

**PREDICT STUDENT PERFORMANCE USING
DATA MINING
AND
MACHINE LEARNING TECHNIQUES**

**A THESIS SUBMITTED TO THE INSTITUTE
OF GRADUATE STUDIES**

OF

NEAR EAST UNIVERSITY

BY

ABDUL RHEMAN SIDDIQUI

**In Partial fulfillment of the requirements for the
Degree of
Master of Science in artificial intelligence**

NICOSIA, 2022

**ABDUL REHMAN
SIDDIQUI**

**PREDICT STUDENT
PERFORMANCE
USING DATA
MINING**

MASTER THESIS

2022

**PREDICT STUDENT PERFORMANCE USING
DATA MINING
AND
MACHINE LEARNING TECHNIQUES**

**A THESIS SUBMITTED TO THE INSTITUTE
OF GRADUATE STUDIES**

OF

NEAR EAST UNIVERSITY

BY


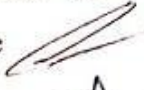

ABDUL RHEMAN SIDDIQUI

**In Partial fulfillment of the requirements for the
Degree of
Master of Science in artificial intelligence**

NICOSIA 2022


Approval

We certify that we have read the thesis submitted by Abdul Rehman Siddiqui titled "**Predict Student Performance Using Data Mining And Machine Learning Techniques**" and that in our combined opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Educational Sciences.


Examining Committee	Name-Surname	Signature
Head of the Committee:	Assist. Prof. Dr. Elbrus Imanov	
Committee Member*:	Assoc. Prof. Dr. Sertan Serte	
Supervisor:	Pro. Dr. Fadi Al-Turjman	

Approved by the Head of the Department

04...10/2022


Pro. Dr. Fadi Al-Turjman
Head of Department

Approved by the Institute of Graduate Studies


Prof. Dr. Kemal Hüsnü Can Başer
Head of the Institute



Declaration

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 materials and results that are not original to this work.

Name: abdul rehman

Surname: siddiqui

Signature:

Date:

Acknowledgment

I want to express my profound gratitude to my supervisor, Prof. Dr. Fadi AL-TURJMAN, for all of his support, advice, and understanding during my graduate studies at Near East University. His oversight was crucial in giving me a well-rounded experience and helping me to project my long-term career aspirations. He advised me to always follow up with calls, asking for regular updates on this work, and to be confident in all I do. Prof. Dr. Fadi AL-TURJMAN, you have my sincere gratitude for what you have done for me. Finally, I want to express my gratitude to my family for their unwavering support and prayers throughout the period I was away.

Abstract

Machine learning (ML) is the ability of a system to acquire and integrate knowledge based on large-scale observations, as well as to develop and extend itself by acquiring new knowledge rather than being programmed with it. In today's competitive world, an individual's academic qualifications are a must. Employers' selection criteria of potential applicants are frequently based on the grade point average (GPA). Planning is an essential step in attaining a decent GPA, regardless of a student's intellectual capacity. Many students, on the other hand, lack the essential skills and time to plan and manage their GPA. An automated education planner system would be extremely beneficial in assisting students in achieving the best CGPA possible depending on their existing skills. Despite the existence of course planning systems, students continue to fall short of their objectives. Educational data mining tools have recently been used to learn more about the educational environment and to help students perform better. Artificial Neural Networks, Support Vector Machines, Naive Bayesian, Decision trees, and other approaches are utilized in educational data mining. The goal of this thesis is to see how these algorithms can help university students plan and improve their academic performance. The suggested personalized web-based academic planner is designed to serve as an online record storage system for students' academic data. The system will provide the optimal path for undergraduates to attain their goals utilizing GA based on their present achievement.

KeyWords: gan, deepfakes, ml, ai, lstm, deeplearning

TABLE OF CONTENT

Declaration.....	i
Acknowledgment.....	ii
Dedication.....	iii
Abstract.....	iv
Table of Contents.....	v
List of Figures.....	vi
List of Tables.....	x
CHAPTER 1:INTRODUCTION.....	1
1.1 Background Of The Study	4
1.2 Statement of problem.	5
1.3 Purpose and Research Question.....	5
1.4 Approach and Methodology.....	5
Chapter 2: LITERATURE REVIEW.....	8
2.2 Related work.....	8
2.3 Student Information System.....	9
2.2.1 Smart Advisory system using Machine learning.....	11
2.3 Types of Machine Learning	13
2.3.1 Types of MI Algorithms	15
2.4 Web application.....	18
CHAPTER 3 :SYSTEM ARCHITECTURE.....	20
3.4 Linear Regression	26
3.5 Apriori	28
3.5 SVM (Support Vector Machine)	29
CHAPTER 4:USER INTERFACE.....	31
4.1 Course Selection.....	33
4.2 Automate Selection	34
4.3 Cgpa Calculator.....	35
4.4 Cgpa Predictor	36
CHAPTER 5:RESULT ANALYSIS.....	37
5.1 Test Environment and Test Plan.....	37

5.2 Browser testing.....	38
CHAPTER 6:CONCLUSION.....	40

LIST OF TABLES

<i>Table 2.1: App Comparisons</i>	18
<i>Table 5.1: Test case</i>	36
<i>Table 5.2: linear regression test case</i>	39
<i>Table 5.2: SVM test case</i>	239

LIST OF FIGURES

FIGURE 3.4: DATA GIVEN	26
FIGURE 3.5: FINAL DATA	26
FIGURE 3.6: LINEAR REGRESSION PLOT	27
FIGURE 3.7 : DATA FOR APRIORI	28
FIGURE 3.8: RULES	28
FIGURE 4.2 :DASHBOARD	31

CHAPTER 1

INTRODUCTION

1.1 Background of the Study

Machine learning is a collection of techniques that, in general, allow us to "train" machines how to accomplish tasks by demonstrating how they should be done[1]. We want to create a system that can determine a student's final grade point average based on the semester they are currently enrolled in. Using the prior data of students who have graduated, we may try to determine the final cgpa and use machine learning algorithms to train the model..

Machine learning is a subfield of computer science that differs from traditional computational approaches. In traditional computing, algorithms are collections of management instructions used by computers to compute or solve problems[3]. Machine learning, on the other hand, allows computers to learn from data inputs and then use statistical analysis to provide outputs that are within a specific range. Machine learning makes it easier for computers to develop models using data and automate procedures based on inputs as a consequence. Everyone who utilizes today's technology has profited from machine learning. Face recognition technology can be used by social media networks to assist users tag and share images of friends. Optical character (OCR) technology converts text images into movable type[4]. Recommender system based on machine learning offer recommendations based on user likes, which movies or TV series to watch next Consumers may soon be able to buy self-driving cars that navigate using machine learning. It's a field that's always evolving. As a result, whether dealing with machine learning methodology or examining the impact of machine learning procedures, there are a few factors to consider.

1.2 Statement of problem.:

Various student management systems make use of outdated technology. Some schools still use paper-based or manual to store student information.

It is possible to make these systems smarter and easier to use by utilizing machine learning and data mining algorithms that make use of existing data. These algorithms will make the work of student advisors easier by assisting students in selecting courses and providing results for their performance, scheduling classrooms, and assigning lecturers to courses.

As a result, the primary goal of this research is to broaden the scope of research by exploring machine learning and data mining and seeing all of the possibilities that these technologies provide.

1.3 Purpose and Research Question

Machine learning and data mining technique will be utilized to predict student success in this thesis. Also included is a method for assessing the student's performance and the quality of the final product. We address two important research questions in this regard:

- How will the student performance be calculated, given the previous data is available?
- How accurate will the prediction be?

1.4 Research Aim and Objectives

This thesis focuses on evaluating student performance and predicting course choices using machine learning approaches and algorithms. Within the scope of this thesis, two distinct challenges will be addressed. First, with access to earlier records, the machine learning system will classify and assess previous data of students and their cgpa each semester. In a second try, an algorithm will be optimized for the same goal without knowing the cgpa of the student.

The following is the procedure for this master's thesis:

An iterative knowledge discovery process will be begun to answer the specified research questions based on a study on the existing techniques and metrics. This procedure entails establishing quality criteria for predicting cgpa, as well as implementing any necessary changes. To solve the specified task optimally, measurements and algorithms, as well as the optimization of machine learning approaches, are used. It's worth noting that this is an iterative process, as well as their impact on translation quality and classification possibilities, will be found by testing the algorithms' results against a database of past data. The data set used will span the years 2015 through 2021. Furthermore, the approaches and algorithms employed will be continuously modified and adjusted during this iterative process to produce the best potential results. Finally, the method and outcomes will be analyzed, assessed, and compared with other machine learning algorithms (etc **svm linear regression**).

1.5 Proposal Organization

The dissertation is split into six chapters, plus references, conclusion, appendices. Chapter 1: Provides an overview of the research and its context. The study's Aim, research methods, etc) Chapter 2: Literature Survey: Include a review of the literature on the major subject of research. This chapter introduces the conceptual framework that will be used to comprehend the remainder of this thesis. Citation of related works and comparison Chapter 3: System Architecture, include a brief overview of research techniques and a description of the methodology used in this study Chapter 4: User Interface and Implementation, this gives an overview of how the application looks like and also talks about the UI. Chapter 5: Performance and Result Analysis Conclusion and Recommendation

CHAPTER 2

LITERATURE REVIEW

The literature review on the relevant studies is presented in this chapter. The chapter begins by reviewing related work about the AI in education, followed by Student Information System and the technology used in making this student management system more effective.

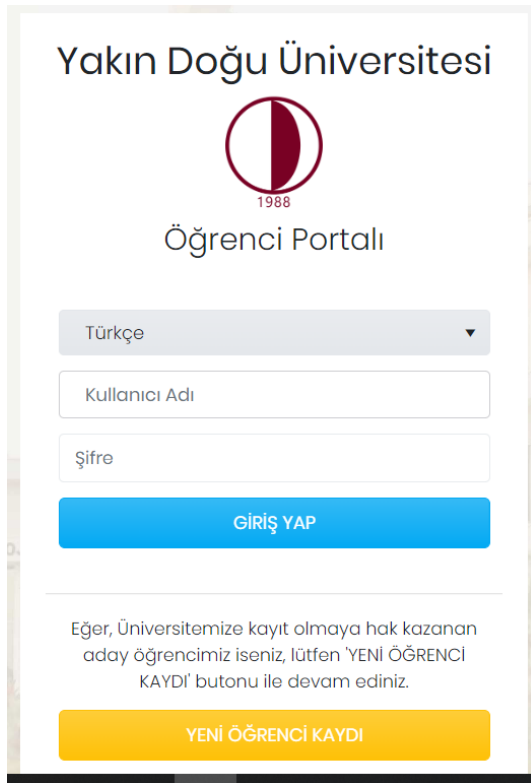
2.1 Related Work

In higher education, several studies have devoted to this project. of data mining. The researchers will provide an overview of a few sample studies in this section. A complete test case from the stage of higher education was presented by Abu Tair and El-Halees [6]. Their study's major goal is to demonstrate how valuable data mining can be in the education field. Using several educational data mining approaches, they identified a variety of information from the graduate student dataset, including classification, clustering, association rules, and outlier identification. On existing higher education students, [7]. The major purpose of their research is to forecast the number of new students who will enroll in the following based on the amount of pupils per year who have enrolled in prior years. This research aids decision-makers in managing the resources and personnel required to administer a student's results. This research also aids teachers in early detection.

Identify the pupils who require additional attention in order to facilitate taking the appropriate action at the appropriate moment to reduce academic failure and improve the student's academic performance. From the higher education level, Brijesh Kumar Bhardwaj and Saurabh Pal [8] used Bayesian classification using student database. This study aimed to identify students who required further attention in order to lower dropout rates and take action at the appropriate moment.

2.2 SIS: Student Information System

This is a web application that was developed to serve a similar purpose. This web application (student information systems **Figure 2.1**, assist universities and other educational institutions in moving student data online for greater management and transparency. That is the essence of it. This web program may gather information on the entire school online so that instructors, parents, students, and administrators can readily access it. This contains private student data such as names, grades, test results, attendance, performance reviews, and many other things. In essence, a SIS gives the school the ability to gather data points for many different areas in one location, making it simple to monitor progress and performance.



