = "Select * From grading Where Grader =@ID and StudentNO=@SNO "
```

```
cmd.Parameters.AddWithValue("@ID", Application("LoggedID"))
cmd.Parameters.AddWithValue("@SNO", Session("StudentNumber"))
Try
cn.Open()
DRead = cmd.ExecuteReader
If DRead.Read Then
graded = 1
End If
DRead.Close()
cn.Close()
Catch generatedExceptionName As Exception
MsgBox(generatedExceptionName.Message)
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub GVGrades_SelectedIndexChanged(sender As Object, e As EventArgs) Handles GVGrades.SelectedIndexChanged
Application("GradeID") = GVGrades.SelectedRow.Cells(1).Text.ToString
Session("StudentNumber") = GVGrades.SelectedRow.Cells(2).Text.ToString
txtname.Text = GVGrades.SelectedRow.Cells(3).Text.ToString
lblplan.Text = GVGrades.SelectedRow.Cells(4).Text.ToString
lblbackground.Text = GVGrades.SelectedRow.Cells(5).Text.ToString
lblDesign.Text = GVGrades.SelectedRow.Cells(6).Text.ToString
lblpresentation.Text = GVGrades.SelectedRow.Cells(7).Text.ToString
lblreport.Text = GVGrades.SelectedRow.Cells(8).Text.ToString
lblTotal.Text = GVGrades.SelectedRow.Cells(9).Text.ToString
txtFeedback.Text = GVGrades.SelectedRow.Cells(10).Text.ToString
btnsubmit.Enabled = False
btndelete.Enabled = True
btnupdate.Enabled = True
End Sub
Protected Sub btnupdate_Click(sender As Object, e As EventArgs) Handles btnupdate.Click
Checkifgraded()
If graded = 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Sorry! You haven't  graded this student!'
});", True)
lblerror.Text = "Sorry! You haven't  graded this student..."
Return
Exit Sub
End If
```

```
Dim totalgrade As Integer = CInt(lblplan.Text) + CInt(lblDesign.Text) + CInt(lblbackground.Text) + CInt(lblpresentation.Text) + CInt(lblreport.Text)
connection()
cmd.Connection = cn
cmd.CommandText = " BEGIN TRANSACTION " &
" Update Grading set StudentNO=@SNO,StudentName=@Name,ProjectPlan=@Plan,RelatedWork=@relatedwork,Designandsolution=@design,Presentation=@presentation,Report=@report,Grader=@grader,Total=@total,Feedback=@Feedback,GradedDate=@gdate  " &
" where grader=@grader and studentNO=@SNO" &
" COMMIT"
cmd.Parameters.AddWithValue("@SNO", ddlStudentNumber.SelectedValue)
cmd.Parameters.AddWithValue("@Name", txtname.Text)
cmd.Parameters.AddWithValue("@Feedback", txtFeedback.Text)
cmd.Parameters.AddWithValue("@Plan", CInt(lblplan.Text))
cmd.Parameters.AddWithValue("@relatedwork", CInt(lblbackground.Text))
cmd.Parameters.AddWithValue("@design", CInt(lblDesign.Text))
cmd.Parameters.AddWithValue("@presentation", CInt(lblpresentation.Text))
cmd.Parameters.AddWithValue("@report", CInt(lblreport.Text))
cmd.Parameters.AddWithValue("@grader", Application("LoggedID"))
cmd.Parameters.AddWithValue("@total", totalgrade)
cmd.Parameters.AddWithValue("@gdate", System.DateTime.Now)
Try
cn.Open()
cmd.ExecuteNonQuery()
cn.Close()
lblerror.Text = "Updated Successfully.."
Logfile(Application("LoggedID"), "Updated Grade")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Updated...',
text: 'Grade Updated Successfully.'
});", True)
BindGrid()
clear()
Catch generatedExceptionName As Exception
MsgBox(generatedExceptionName.Message)
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub btndelete_Click(sender As Object, e As EventArgs) Handles btndelete.Click
Checkifgraded()
If graded = 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Sorry! You haven't  graded this student!'
