HÜSEYİN UZUNKOŞAR

GEOGRAPHIC MAPPING OF GROUND WATER IN NORTHERN CYPRUS

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

By
HÜSEYİN UZUNKOŞAR

In Partial Fulfillment of the Requirements for
The Degree of Master of Sciene
in
Biomedical Engineering

GEOGRAPHIC MAPPING OF GROUND WATER IN NORTHERN CYPRUS

NEU
2018

NICOSIA, 2018

GEOGRAPHIC MAPPING OF GROUND WATER IN NORTHERN CYPRUS

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

By
HÜSEYİN UZUNKOŞAR

In Partial Fulfillment of the Requirements for The Degree of Master of Sciene in Biomedical Engineering

NICOSIA, 2018

Hüseyin UZUNKOSAR: GEOGRAPHIC MAPPING OF GROUND WATER IN NORTHERN CYPRUS

Approval of Director of Graduate School of Applied Sciences

Prof. Dr. Nadire CAVUS

We certify this thesis is satisfactory for the award of the degree of Master of Science in Biomedical Engineering

Examining Committee in Charge:

Assoc. Prof. Dr. Rasime Kalkan	Committee Chairman, Department of Medical Genetics, NEU
Assoc. Prof. Dr. Terin Adalı	Supervisor, Department of Biomedical Engineering, NEU
Assist. Prof. Dr. Melis Sümengen Özdenefe	Department of Biomedical 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: Hüseyin Uzunkoşar

Signature:

Date:

ACKNOWLEDGEMENTS

First and foremost, I would like to sincerely thank Assoc. Prof. Dr. Terin Adalı for being my advisor for this thesis and thank for helping me find the thesis topic and for always supporting me. I am so proud of being her master student. Under her guidance, I successfully overcome many difficulties. In each discussion, she answered my questions patiently and she helped me a lot during the progress of the thesis.

To all my colleagues and friends of whom I will mention few, Nasire Uluç, Hasan Özyapıcı and everyone that has gone beyond limit to support me in every way possible to seeing that this work is a reality, I want to say a big thank for.

Finally, I would like to thank my family who supported me both materially and morally in every step of my education.

To my parents...

ABSTRACT

In Northern Cyprus,various kind of pesticide residue analysis has been performed to specify the state of ground water sources. As the results of these analysis, too much data was collected. By applying this project, pesticide residue analysis results was stored in a specially designed database by using the specially designed user-friendly graphical interface.

The database of developed software was designed for the analyze results of pesticide residues in ground water of Northern Cyprus and the analyze results were collected under one schema. The tables of database were obtained by using Microsoft SQL Server 2008. SQL Server was used to obtain system stability and easy-use.

Visual Studio 2010 has been used to develop the graphical user interface and the data access application programming interface. Labels, text boxes, buttons, a tab control, a data grid view, a gmap control and their attributes were used to develop a user friendly graphical user interface. LINQ technology were used to set up connection between database and graphical user interface.

GMAP is the external application reference tool which contains the core of global mapping systems, such as Google Maps, Bing Maps, ArcGis and Yandex Maps. In addtion, GMAP has been used to mark groundwater sources.

The results reveal that; the input data can be easily done and also provides to enter different kind of data beside the input data of the underground resources. In addition, it has been observed that, these data have been processed rapidly. Moreover, usage tests showed that the stability and reliability of the system has been at the highest level.

Keywords: GMap, Microsoft; Sql Server; Linq; North Cyprus; Groundwater

ÖZET

Kuzey Kıbrıs'ta yeraltı kaynaklarının durumunu belirlemek için bir çok pestisit analizi yapılmaktadır.Yapılan analizler sonucunda bir çok veri elde edilmiştir. Bu projenin gerçekleştirilmesiyle birlikte elde edilen veriler, özel olarak tasarlanmış veritabanına, yine özel olarak tasarlanmış kullanıcı dostu grafik arayüzü ile kaydedilmişlerdir.

Geliştirilmiş yazılımın veritabanı, Kuzey Kıbrıs yeraltı kaynakları pestisit kalıntı analiz raporu sonuçları için tasarlanmıştır ve analiz sonuçları bir şema altında toplanmıştır. Veritabanı tabloları Microsoft SQL Server 2008 kullanılarak elde edilmiştir. SQL Server sistem kararlılığını ve kolay kullanımı sağlamak için kullanılmıştır.

Visual Studio 2010, grafik kullanıcı arayüzü ve veri erişim uygulama programlama arabirimini geliştirmek için kullanılmıştır. Kullanıcı dostu grafik arayüzü geliştirmek için Labels, text boxes, buttons, tab control, data grid view, gmap control araçları ve bu araçların kendilerine has özellikleri kullanılmıştır. LINQ teknolojisi veritabanı ile kullanıcı arayüz birimi arasındaki bağlantıyı kurmak için kullanılmıştır.

GMAP, Google Haritalar, Bing Haritalar, ArcGis ve Yandex Haritalar gibi küresel haritalama sistemlerinin çekirdeklerini içeren bir dış kaynak uygulama referans aracıdır.
Sonuçlar, verilerin girişinin kolaylıkla yapılabildiği ve bu verileri işlerken kullanıcısına hız kazandırdığı gözlemlenmiştir. Buna ek olarak, sadece yeraltı kaynaklarının durumu ile ilgili verilerinin girişinin yanında farklı verilerinde girişinin yapılabildiğini kanıtlamıştır. Ayrıca, Kullanım testleri ile sistemin kararlılığının ve güvenilirliğinin en üst düzeyde olduğu gözlemlenmiştir.

Anahtar Kelimeler: Gmap; Microsoft; Sql Server; Kuzey Kıbrıs; Yeraltı Kaynakları

TABLE OF CONTENTS

CHAPTER 3: RESULTS AMD DISCUSSIONS

CHAPTER 4: CONCLUSION

APPENDICES

LIST OF FIGURES

LIST OF ABBREVATIONS

.NET:	Dot Net
C++:	C Plus Plus
C#:	C Sharp
CPU:	Centrap Processing Unit
DB:	Database
F#:	Fortran Sharp
GIS:	Geographical Information System
JDBC:	Java Database Connectivity
Linq:	Language Integrated Query
ODBC:	Open Database Connectivity
RAM:	Random Access Memory
SQL:	Structed Query Langueage
T-SQL:	Transaction Structed Query Language
VGA:	Video Graphics Array

CHAPTER ONE

THE UNDERGROUND WATER DATABASES

1.1 Underground Water of Northern Cyprus

The underground water of Northern Cyprus has become very important for all people who has lived in Northern Cyprus, because underground water has been used for agriculture, farming, and municipal water for people. The incremental usage of underground water and global climate change has been increasing the important of underground water of Northern Cyprus. Beside high usage of underground water, global pollution has affected the quality of underground water. Chemical precautions for agriculture, and cities, factory wastes, petroleum wastes and etc. are the most popular examples of global pollution (Ajit et al., 2012).

1.2 Underground Water Analysis

From the starting time of the human race, people tried to learn about how to reach to pure water. When the human population started to expand in size, people noticed that underground water makes people sick and causes to people death since the underground water contains some harmful matters. Then people started to examine the water quality and how to the impurity of water happens (Stiff, 1951).

There are many ways to measure impurity of water. The most popular analysis are Pesticide Residue Analysis, Radiological and Environmental Analysis, Microbiological Analysis, Pharmaceutical and Chemical Analysis. In this project, the speacial software and it's database have been developed for only pesticide residue analysis.

1.2.1 Pesticide Residue Analysis

A pesticide is a chemical material or a mixture of chemical materials used for killing pests. Pesticides are used when organisms are dangerous to plants or to animals. Although there are various kind of pesticides, the most common examples are insecticide, fungicide, herbicide, and nematocide. Unconscious applications or poor quality of pesticide

applications to crops and animals may leave residues in or on food and those specified residues have toxicological significance (Oymen, 2014).

After implementation of pesticides to food crops, some residues may remain in or on them. The levels of these residues in foods are often stipulated by regulatory bodies. In many countries, the level of residues are being supervised by some institutions. Humen, plants, animals are affected from pesticide residues (Randall, 2013).

1.2.2 Radiological Analysis

Radiological analysis employ comprehensive measurement programs designed to present levels of radioactivity in the environment. These analyses provide to determine level of potential contamination in food and feed stuff. The measurement programs including some specific techniques such as gamma spectrometry, alpha spectrometry, alpha scintillation, gas flow proportional counting, and liquid scintillation counting facilities are used to assess environmental radioactivity and food safety via various radio analytical and mass spectrometric methods. (Siegbahn, 1968)

1.2.3 Microbiological Analysis

Scientists developed a broad portfolio of dyes and kits for detecting and identifying microbes and other organisms in the enviroment, and determination of structural composition and integrity. These properties enable to change analytical platform including microplates, fluorescence-based microscopy and cytometry (Madigan et al, 2009).

1.2.4 Pharmaceutical and chemical analysis

Pharmaceutical Analysis determines the safety, purity and quality of chemicals and drugs. Pharmaceutical Analysis is also called as quantitative pharmaceutical chemistry. Pharmaceutical Analysis has both quantitative and qualitative analysis of drugs. Pharmaceutical substances start from bulk drugs to the finished dosage forms. In the modern practice of medicine, the analytical methods are used in the analysis of chemical constituents found in human body whoose altered concentrations during disease states serve as diagnostic aids and also used to analyze the medical agents and their metabolites found in biological system (Dheeravath et al., 2013).

Chemical Analysis provides an expert analysis in all areas of Pharmaceutical testing and analysis such as raw materials, stability testing, related substances, impurities, and active assays for the pharmaceutical industry.

1.2.5 Enviromental analysis

Environmental monitoring is necessary to characterise and monitor the quality of the environment. Enviromental assessments utilize environmental monitoring, in many circumstances in which human activities become a threat for the natural environment. All monitoring techniques have justifications and reasons which are designed to build up the actual status of an environment or to build up trends in environmental parameters. At the end, all case results of monitoring will be reviewed, analyzed statistically, and published (Reeve, 1994).

CHAPTER TWO

MATERIALS AND METHODS

2.1 Software Requirements

Visual Studio 2010, Sql Server 2008, Dot Net Framework 4, GMap, and Linq were used for developing the software. Visual C# was used for developing the Graphical User Interface (GUI) and custom code in Visual Studio 2010. Sql Server 2008 was used to develop the database. Linq was used to connect specially designed GUI and database. Dot Net Framework 4 and GMap were used to provide map and specially designed properties to the project.

2.1.1 Visual Studio 2010

Microsoft Visual Studio was used to develop the project, because Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs for Microsoft Windows operating systems. Visual Studio is also used to develop web sites, web services, and web applications. Visual Studio has Microsoft software development platforms such as Windows Presentation Foundation, Windows Application Program Interface (API), Windows Store, Windows Forms, and Microsoft Silverlight (Randolp et al., 2010).

Visual Studio includes a code editor which support IntelliSense like code refactoring. Visual Studio works with two integrated debugger such as source-level debugger and machine-level debugger. There are also other built-in tools including a forms designer for building web designer, database schema designer, GUI applications, and class designer. Visual Studio has ability to accepts plug-ins that enhance the functionality (Randolp et al., 2010).

Visual Studio supports different programming languages and Visual Studio have built-in languages such as C, C++, C++/CLI (via Visual C++), VB.NET (via Visual Basic .NET), C# (via Visual C#), and F# (as of Visual Studio 2010). Visual Stuido's support for other languages such as Ruby, Python, and M among others has availability with the language services installed separately. It also supports XML/XSLT, HTML/XHTML, JavaScript ,and CSS (Randolp et al., 2010).

2.1.2 Dot Net Framework 4

The Dot Net Framework is an application development platform. This platform provides many services which are developed for building, deploying, running desktop, web, and phone applications and web services (Richter, 2002).

2.1.3 Microsoft Structured Query Language Server 2008

Microsoft Structured Query Langueage (SQL) Server is a relational database management system developed by Microsoft. Microsoft SQL Server's primary function is to store and retrieve data as requested by other software applications. MS SQL can be on the same computer or those running on another computer across a network (including the Internet). There are many versions of Microsoft SQL Server for different audiences and for different workloads. Microsoft SQL Server's primary query languages are Transact SQL (T-SQL) and American National Standarts Institute SQL (ANSI SQL) (Vieria, 2008).

2.1.4 Visual C Sharp (C#)

C# is a programming language that runs on the .NET Framework. C# is simple, powerful, type-safe, and object-oriented. C# helps developer to increase development speed and quality of the product. C# can be used in Visual Studio. Moreover, Visual Studio provides many rich libraries to develop much more powerful programs to users on Windows Operating Systems (Williams, 2002).

2.1.5 Language Integrated Query (LINQ)

Language Integrated Query (LINQ) has magnificient ability to connect database with user program in a short time with rich features that extends powerful query capabilities to the language syntax of C# and Visual Basic. LINQ is the fastest and reliable way to insert, delete, update, and select operation. Visual Studio consists of LINQ provider assemblies enable the use of LINQ with SQL Server databases, .NET Framework collections, ADO.NET Datasets, and Extensible Markup Language (XML) documents. When traditional queries are used, they become slower than LINQ thus, performance problems occur. Becouse of the fact that LINQ checks it's own compile time and gives developer an Intellisense support. In traditional queries, it is necessary to learn different query languages or different

programming languages such as SQL databases, XML documents, various Web services, etc., but in LINQ, it is not necessary to learn all of them. If developer knows C# and Visual Basic, using LINQ will be easier (Rattz, 2007).

2.1.6 GMAP.NET

Gmap is a .Net project that enables maps to reference them in a Visual Studio C# Project. GMap core connects projects to Google Maps, Bing Maps, Yahoo Maps, ArcGis, and etc. It also has compability of storing the pinned points to a database.

2.2 Computer requirements

The hardware requirements for developing and running the specially designed software system are:

- Windows 7,8,8.1,10
- At least 2 GB RAM
- At least 1.6 Ghz Dual Core CPU
- VGA Graphic Cards
- Network Connection Support

2.3 Setting Up Visual Studio 2010

There are some preparations before setting up Visual Studio 2010.

- Remove all pre-release versions of components on your machine.
- Disable Anti-Virus program.
- Update your computer.
- Make sure that your computer’s system is healthy

After preparations, insert Visual Studio 2010 DVD on your DVD/CD ROM. Find setup.exe in your Visual Studio 2010 DVD and double click on to the setup.exe to start Visual Studio 2010 products installation.

Figure 2.1: Click on install microsoft visual studio 2010 button

Select "Install Microsoft Visual Studio 2010" option and wait for loading installation components.

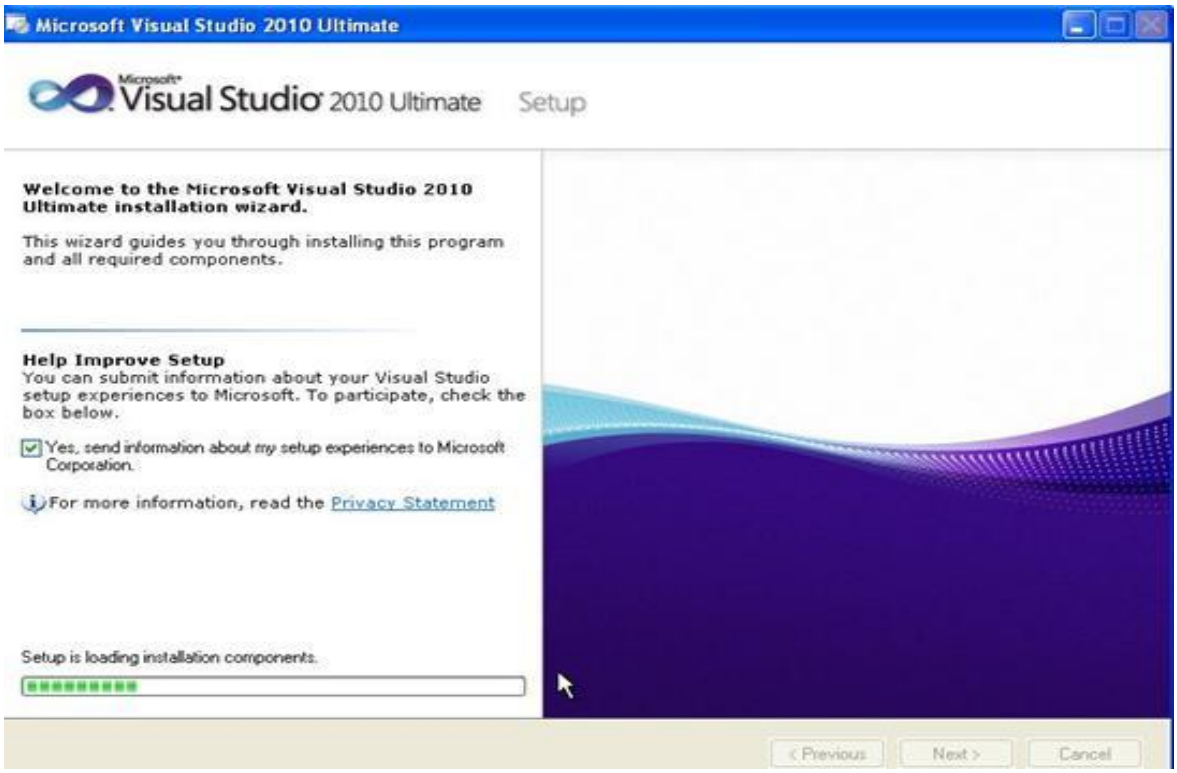

Figure 2.2: Loading setup screen

After the loading is completed, Click Next Button.