The image shows a screenshot of the login page for the Yakın Doğu Üniversitesi Öğrenci Portalı. The page features the university's logo, which is a red circle with a white crescent and star, and the year 1988 below it. The text 'Yakın Doğu Üniversitesi' is at the top, and 'Öğrenci Portalı' is below the logo. There is a language dropdown menu set to 'Türkçe'. Below that are input fields for 'Kullanıcı Adı' (Username) and 'Şifre' (Password). A blue button labeled 'GİRİŞ YAP' (Login) is positioned below the password field. At the bottom, there is a yellow button labeled 'YENİ ÖĞRENCİ KAYDI' (New Student Registration) and a line of text: 'Eğer, Üniversitemize kayıt olmaya hak kazanan aday öğrencimiz iseniz, lütfen 'YENİ ÖĞRENCİ KAYDI' butonu ile devam ediniz.'

Figure 2.2: Near east Student information system

2.3 Smart Advisory system using Machine learning

Many academics attempted to create advising systems to simplify the registration process for students. There are two distinct approaches for developing advising systems: one relies exclusively on the advisor, while the other depends on both the counselor and the student. The first one lacks interaction because the student must rely on the advisor's recommendations for which courses to register and then follow them. [9] is an illustration of this kind. The second paradigm, in contrast, is more engaging because the advisor bases his or her recommendations on the preferences the student expresses [10]. Compared to the previous type, this one is more prevalent and typically yields superior results. The authors of [11] created a Web-based advising system for undergraduate CS and CE students that takes these preferences as input and bases its recommendations on them. The advisors provide their recommendations once the students input their choices using a web browser. As a result, the process of advising is made easier, faster, and more trustworthy. (This system is not automated; as a result, in order to develop its recommendations, it requires input from the students, which limits the system's potential.) The authors of [12] created and designed a prototype for a rule-based expert advising system with an object-oriented database in yet another attempt to create a smart advising system. The authors based their system on two categories: academic rules and student preferences. All student choices are entered into the system, which then checks them against the regulations. The authors failed to use their expert system to create an automated system that automatically generates recommendations based on noticed patterns among previously registered courses taken by previous students, even though this system is an advancement towards building a smart system by integrating students' preferences with a process of validating these preferences (which refines those preferences).

2.4 Full stack Web Application

When working on a project, the first thing a developer should think about is the project structure, which includes the system's UI/UX design. React is an open and free front-end JavaScript framework for building user interfaces that leverage UI [13]. It is sponsored by Meta (formerly Facebook) as well as a community of developers and businesses. React can be used as a foundation for single-page, mobile, or http applications thanks to frameworks like Next.js. Because React is mainly concerned with state management and displaying that state to the DOM, React apps frequently require the use of additional frameworks for navigation and client-side features. Node.js is a cross-platform, open-source back-end JavaScript runtime environment that runs JavaScript code outside of a web browser using the V8 engine. Node.js enables developers to construct command-line tools and server-side scripting using JavaScript. Before transmitting the page to the user's browser, which includes running scripts on the server. As a result, Node.js represents a "JavaScript everywhere" paradigm, uniting online application development under a single programming language rather than two separate technologies for server-side and client-side scripts. Asynchronous I/O is possible because of Node.js' event-driven architecture. These design choices are aimed at increasing throughput and scalability in web applications with a lot of input/output activities, as well as real-time Web applications.

2.5. Android Studio code

This is free software designed specifically for creating Android apps. After you've installed the software, you'll need to download the essential plugins. The next step is to open the app and create a new project, as shown in **Figure 2.2** below.

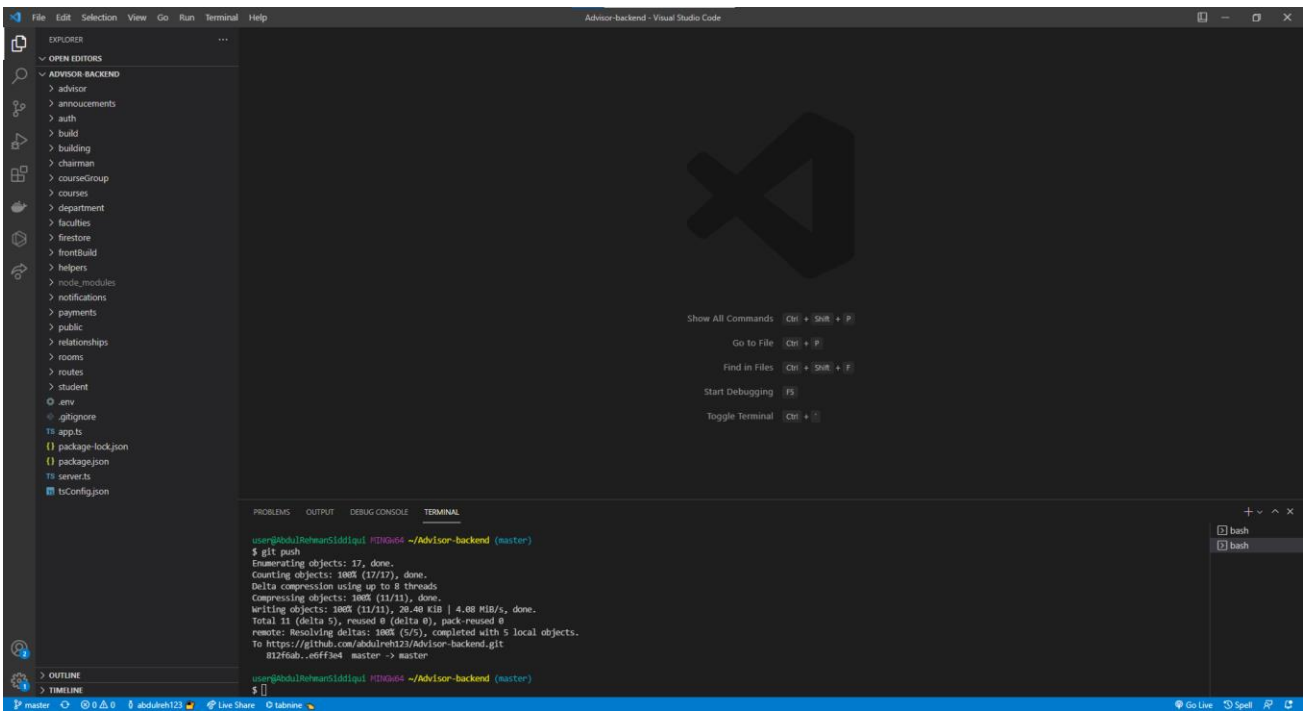


Figure 2.2: Android Studio Interface

Each Android Studio project has one or more modules that contain source code and resource files.

Modules can be of various types:

- Modules for Android apps
- Modules for libraries
- Modules for Node Engine

As seen in Figure 2.2, Android Studio displays your project files in the Android project window by default. This view is grouped by modules to make it easy to find the important source files for your project.

2.6 Tensor flow

TensorFlow is a machine learning system that works in diverse situations and at scale[14]. Dataflow graphs are used by TensorFlow to describe computation, shared state, and the operations that modify it. It distributes the nodes of a dataflow graph among numerous machines in a cluster and among

various computing units within a single system, such as multicore CPUs, general-purpose GPUs, and specially created ASICs known as Tensor Processing Units (TPUs)[15]. This architecture allows the application developer flexibility because, unlike earlier "parameter server" systems, which managed shared state internally, TensorFlow enables developers to test out cutting-edge optimizations and training techniques. With an emphasis on deep neural network training and inference, TensorFlow serves a wide range of applications. TensorFlow has been widely adopted for machine learning research and is currently utilized in several Google services. It was also made available as an open-source project by us. In this article, we present the TensorFlow dataflow model and show the impressive results that Tensor-Flow produces for a number of practical applications.

2.7 Data Mining

The act of sifting through large data sets in order to find patterns is known as data mining, and to find patterns and relationships that may be used to address business problems. Enterprises can use data mining techniques and technologies to predicting future trends and make better business decisions.

It is an important element of data analytics and the fundamental disciplines in data science, in which advanced analytics techniques are used to extract meaningful information from large data sets. Data mining is a step in the knowledge discovery in databases (KDD) process, a data science methodology for obtaining, processing, and analyzing data, at a more granular level. Although data mining and KDD are often used interchangeably, they are more frequently considered as separate concepts.

The four basic stages of the data mining process are as follows as shown in **figure 2.3**:

1. **Obtaining information.** Data that is relevant to an application for data analysis is recognized and gathered. The data could be in many source systems, a data warehouse, which is becoming increasingly popular in huge data environments with a mix of structured and

unstructured data. It's also possible to use external data sources. A data scientist commonly transports data to a data lake for the rest of the phases in the process, regardless of where it came from.

2. **Preparation** of data This stage consists of a series of activities that prepare the data for mining. Data exploration, profiling, and pre-processing are the first steps, followed by data cleaning to correct mistakes and other quality issues. If a data scientist is attempting to evaluate unfiltered raw data for a specific application, data transformation is also done to make data sets consistent.
3. **Mining the data.** determining the right technique and then implementing one or more algorithms to execute the mining when the data is prepared. Before being run on the entire set of data in ML applications, the algorithms must often be trained on sample data sets to hunt for the information being sought.
4. **Data interpretation and analysis,**The results are used to develop models that can aid in decision making and other commercial activities. The data science team must also convey the findings to company executives and users, which is commonly accomplished through data visualization and data storytelling methodologies.

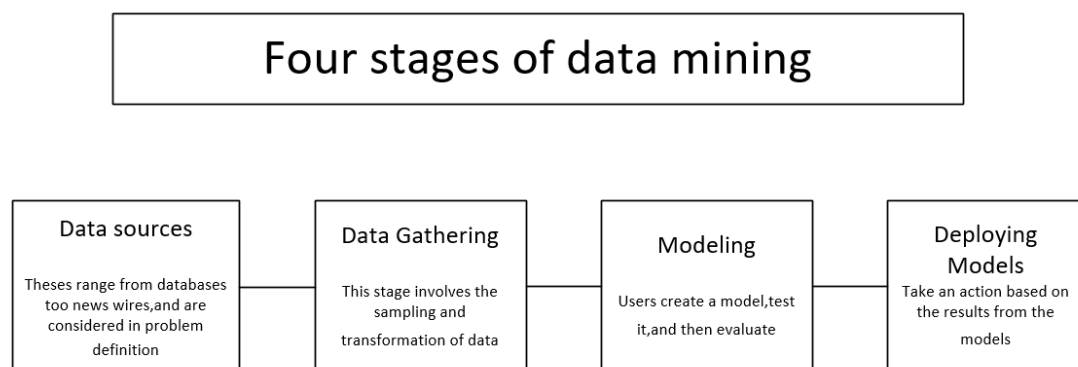


Figure 2.3: Stages of data mining.

2.9 Apriori

It's a method for extracting Boolean association rules. The association rules between the data are mined from the generated frequent item sets, giving us decision support, and the full database is inspected after each set of frequent item sets is formed. An item-set is a collection of 0 or more items, and a frequent item-set is one that has more support than the specified minimum support count [6]. Frequently used item-set evaluation criteria:

- Support: one of the two core components of association rules. It is the proportion of transactions in the dataset D's All sample that include both x and y to all transactions. The corresponding support level for two data sets (x and y) that require correlation analysis is:

$$\text{Support}(x,y) = P(xy) = \text{num}(xy) / \text{num}(\text{Samples})$$

$$\text{Support}(X \& Y) = P(X \cup Y) = \text{count}(XUY) / |D|$$

A support rating of 28%, for example, indicates that there is a 28% chance that a person in the population will include both X and Y.

- Confidence: The proportion of transactions with x and y included to those with y included is the conditional probability. $\text{Confidence}(x \Rightarrow y) = P(X | Y) = P(xy) / P(y)$

$$\text{Confidence}(X \& Y) = P(X|Y) = \text{Support}(XUY) / \text{Support}(X)$$

The confidence is 52 percent if 52 percent of the phrases containing X contain Y.

- lift: It's the ratio of the total number of transactions occurring in x to the number of transactions containing x under the assumption of include y.

$$\text{Lift}(X \Rightarrow Y) = P(X|Y) / P(x) = \text{Confidence}(x \& y) / P(x)$$

2.10 Work Comparison

This thesis describes the application in terms of its functionality, user interface, and supporting architecture.