```

```
});", True)
lblerror.Text = "Sorry! You haven't  graded this student..."
Return
Exit Sub
End If
connection()
cmd.Connection = cn
cmd.CommandText = " BEGIN TRANSACTION " &
" delete grading where studentNO=@stno and grader=@gr" &
" COMMIT"
cmd.Parameters.AddWithValue("@stno", ddlStudentNumber.SelectedValue)
cmd.Parameters.AddWithValue("@gr", Application("LoggedID"))
Try
cn.Open()
cmd.ExecuteNonQuery()
cn.Close()
lblerror.Text = "Deleted Successfully.."
Logfile(Application("LoggedID"), "Deleted Grade")
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Deleted...',
text: 'Student Grade Deleted Successfully.'
});", True)
BindGrid()
clear()
Catch generatedExceptionName As Exception
MsgBox(generatedExceptionName.Message)
End Try
cmd.Parameters.Clear()
End Sub
Sub clear()
btnsubmit.Enabled = True
btnupdate.Enabled = False
btndelete.Enabled = False
txtFeedback.Text = ""
txtname.Text = ""
lblplan.Text = 0
lblbackground.Text = 0
lblDesign.Text = 0
lblerror.Text = ""
lblpresentation.Text = 0
lblreport.Text = 0
lblTotal.Text = 0
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
```

```
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

View my grades code

```
Imports System.Collections.Generic
Imports System.Linq
Imports System.Web
Imports System.Web.UI
Imports System.Web.UI.WebControls
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Imports System.IO
Public Class ViewMyGrades
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim dr As SqlDataReader
Dim strings As New ConnectionClass
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not IsPostBack Then
getstudentDetails()
getDepartment()
getThesisTitle()
getgrades()
gettotal()
Summary()
End If
End Sub
Dim departmenID As String = ""
Sub getstudentDetails()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select StudentNO,Name,Nationality,Gender,Email,AcademicYear,Level,Department from Student where StudentNO=@SNO"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@SNO", Application("LoggedID"))
```

```
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
lblStudentNO.Text = DRead(0).ToString
lblStudentName.Text = DRead(1).ToString
lblNationality.Text = DRead(2).ToString
lblGender.Text = DRead(3).ToString
lblEmail.Text = DRead(4).ToString
lblAcademicYear.Text = DRead(5).ToString
lbllevel.Text = DRead(6).ToString
departmenID = DRead(7)
lblDate.Text = System.DateTime.Now.ToShortDateString
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
Sub getDepartment()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select Department.name,Faculty.name from Department inner join Faculty on Faculty.ID=Department.ID where Department.ID='" & departmenID & "'"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
lblDepartment.Text = DRead(0)
lblFaculty.Text = DRead(1)
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
```

```
Sub getThesisTitle()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select Title  from ProposedTitle  where StudentNO='" & Application("LoggedID") & "' and status='Accepted'"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
lblTitle.Text = DRead(0)
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
Sub getgrades()
Using con As New SqlConnection(cnstring)
Using cmd As New SqlCommand("Select StudentNO,StudentName,ProjectPlan,RelatedWork,Designandsolution,Presentation,Report,Total,Feedback from Grading where StudentNO='" & Application("LoggedID") & "' order by GradedDate desc")
Using sda As New SqlDataAdapter()
cmd.Connection = con
sda.SelectCommand = cmd
Using dt As New DataTable()
sda.Fill(dt)
GVGrades.DataSource = dt
GVGrades.DataBind()
End Using
End Using
End Using
End Using
End Sub
Dim Total As String = ""
Sub gettotal()
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "select  Sum(Total) from Grading WHERE STUDENTNO ='" & Application("LoggedID") & "'"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
```

```
Try