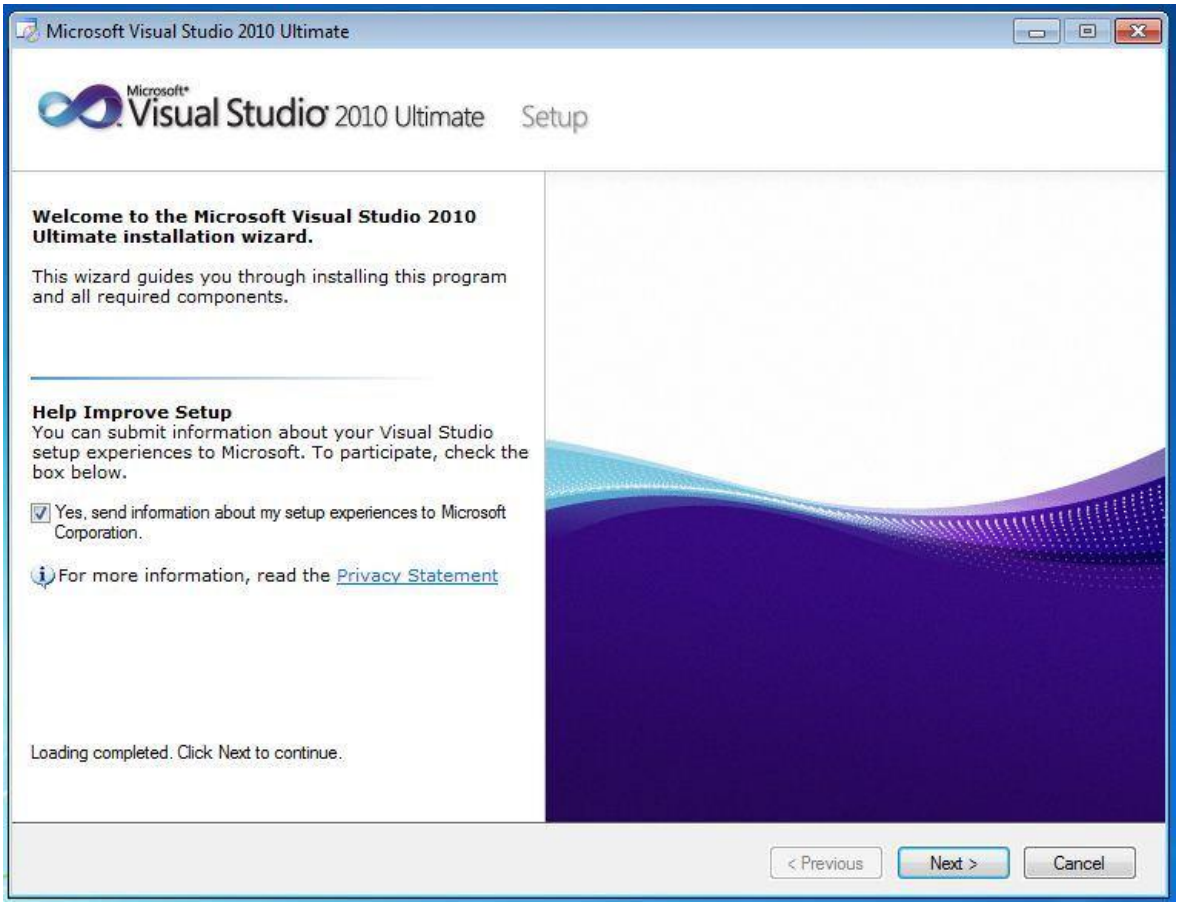

Figure 2.3: Welcome to installation wizard screen

Before clicking Next Button, read licence aggrement carefully. If you accept these license terms, select the option which named “I have read and accept the license terms”. Then click Next Button.

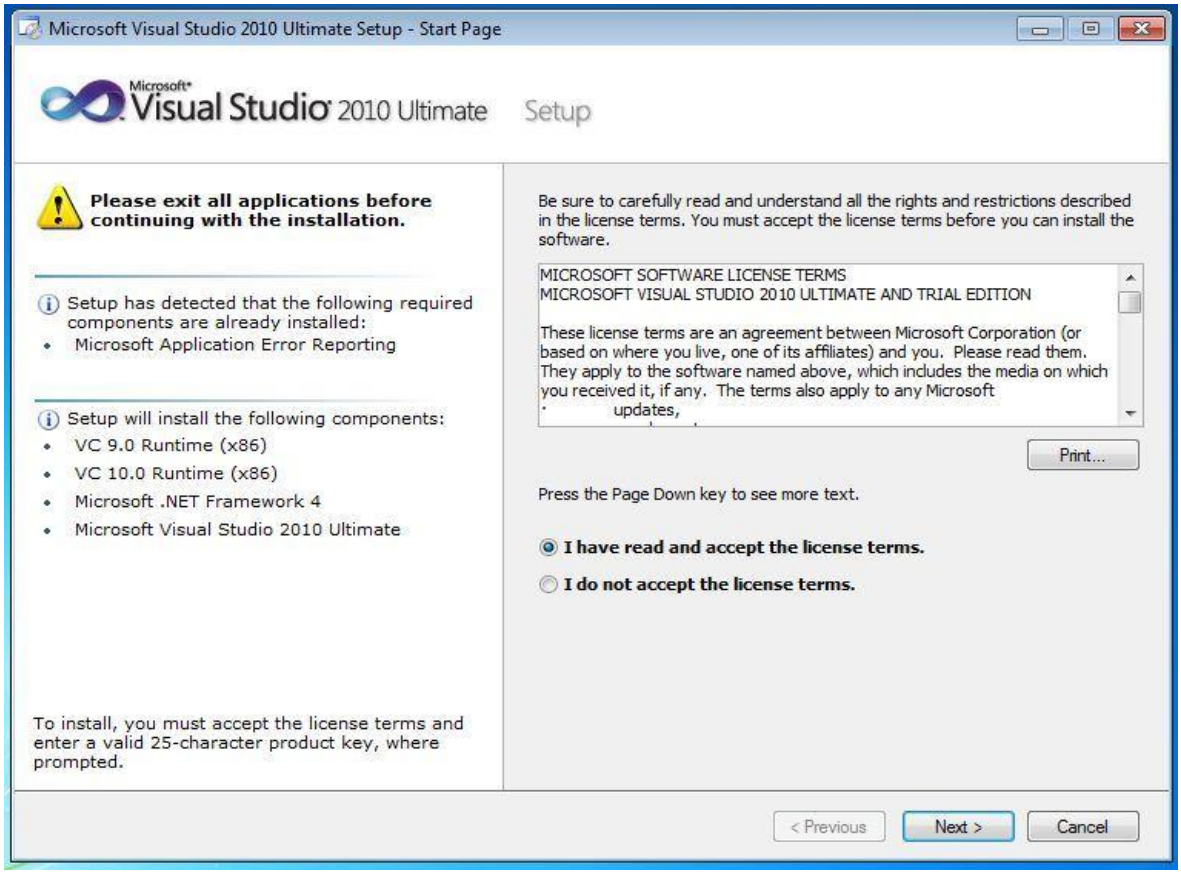

Figure 2.4: Accept licence terms screen

We need all languages and tools in order not to occur an unhandled exception, so select "Full" named option. Then, Select a path where to install the Visual Studio 2010. In addition, There are informations about disk spaces. If there is no problem about installization path, click on Install Button.

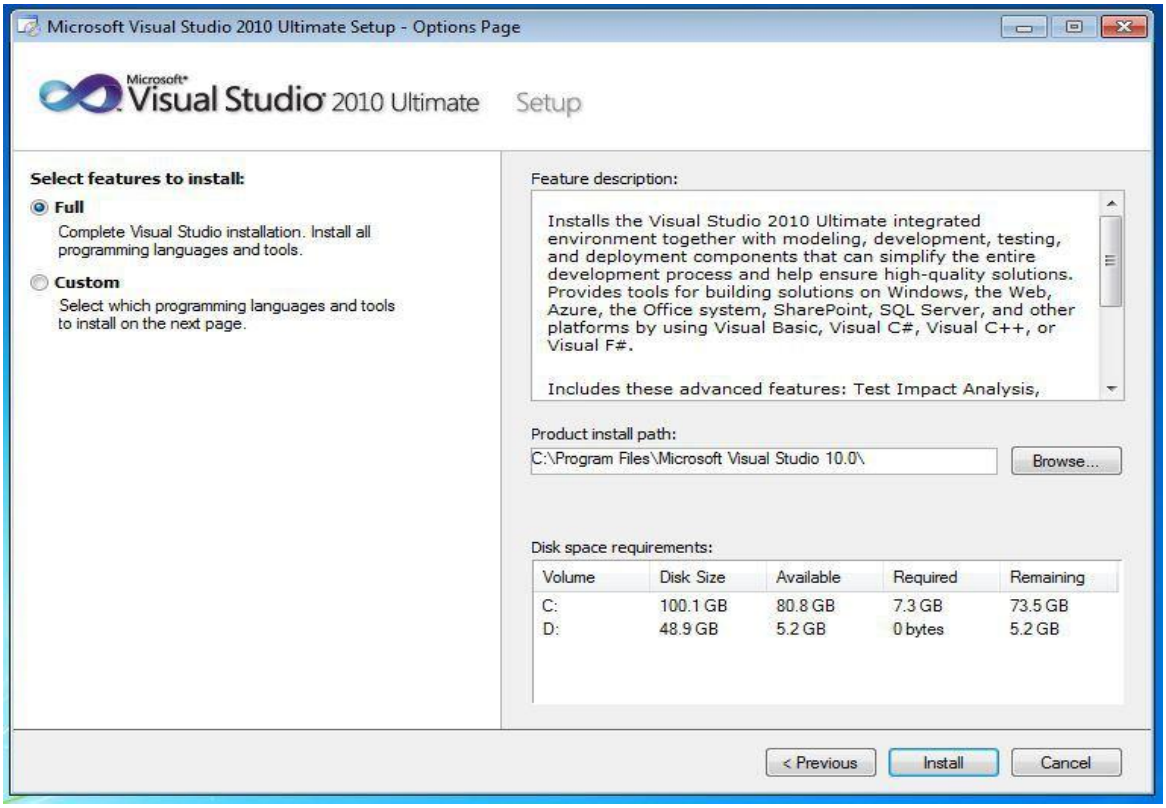

Figure 2.5: Installation features screen

The progress is shown step by step in the picture below.

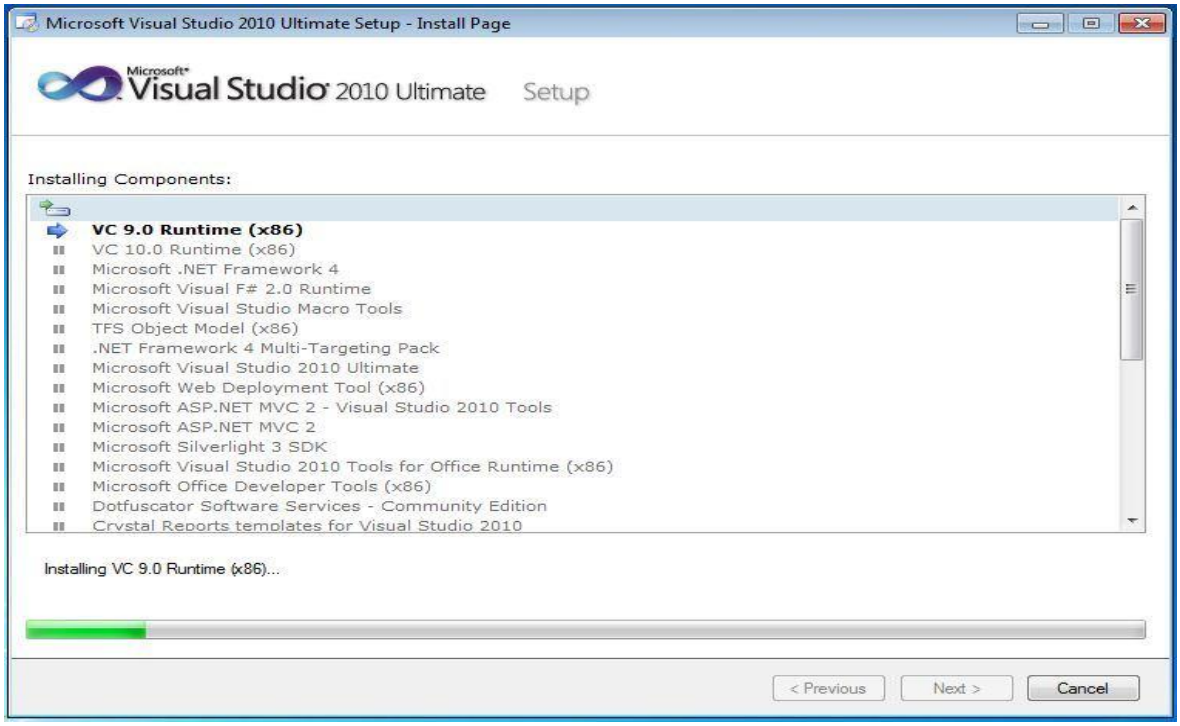

Figure 2.6: Installing components screen

After the installization finishes, this is the first screen that programs shown in below to users for five seconds approximately.

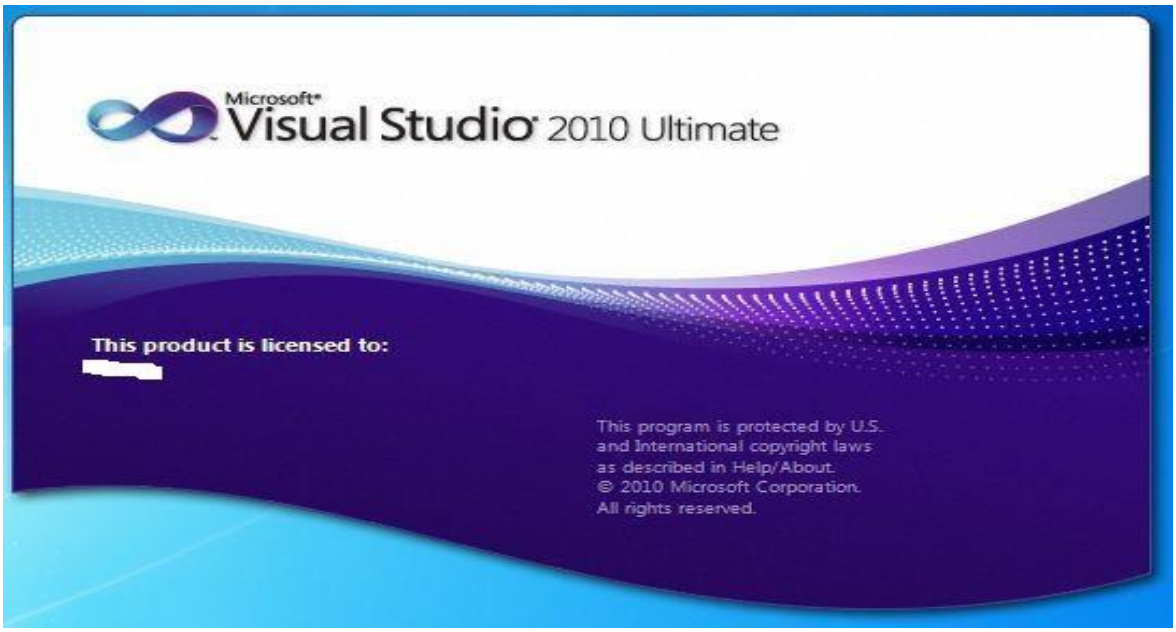

Figure 2.7: First loading page

There is a selection screen about program languages. Most people select Visual C# and Basic Development Settings. After selection is completed, click on Start Visual Studio button.

Figure 2.8: Choose default environment settings screen

2.4 Setting up Microsft Structured Query Language Server 2008

Prepatations:

- .Net Framework 3.5 or above.
- Update your computer.

After finishing the preparations, insert SQL Server 2008 DVD in your computer's DVD-ROM.

Open your SQL Server 2008 DVD and double click on setup.exe file.

Figure 2.9: Click on setup.exe screen

If there is no problem about your computer's system. This window appears with the selection of planning menu. It provides informations and helps to optimize system compatible with SQL Server 2008.

Figure 2.10: Planning screen

Maintenance menu helps to Ugrade your edition, repair your SQL Server and remove a node from your SQL Server.

Figure 2.11: Maintenace screen

Tools menu collects data about all installed SQL Server tools.

Figure 2.12: Tools screen

In Resources menu, users can reach help documents and online resource pages.

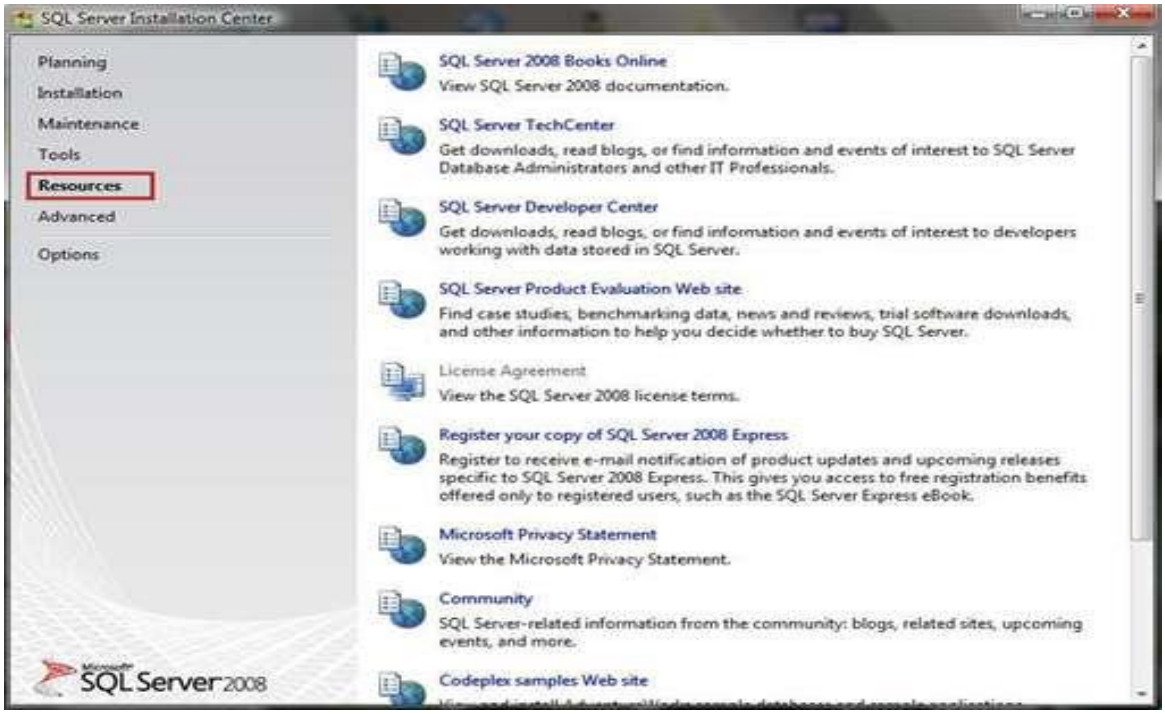

Figure 2.13: Resources screen

In Advanced menu, different kind of installizations can be done by the advanced tools.