It has six functionality

- Automated scheduling

- Prediction of Cgpa
- Cgpa calculator
- Course selection automation
- Timetable
- Course clash notifier

Application Name	Automated scheduling	Cgpa Prediction	Cgpa Calculator	Course Automation	Time table	Course clash notifier
NEU SIS System	X	X	X	X	X	X
EMU STD Portal	X	X	✓	X	✓	✓
Advisor App	✓	✓	✓	✓	✓	✓

Table 2.1: App Comparisons

CHAPTER 3

SYSTEM ARCHITECTURE

the architecture of the system will be discussed here, which technologies were used and how it was implemented

3.1 The structure of Advisor system

Visual studio code, nodejs, reactjs, and SQL were used to create the NEU Advisor system. The project is complicated, yet the user interface is simple. TypeScript, Python, and HTML files are included. The software uses normal web applications and Python for machine learning and data mining methods, with a dynamic SQL backend that allows the administrator to update and maintain backend data without the need for a separate server. **Figure 3.1** depicts the life cycle and operation of an Web application.

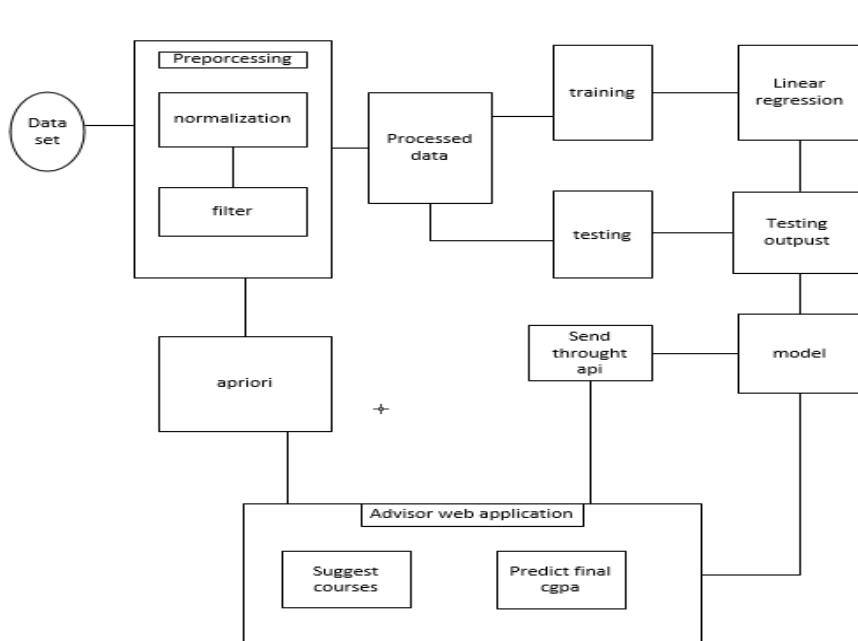


Figure 3.1: The structure of Advisor system

3.3. Backend customization

First create an empty folder, navigate into and open in VS Code, From the File Explorer toolbar, press the New File button:

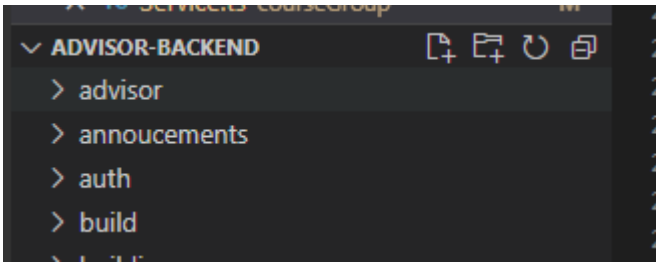


Figure 3.1: Android Studio folder structure

and name the file:



Figure 3.2: Android Studio file structure

and then open the folder in terminal and write `npm init` which will create the `package.json` file

which will contain all our libraries which will be used in the project.

Now the next step is to create a server for our app. `express` will be used for generating api endpoints,

3.3.1 Database Architecture

The MySQL design shows how the various components of a MySQL framework interact. MySQL engineering is essentially a client-server architecture. MySQL information base server refers to the server as well as the apps that connect to it. are customers.

This mobile app requires an internet connection and a centralized database (MySQL). This app is a client/server application. Clients are Web apps, whereas the server is made up of MySQL databases and Typescript API scripts. The Typescript APIs script connects the Web application to the MySQL database. The structure of this application is shown in **Figure 3.4**. The Typescript API's primary functions are as follows:

- Accepting the read and write request from the clients.
- Run and manipulate the request to MySQL database
- Formatting the output as JavaScript Object Notation (JSON)

Operations that are applied on the app: The developed application presents some operations that have been created to make some interaction with users. These operations are:

- Add new users
- Add courses and background information
- User authentication
- Add rooms to courses

A database is a structured collection of data that is often stored and accessible electronically through a computer system. Where databases are more complicated, formal design and modeling techniques are frequently used. The database management system (DBMS) is the software that captures and analyzes data through interacting with end-users, applications, and the database itself. The database is required for data storage in this project. A user is established after the database is setup to enable permissions for reading, writing, and other functions. We access the phpMyAdmin dashboard to build tables after we've finished creating the database and user.

Table	Action	Rows	Type	Collation	Size	Overhead
advisor	Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	32.0 K1B	-
announcements	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	32.0 K1B	-
buildings	Browse Structure Search Insert Empty Drop	19	InnoDB	utf8mb4_general_ci	16.0 K1B	-
chairman	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	32.0 K1B	-
coursegroup	Browse Structure Search Insert Empty Drop	67	InnoDB	utf8mb4_general_ci	48.0 K1B	-
courserooms	Browse Structure Search Insert Empty Drop	7	InnoDB	utf8mb4_general_ci	48.0 K1B	-
courses	Browse Structure Search Insert Empty Drop	52	InnoDB	utf8mb4_general_ci	48.0 K1B	-
departments	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	64.0 K1B	-
faculties	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	32.0 K1B	-
notifications	Browse Structure Search Insert Empty Drop	26	InnoDB	utf8mb4_general_ci	16.0 K1B	-
rooms	Browse Structure Search Insert Empty Drop	173	InnoDB	utf8mb4_general_ci	32.0 K1B	-
sessions	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	16.0 K1B	-
studentpayments	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	32.0 K1B	-
students	Browse Structure Search Insert Empty Drop	28	InnoDB	utf8mb4_general_ci	64.0 K1B	-
studentscourses	Browse Structure Search Insert Empty Drop	40	InnoDB	utf8mb4_general_ci	48.0 K1B	-
user	Browse Structure Search Insert Empty Drop	39	InnoDB	utf8mb4_general_ci	64.0 K1B	-
16 tables	Sum	463	InnoDB	utf8mb4_general_ci	624.0 K1B	0 B

Figure 3.3: Adding tables to the database

sixteen tables were created which are advisor, buildings, chairman, courseGroup, courseRooms, courses, departments, faculties, rooms, students, studentCourses

3.4 Linear Regression

In this project, historical data on students' cgpa was available from 2015, and the data was extracted using excel and javascript. The goal was to predict student's final cgpa based on which semester they were currently in, and seven models were developed using python and the tensorflow library., here are the following steps.

- Get the required data from the dataset
- Remove null values
- Remove repeated data
- filter column for each model
- send the prediction using api

StudentN	AcademicYear	Term	CourseTitleNameTur	TermAVG
20147176	2015-2016	Güz	DİSKRİT YAPILAR (G1)	1.590909
20147176	2015-2016	Güz	İNGİLİZCE II (G3)	1.590909
20147176	2015-2016	Güz	MATEMATİK II (G3)	1.590909
20147176	2015-2016	Güz	GENEL KİMYA (G2)	1.590909
20147176	2015-2016	Güz	PROGRAMLAMA VE SORU ÇÖZME (G1)	1.590909
20147176	2015-2016	Güz	GENEL FİZİK II (G2)	1.590909
20147176	2014-2015	Bahar	PROGRAMLAMAYA GİRİŞ (G1)	2.026315
20147176	2014-2015	Bahar	MATEMATİK I (G1)	2.026315
20147176	2014-2015	Bahar	İNGİLİZCE I (G1)	2.026315
20147176	2014-2015	Bahar	GENEL KİMYA (G1)	2.026315
20147176	2014-2015	Bahar	TÜRKÇE I (G1)	2.026315
20147176	2014-2015	Bahar	BİLGİSAYAR MÜHENDİSLİĞİNE YÖNLENDİRİM	2.026315
20147176	2014-2015	Bahar	GENEL FİZİK I (G1)	2.026315
20150070	2016-2017	Yaz	DEVRE TEORİSİ II (G1)**	3.171052
20150070	2016-2017	Bahar	MÜHENDİSLİKTE PROJE YÖNETİMİ (G1)	3.171052
20150070	2016-2017	Bahar	ATATÜRK İLKELERİ VE İNKILAP TARİHİ II	3.171052
20150070	2016-2017	Bahar	ELEKTROMANYETİK TEORİ (G1)	3.171052
20150070	2016-2017	Bahar	DEVRE TEORİSİ I (G1)	3.171052
20150070	2016-2017	Bahar	DOĞRUSAL CEBİR I (G1)	3.171052
20150070	2016-2017	Bahar	KARMAŞIK ANALİZ I (G1)	3.171052

Figure 1: data given

cgpa_one	cgpa_two	cgpa_three	cgpa_four	cgpa_five	cgpa_six	cgpa_seven	final
2.8	2.7	2.5	2.2	2.1	2.2	2.6	3
2.9	2.8	2.6	2.3	2.2	2.3	2.7	3.05
3	2.9	2.7	2.4	2.3	2.4	2.8	3.1
3.1	3	2.8	2.5	2.4	2.5	2.9	3.15
3.2	3.1	2.9	2.6	2.5	2.6	3	3.2
3.3	3.2	3	2.7	2.6	2.7	3.1	3.25
3.4	3.3	3.1	2.8	2.7	2.8	3.2	3.3
3.5	3.4	3.2	2.9	2.8	2.9	3.3	3.35
3.6	3.5	3.3	3	2.9	3	3.4	3.4
3.7	3.6	3.4	3.1	3	3.1	3.5	3.45
2.7	2.6	2.4	2.1	2	2.1	2.5	2.95
2.8	2.7	2.5	2.2	2.1	2.2	2.6	3
2.9	2.8	2.6	2.3	2.2	2.3	2.7	3.05
3	2.9	2.7	2.4	2.3	2.4	2.8	3.1
3.1	3	2.8	2.5	2.4	2.5	2.9	3.15
3.2	3.1	2.9	2.6	2.5	2.6	3	3.2
3.3	3.2	3	2.7	2.6	2.7	3.1	3.25
3.4	3.3	3.1	2.8	2.7	2.8	3.2	3.3
3.5	3.4	3.2	2.9	2.8	2.9	3.3	3.35
3.6	3.5	3.3	3	2.9	3	3.4	3.4
2.6	2.5	2.3	2	1.9	2	2.4	2.9
2.7	2.6	2.4	2.1	2	2.1	2.5	2.95
2.8	2.7	2.5	2.2	2.1	2.2	2.6	3
2.9	2.8	2.6	2.3	2.2	2.3	2.7	3.05
3	2.9	2.7	2.4	2.3	2.4	2.8	3.1
3.1	3	2.8	2.5	2.4	2.5	2.9	3.15
3.2	3.1	2.9	2.6	2.5	2.6	3	3.2
3.3	3.2	3	2.7	2.6	2.7	3.1	3.25

Figure 2: final data

Based on the training dataset, the multiple regression analysis technique was used to create seven predictive models. Each forecasting model's mathematical formula is as follows:

Students done with one semesters:

$$\text{Final} = 0.64865811(\text{cgpa_one}) + 0.8702897954289899$$

Students done with two semesters:

$$\text{Final} = -0.3767197(\text{cgpa_one}) + 1.07788012(\text{cgpa_two}) + 0.7319795607738113$$

Students done with three semesters:

$$\text{Final} = -0.05800013 (\text{cgpa_one}) - 0.43605543 (\text{cgpa_two}) + 1.34217604(\text{cgpa_3}) + 0.3978674914422706$$

Students done with four semesters:

$$\text{Final} = -0.01496279 (\text{cgpa_one}) + 0.14194983 (\text{cgpa_two}) - 0.59270694 (\text{cgpa_3}) + 1.3738597(\text{cgpa_four}) + 0.2785981385730869$$