conn.Open()
DRead = cmd.ExecuteReader()
If DRead.HasRows Then
While DRead.Read
Total = DRead(0)
lblTotal.Text = DRead(0)
lblPercentage.Text = DRead(0)
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'success',
title: 'Your Total Grade is:',
text: '" & DRead(0).ToString & "'
});", True)
End While
End If
conn.Close()
Catch ex As Exception
conn.Close()
End Try
cmd.Parameters.Clear()
End Using
End Using
End Sub
Sub Summary()
Select Case Total
Case 0 To 49
lblGPA.Text = 0
lblLetter.Text = "F"
LblStatus.Text = "Fail"
lblstatusH.Text = "Fail"
Case 50 To 54
lblGPA.Text = 1.67
lblLetter.Text = "C"
LblStatus.Text = "Weak"
lblstatusH.Text = "Weak"
Case 55 To 59
lblGPA.Text = 2
lblLetter.Text = "C+"
LblStatus.Text = "Waek"
lblstatusH.Text = "Weak"
Case 60 To 64
lblGPA.Text = 2.33
lblLetter.Text = "B-"
LblStatus.Text = "Weak"
lblstatusH.Text = "Weak"
Case 65 To 69
lblGPA.Text = 2.67
```

```
lblLetter.Text = "B+"
LblStatus.Text = "Good"
lblstatusH.Text = "Good"
Case 70 To 74
lblGPA.Text = 3
lblLetter.Text = "B "
LblStatus.Text = "very good"
lblstatusH.Text = "very good"
Case 75 To 79
lblGPA.Text = 3.33
lblLetter.Text = "A-"
LblStatus.Text = "Great"
lblstatusH.Text = "Great"
Case 80 To 89
lblGPA.Text = 3.67
lblLetter.Text = "A"
LblStatus.Text = "Excellent"
lblstatusH.Text = "Excellent"
Case 90 To 100
lblGPA.Text = 4
lblLetter.Text = "A+"
LblStatus.Text = "Excellent With Honor"
lblstatusH.Text = "Excellent With Honor"
Case Else
lblGPA.Text = 0
lblLetter.Text = ""
LblStatus.Text = ""
lblstatusH.Text = ""
End Select
End Sub
End Class
```

Compose mail code

```
Imports System.IO
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class composemail
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_PreInit(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.PreInit
If Application("role").Equals("Admin") Then
Me.MasterPageFile = "~/AdminMaster.Master"
ElseIf Application("role").Equals("Student") Then
Me.MasterPageFile = "~/StudentMaster.Master"
ElseIf Application("role").Equals("Supervisor") Then
Me.MasterPageFile = "~/SupervisorMaster.Master"
ElseIf Application("role").Equals("Advisor") Then
Me.MasterPageFile = "~/AdviserMaster.Master"
ElseIf Application("role").Equals("Examiner") Then
Me.MasterPageFile = "~/AdviserMaster.Master"
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not Me.IsPostBack Then
lblusernamebody.Text = Application("username")
lblloggeddatebody.Text = Application("logindate")
End If
End Sub
Protected Sub btnsubmit_Click(sender As Object, e As EventArgs) Handles btnsubmit.Click
If txtTO.Text <> "" OrElse txtSubject.Text <> "" OrElse txtmessage.Text <> "" Then
Checkrecipient()
If chechreciever = 0 Then
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
```

```
icon: 'error',
title: 'Oops...',
text: 'The Reciever Email is not Exist!'
});", True)
lblerror.Text = "The Reciever Email is not Exist"
lblerror.ForeColor = Drawing.Color.Red
Else
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
" insert into sentmail(recipient,from_address,subject,body,maildate)
values(@Rec,@from,@subject,@body,@maildate) " &
" insert into Mail(recipient,from_address,subject,body,maildate)
values(@MRec,@Mfrom,@Msubject,@Mbody,@Mmaildate) " &
" COMMIT"
cmd.Parameters.AddWithValue("@Rec", txtTO.Text)
cmd.Parameters.AddWithValue("@from", Application("LoggedID"))
cmd.Parameters.AddWithValue("@subject", txtSubject.Text)
cmd.Parameters.AddWithValue("@body", txtmessage.Text)
cmd.Parameters.AddWithValue("@maildate", System.DateTime.Now)
cmd.Parameters.AddWithValue("@MRec", txtTO.Text)
cmd.Parameters.AddWithValue("@Mfrom", Application("LoggedID"))
cmd.Parameters.AddWithValue("@Msubject", txtSubject.Text)
cmd.Parameters.AddWithValue("@Mbody", txtmessage.Text)
cmd.Parameters.AddWithValue("@Mmaildate", System.DateTime.Now)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
cn.Close()
Logfile(Application("LoggedID"), "Sent a Message!")