Figure 2.14: Advanced menu screen

In Installation menu, new installation, adding features or upgrading old versions can be done.

For the new installation, select the New SQL Server stand-alone installation or add features to an existing istallation link.

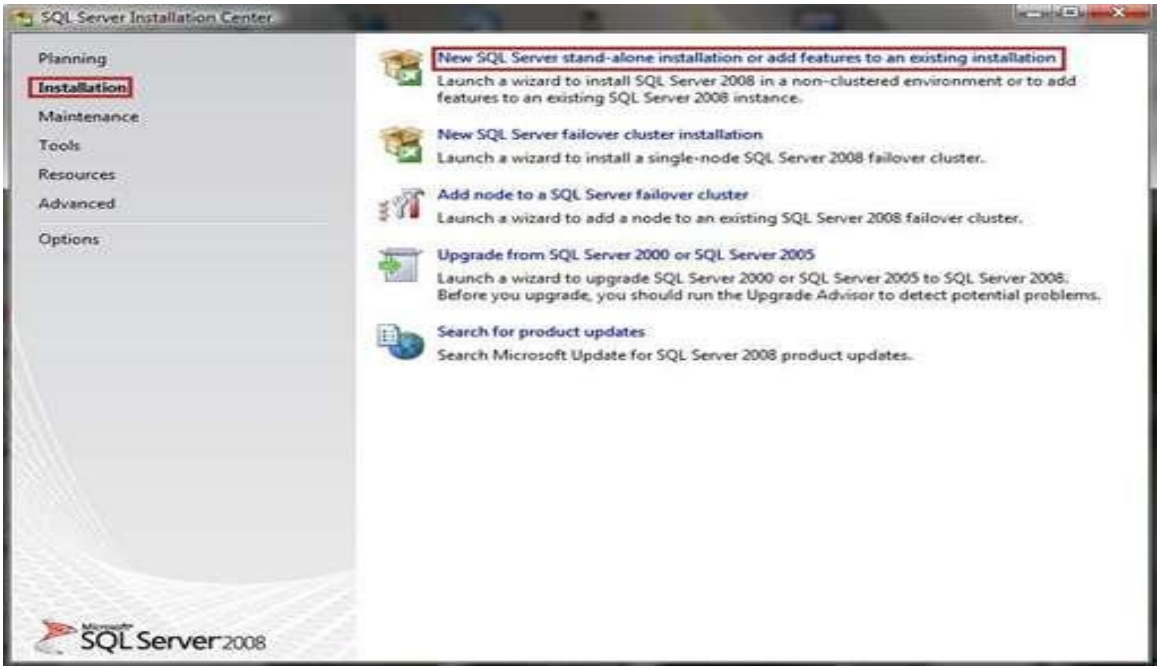

Figure 2.15: Installation screen

In this windows, setup support rules examine the system for errors. If there is no problem, user can click on "Ok" button. If there are problems, user can not continue to next windows.

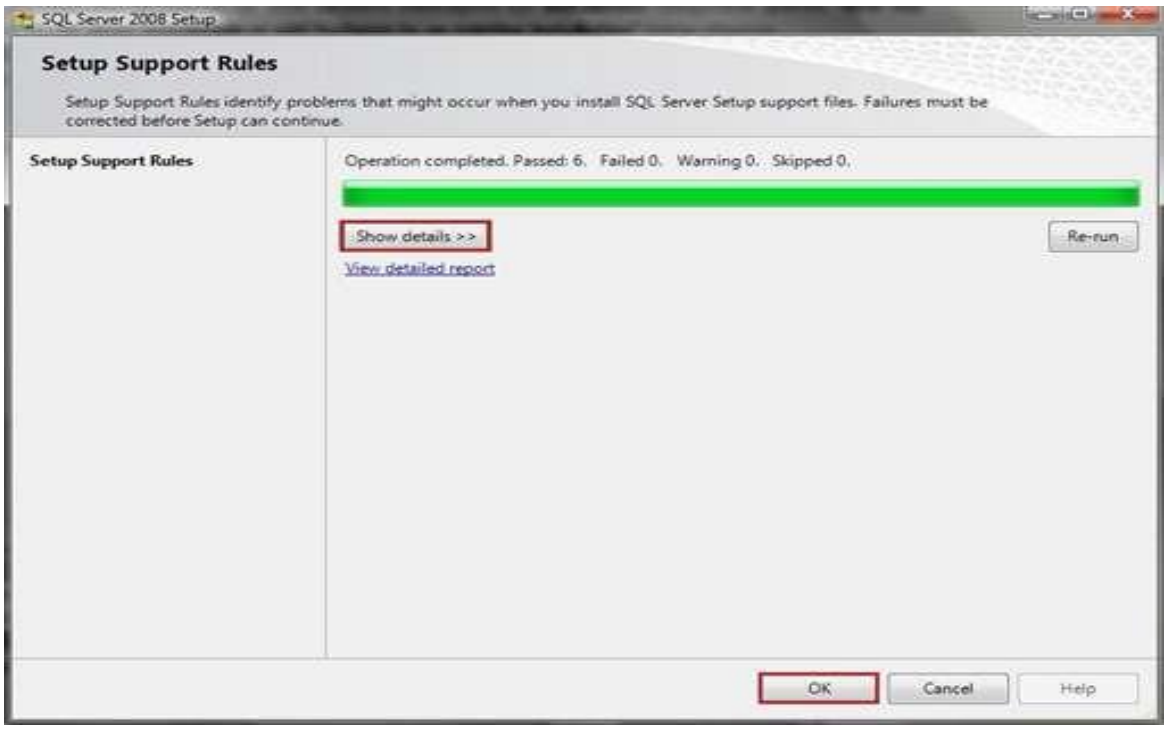

Figure 2.16: Setup support rules

By clicking on Show details button, details can be shown below.

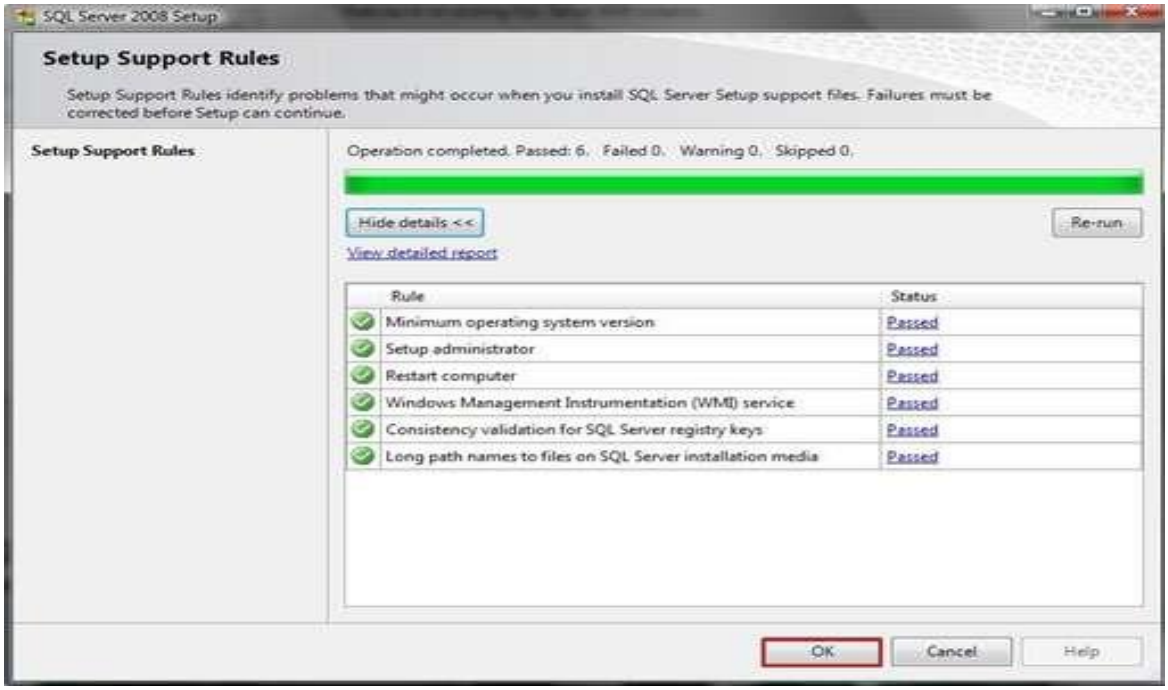

Figure 2.17: Setup support rules detailes

There are two option for users. If the first option is selected, free editions can be selected and they can be installed. If the users has a product key, the users must select second option to enter key and install the full version of SQL Server 2008. Then click on "Next" button to continue.

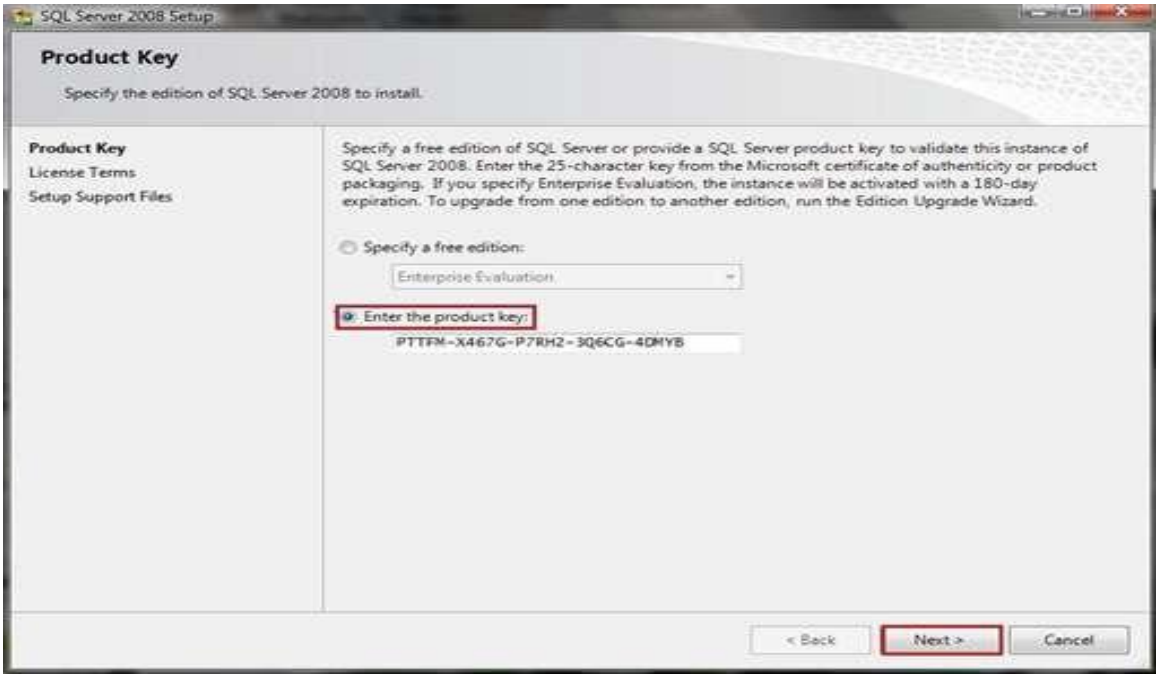

Figure 2.18: Product key

Accept the license terms and click on Next Button.

Figure 2.19: Licence terms

Setup support files can be installed by clicking on Install button.

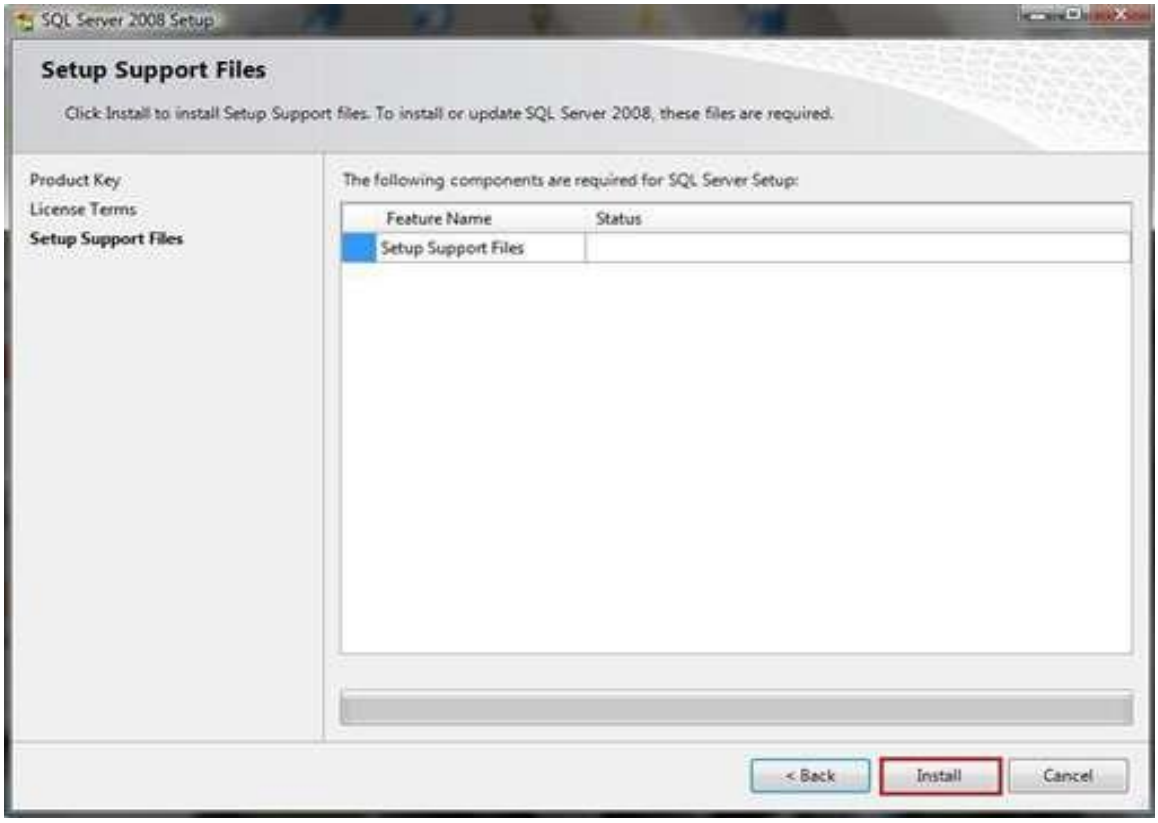

Figure 2.20: Setup support files

The Setup Support Rules identify if there is a problem or not. Windows Firewall has a warning becouse of its configuration. But it doesn't affect the installation.

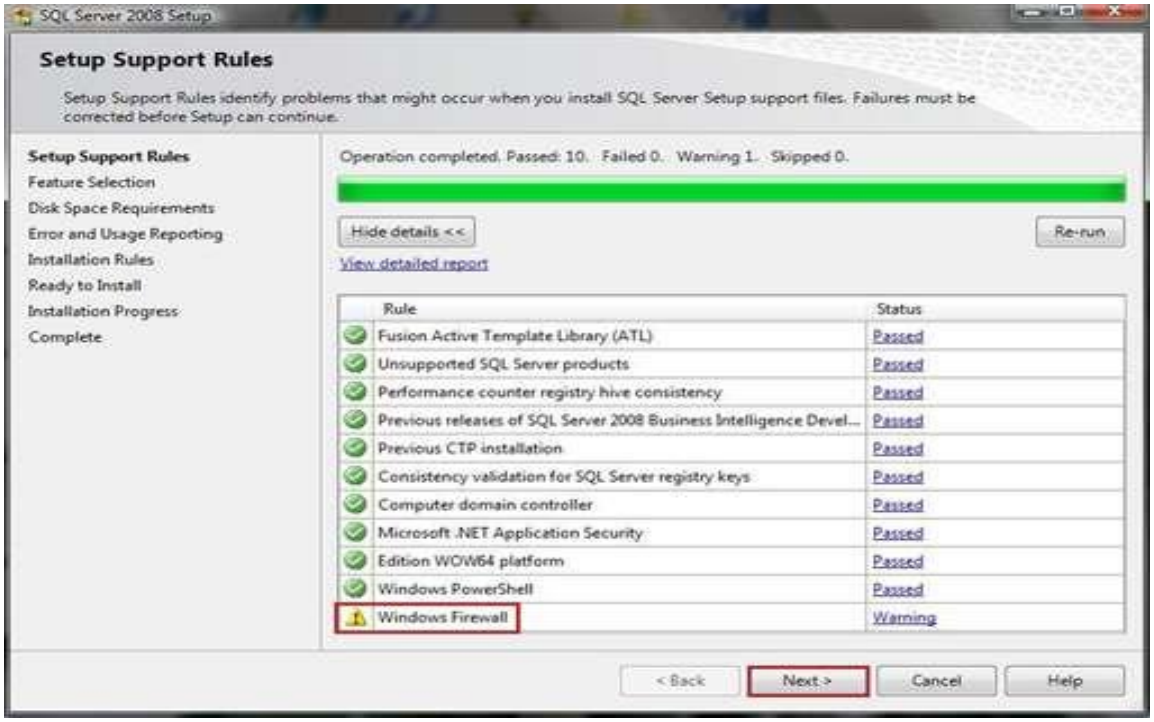

Figure 2.21: Setup support rules details

For feature selection click on Select All button. Then Click on “Next” button.