Students done with five semesters:

$$\text{Final} = -0.01199102(\text{cgpa_one}) - 0.0020485 (\text{cgpa_two}) + 0.1369418(\text{cgpa_3}) - 0.68974443 (\text{cgpa_four}) + 1.51859425(\text{cgpa_five}) + 0.16427075754551002$$

Students done with six semesters:

$$\text{Final} = -0.002209(\text{cgpa_one}) + 0.02272607(\text{cgpa_two}) + 0.02647803(\text{cgpa_3}) - 0.09801539 (\text{cgpa_four}) - 0.41102471 (\text{cgpa_five}) - 0.41102471(\text{cgpa_six}) + 0.07259407553354569$$

Students done with seven semesters:

$$\text{Final} = -0.00929751(\text{cgpa_one}) + 0.00617765 (\text{cgpa_two}) + 0.02967188 (\text{cgpa_3}) - 0.05704418 (\text{cgpa_four}) + 0.09269362 (\text{cgpa_five}) - 0.57070776 (\text{cgpa_six}) + 1.49952976(\text{cgpa_seven}) + 0.030531324906364343$$

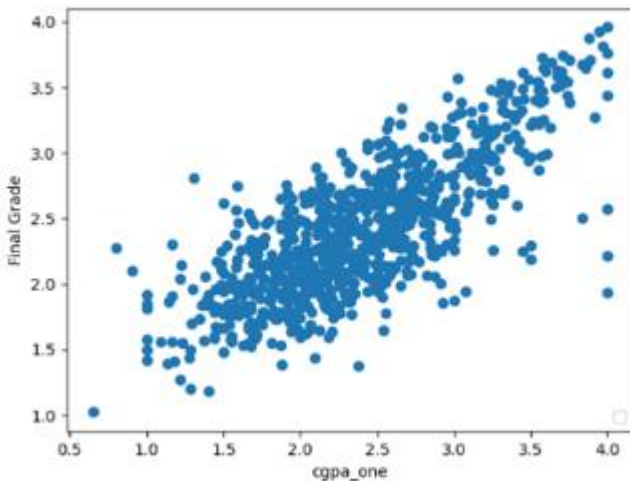


Figure 3: Linear regression plot

3.5 Apriori

On the data set, we used an apriori technique to count the number of times each course was taken. The minimal support was set to 0.0050, and itemsets with occurrences that satisfy the min sup were found. Only courses with a score greater than or equal to min sup advance to the next iteration, while the others are ignored.

The data set:

?NG?L?ZCE II	MATEMATİK II	GENEL K?MYA	PROGRAMLAMA V GENEL F?Z?K II			
MATEMATİK I	?NG?L?ZCE I	GENEL K?MYA	T?RK?E I	B?LG?SAYAR M?HEND?SL?P??NE Y?N GENEL F?Z?K I		
?NG?L?ZCE I	GENEL F?Z?K I	B?LG?SAYAR PROGRAMLAMA	GENEL K?MYA	MATEMATİK I		
TEKNİK F?Z?M	MATEMATİK I	ELEKTRİK VE ELEKTRONİK M?H. G?R??	?NG?L?ZCE II			
FRANSIZCA I	DEVRE TEOR?S? I	B?LG?SAYAR UYGULAMALARI	?NG?L?ZCE ?LET?? D?FERANS?YEL DENKLEMLER			
ATAT?RK ?L?KELER? VE ?NKILAP TAR?H? I	ELEKTRONİK TEOR?S? I	OLASILIK VE ?STAT?STİK I	DO?RUSAL CEB?R I KARMA?IK ANALZ I			
S?NYALLER VE S?STEMLER	S?N?R A?LARI	STAJ I	SAYISAL ANALZ I			
?LET??M S?STEMLER?	ELEKTRONİK I	AYDINLATMA TEKN?? VİSTAJ II	KONTROL S?STEMİ ELEKTRİK MAK?NALARI II			
ELEKTRONİK II	AYDINLATMA TEKN?? VİSTAJ II	?LET??M S?STEMLER?	G?? S?STEMLER? A M?HEND?SL?K TASARIM I	Y?KSEK GER?LM TEKN?? I		
ELEKTRİK DA?TITIM S?STEMİ T?RK D?L? I	?LET??M S?STEMLER?	PROGRAMLANAB?L?R MANTIKSAL DENETİM M?HEND?SL?K TASARIM	T?RK D?L? II	G?? S?STEMLER? ANALZ? II	PROGRAMLANAB?L?R MANTIKSAL DENETİM M?HEND?SL?K TASARIM	M?KRO?LEMC?LER
MATEMATİK I	ATAT?RK ?L?KELER? VE ?NKILAP TAR?H? I	GENEL K?MYA	GENEL F?Z?K I	?NG?L?ZCE I		
ELEKTRİK VE ELEKTRONİK K? ?NG?L?ZCE II	TEKNİK F?Z?M	ELEKTRİK MALZEMELER?	GENEL F?Z?K I	ATAT?RK ?L?KELER? VE ?NKILAP TAR?H? I		
ELEKTRİK MALZEMELER?	B?LG?SAYAR UYGULAMALARI ?NG?L?ZCE ?LET??M TEKN?KLER?	DO?RUSAL CEB?R I	GENEL F?Z?K II	FRANSIZCA I		
DO?RUSAL CEB?R I	ELEKTRONİK TEOR?S? I	SAYISAL ANALZ I	ATAT?RK ?L?KELER? KARMA?IK ANALZ I			
SAYISAL ANALZ I	S?N?R A?LARI	OLASILIK VE ?STAT?STİK I	STAJ I	ELEKTRİK MAK?NALARI I		
ELEKTRİKSEL ?L?ME	ELEKTRONİK I	?LET??M S?STEMLER?	ELEKTRİK MAK?N?STAJ II			
Y?KSEK GER?LM TEKN?? I	G?? S?STEMLER? ANALZ? SAYISAL MANTIK S?STEMLER?	AYDINLATMA TEKN?M?HEND?SL?K TASARIM I	M?HEND?SL?K TAS M?KRO?LEMC?LER	B?LG?SAYARLARLA G?RMEYE G?R??	ELEKTRİK DA?TITIM S?STEMLER?	
T?RK D?L? II	PROGRAMLANAB?L?R M?T?RK D?L? I	GENEL K?MYA	GENEL F?Z?K I	ATAT?RK PRENS?PLER? VE REFORM? D?SKR?T YAPILAR		
PROGRAMLAMAYA G?R??	?NG?L?ZCE I	MATEMATİK I	GENEL F?Z?K I	ATAT?RK PRENS?PLER? VE REFORM? D?SKR?T YAPILAR		
?NG?L?ZCE II	GENEL K?MYA	MATEMATİK I	GENEL F?Z?K I	ATAT?RK PRENS?PLER? VE REFORM? D?SKR?T YAPILAR		
MATEMATİK I	D?SKR?T YAPILAR	ATAT?RK PRENS?PLER? VE REFORMLARI	GENEL F?Z?K II	?NG?L?ZCE II		
GENEL K?MYA	KONU?MA BECER?LER?	SAYISAL MANTIK S?STEMLER?	ATAT?RK ?L?KELER? Matematik II			
MATEMATİK II	FRANSIZCA I	GENEL F?Z?K I	GENEL K?MYA			
NESNEYE Y?NEL?ML? PROGİS?STEM S?M?LASYON	SAYISAL MANTIK S?STEMLER?		MATEMATİK II			
FRANSIZCA I	VER? YAPILARI VE ALGORİTMA MATEMATİK II	SAYISAL MANTIK S?STEMLER?				
NESNEYE Y?NEL?ML? PROGİGENEL F?Z?K I	SAYISAL MANTIK S?STEMLER?	DO?RUSAL CEB?R I				
T?RK D?L? I	D?? MORFOLOJ?S? VE M?ORGANİK K?MYA VE B?YOK?MYA	B?YOF?Z?K	TİBB? B?YOLOJ? VE GENETİK	F?Z?K	SE?MEL? DERS (BEDEN E? B?LG?SAYAR	
ANATOM?	PROTETİK D?? TEDAV?S? TEMEL YA?AM DESTE??	ENDODONT?	ANATOM?	TİBB? B?YOLOJ? VE GENETİK	H?STOLOJ?	M?KROB?YOLOJ?
GENEL K?MYA I	B?LG?SAYAR M?HEND?SL GENEL F?Z?K I	YABANCI D?L I	PROGRAMLAMAYA G?R??	ATAT?RK ?L?KELER? VE ?NKILAP TAR?H? I		

Figure 4: data for apriori

Only the most appealing courses, or at least 30 occurrences, are subject to the rules. $30/5745=0.0050$ is the amount of support for those things. The regulations have a minimum confidence level of 20%, or 0.2. In the same way, we set lift to 3 and min length to 2 because we want at least two courses in our rules.

Association Rules:

Rule	Support	Confidence	Lift
SYSTEMLER? -> ELEKTRİK MAK?NALARI II	0.015143603133159269	0.4484536082474227	14.39310603006393
YABANCI DİL I -> MATEMATİK I	0.006788511749347258	0.22033898305084745	5.651104721549636
YABANCI DİL I -> MATEMATİK I	0.012010443864229765	0.24295774647887325	6.23121541750503
PROGRAMLAMAYA -> GENEL KİMYA I	0.007310704960835509	0.23728813559322032	15.85139929050059

Figure 5: rules

3.5 SVM (Support Vector Machine)

Scikit-learn is a popular library for implementing machine learning algorithms in Python. SVM is also available in the scikit-learn library,

Let's have a look at how SVM can be used for categorization, the final cgpa is converted to string as if the cgpa is 2.2 the result will be 2 – 2.5.

Using the following criterion, the data leads to a hyperplane that separates the seven classes: The data is projected onto the hyperplane in such a way that the between-class variance is maximized while the within-class variance is minimized.

CHAPTER4

USER INTERFACE

In this section the user interface of the system and key features will be discussed here

4.1 App features

Because the room is a group to their faculty, the Advisor app interface is designed to allow users of small-screen devices to quickly screw down or navigate for a specific query like a courses in their departments. To achieve this, the app makes use of the app interaction model's main elements. The characteristics of student management are as follows:

- The app will automate which courses to be chosen.
- The app will automate timetable.
- The app will automatically offer semester courses.
- Automated scheduling.
- CGPA calculater.
- Cgpa Predector.

The screenshots of the user interface are listed below in **figure 4:1** screenshot A the login page, screenshot B shows the dashboard, and screenshot C shows the transcript.