Catch ex As Exception
cn.Close()
End Try
cmd.Parameters.Clear()
clear()
End If
Else
ScriptManager.RegisterStartupScript(Me, [GetType](), "alert", "Swal.fire({
icon: 'error',
title: 'Oops...',
text: 'Plz fill the blanks..!'
});", True)
lblerror.Text = "Plz fill the blanks.."
lblerror.ForeColor = Drawing.Color.Red
End If
```

```
End Sub
Dim chechreciever As Integer = 0
"check the reciever user
Sub Checkrecipient()
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
" select * from users " &
" where ID=@RID" &
" COMMIT"
cmd.Parameters.AddWithValue("@RID", txtTO.Text)
Try
connection()
cn.Open()
chechreciever = CInt(cmd.ExecuteScalar())
cn.Close()
Catch ex As Exception
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
"clear function
Sub clear()
txtmessage.Text = ""
txtSubject.Text = ""
txtTO.Text = ""
lblerror.Text = ""
txtTO.Focus()
End Sub
Protected Sub btnreset_Click(sender As Object, e As EventArgs) Handles btnreset.Click
clear()
End Sub
Sub Logfile(user As String, descripttion As String)
Using conn As SqlConnection = New SqlConnection(cnstring)
Dim sql As String = "Insert into logfile values (@user, @description, @logdate)"
Using cmd As SqlCommand = New SqlCommand(sql, conn)
cmd.Parameters.AddWithValue("@user", user)
cmd.Parameters.AddWithValue("@description", descripttion)
cmd.Parameters.AddWithValue("@logdate", Date.Now.ToShortDateString)
Try
conn.Open()
cmd.ExecuteNonQuery()
conn.Close()
Catch ex As Exception
conn.Close()
End Try
```

```
cmd.Parameters.Clear()
End Using
End Using
End Sub
End Class
```

Mailbox code

```
Imports System.IO
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class Mailbox
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_PreInit(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.PreInit
If Application("role").Equals("Admin") Then
Me.MasterPageFile = "~/AdminMaster.Master"
ElseIf Application("role").Equals("Student") Then
Me.MasterPageFile = "~/StudentMaster.Master"
ElseIf Application("role").Equals("Supervisor") Then
Me.MasterPageFile = "~/SupervisorMaster.Master"
ElseIf Application("role").Equals("Advisor") Then
Me.MasterPageFile = "~/AdviserMaster.Master"
ElseIf Application("role").Equals("Examiner") Then
Me.MasterPageFile = "~/AdviserMaster.Master"
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not Me.IsPostBack Then
lblloggeddatebody.Text = Application("logindate")
lblusernamebody.Text = Application("username")
BindGView()
End If
End Sub
"bind to gridview
Sub BindGView()
Using conn As SqlConnection = New SqlConnection(cnstring)
```

```
Using sda As SqlDataAdapter = New SqlDataAdapter("SELECT MessageID,from_address,subject,maildate from Mail where recipient ='" & Application("LoggedID") & "' order by maildate desc ", conn)
Dim dt As DataTable = New DataTable()
sda.Fill(dt)
GVInbox.DataSource = dt
GVInbox.DataBind()
End Using
End Using
End Sub
Protected Sub btnsearch_Click(sender As Object, e As EventArgs) Handles btnsearch.Click
If txtsearchmail.Text <> "" Then
Using conn As SqlConnection = New SqlConnection(cnstring)
Using sda As SqlDataAdapter = New SqlDataAdapter("SELECT MessageID,from_address,subject,maildate from Mail where recipient='" & Application("LoggedID") & "' and from_address='" & txtsearchmail.Text & "' order by maildate desc ", conn)
Dim dt As DataTable = New DataTable()