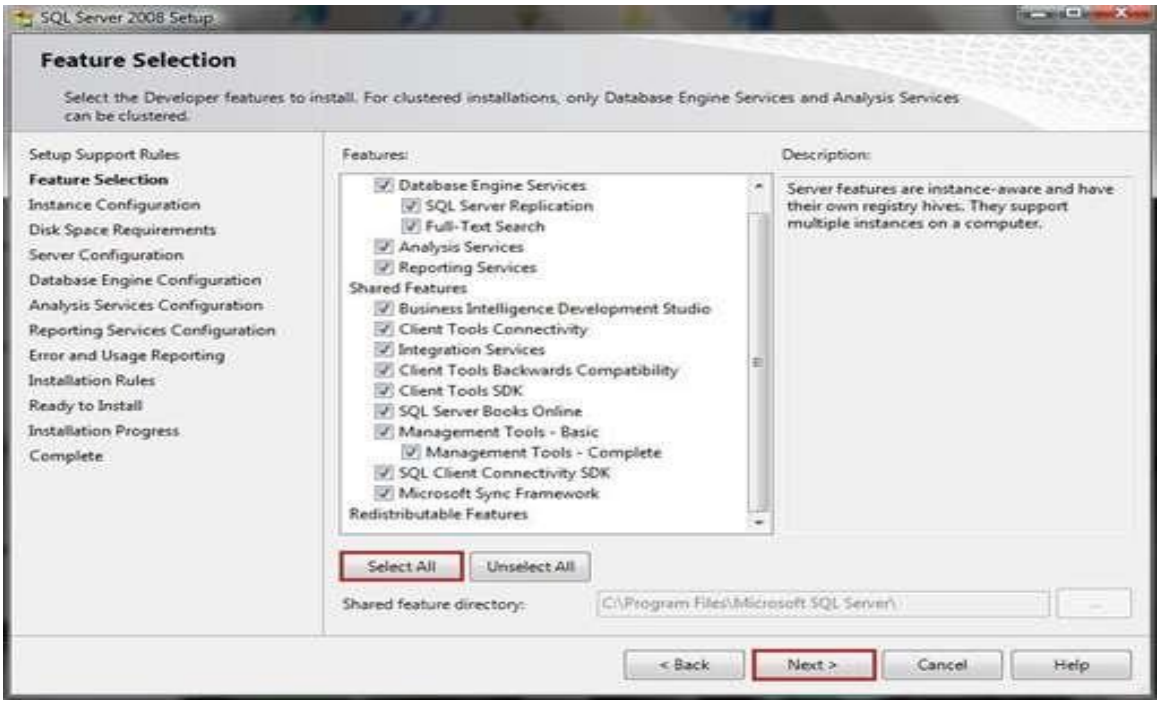

Figure 2.22: Feature selection

Select “Named Instance”, then enter your Instance Name and choose your specific instance root directory (An instance name maintains a personal SQL Server Engine.). Then click on “Next”.

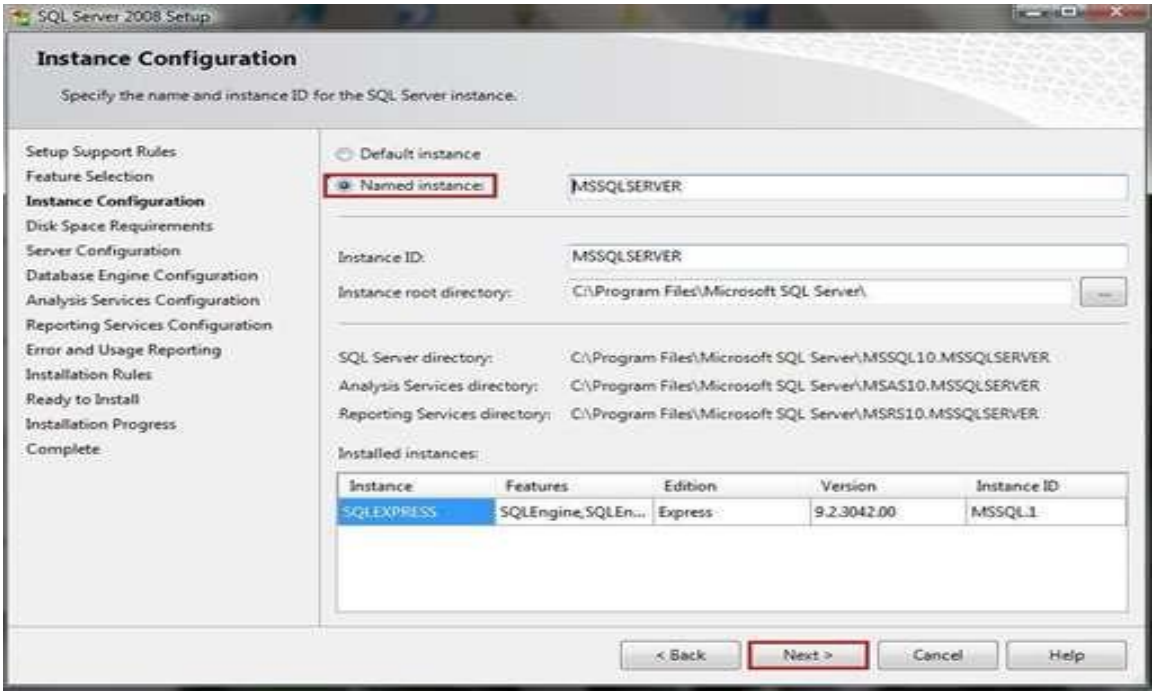

Figure 2.23: Instance configuration

If there is no problem about your amout of disk space, click “Next” Button.

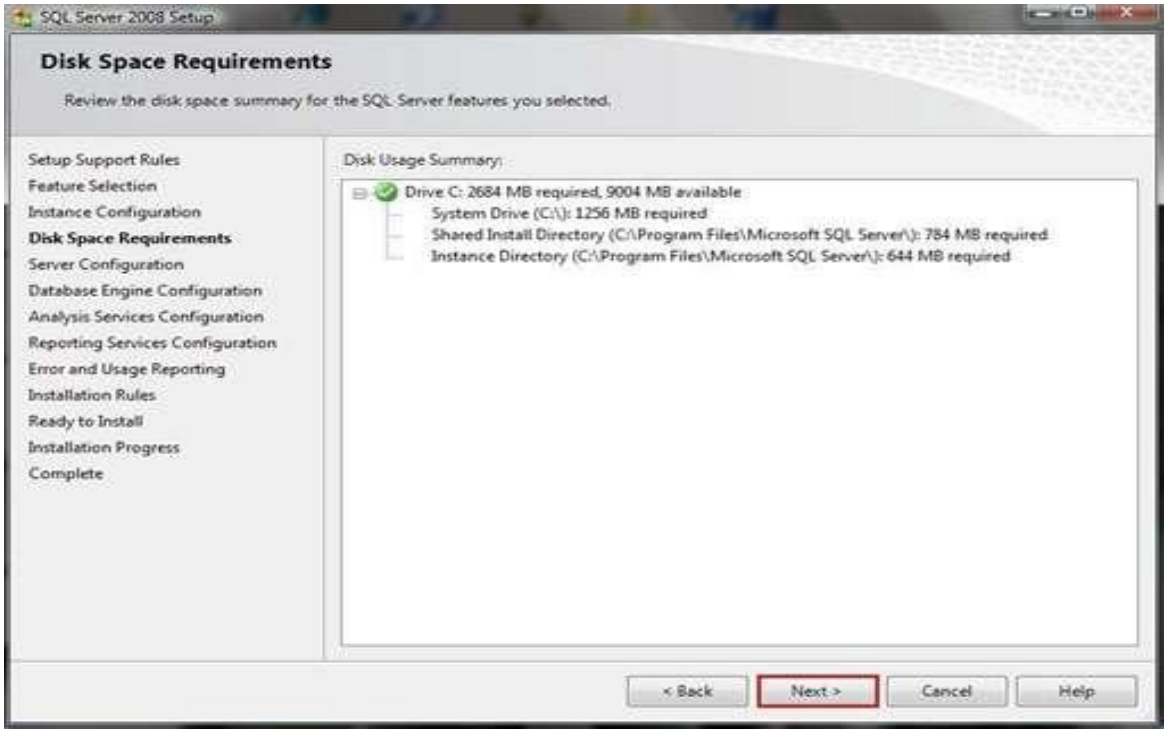

Figure 2.24: Disk space requirements

Make your personal Server Configuration and click on Use the same account for all Server Service button to combine all services under same account. Then click on "Next" button.

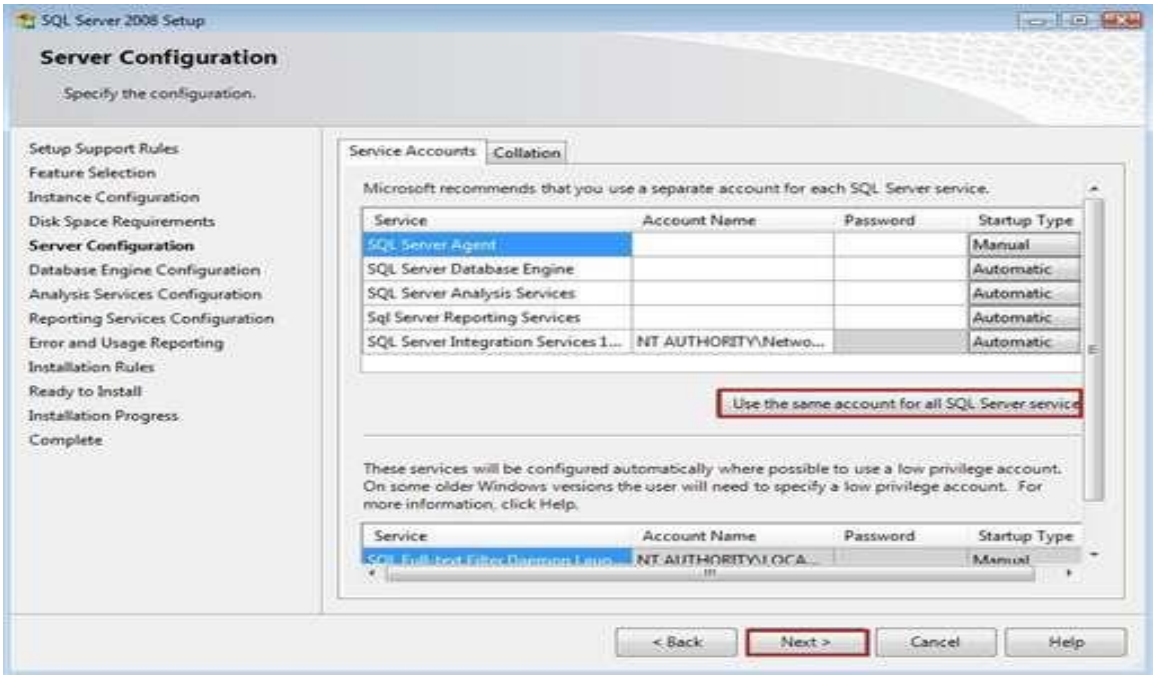

Figure 2.25: Server configuration

In Database Engine Configuration windows, select mixed mode (SQL Server authentication and Windows authentication) selection and type your personal password. Then click on Add Current User button to set the typed password to the current user. Then click on “Next”.

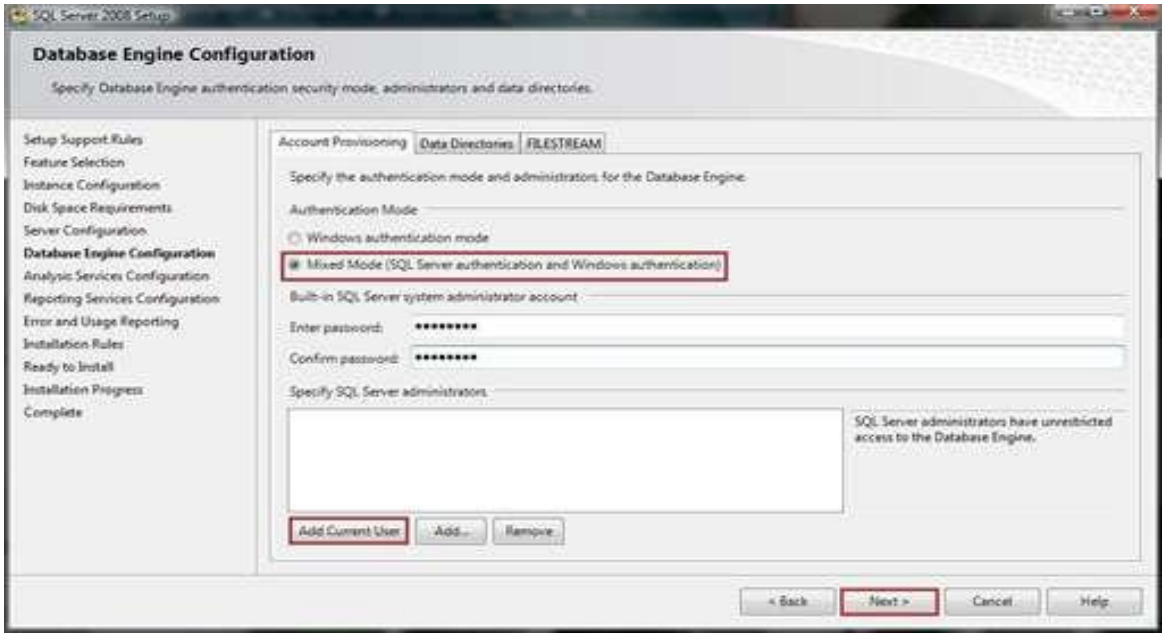

Figure 2.26: Database engine configuration

In Analysis Services Configuration windows, click on Add Current User button to get all the informations about the user. Then Click on “Next” Button.

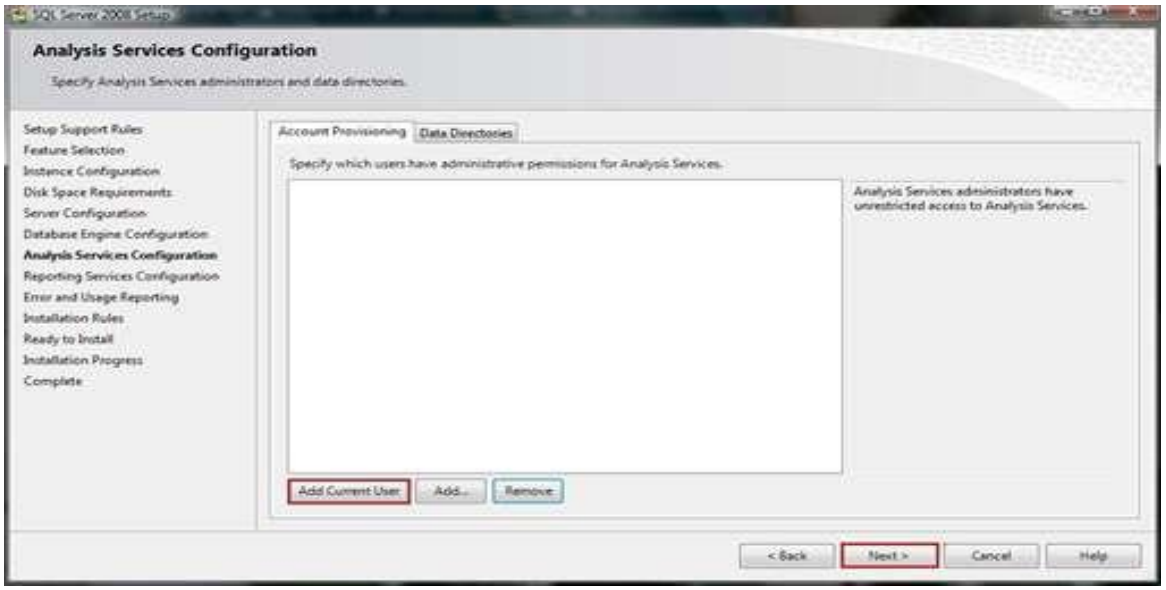

Figure 2.27: Analysis services configuration

In Reporting Services Configuration windows, select first selection (Install the native mode default cofiguration) to install the report server and configure it in Native mode to use the default values.

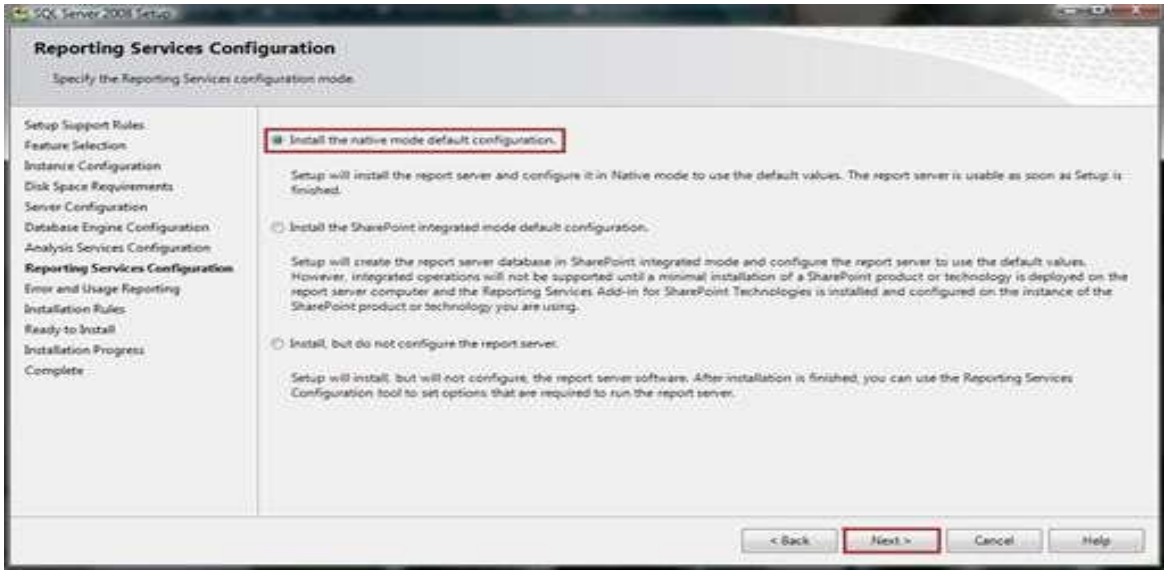

Figure 2.28: Reporting services configuration

Error and Usage Reporting menu, offers users to help Microsoft to improve SQL Server features and services. To continue click on “Next” button.