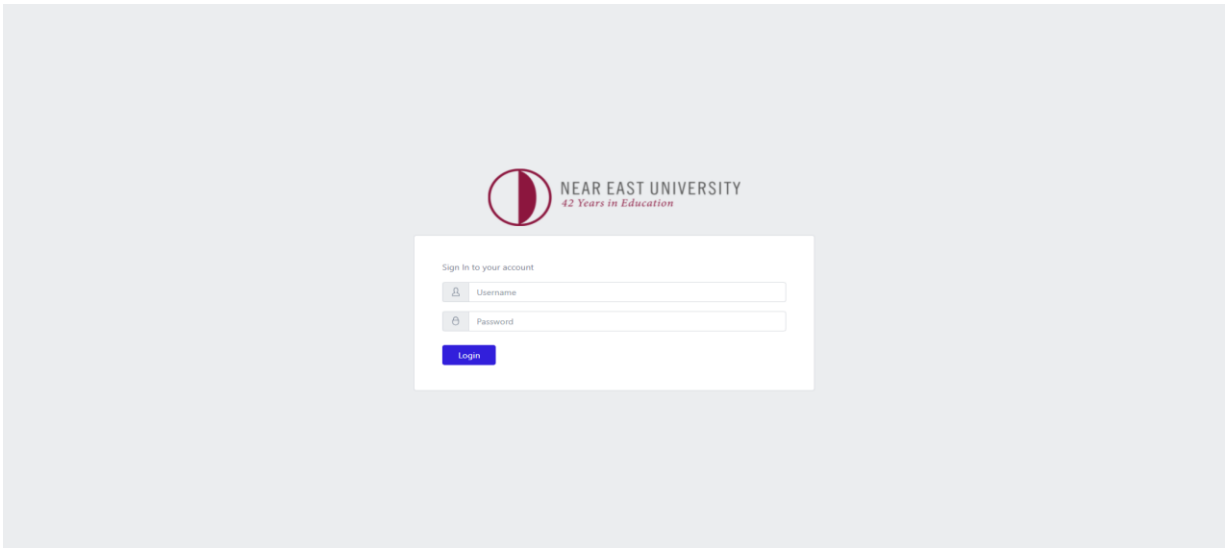


Figure 4.1: login page

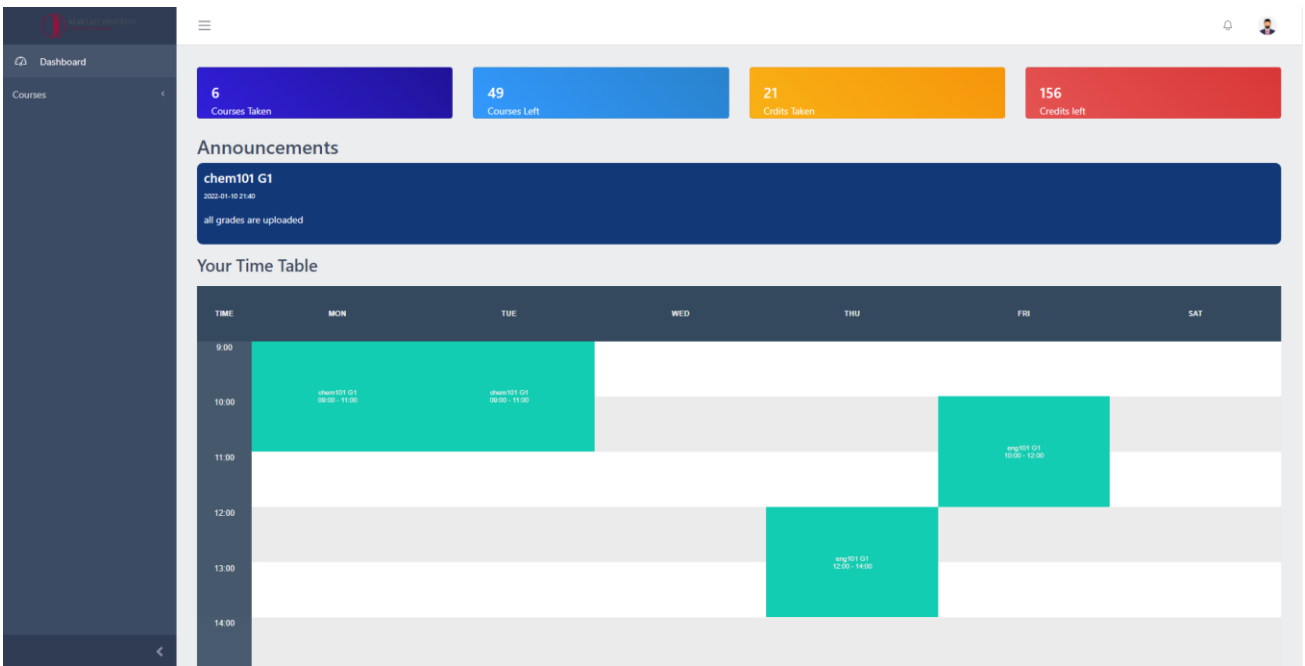


Figure 6: dashboard

Code	Name	Credit	grade	CrPts
CHM101	GENERAL CHEMISTRY	4	AA	12
ENG101	English I	3	AA	12
MTH101	Calculus I	4	BA	14
PHY101	General Physics I	2	CC	4
Y7T101	Turkish	4	AA	16
ECC106	Introduction to Programming	4	CB	10

GPA: 3.24 / CGPA: 3.24 / STATUS : Successful / TOTAL CREDIT: 21

Figure 4.3: Transcript

When a user first launches the app, the first screen that appears is a login screen with the app logo. The user then enters their credentials; if they are not approved, the app will deny access; the user's information will be retrieved; and the dashboard will provide a list of equipment according to the faculty. Users can also access the adviser system, which displays a variety of features.

4.2 Course Selection

To add a course, the student must first go to their profile, then click add courses, select the semester, and if there is a conflict, the system will provide an error.

MANAR AL ISLAM MARWAN HAKAM
Student

✉ manar al islam marwan hakam.manar al islam marwan hakam@gmail.com

🏠 Artificial intelligence

📄

General | **Contact**

Roll	:	125
Academic Year	:	2020
Gender	:	Male
Predicted Graduation Cgpa	:	2.966710237460468
blood	:	B+

[Add courses](#) [Automate Selection](#)

Payments

There are no records to display

Figure 4.4: student profile

Add courses ✕

Academic Year :

Please select

- Please select
- 2021-2022 - Fall
- 2021-2022 - Spring
- 2022-2023 - Fall
- 2022-2023 - Spring
- 2023-2024 - Fall
- 2023-2024 - Spring
- 2024-2025 - Fall
- 2024-2025 - Spring

Figure 4.5: course selection

AA	12
ecc102 G1 has a clash with eng101 G1 ✕	
CC	4

Figure 4.6: clash error

4.3 Automate Selection

Course automation will reduce advisor and student involvement in course selection. Predetermining decision criteria, subprocess linkages, and related activities — and encoding those predeterminations in computers – reduces intervention.

Course automation entails the utilization of a variety of data, such as which semester it is, whether the student has met the prerequisites, whether the course opens other courses, and so on.

To automate the user just has to click on the automation button and select the semester.

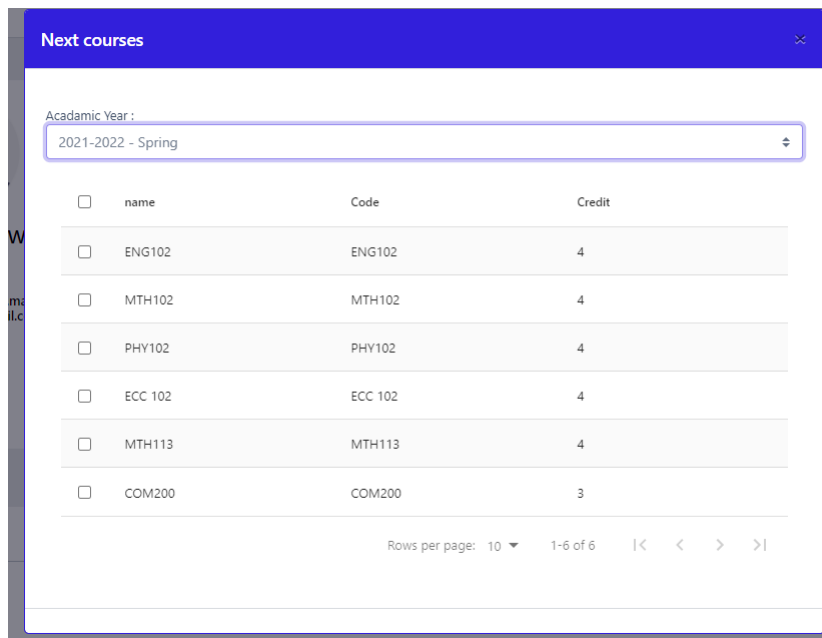


Figure 4.7: automate selection

4.4 Cgpa Calculator

CGPA Calculator is a unique calculator for calculating GPA or CGPA. Calculating GPA and CGPA by semester or year is quite simple. How to use the CGPA Calculator is a question that many students have. What Does the CGPA Stand For? See also the CGPA Calculator for Honours and the CGPA Calculator for Degrees. I'll explain how to compute CGPA as well as the whole meaning of CGPA. Let's look at how to calculate CGPA and the NEU GPA/CGPA Grading System.

CGPA Grading System

Number	Grade	Point
80% and above	AA	4.00
75% to less than 80%	BA	3.50
70% to less than 75%	BB	3.00
65% to less than 70%	CB	2.50
60% to less than 65%	CC	2.00
55% to less than 60%	DC	1.50
50% to less than 55%	DD	1.00
45% to less than 50%	FD	0.50
40% to less than 45%	FF	0.00

Figure 4.8: grade points

The CGPA Calculator is simple to operate. You can determine your GPA or CGPA by semester or year. The system compute your first-year or semester GPA before proceeding to the next year or semester.

PHY101	General Physics I	2	CB	5
Y2T101	Turkish	4	DD	4
ECC106	Introduction to Programming	4	CC	8
GPA: 2.06 / CGPA: 2.06/ STATUS : Successful / TOTAL CREDIT: 17				
Academic year 2021-2022 - Spring				
Code	Name	Credit	grade	CrPts
ENG102	English II	4	AA	16
MTH102	Calculus II	4	Predict Grade	
PHY102	General Physics II	4	CB	10
ECC 102	Programming &Problem Solving	4	Predict Grade	
MTH113	Linear Algebra	4	Predict Grade	
COM200	Summer Practice I	3	Predict Grade	
GPA: 4.00 / CGPA: 2.43/ STATUS : / TOTAL CREDIT: 23				

Figure 4.9: Cgpa Calculator

4.5 Cgpa Predictor

The Predictor takes inputs depending on how many semesters has been finished by the student Then gives the prediction.

Roll	:	125
Academic Year	:	2020
Gender	:	Male
Predicted Graduation Cgpa	:	2.203167295318509
blood	:	B+

Figure 4.10: Cgpa predictor

CHAPTER5

RESULT ANALYSIS

Programs are tested and debugged to ensure that the general operation of the program satisfies the overall system objectives, which will be detailed here. The computer target requirements are also detailed in this chapter, as are maintenance difficulties.

Performance is crucial for every web application. We make certain that the system's performance is up to par. The proposed web app is properly optimized since if it is slow, the end-user will look for another related app that performs better. Our system is compatible with other systems and capable of utilizing existing features in today's web application. This is the option that was picked. Because of cgpa prediction and course management sections of the application,

5.1 Test Environment and Test Plan

The test environment consists of the two environments where the tests will be executed, as well as the tools and requisite hardware.

The following is the test environment used in this study:

I. Browser evaluation

II. Monitoring tool for hardware resource consumption in devices

The web server is equipped with 2GB of RAM, a 5GB hard drive, and a Quad-core 3.3GHz CPU.

Table 1.2 shows the list of web browsers and specifications that were used to test the app.

Device	Browser	Ram
Samsung Galaxy A21s	Google Chrome	3Gb
Lenovo Desktop	Microsoft Edge	8gb
Monster Notebook	Opera	8Gb

Table 5.1: Test cases

For each test, we record the amount of time it takes to complete a job, as well as the amount of RAM and CPU used. The tests were conducted over both a Wi-Fi and a 4G network. The total time required to complete the request. In addition, the processing time required by the server will be measured so that we can track the delay for each cloud request. To determine how well the server's resources, RAM, and processor are being used,

5.2 Browser testing

The fundamental goal of testing is to take the tiniest bit of the tested section of the application and evaluate whether it behaves exactly as it should and meets the thesis' objectives.