sda.Fill(dt)
GVInbox.DataSource = dt
GVInbox.DataBind()
End Using
End Using
txtsearchmail.Text = ""
End If
End Sub
Protected Sub chckall_CheckedChanged(sender As Object, e As EventArgs) Handles chckall.CheckedChanged
If chckall.Checked = True Then
For Each row As GridViewRow In GVInbox.Rows
If row.RowType = DataControlRowType.DataRow Then
Dim chkRow As CheckBox = TryCast(row.Cells(0).FindControl("chkRow"), CheckBox)
If chkRow.Checked = False Then
chkRow.Checked = True
End If
End If
Next
End If
End Sub
Protected Sub btnrefresh_Click(sender As Object, e As EventArgs) Handles btnrefresh.Click
BindGView()
End Sub
Protected Sub btnreferesh2_Click(sender As Object, e As EventArgs) Handles btnreferesh2.Click
BindGView()
```

```
End Sub
Protected Sub btndeletemail_Click(sender As Object, e As EventArgs) Handles btndeletemail.Click
For Each row As GridViewRow In GVInbox.Rows
If row.RowType = DataControlRowType.DataRow Then
Dim chkRow As CheckBox = TryCast(row.Cells(0).FindControl("chkRow"), CheckBox)
If chkRow.Checked Then
Dim MsgID As String = row.Cells(1).Text
delete(MsgID)
End If
End If
Next
BindGView()
End Sub
Sub delete(ID As String)
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
" Delete Mail " &
" where MessageID= @ID " &
" COMMIT"
cmd.Parameters.AddWithValue("@ID", ID)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
cn.Close()
Catch ex As Exception
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub btndelete2_Click(sender As Object, e As EventArgs) Handles btndelete2.Click
For Each row As GridViewRow In GVInbox.Rows
If row.RowType = DataControlRowType.DataRow Then
Dim chkRow As CheckBox = TryCast(row.Cells(0).FindControl("chkRow"), CheckBox)
If chkRow.Checked Then
Dim MsgID As String = row.Cells(1).Text
delete(MsgID)
End If
End If
Next
BindGView()
End Sub
```

```
Protected Sub Send(ByVal sender As Object, ByVal e As EventArgs)
'Reference the Button.
Dim btnread As Button = CType(sender, Button)
'Reference the GridView Row.
Dim row As GridViewRow = CType(btnread.NamingContainer, GridViewRow)
'Save the GridView Row in Session.
Application("mailRow") = row
'Redirect to other Page.
Response.Redirect("ReadMail.aspx")
End Sub
End Class
```

Sent mail code

```
Imports System.IO
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration
Public Class sentmail
Inherits System.Web.UI.Page
Dim cmd As New SqlCommand()
Dim DRead As SqlDataReader
Dim DataSet As New DataSet
Dim DataAdapter As New SqlDataAdapter
Public cnstring As String = "Server=DESKTOP-77ANH2V; database=GPREPOSITORY;integrated security=true"
Dim cn As New SqlConnection(cnstring)
Sub connection()
If ConnectionState.Open Then
cn.Close()
End If
End Sub
Protected Sub Page_PreInit(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.PreInit
If Application("role").Equals("Admin") Then
Me.MasterPageFile = "~/AdminMaster.Master"
ElseIf Application("role").Equals("Student") Then
Me.MasterPageFile = "~/StudentMaster.Master"
ElseIf Application("role").Equals("Supervisor") Then
Me.MasterPageFile = "~/SupervisorMaster.Master"
ElseIf Application("role").Equals("Advisor") Then
Me.MasterPageFile = "~/AdviserMaster.Master"
ElseIf Application("role").Equals("Examiner") Then
Me.MasterPageFile = "~/AdviserMaster.Master"
End If
End Sub
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