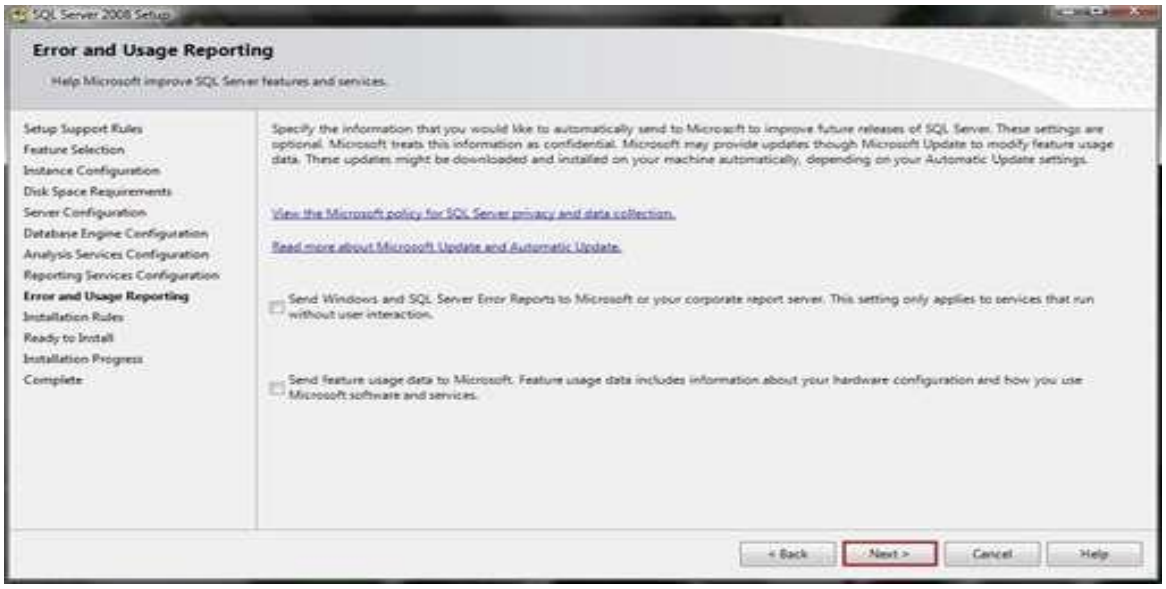

Figure 2.29: Error and usage reporting

In Installation Rules window, setup is running rules to determine if the installation process will be blocked. If there is no error, click on “Next” button.

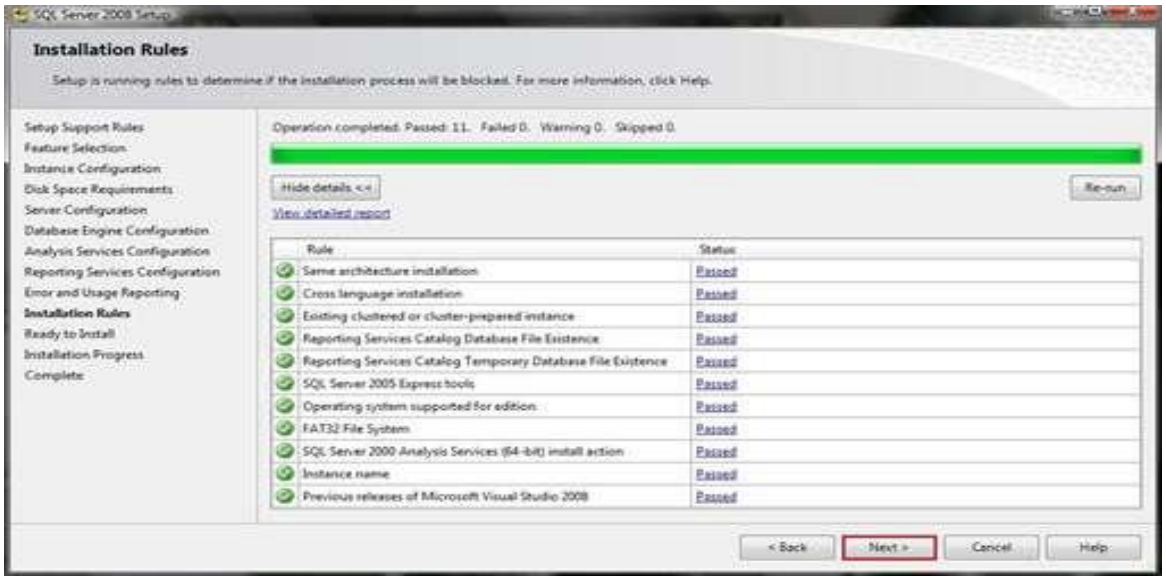

Figure 2.30: Installing rules

In Ready to Install window, all configurations and tests are done. Click on “Install” button to install the SQL Server 2008.

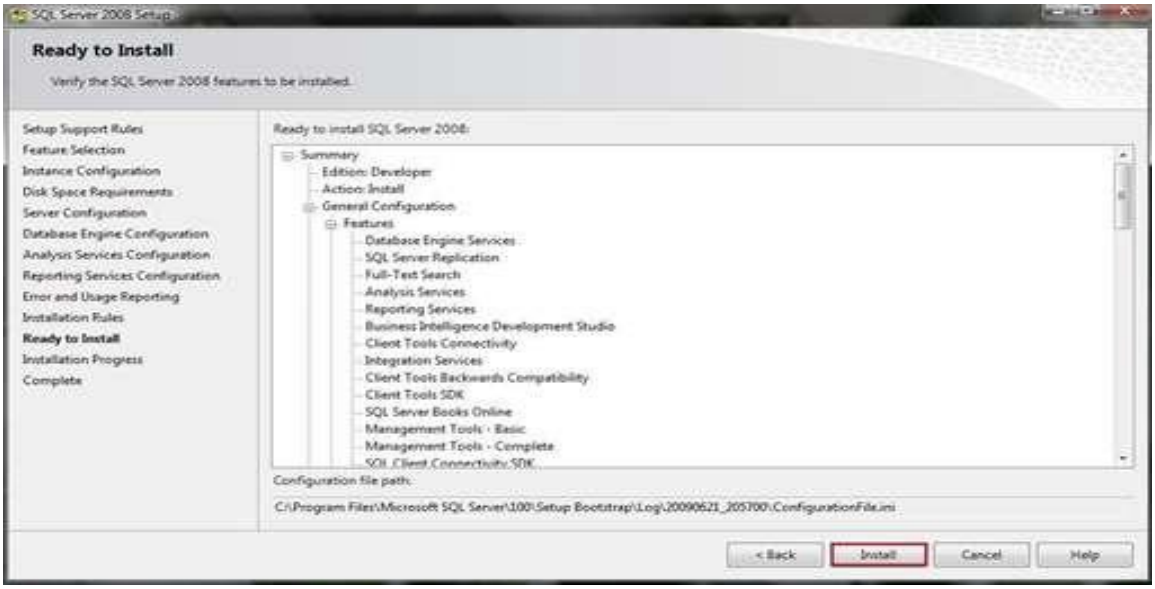

Figure 2.31: Ready to install

When the progress is completed. Click on “Next” button to finalize the installation.

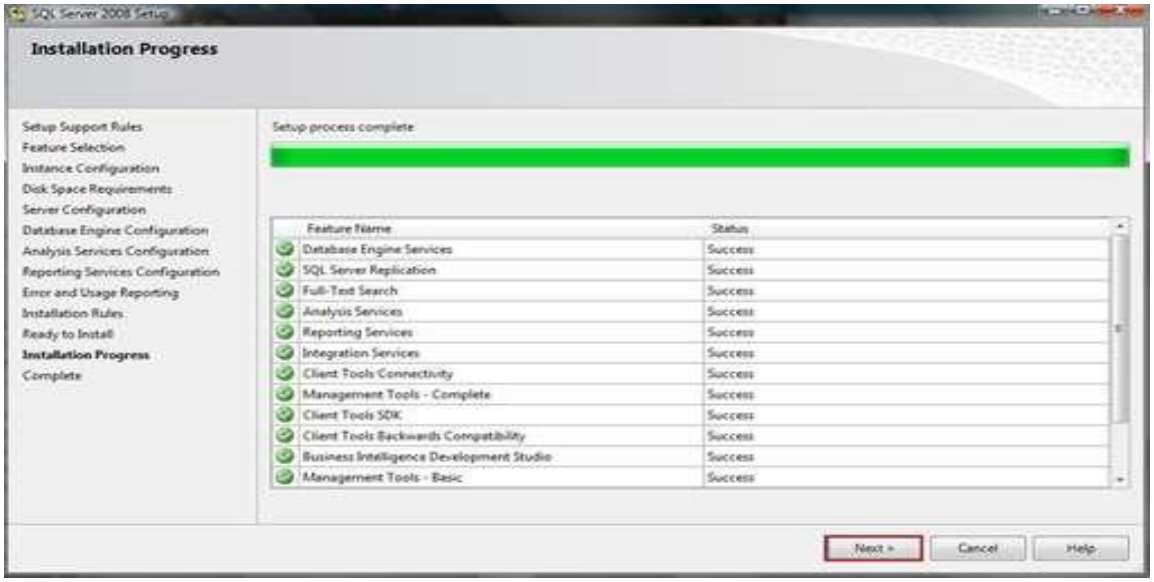

Figure 2.32: Installing progress

CHAPTER THREE

RESULTS AND DISCUSSIONS

3.1 Design of Database

Due to the results of Pesticide Residues in Underground Water of Northern Cyprus, 9 tables were designed for main purposes which are wells, well positions, water reports, pesticides, pesticides blacklist, foods, analysis types, tests, and food analysis reports. These tables have logical relationships between each other.

3.1.1 Design of Wells Table

In wells table, we have 5 columns which are well_id, well_name, well_start_date, well_end_date, and well_navi_date. Generally well_start_date column and well_navi_date have same date time when they are inserted into the database because we don't know when the wells were digged. Well_id icolumn holds the unique, incremental identity data. Well_name stores the town or city data. Moreover, Well_end_date holds the datetime when well is deleted from database. Any columns don't allow any null data in cells. The data types are shown in the figure 3.1.

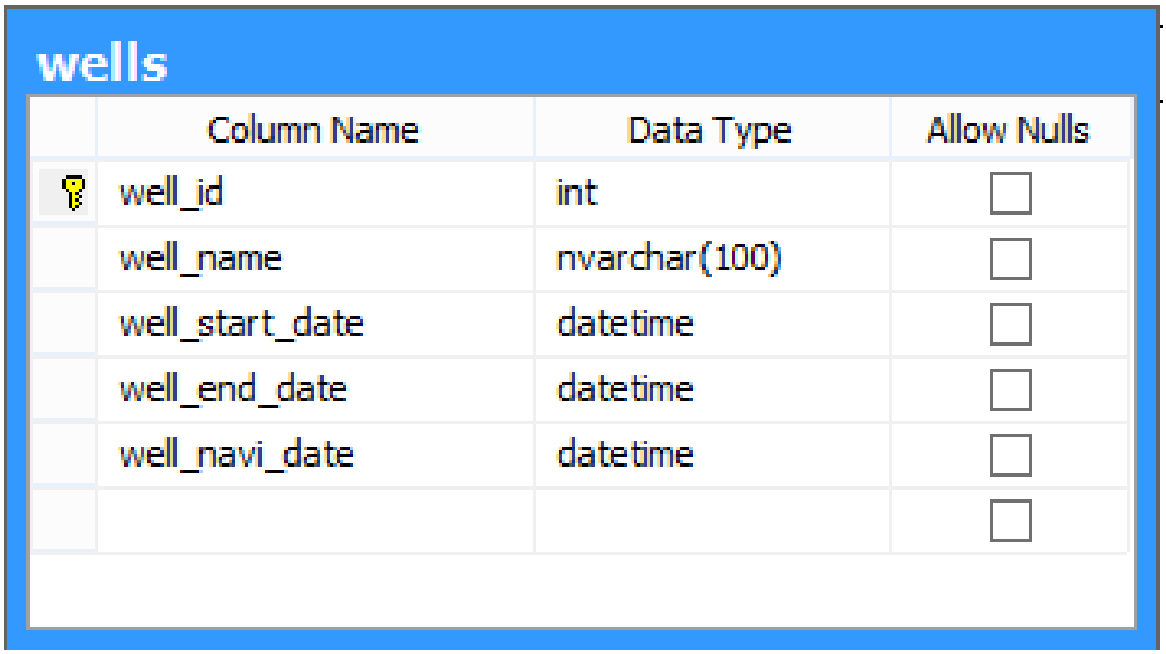

Figure 3.1: Wells table

3.1.2 Design of Well Positions Table

In well positions table, whe have 8 columns which they are w_position_id, w_x, w_y, w_start_date, w_end_date, w_navi_date, well_well_id, and well_depth columns. W_position_id is the unique, incremental identity column. This column has the primary key. W_x and w_y columns accept the longitude and latitude informations of the wells. W_start_date and w_end_date store the date time information when the well is inserted in to the database. Well_well_id store the information of unique well_id of wells table and this columns have the foreign key. Well_depth column store the depth data of the wells and this column accepts null data (if the depth of well don't known). Figure 3.2 shows the data types of the columns.

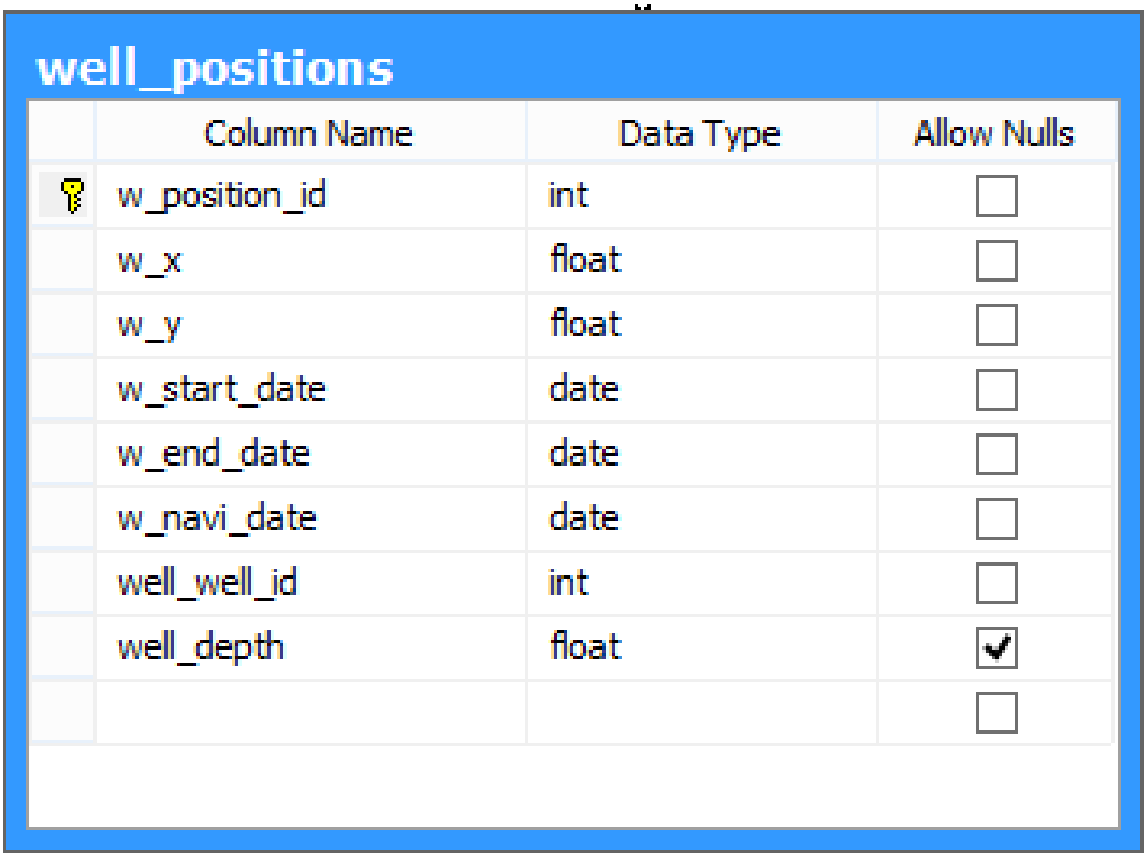

Figure 3.2: Well positions table

3.1.3 Design of Foods Table

Foods table store the unique incremental identities of foods, food name, insertion date, and deletion date. The foods table rows are food_id (Primary Key), food_name, navi_date, and end_date. Only end_date column accepts null data. Figure 3.3 shows the data types of the columns.