- **5.2.1 Speed test.**

The speed test assists us in determining the ideal network conditions in which to run the application. The y-axis shows the application's performance in milliseconds, while the x-axis shows the devices and their parameters, which we call devices, as well as the browser type. We tested connection speeds using three distinct browsers: Google Chrome, Microsoft Edge, and Safari. We choose to use these browsers since they are the most frequently used and popular. After that, we execute the app in each of these three browsers, and the results are shown in Figure (5.1). According to our findings, the

best connection is WIFI, which implies that our software can function at the fastest connection speed, but LTE connections are still within the limit. and it was tested with different internet speed

We used **3g fast** and **3g slow** which can be selected in the browser

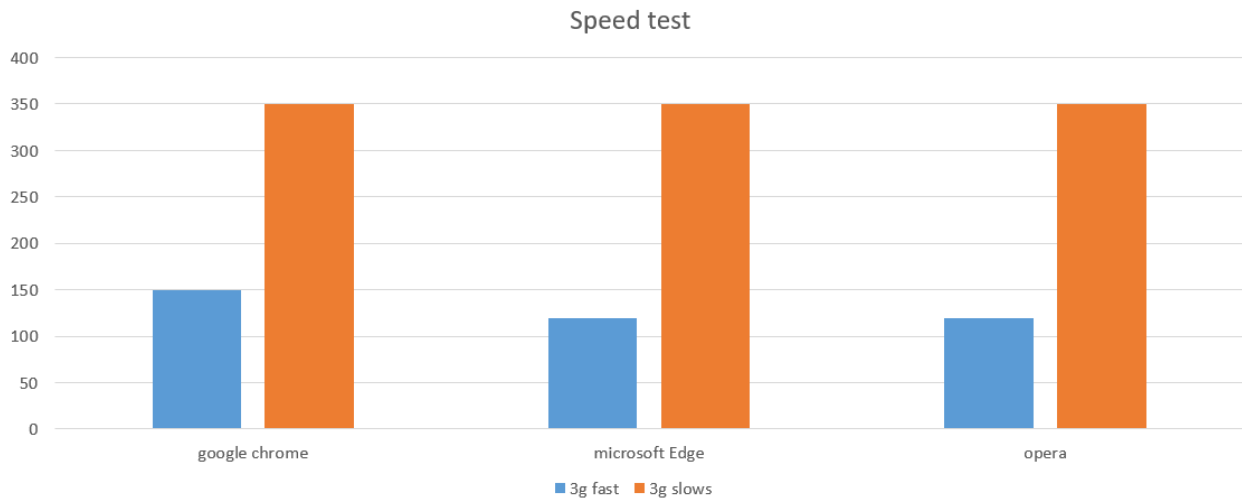


Figure 5.1 :Execution Time

The testing were carried out on a network using the following api response speeds: 1800kbps/100kbps for 3g fast, and 1200kbps/700kbps for 3g slow. For each test, the regular execution time is used. The times are displayed in milliseconds. The execution time of going to a component is shown in Figure 5.1 for nearby, 3g quick, and 3g slow. A cluster of predefined lengths is created in this test, and a component is returned to begin with.

- **5.2.2 Results from Cgpa Prediction(linear regression).**

The application's Cgpa prediction (linear regression) algorithm allows the user to determine their final cgpa. It was put to the test by entering students' transcripts into the system; the algorithm requires 1 to 7 inputs in this procedure.

If the student has completed two semesters, the model will be applied using the input presented in table 2

Semester	Cgpa	Prediction(final)	Actual(final)
1	2.52	2.50	3.72
2	3.13	3.15	3.72
3	3.56	3.67	3.72
4	3.66	3.69	3.72
5	3.72	3.74	3.72
6	3.76	3.75	3.72
7	3.72	3.71	3.72

Table 5.2: linear regression tes case

Prediction	Number of error per 20 cases	Percentage
With 1 semester	15	75%
With 2 semester	11	55%
With 3 semester	8	41%
With 4 semester	7	30.5%
With 5 semester	4	20%
With 6 semester	2	12%
With 7 semester	1	10%

Table 5.3: Error analysis

- **5.2.2 Results from Cgpa Prediction(SVM).**

SVM algorithm was put to the test by entering students' transcripts into the test data; the algorithm requires 1 to 7 inputs in this procedure.

If the student has completed two semesters, the model will be applied using the input presented in table 5.2

Semester	Cgpa	Prediction(final)	Actual(final)
1	2.52	2 - 2.5	3.72
2	3.13	2 - 2.5	3.72
3	3.56	2.5 - 3	3.72
4	3.66	2.5 - 3	3.72
5	3.72	3 - 3.5	3.72
6	3.76	3.5 - 4	3.72
7	3.72	3.5 - 4	3.72

Table 5.4: SVM test case

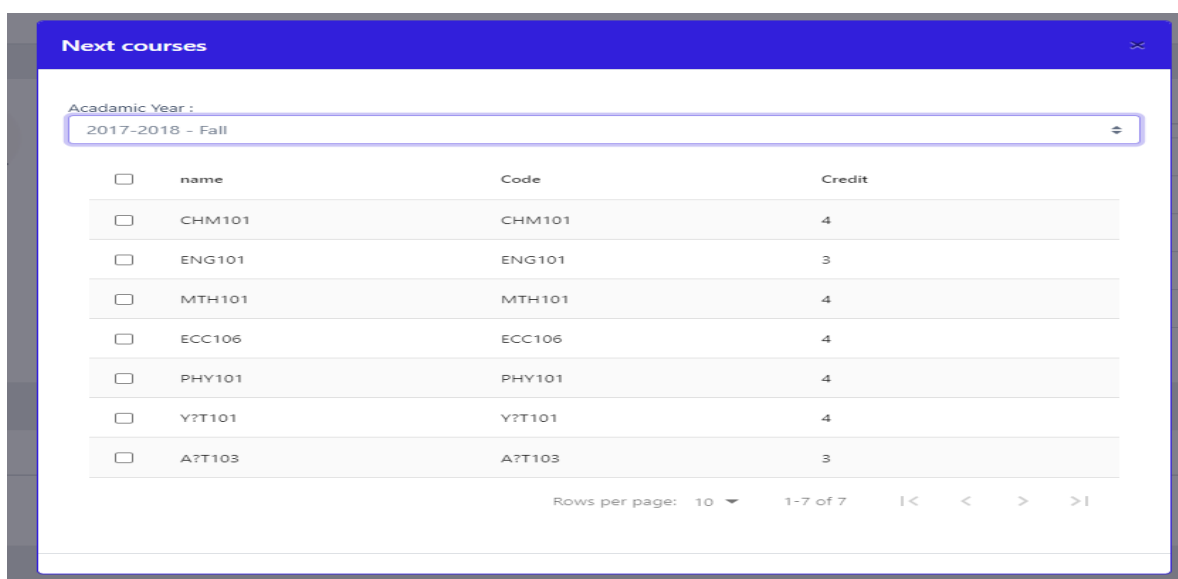
Prediction	Number of error per 20 cases	Percentage
With 1 semester	19	85%
With 2 semester	17	75%
With 3 semester	12	60%
With 4 semester	9	45%
With 5 semester	6	30%
With 6 semester	4	22%
With 7 semester	3	18%

Table 5.3: Error analysis svm

- **5.2.3 Results from course selection:**

The application suggests courses to take based on the student's department and previous courses. The student can then choose among the suggestions, and the system will ensure that there are no conflicts.

Figure 5.2 depicts the outcome.



The screenshot shows a window titled "Next courses" with a blue header. Below the header is a dropdown menu for "Academic Year" set to "2017-2018 - Fall". The main content is a table with columns for "name", "Code", and "Credit". Each row has a checkbox on the left. The table lists seven courses: CHM101 (4 credits), ENG101 (3 credits), MTH101 (4 credits), ECC106 (4 credits), PHY101 (4 credits), Y?T101 (4 credits), and A?T103 (3 credits). At the bottom, there is a pagination control showing "Rows per page: 10" and "1-7 of 7" with navigation arrows.

<input type="checkbox"/>	name	Code	Credit
<input type="checkbox"/>	CHM101	CHM101	4
<input type="checkbox"/>	ENG101	ENG101	3
<input type="checkbox"/>	MTH101	MTH101	4
<input type="checkbox"/>	ECC106	ECC106	4
<input type="checkbox"/>	PHY101	PHY101	4
<input type="checkbox"/>	Y?T101	Y?T101	4
<input type="checkbox"/>	A?T103	A?T103	3

Figure 5.3 :Results from course selection

CHAPTER 6: CONCLUSION

This research suggests a "Smart Advisory Application" to simplify the work of students and advisers. We employed a linear regression technique to estimate the students' Cgpa in this study. To recommend courses to the students, we employed the apriori method. The program was put to the test on a real case study from the Faculty of Engineering at Near East University. The proposed application's usability and efficiency were demonstrated by the results.

The application can be improved in the future by adding personality prediction and entirely automating the assignment of students to advisers. Adding different layers to the web app for Handling high traffic.

References

1. A.M. El-Halees, and M.M. Abu Tair, "Mining educational data to improve students' performance: A case study," International Journal of Information and Communication Technology Research, 2011, pp. 140-146.
2. S.Karthik M.Sukanya, S.Biruntha and T.Kalaikumaran, "Mining Data mining: Performance improvement in education sector using classification and clustering algorithm," ICCCE In Proceedings of the International Conference on Computing and Control Engineering, 2012
3. M. tech Er. Rimmy Chuchra. "Use of data mining techniques for the evaluation of student performance: a case study". International Journal of Computer Science and Management Research, 1(3):425–433, October 2012
4. Brijesh Kumar Bhardwaj and Saurabh Pal. Data mining: " A prediction for performance improvement using classification". (IJCSIS) International Journal of Computer Science and Information Security, 9(4), April, 2011.
5. Kabakchieva, D., Predicting student performance by using data mining methods for classification. Cybernetics and information technologies, 2013. 13(1): p. 61-72
6. Ikbal, S., et al., On early prediction of risks in academic performance for students. IBM Journal of Research and Development, 2015. 59(6): p. 5: 1-5: 14.
7. Ktona, A., D. Xhaja, and I. Ninka. Extracting Relationships between Students' Academic Performance and Their Area of Interest Using Data Mining Techniques. in Computational Intelligence, Communication Systems and Networks (CICSyN), 2014 Sixth International Conference on. 2014. IEEE

8. Bunkar, K., et al. Data mining: Prediction for performance improvement of graduate students using classification. in *Wireless and Optical Communications Networks (WOCN)*, 2012 Ninth International Conference on. 2012. IEEE.
9. B. B. Crookston, "A developmental view of academic advising as teaching", *Journal of College Student Personnel*, Vol. 13, No. 1, pp. 12-17, 1972. A. A. AL-SHARGABI and A. N. NUSARI, Discovering vital patterns from UST students data by applying data mining techniques, in *2010 The 2nd International Conference on Computer and Automation Engineering, ICCAE 2010*, 2010, 2, pp. 547–551
10. C. M. Chando, "Predicting advising style preference from student characteristics", Doctoral dissertation, University of Memphis, UAS, 1997. H. JIAWEI, M. KAMBER, J. HAN, M. KAMBER, and J. PEI, *Data Mining: Concepts and Techniques*, 2012.
11. S. Hsu, O. Marques, M. Ilyas, and X. Ding, "Web-Based Undergraduate Academic Advising System", *International*
12. E. N. OGOR, Student Academic Performance Monitoring and Evaluation Using Data Mining Techniques, in *Electronics, Robotics and Automotive Mechanics Conference (CERMA 2007)*, 2007, pp. 354–359.
13. HARWATI, A. P. ALFIANI, and F. A. WULANDARI, Mapping Student's Performance Based on Data Mining Approach (A Case Study), *Agric. Agric. Sci. Procedia*, 3, pp. 173–177, 2015.
14. W. Ham" al" ainen and M. VINNI, "Classifiers for Educational Data Mining, *Handb. Educ. Data Mining, Data Min. Knowl. Discov. Ser.*, pp. 57–71., 2010.
15. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.
16. B. JANTAWAN and C.-F. TSAI, The Application of Data Mining to Build Classification Model for Predicting Graduate Employment, *IJCSIS) Int. J. Comput. Sci. Inf. Secur.*, 11(10), 2013.

17. A. A. AL-SHARGABI and A. N. NUSARI, Discovering vital patterns from UST students data by applying data mining techniques, in 2010 The 2nd International Conference on Computer and Automation Engineering, ICCAE 2010, 2010, 2, pp. 547–551.