If Not Me.IsPostBack Then
lblloggeddatebody.Text = Application("logindate")
lblusernamebody.Text = Application("username")
BindGView()
End If
End Sub
"bind to gridview
Sub BindGView()
Using conn As SqlConnection = New SqlConnection(cnstring)
```

```
Using sda As SqlDataAdapter = New SqlDataAdapter("select MessageID,recipient,subject,maildate from SentMail where [from_address] ='" & Application("LoggedID") & "' order by maildate desc ", conn)
Dim dt As DataTable = New DataTable()
sda.Fill(dt)
GVsent.DataSource = dt
GVsent.Columns(3).ControlStyle.Height = 50
GVsent.DataBind()
End Using
End Using
End Sub
Protected Sub btnsearch_Click(sender As Object, e As EventArgs) Handles btnsearch.Click
If txtsearchmail.Text <> "" Then
Using conn As SqlConnection = New SqlConnection(cnstring)
Using sda As SqlDataAdapter = New SqlDataAdapter("SELECT MessageID,recipient,subject,maildate from SentMail where from_address='" & Application("LoggedID") & "' and recipient='" & txtsearchmail.Text & "' order by maildate desc ", conn)
Dim dt As DataTable = New DataTable()
sda.Fill(dt)
GVsent.DataSource = dt
GVsent.DataBind()
End Using
End Using
txtsearchmail.Text = ""
End If
End Sub
Protected Sub chckall_CheckedChanged(sender As Object, e As EventArgs) Handles chckall.CheckedChanged
If chckall.Checked = True Then
For Each row As GridViewRow In GVsent.Rows
If row.RowType = DataControlRowType.DataRow Then
Dim chkRow As CheckBox = TryCast(row.Cells(0).FindControl("chkRow"), CheckBox)
If chkRow.Checked = False Then
chkRow.Checked = True
End If
End If
Next
End If
End Sub
Protected Sub btnrefresh_Click(sender As Object, e As EventArgs) Handles btnrefresh.Click
BindGView()
End Sub
Protected Sub btnreferesh2_Click(sender As Object, e As EventArgs) Handles btnreferesh2.Click
```

```
BindGView()
End Sub
Protected Sub btndeletemail_Click(sender As Object, e As EventArgs) Handles btndeletemail.Click
For Each row As GridViewRow In GVsent.Rows
If row.RowType = DataControlRowType.DataRow Then
Dim chkRow As CheckBox = TryCast(row.Cells(0).FindControl("chkRow"), CheckBox)
If chkRow.Checked Then
Dim MsgID As String = row.Cells(1).Text
delete(MsgID)
End If
End If
Next
BindGView()
End Sub
Sub delete(ID As String)
cmd = New SqlCommand()
cmd.Connection = cn
cmd.CommandText =
" BEGIN TRANSACTION " &
" Delete SentMail " &
" where MessageID=@Id " &
" COMMIT"
cmd.Parameters.AddWithValue("@Id", ID)
Try
connection()
cn.Open()
cmd.ExecuteNonQuery()
cn.Close()
Catch ex As Exception
cn.Close()
End Try
cmd.Parameters.Clear()
End Sub
Protected Sub btndelete2_Click(sender As Object, e As EventArgs) Handles btndelete2.Click
For Each row As GridViewRow In GVsent.Rows
If row.RowType = DataControlRowType.DataRow Then
Dim chkRow As CheckBox = TryCast(row.Cells(0).FindControl("chkRow"), CheckBox)
If chkRow.Checked Then
Dim MsgID As String = row.Cells(1).Text
delete(MsgID)
End If
End If
Next
BindGView()
End Sub
```

```
Protected Sub Send(ByVal sender As Object, ByVal e As EventArgs)
'Reference the Button.
Dim btnread As Button = CType(sender, Button)
'Reference the GridView Row.
Dim row As GridViewRow = CType(btnread.NamingContainer, GridViewRow)
'Save the GridView Row in Session.
Application("mailsentRow") = row
'Redirect to other Page.
Response.Redirect("ReadSentMail.aspx")
End Sub
End Class
```