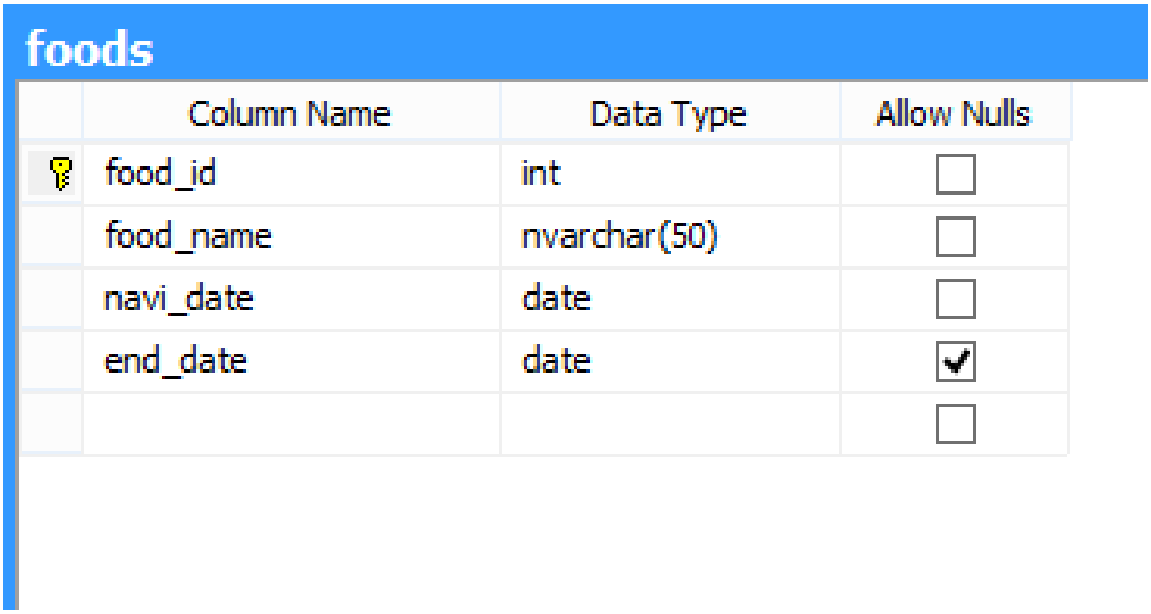

Figure 3.3: Foods table

3.1.4 Design of Pesticides Table

Pesticides table has 5 columns which are callled as pesticide_id (Primary key), pesticide_name, navi_date, limit, and end_date. Pesticide_id column is the unique, incremental identity column. Navi_date store the insertion date time data and end_date store the deletion time data. Both limit and end_date table accepts null data. Figure 3.4 shows the data types of the columns.

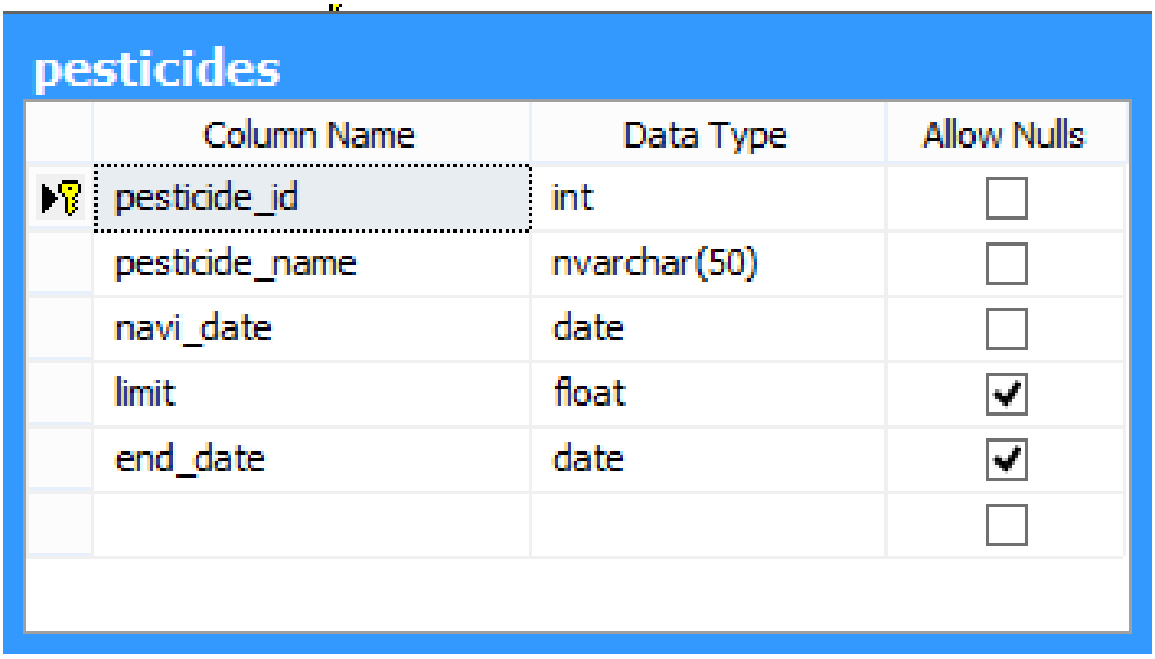

Figure 3.4: Pesticides table

3.1.5 Design of Pesticides Blacklist Table

Pesticides black list table store the pesticides which are accepted as forbidden. This table has unique incremental pblacklist_id (Primary Key) column, pesticide_pesticide_id as foreign key, navi_date and start_date for insertion date and end_date for deletion date. Figure 3.5 shows the data types of the columns.

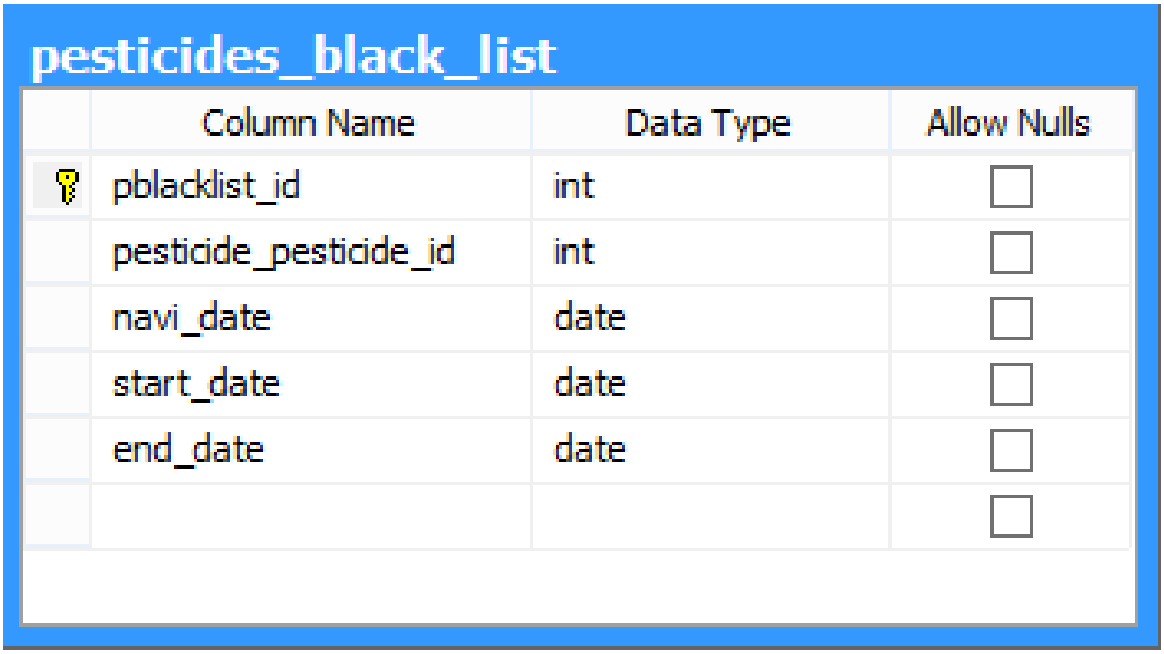

Figure 3.5: Pesticides blacklist table

3.1.6 Design of Water Reports Table

In water reports table, there are 6 columns which are unique incremental identity column called as report_id (Primary key), well_well_id (foreign key for wells table), pesticide_pesticide_id (foreign table for pesticides table), pesticide_amout for storing amout of pesticide, navi_date for insertion date time, end_date for deletion date time. Only end_date column accepts null data. Figure 3.6 shows the data types of water reports table columns.

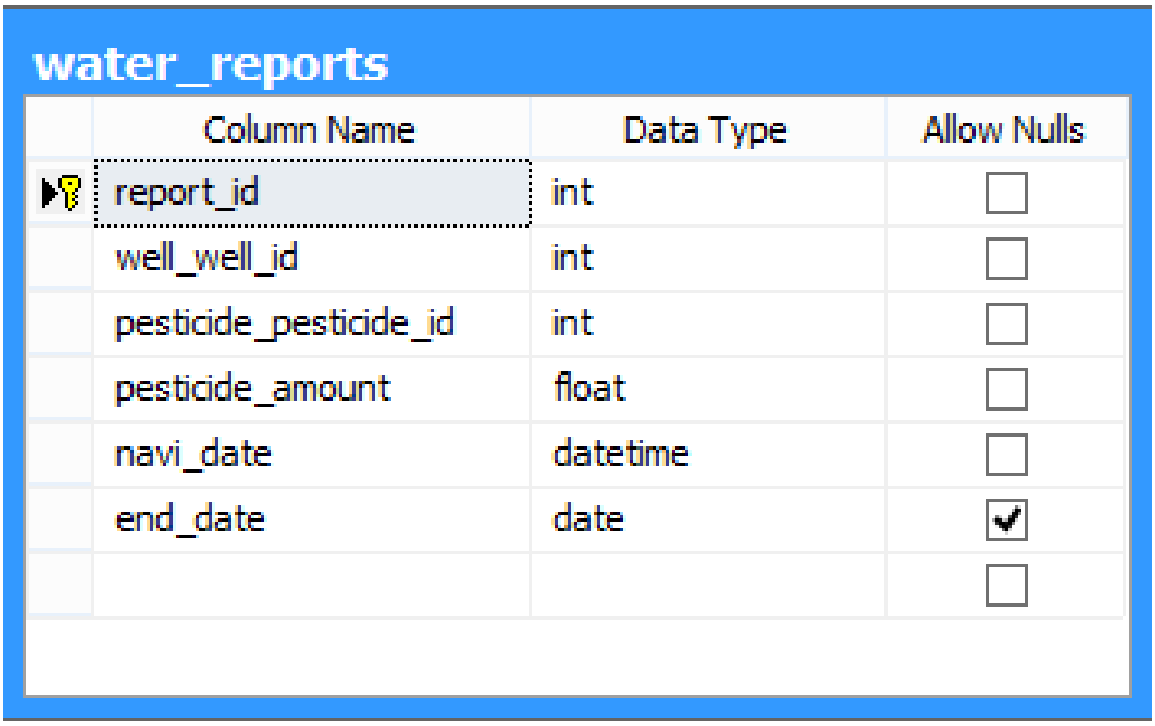

Figure 3.6: Water reports table

3.1.7 Design of Analysis Types Table

In this table, there are 5 columns which are called anayisis_id, analysis_name, analysis_starttime, anaylsis_endtime ,and analysis_comment. Analysis_id is the unique, incremental primary key of this table. Analysis_name is define the analysis. Analysis_starttime is defined as insertion datetime and analysis_endtime is also defined as deletion time. Analysis_comment (accepts null data) is for make comment about analysis. Figure 3.7 shows the data types of the columns.

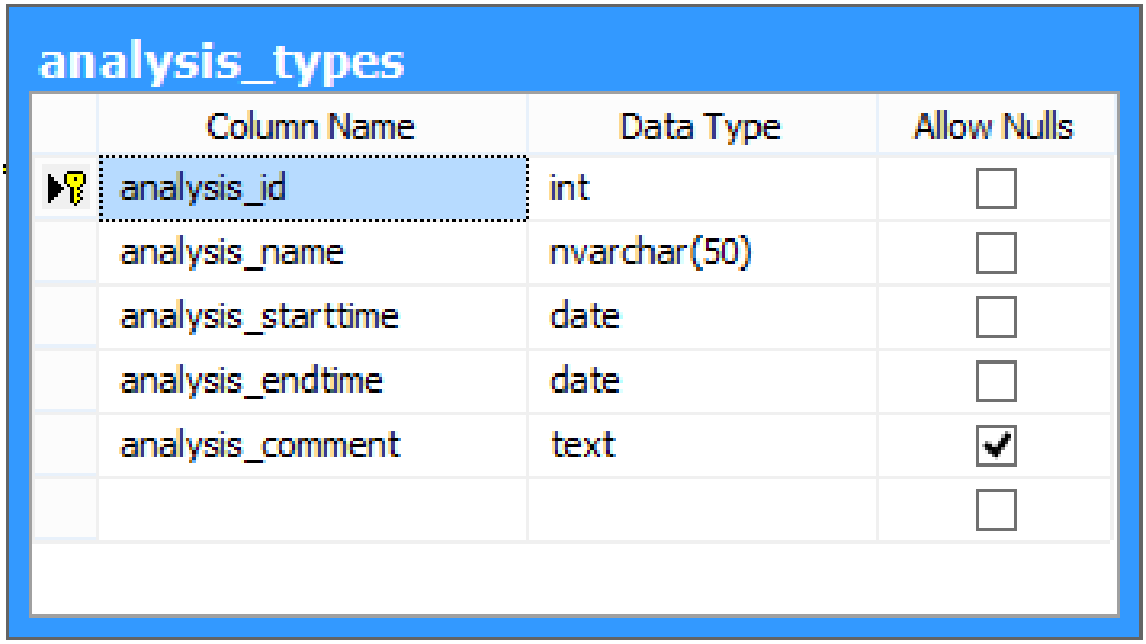

Figure 3.7: Analysis types table

3.1.8 Design of Tests Table

Tests table contains unique incremental identity column which is called test_id (Primary key), analysis test_name to define tests, test_start_time for insertion date time, test_endtime for deletion date time, test_comment (accepts null data) to include extra comments ,and analysis_analysis_id (Foreign key) to connection to analysis types table. Figure 3.8 shows the data types of the columns.

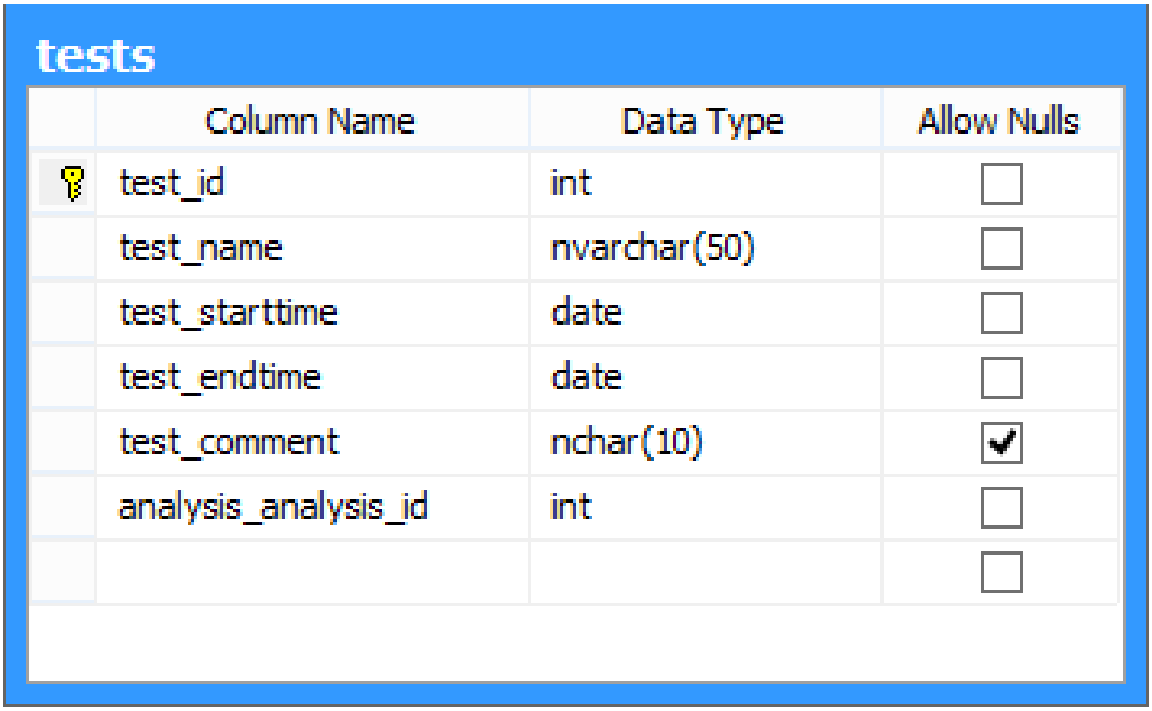

Figure 3.8: Tests table

3.1.9 Design of Food Analysis Reports Table

Food analysis reports table which contains unique incremental identity column called report_id (Primary key), report_name which defines the report, report_text for storing report details (accepts null data), food_food_id (Foreign key) including connection to foods table, test_test_id (Foreign key) including connection to tests table, test_result_amount which species amount of test result (accepts null data) , test_result_comment determining comment about test result (accepts null data), pesticide_pesticide_id (Foreign key, accepts null data) which connects to pesticides table, pesticide amount (accepts null data), navi_date for insertion date time, pesticide_limit_status for specify pesticide limit, food_coordinate_x and food_coordinate_y for longitiude and latitude food data (accepts null data), well_well_id including connection to wells table. Figure 3.9 shows the data types of the columns.