APPENDIX 1

```
const bcrypt = require("bcrypt");
const Advisormodel = require("../advisor/model");
const Annoucementmodel = require("../annoucements/model");
const departmentModel = require("../department/model");
const Student = require("../Model");
const StudentCourses = require("../StudentCourses.model");
const Notification = require("../notifications/model");
const Courses = require("../courses/model");
const CourseRooms = require("../rooms/courseRooms.model");
const Group = require("../courseGroup/model");
import GroupService from "../courseGroup/Service";
const GroupServices = new GroupService();
const user = require("../auth/model")
const { Op, Sequelize } = require("sequelize");
import firestoreService from '../firestore/firebase'
export default class DepartmentService {
  constructor() { }
  public WEEK_DAYS = ["monday", "tuesday", "wednesday", "thursday", "friday", "saturday"]
  private hashPassword = async (password: string): Promise<string> => {
    const salt = await bcrypt.genSalt(10);
    const hash = await bcrypt.hash(password, salt);
    return hash;
  };
  private getAcademicYear = async() => {
    const year = await firestoreService.get(
      'academic',
      'qYX8QXS3XW564eKdfPTP'
    )
    return year.data.year
  }
}
```



```

}
// Create Student
createStudent = async (data: any): Promise<any> => {
  try {
    const department = await Student.create({ ...data });
    if (data.user) {
      const password = await this.hashPassword(data.user.password)
      await user.create({ userName: data.user.userName, password: password, userStudent:
department.id })
    }
    return department;
  } catch (error) {
    throw error;
    // throw new Error("An Error occurred while creating department!");
  }
};
// Get Student
getStudents = async (): Promise<any> => {
  try {
    const departments = await Student.findAll({
      include: [
        {
          model: Advisormodel,
          as: "advisor"
        },
      ],
    });
    return departments;
  } catch (error) {
    throw error;
  }
}

```

```

};
// Get Transcript
getTranscript = async (studentId: number): Promise<any> => {
  try {
    const result = await Student.findOne({
      where: {
        userId: studentId
      },
      include: [
        {
          model: Advisormodel,
          as: "advisor"
        },
        {
          model: Group,
          as: "Group",
          include: [{
            model: Courses,
            as: "Course",
          }],
        }
      ],
    });
    let transcript: any = []
    let totalPts:number=0
    let totalcredits:number=0
    const groups = result.Group
    const academicYears = await groups.map((group: any) => group.studentscourses.academicYear)
    const uniqueArray = academicYears.filter(function (item: any, pos: any) {
      return academicYears.indexOf(item) == pos;
    });
  }
};

```

```

})
await Promise.all(await uniqueArray.map(async (group: any) => {
  let status
  const year = await groups.filter((year: any) => year.studentscourses.academicYear === group)
  const approved = year.filter((course: any) => course.studentscourses.approvedBy !== null)
  const totalcrPts = await approved.map((item: any) =>
  parseInt(item?.studentscourses?.CrPts)).reduce((prev: number, next: number) => prev + next);
  const totalcredit = await approved.map((item: any) =>
  parseInt(item.Course.credit)).reduce((prev: number, next: number) => prev + next);
  totalPts=totalPts+totalcrPts
  totalcredits=totalcredits+totalcredit
  if(totalcrPts / totalcredit>3){
    status='Honours'
  }
  if(totalcrPts / totalcredit>3.5){
    status='High honours'
  }
  if(totalcrPts / totalcredit>2){
    status='Successful '
  }
  if(totalcrPts / totalcredit<2){
    status='Unsuccessful '
  }
  const data = {
    year: group,
    courses: approved,
    totalcrPts: totalcrPts,
    totalcredit: totalcredit,
    status:status,
    gpa: totalcrPts / totalcredit,
    cgpa: totalPts / totalcredits

```

```
    }
    transcript.push(data)
  )))
  return transcript;
} catch (error) {
  throw error;
}
};

getStudent = async (studentId: number): Promise<any> => {
  try {
    const result = await Student.findByPk(studentId, {
      include: [
        {
          model: Advisormodel,
          as: "advisor"
        },
        {
          model: departmentModel,
          as: "Department"
        },
        {
          model: Group,
          as: "Group",
          include: [{
            model: Courses,
            as: "Course",
          }]
        }
      ]
    });
  }
```

```

    return result;
  } catch (error) {
    throw error;
  }
};

getTimeTable = async (studentId: number, year: string): Promise<any> => {
  try {
    const year = await this.getAcademicYear()
    const result = await Student.findByPk(studentId, {
      include: [
        {
          model: Group,
          as: "Group",
          where: { year: year },
          include: [{
            model: CourseRooms,
            as: "CourseRooms",
          }]
        }
      ]
    });

    const groups = await result?.Group

    const timetable = await Promise.all(this.WEEK_DAYS.map(async (days: string) => {
      let weekDay: any = []
      const data = await groups.filter((f: any) =>
        f.CourseRooms.some((o: any) => days?.includes(o.day))
      )
      if (data.length > 0) {
        const x = await Promise.all(await data.map(async (day: any) => {
          const table = await day.CourseRooms.filter((dayt: any) => dayt.day === days)

```

```

const week = await table.map((time: any) => {
  return {
    name: day.name,
    type: "custom",
    startTime: `2018-02-24T${time.timeStart}`,
    endTime: `2018-02-24T${time.timeEnd}`,
  }
})
return week
}))

```

```

weekDay.push(x)
}
const result = {
  [`${days}`]: weekDay
}
return result
}))

```

```

return timetable;
} catch (error) {
  throw error;
}
};

```

```

getStudentStats = async (studentId: number, departmentId: number): Promise<any> => {
  try {
    const result = await Student.findOne({
      where: {
        id: studentId
      },
    },

```

```

include: [
  {
    model: Group,
    as: "Group",
    include: [{
      model: Courses,
      as: "Course",
    }]
  }
]
});

const allCourses = await Courses.findAll({
  where: {
    departmentId: {
      [Op.or]: [departmentId, 4]
    }
  }
})

const stats = await result?.Group.filter((course: any) => course.studentscourses.grade !== null
&& course.studentscourses.grade !== 'FF')

const totalDepartmentcredit = await allCourses.map((item: any) =>
parseInt(item.credit)).reduce((prev: number, next: number) => prev + next);

const totalcredit = await stats.map((item: any) => parseInt(item.Course.credit)).reduce((prev:
number, next: number) => prev + next);

const data = {
  courses: allCourses.length,
  coursesTaken: stats.length,
  credit: totalDepartmentcredit,
  creditTaken: totalcredit
}

return data;

```

```
    } catch (error) {
      throw error;
    }
  };
  getStudentByAdvisor = async (advisorId: number): Promise<any> => {
    try {
      const result = await Student.findAll({
        where: {
          advisorId: advisorId
        },
        include: [
          {
            model: Advisormodel,
            as: "advisor"
          },
          {
            model: Group,
            as: "Group",
            include: [{
              model: Courses,
              as: "Course",
            }]
          }
        ]
      });
      return result;
    } catch (error) {
      throw error;
    }
  };
};
```



```
CalculateCrPoints = async (grade: string, credit: number) => {  
  let points: number | undefined;  
  switch (grade) {  
    case "AA":  
      points = 4 * credit  
      break;  
    case "BA":  
      points = 3.5 * credit  
      break;  
    case "BB":  
      points = 3 * credit  
      break;  
    case "CB":  
      points = 2.5 * credit  
      break;  
    case "CC":  
      points = 2 * credit  
      break;  
    case "DC":  
      points = 1.5 * credit  
      break;  
    case "DD":  
      points = 1 * credit  
      break;  
    case "FD":  
      points = 0.5 * credit  
      break;  
    case "FF":  
      points = 0 * credit  
      break;
```

```

    }
    return points
  }
// Update Student
updateStudent = async (
  studentId: number,
  data: any,
): Promise<any> => {
  try {
    await Student.update(
      { ...data },
      { where: { id: studentId } }
    );
    const department = await this.getStudent(studentId);
    console.log(data.englishScore)
    if(data.englishScore<60){
      await Student.update({ departmentId:5},{ where: { id:studentId } })
    }
    return department;
  } catch (error) {
    throw error;
  }
};

addRemoveCourse = async (
  studentId: number,
  data: any,
  status: string,
  name: string
) => {
  try {

```

```

let error: string | undefined;

const student = await this.getStudent(studentId);

if (data.type === "add") {
  await Promise.all(await data.courses.map(async (course: number) => {
    const group = await GroupServices.getGroup(course)
    const prerequisites = group?.Course?.prerequisites
    if (prerequisites) {
      const allCode = student?.Group.map((group: any) => {
        if (group?.studentscourses.grade && group?.studentscourses.grade !== 'FF') {
          return (group?.Course?.code)
        }
      })
      if (allCode.includes(prerequisites)) {
        if (status === 'Student') {
          await StudentCourses.create({ studentId: studentId, courseGroupId: course, academicYear:
data?.year })
        } else {
          await StudentCourses.create({ studentId: studentId, courseGroupId: course, academicYear:
data?.year, approvedBy: name })
        }
      } else {
        error = `student did not take the required prerequisites for ${group.Course.name}`
      }
    } else {
      if (status === 'Student') {
        await StudentCourses.create({ studentId: studentId, courseGroupId: course, academicYear:
data?.year })
      } else {
        await StudentCourses.create({ studentId: studentId, courseGroupId: course, academicYear:
data?.year, approvedBy: name })
      }
    }
  })
}

```

```

    )))
  }
  if(data.type=== "add"){
    if (status === 'Student') {
      await Notification.create({ content: `${student.name} courses are waiting for your approval`,
receiver: student.advisorId, type: "normal" })
    }else{
      await Notification.create({ content: `Your courses has been approved`, receiver:
student.userId, type: "normal" })
    }
  }
  if (data.type == "remove") {
    await data.courses.map(async (course: number) => {
      await StudentCourses.destroy({ where: { studentId: studentId, courseGroupId: course, grade:
null } })
    })
  }
  if (error) {
    throw Error(error)
  }
  return student;
} catch (error) {
  throw error;
}
}

```

```

updateGrade = async (
  studentId: number,
  courseId: number,
  data: any

```

```

): Promise<any> => {
  try {
    const course = await Group.findByPk(courseId, {
      include:[
        {
          model:Courses,
          as:'Course'
        }
      ]
    })
    const points = await this.CalculateCrPoints(data.grade, course.Course.credit)
    await StudentCourses.update(
      {
        grade: data.grade,
        midtermOne:data.midtermOne,
        midtermTwo:data.midtermTwo,
        final:data.final,
        CrPts: points
      },
      { where: { studentId: studentId, courseGroupId: courseId } }
    );
    const student = await this.getStudent(studentId);
    await Notification.create({ content: `${course.Course.name} grades has been uploaded`,
receiver: student.userId, type: "normal" })
    return student;
  } catch (error) {
    throw error;
  }
};

getCourseApproval = async (
  studentId: number,

```

```

): Promise<any> => {
  try {
    const result = await Student.findOne({
      where: {
        userId: studentId,
      },
      include: [
        {
          model: Group,
          as: "Group",
          include: [{
            model: Courses,
            as: "Course",
          }],
        }
      ]
    })
    const group = result.Group
    const notApproved = group.filter((course: any) => course.studentscourses.approvedBy === null)
    return notApproved;
  } catch (error) {
    throw error;
  }
};

approveCourse = async (
  studentId: number,
  courseId: number,
  name: string
): Promise<any> => {
  try {

```

```

await StudentCourses.update(
  {
    approvedBy: name
  },
  { where: { studentId: studentId, courseGroupId: courseId, approvedBy: null } }
);

const student = await this.getStudent(studentId);
return student;
} catch (error) {
  throw error;
}
};

AutomateSelection = async (
  studentId: number,
  year: string
): Promise<any> => {
  try {
    const student = await this.getStudent(studentId);
    const department = await departmentModel.findByPk(student.departmentId);
    const coursesTaken = await student.Group.filter((course: any) => course.studentscourses?.grade
    !== null &&
      course.studentscourses?.grade !== "FF" || course.studentscourses?.academicYear !== year
    ).map((course: any) => {
      return course?.Course.code
    })
    // get offered courses
    const allGroup = await Group.findAll({
      where: {
        year: year
      },
      include: [

```

```

{
  model: Courses,
  as: "Course",
  where: {
    [Op.and]:{
      departmentId: {
        [Op.or]: [student.departmentId,4]
      },
      facultyId: department.facultyId
    }
  }
}
]
})

```

```

if(allGroup.length===0){

```

```

  const allCourses= await Courses.findAll({

```

```

    where: {

```

```

      [Op.and]:{

```

```

        departmentId: {

```

```

          [Op.or]: [student.departmentId,4]

```

```

        },

```

```

        facultyId: department.facultyId

```

```

      }

```

```

    }

```

```

  })

```

```

  //remove courses which are taken

```

```

  const remove = await allCourses.filter((course: any) => !coursesTaken.includes(course.code))

```

```

  const prerequisites = await remove.filter((course: any) =>

```

```

  coursesTaken.includes(course.prerequisites) || course.prerequisites === null)