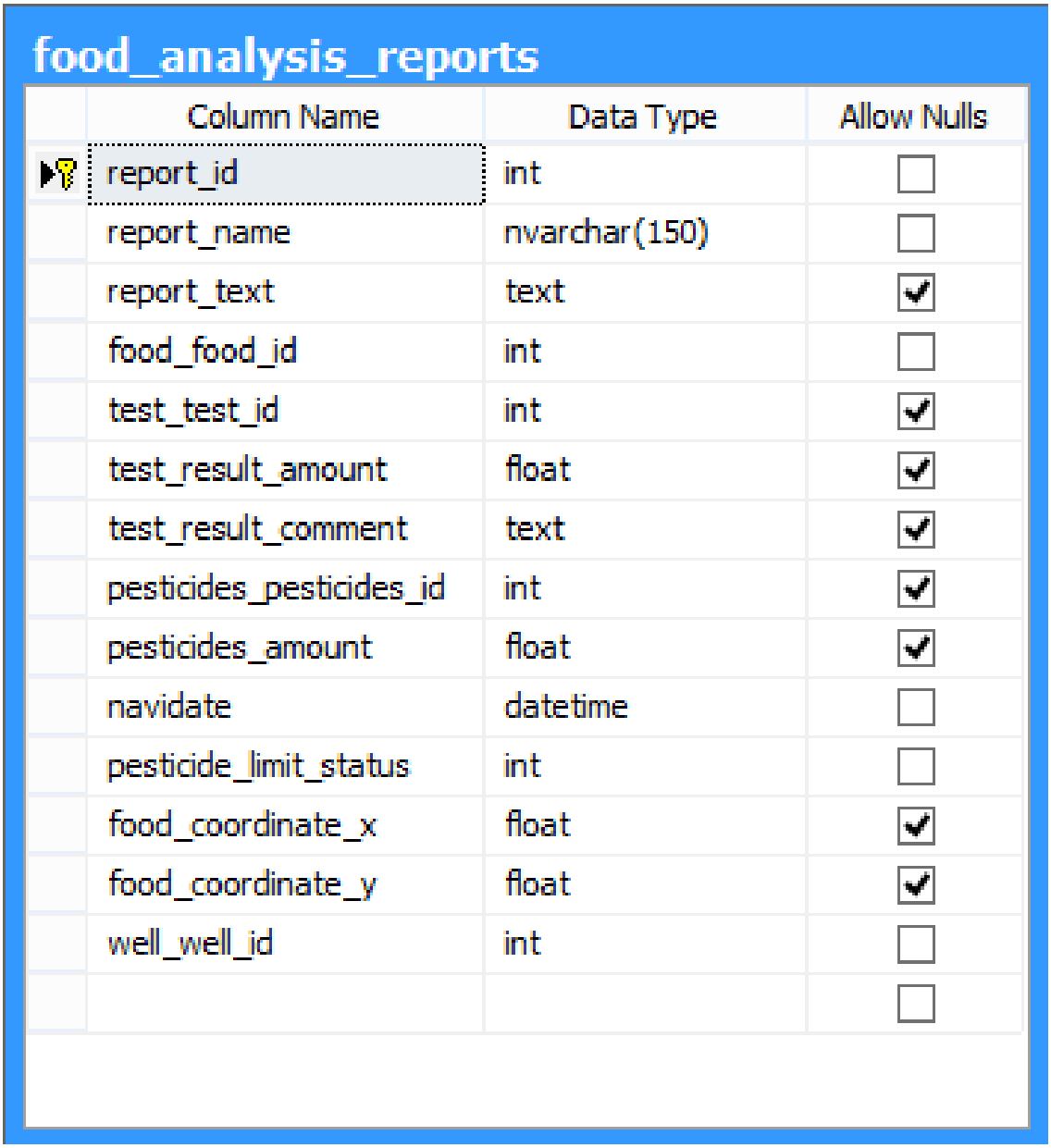

Figure 3.9: Food analysis reports table

3.2 Design of Graphical User Interface

Graphical User Interface is the main method to interacting with a computer. GUI is the graphical images with text that displays on screen. GUIs can be controlled with the keyboard, mouse, touchpad, and etc. All GUIs have a frame which are movable and resizable with the user demands.

In this project, GMAP libraries are citied in to the references to improve the graphical user interface. GMAP libraries maintain google earth features to include in to the project. The most usefull features of GMAP libraries are various kind of pointers, map with local city informations and GPS connectivity.

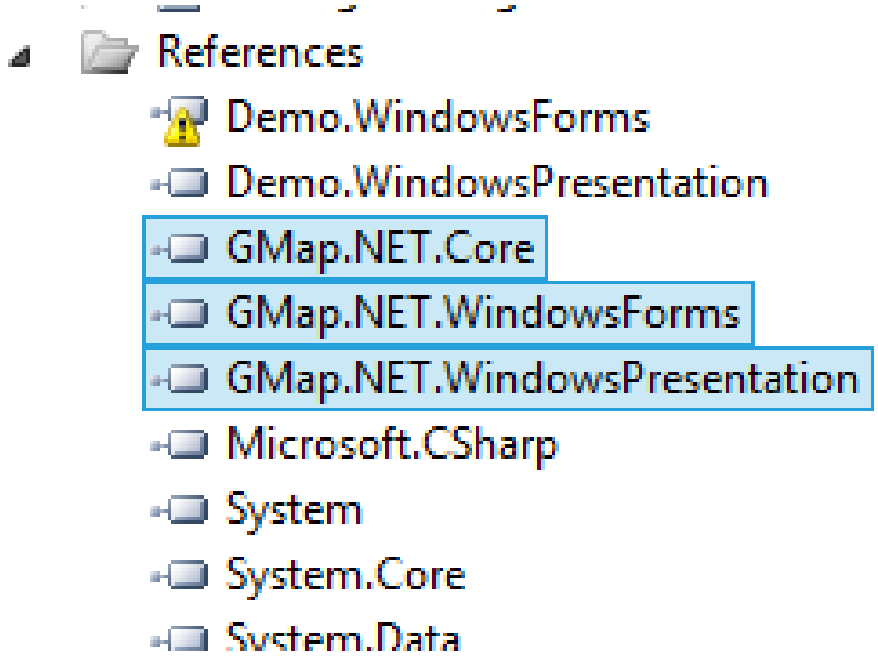

Figure 3.10: GMAP libraries

There are 1 main window which has 3 tabs and 21 sub windows. Main properties such are Google Map Tool, add/remove/update information operation window, and main window of data queries were collected in to the tabs and the specific tools.

3.2.1 GMAP Main Window

GMAP main window is the first openning window when clicking on software shortcut. In this window, there is one tab pages tool with 3 tabs. GMAP main window size is fixed to avoid from image disorders.

In map tab, there are 2 labels to show longitude and latitude. Moreover, a gmap control tool in map page shows google maps and pointers on maps. In this tab, there are some extra properties which mentioned below.

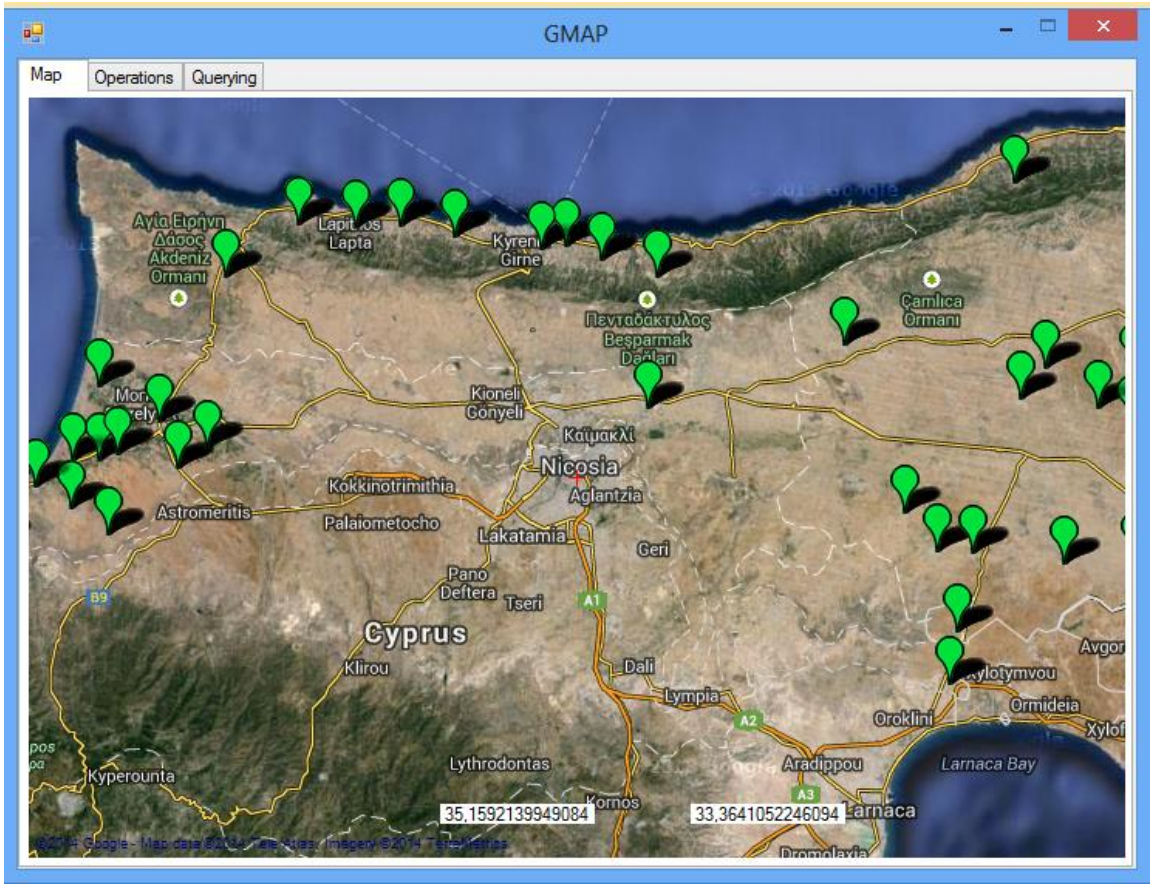

Figure 3.11: GMAP main window

In operations tab, there are add, remove, and update buttons for test types, analysis, wells, pesticides, foods, report operations. Operations tab is a port to make changes on database by openning new add, delete and update forms.

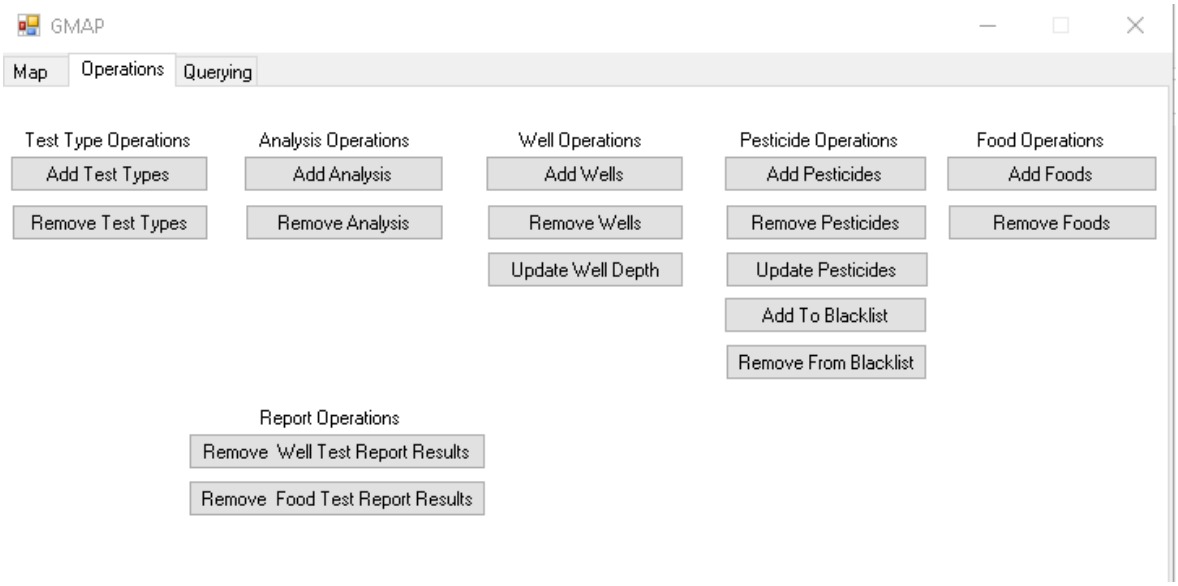

Figure 3.12: Operations tab

There is one more tab named Querying. Querying tab is a quick way to be aware from important informations. This tab has four buttons and one data grid view tool. When clicking on a button, a query command is executed and the query result is shown on the data grid view tool. There are four query in this tab. They are:

- Show wells which have exceeding level of pesticides
- Show Clean Wells
- Show Only Deleted Wells
- Show Black Listed Pesticides

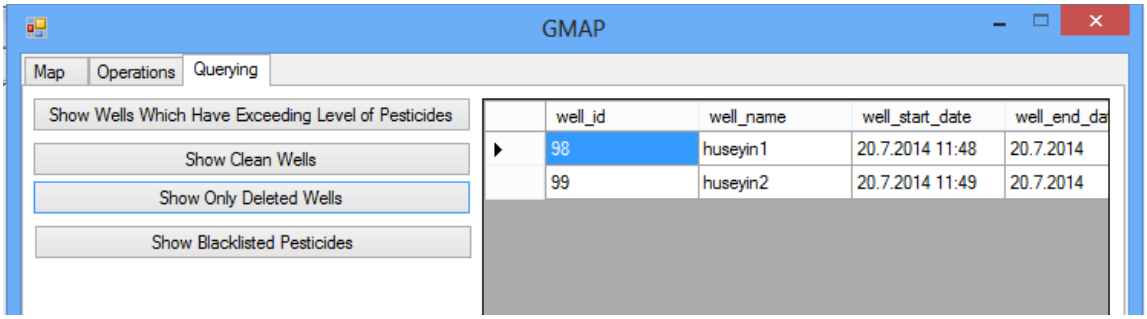

Figure 3.13: Querying tab

3.2.2 GMAP Sub Windows

In GMAP software, there are some sub windows which appear after doing an action. These actions are seen after double clicking and clicking on a tool.

3.2.2.1 Sub Windows of Map Tab

There are 2 action to open sub windows of map tab. Double clicking on google map opens add well sub window.

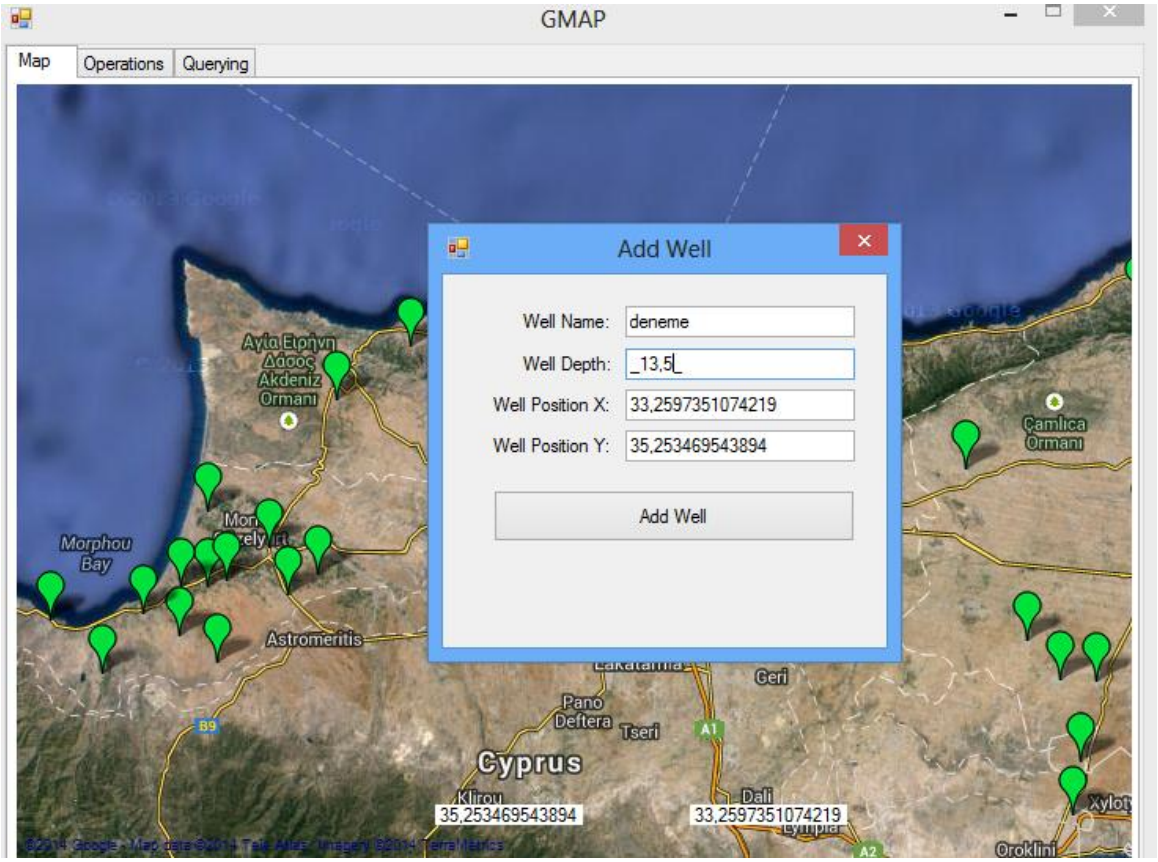

Figure 3.14: Add well sub window

In add well sub window, there are four labels such as three textbox, one button, and one masked text box. Double clicking on gmap controller in map tab, the longitude and latitude are enable to add well sub windows and automatically are being typed in well position x and well position y textboxes. Well name and well depth are manually typed to their own textboxes. After typing the information, clicking on add well button provides to add the well into the database and the system automatically determines the new well pointer, a message

box appears to show adding well action is completed. Action messages accomplished are default in this software.

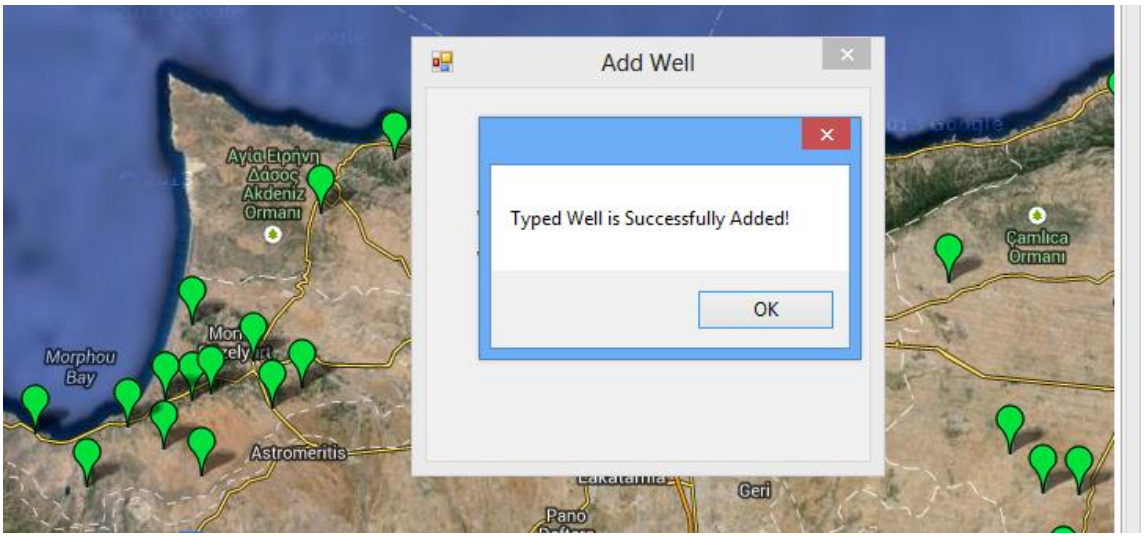

Figure 3.15: Action completed message

When clicking on a google map pointer, the options sub window is seen. In the options sub windows, there are four radio buttons and a button to execute the action. All radio buttons provide to have different actions.