```



```

const totalcredit = await prerequisites.map((item: any) => parseInt(item.credit)).reduce((prev:
number, next: number) => prev + next);

const prerequisitesId = prerequisites.map((group:any)=> group.id)

const result = await allCourses.filter((course: any) => prerequisitesId.includes(course.id))

let credits: number = 0

const creditLimit = year.includes('Summer')? 12:21

const automation = await result.map((courses: any) => {

  if (totalcredit < creditLimit) {

    return courses

  } else {

    credits = credits + courses.credit

    if (credits <= creditLimit) {

      return courses

    }

  }

})

const removeNull = await automation.filter((Course: any) => Course !== undefined)

return removeNull

}else{

  const CoursesOffered = await allGroup?.map((course: any) => course.Course)

  CoursesOffered.sort((a: any, b: any) => parseFloat(a.semester) - parseFloat(b.semester));

  //remove repeated courses

  const filtered = CoursesOffered.filter((v: any, i: any, a: any) => a.findIndex((t: any) => (t.id
=== v.id)) === i)

  //remove courses which are taken

  const remove = await filtered.filter((course: any) => !coursesTaken.includes(course.code))

  //check if prerequisites is done

  const prerequisites = await remove.filter((course: any) =>
coursesTaken.includes(course.prerequisites) || course.prerequisites === null)

  const totalcredit = await prerequisites.map((item: any) => parseInt(item.credit)).reduce((prev:
number, next: number) => prev + next);

```

```

const prerequisitesId = prerequisites.map((group:any)=> group.id)
const result = await allGroup.filter((course: any) => prerequisitesId.includes(course.Course.id))
let credits: number = 0
const automation = await result.map((courses: any) => {
  if (totalcredit < 21) {
    return courses
  } else {
    credits = credits + courses.credit
    if (credits < 21 && credits < 19) {
      return courses
    }
  }
})
const removeNull = await automation.filter((Course: any) => Course !== undefined)
return result;
}
} catch (error) {
  throw error;
}
};
// Delete Student
deleteStudent = async (
  studentId: number,
): Promise<any> => {
  try {
    const department = await Student.findOne({
      where: {
        id: studentId,
      },
      paranoid: false,

```

```

});
department.destroy();
return { message: "Advisor record deleted!" };
} catch (error) {
  throw error
}
};

getStudentAnnouncements = async (studentId:number): Promise<any> => {
  try {
    const studentGroup = await StudentCourses.findAll({
      studentId:studentId
    })
    let groupIds =await studentGroup.map((student:any)=>student.groupId)
    groupIds.push(null)
    const announcements = Announcementmodel.findAll({
      where:{
        groupIdId: {
          [Op.or]: groupIds
        },
      },
      include: [
        {
          model: Group,
          as: "Group"
        },
      ],
    ])
    return announcements;
  } catch (error) {
    throw error;
  }
}

```

```

    }
};
}
const user = require("./model");
const bcrypt = require("bcrypt");
const jwt = require("jsonwebtoken");
const Student = require("../student/Model");
const dayjs = require('dayjs')
const chairman = require("../chairman/model");
const Advisor = require("../advisor/model");
const faculty = require("../faculties/model");
const Department = require("../department/model");
import firestoreService from '../firestore/firebase'
const { Op } = require("sequelize");
export default class AuthService {
  constructor() { }
  /**
   * Authenticate user via form input
   * @param data { companyEmail:string, password:string }
   */

  private getAcademicYear = async() => {
    // let year: string =
    // const month = dayjs().month()
    // const currentyear = dayjs().year()
    // if (month >= 1 && month <= 5) {
    //   year = `${currentyear-1}-${currentyear} - Spring`
    // }
    // if (month > 5 && month <= 8) {
    //   year = `${currentyear-1}-${currentyear} - Summer`

```

```

// }
// if (month > 8 || month < 1) {
//   year = `${currentyear}-${currentyear + 1} - Fall`
// }

const year = await firestoreService.get(
  'academic',
  'qYX8QXS3XW564eKdfPTP'
)

return year.data.year

}

async loginViaForm(data: any) {
  let status

  try {
    const { username, password } = data;
    const users = await user.findOne({
      where: { userName: username },
      include: [
        {
          model: Student,
          as: "Student",
          include: [
            {
              model: Advisor,
              as: "advisor",
            },
            {
              model: Department,
              as: "Department",
              include: [

```

```
{
  model: faculty,
  as: "Faculty"
},
]
},
]],
{
  model: chairman,
  as: "chairman",
  include: [
    {
      model: Department,
      as: "Department",
      include: [
        {
          model: faculty,
          as: "Faculty"
        },
      ]
    },
  ],
}
],
{
  model: Advisor,
  as: "Advisor",
  include: [
    {
      model: Department,
      as: "Department",
```

```
    include: [  
      {  
        model: faculty,  
        as: "Faculty"  
      },  
    ],  
  ],  
},  
]  
},  
]  
});
```

```
if (!users) throw Error("Invalid Credentials");  
const comparePassword = await this.comparePassword(  
  password,  
  users.password  
);  
if (!comparePassword) {  
  throw Error("Invalid Credentials");  
}  
if (users.userAdvisor) {  
  status = "Advisor"  
}  
if (users.userStudent) {  
  status = "Student"  
}  
if (users.userChairman) {  
  status = "Chairman"  
}  
if (users.userSuperAdmin) {
```

```

    status = "SuperAdmin"
  }

  const token = await this.generateToken({
    userId: users.Advisor?.userId || users.Student?.userId || users.Chairman?.userId ||
users.userSuperAdmin,
    surname: users.Advisor?.surname || users.Student?.surname || users.Chairman?.surname ||
users.userSuperAdmin,
    department: users.Advisor?.Department || users.Student?.advisor.Department ||
users.Chairman?.Department || users.userSuperAdmin,
    status: status,
  });

  const result = {
    token,
    user: {
      userId: users.Advisor?.userId || users.Student?.userId || users.chairman?.userId ||
users.userSuperAdmin,
      Id: users.Advisor?.id || users.Student?.id || users.chairman?.id || users.userSuperAdmin,
      username: users.userName,
      name: users.Advisor?.name || users.Student?.name || users.chairman?.name || users.name,
      surname: users.Advisor?.surname || users.Student?.surname || users.chairman?.surname ||
users.userSuperAdmin,
      department: users.Advisor?.Department || users.Student?.Department ||
users.chairman?.Department || users.userSuperAdmin,
      faculty: users.Advisor?.Department?.Faculty || users.Student?.Department.Faculty ||
users.chairman?.Department.Faculty || users.userSuperAdmin,
      status: status,
    },
  };

  return result;
} catch (error) {
  throw (error);
}

```



```

}
protected async comparePassword(
  input: string,
  compare: string
): Promise<boolean> {
  const match = await bcrypt.compare(input, compare);
  return match;
}
/**
 * verify user
 * @param param
 */
async verifyUser(userName: string) {
  try {
    let status:string | undefined;
    const users = await user.findOne({
      where: {
        userName:userName
      },
      include: [
        {
          model: Student,
          as: "Student",
          include: [
            {
              model: Advisor,
              as: "advisor",
            },
            {
              model: Department,

```

```
    as: "Department",
    include: [
      {
        model: faculty,
        as: "Faculty"
      },
    ]
  },
]},
{
  model: chairman,
  as: "chairman",
  include: [
    {
      model: Department,
      as: "Department",
      include: [
        {
          model: faculty,
          as: "Faculty"
        },
      ]
    },
  ]
},
]
},
{
  model: Advisor,
  as: "Advisor",
  include: [
    {
```

```
    model: Department,
    as: "Department",
    include: [
      {
        model: faculty,
        as: "Faculty"
      },
    ]
  },
]
},
]
})
```

```
if (!users) throw Error("User Does not Exist!");
if (users.userAdvisor) {
  status = "Advisor"
}
if (users.userStudent) {
  status = "Student"
}
if (users.userChairman) {
  status = "Chairman"
}
if (users.userSuperAdmin) {
  status = "SuperAdmin"
}
// If user exist and activated
// Generate jwt token
const token = await this.generateToken({
```

```

    user: {
        userId: users.Advisor?.userId || users.Student?.userId || users.Chairman?.userId ||
users.userSuperAdmin,
        surname: users.Advisor?.surname || users.Student?.surname || users.Chairman?.surname ||
users.userSuperAdmin,
        department: users.Advisor?.Department || users.Student?.advisor.Department ||
users.Chairman?.Department || users.userSuperAdmin,
        status: status,
    },
});
const result = {
    token,
    user: {
        userId: users.Advisor?.id || users.Student?.userId || users.chairman?.userId ||
users.userSuperAdmin,
        Id: users.Advisor?.id || users.Student?.id || users.chairman?.id || users.userSuperAdmin,
        username: users.userName,
        name: users.Advisor?.name || users.Student?.name || users.chairman?.name || users.name,
        surname: users.Advisor?.surname || users.Student?.surname || users.chairman?.surname ||
users.userSuperAdmin,
        department: users.Advisor?.Department || users.Student?.Department ||
users.chairman?.Department || users.userSuperAdmin,
        faculty: users.Advisor?.Department?.Faculty || users.Student?.Department.Faculty ||
users.chairman?.Department.Faculty || users.userSuperAdmin,
        year:await this.getAcademicYear(),
        status: status,
    },
};
return result;
} catch (error) {
    throw (error);
}
}


```

```
// Generate jwt token
protected generateToken = async (data: any): Promise<any> => {
  try {
    const token = await jwt.sign(data, "neuAdvisor", {
      expiresIn: 36000,
    });
    return token;
  } catch (error) {
    throw (error);
  }
};
}
```

APPENDIX 2

SIMILARITY REPORT

Auwali Saleh Mubarak | User Info | Messages (2 new) | Instructor | English (International) | Community | Help | Logout



















All Classes | Join Account (TA) | Quick Submit

NOW VIEWING: HOME > QUICK SUBMIT

About this page
This is your assignment inbox. To view a paper, select the paper's title. To view a Similarity Report, select the paper's Similarity Report icon in the similarity column. A ghosted icon indicates that the Similarity Report has not yet been generated.

Yakın Doğu Üniversitesi
QUICK SUBMIT | NOW VIEWING: NEW PAPERS

Submit

<input type="checkbox"/>	AUTHOR	TITLE	SIMILARITY		FILE	PAPER ID	DATE
<input type="checkbox"/>	Siddiqui Abdul_rehma...	CHAPTER 4	0% 			1880415308	09-Aug-2022
<input type="checkbox"/>	Siddiqui Abdul_rehma...	CHAPTER 5	0% 			1880415312	09-Aug-2022
<input type="checkbox"/>	Siddiqui1 Abdul_rehm...	BSTTRACT	0% 			1880695880	09-Aug-2022
<input type="checkbox"/>	Siddiqui Abdul_rehma...	CONCLUSION	0% 			1880415318	09-Aug-2022
<input type="checkbox"/>	Siddiqui Abdul_rehma...	CHAPTER 3	5% 			1880415304	09-Aug-2022
<input type="checkbox"/>	Siddiqui Abdul_rehma...	CHAPTER 1	11% 			1880415296	09-Aug-2022
<input type="checkbox"/>	Siddiqui3 Abdul_rehm...	THESIS	12% 			1880800053	10-Aug-2022
<input type="checkbox"/>	Siddiqui2 Abdul_rehm...	CHAPTER 2	14% 			1880796583	10-Aug-2022

APPENDIX 3 CURRICULUM VITAE

CURRICULUM VITAE

Abdul Rehman Siddiqui in 2020, he received a bachelor's degree in Computer Engineering from Near East University. Abdul Rehman Siddiqui is a master's student at Near East University studying Computer Engineering with a specialization of Artificial Intelligence.

PERSONAL DATA

Name: Abdul Rehman Siddiqui

Date of Birth: 12th April, 1996

Nationality: Pakistani

Place of Birth: Dubai, UAE.

Permanent Address:

Current Address:

Mobile Number: +90 548 8584 254

Marital Status: Single

Email: abdulrehmansiddiqui123@gmail.com

QUALIFICATIONS

Near East University

Bachelors of Computer Engineering

Dates 2016-2020

WORK EXPERIENCE

GUNSEL, Nicosia— Backend Team Leader December 2020 to Present

Implemented and updated application modules under the direction of Senior Software Developers.

Successfully worked at an independent level, while also serving as an effective and enthusiastic collaborator. Currently developing a system for HR and Production using React.js, node.js, Redux, Redis.

SKILLS

React

Node.js (redis, express, socket.io)

MySQL

Nest

Typescript/Javascript

Python

Apriori

Linear Regression

Support Vector Machine