Figure 3.16: Options sub window

The radio buttons enable to add water and food test report,to delete selected well and close button. The close named radio button only closes the option sub window without any change. Furthermore, delete selected well radio button deletes (sets the end_date to datetime) only a

well which is selected at first. These two radio button don't require any sub window to realize an action. Add water test report and add food test report radio buttons require their own sub windows to realize an action.

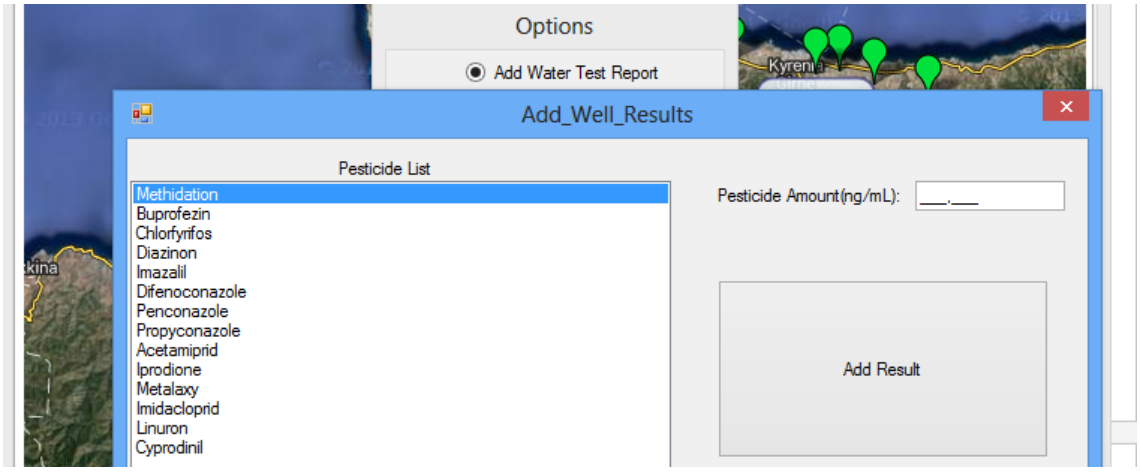

Figure 3.17: Add water test report

There are two labels such as one listbox, one masked text box, and one button in add well results sub form. List box gets the pesticides from database. To add water test reports, select add water test report radio button. Then click on action button. The window named Add Well Results select a pesticide from pesticide list, then shows presticide amount. At last click on add result button completes the action.

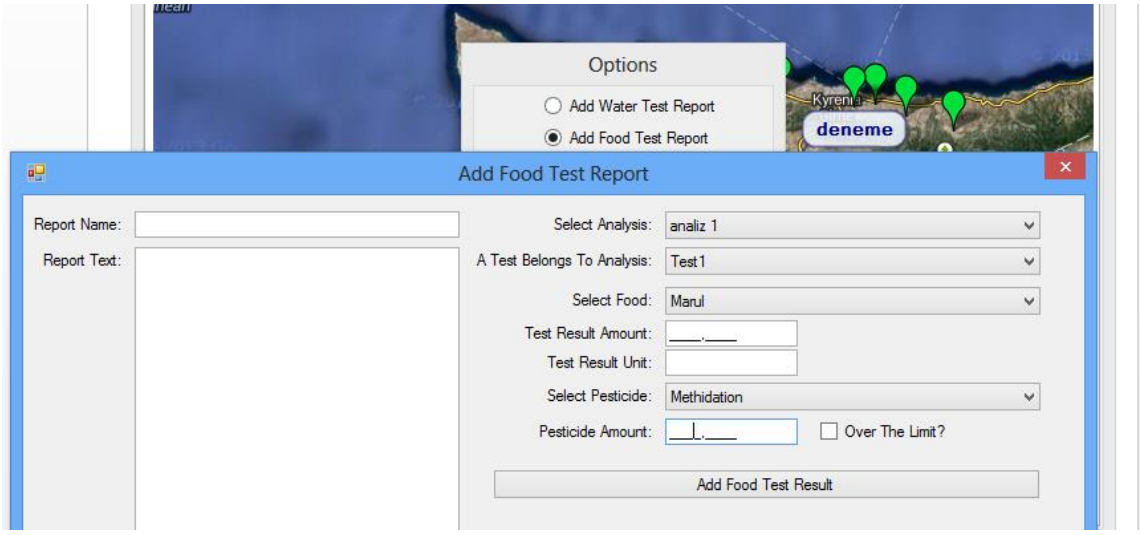

Figure 3.18: Add food test report

In add food test report sub windows, There are four combo box, three textbox,two masked box, nine labels,one check box and one button. To access add food test report sub window, select add food test report radio button , then click on do action button. To add food test reports, fill the textboxes, select an analyse and a test, check if over the limit, then click on add food test result button to complete the action.

3.2.2.2 Sub Windows of Operations Tab

Operation tab is a port to make changes on database by openning new add, delete and update forms, which was mentioned above. There are sixteen buttons to open sixteen different sub forms (windows) which is shown in Figure 3.12.

Add Analysis sub window has two labels, two textbox, and one button. In order to add analysis, write an analyse name and comment, then click on add analyse button.

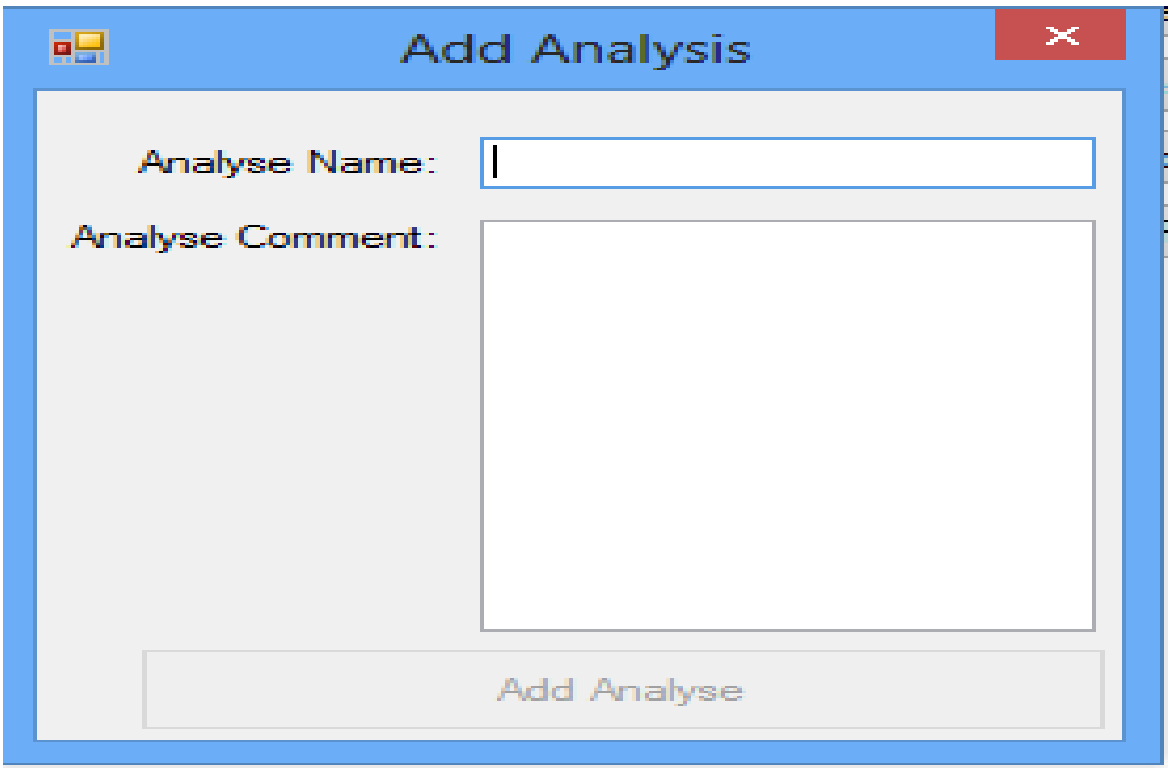

Figure 3.19: Add analysis sub window

Remove Analysis sub window has one label, one data grid view, and one button. In order to remove an analysis, select an analyse and click on remove analyse button, respectively.

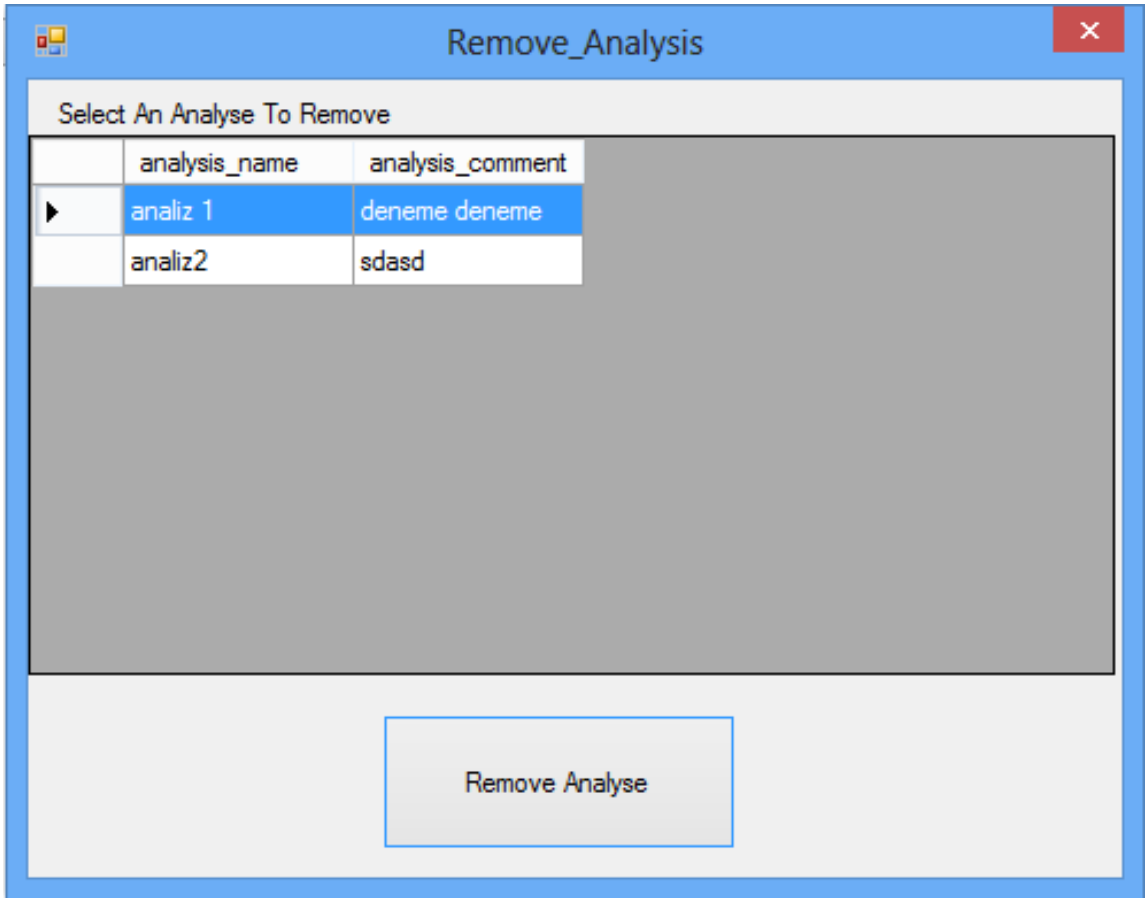

Figure 3.20: Remove analysis sub window

Add test types sub window has one data grid view, three label, two text box, and one button. In order to add test types, select an analyses from data grid view, write test name and comment and click on add test button.

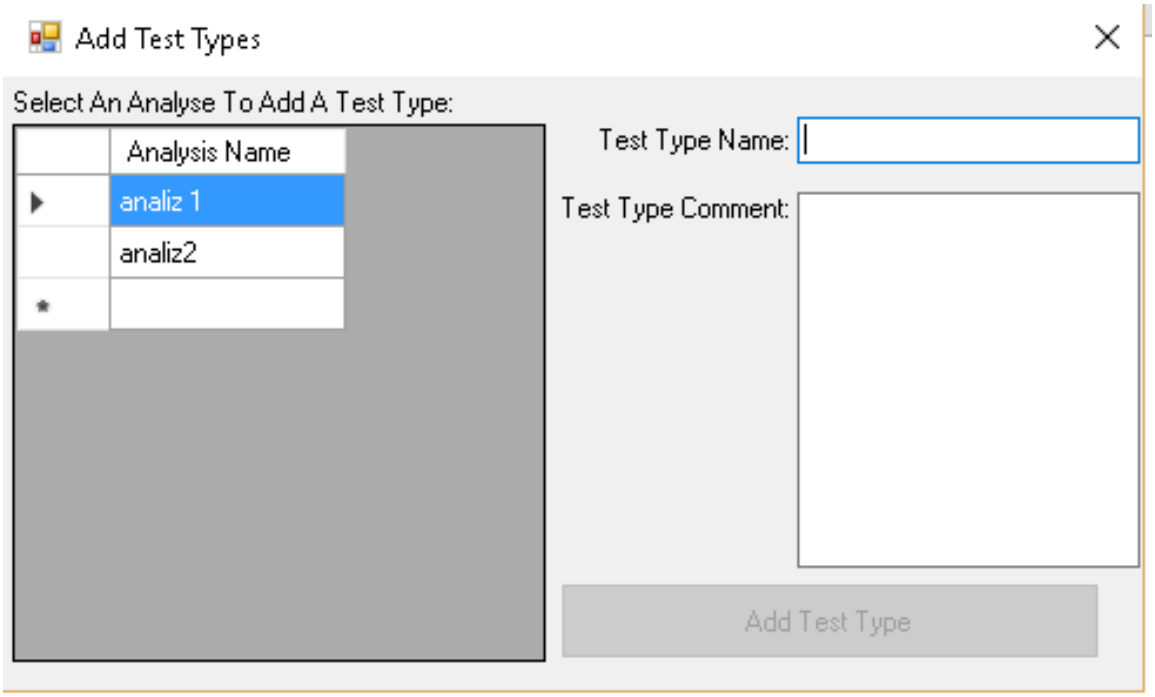

Figure 3.21: Add test types sub window

Remove test types sub window has one label, one data grid view, and one button. In order to remove a test type, select a test type from data grid view and click on remove test button.

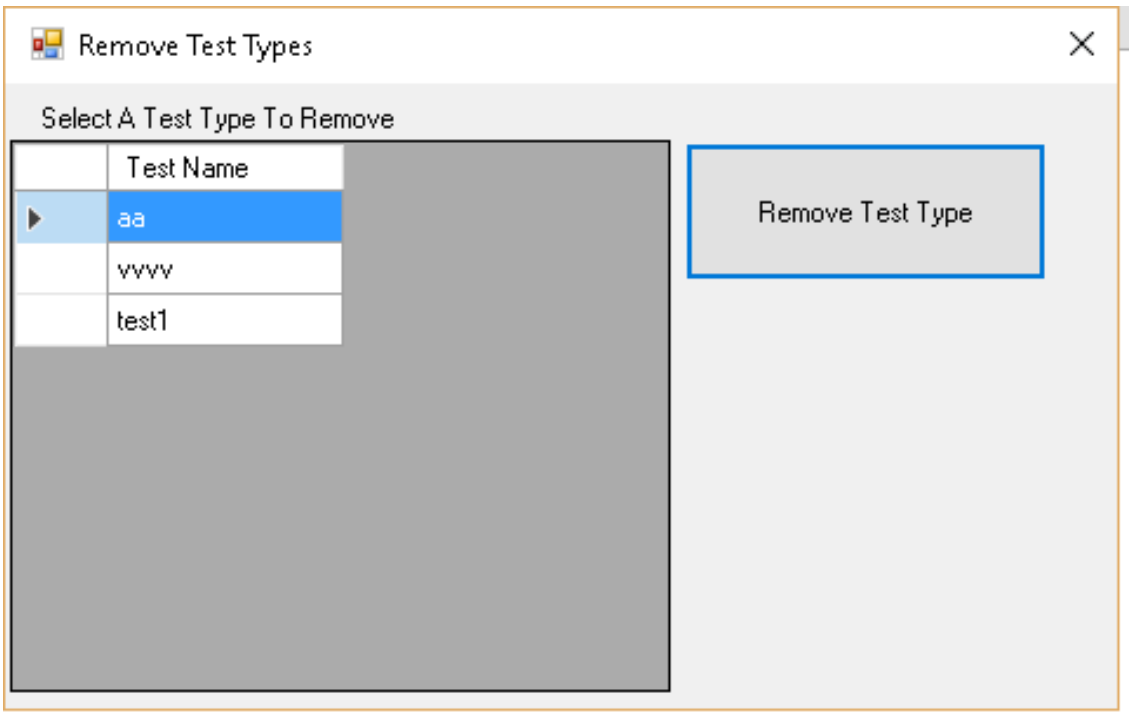

Figure 3.22: Remove test types sub window

In add wells sub window, there are four labels, three text box, one masked text box, and one button. In order to add a well, fill all text boxes and click on add well button.

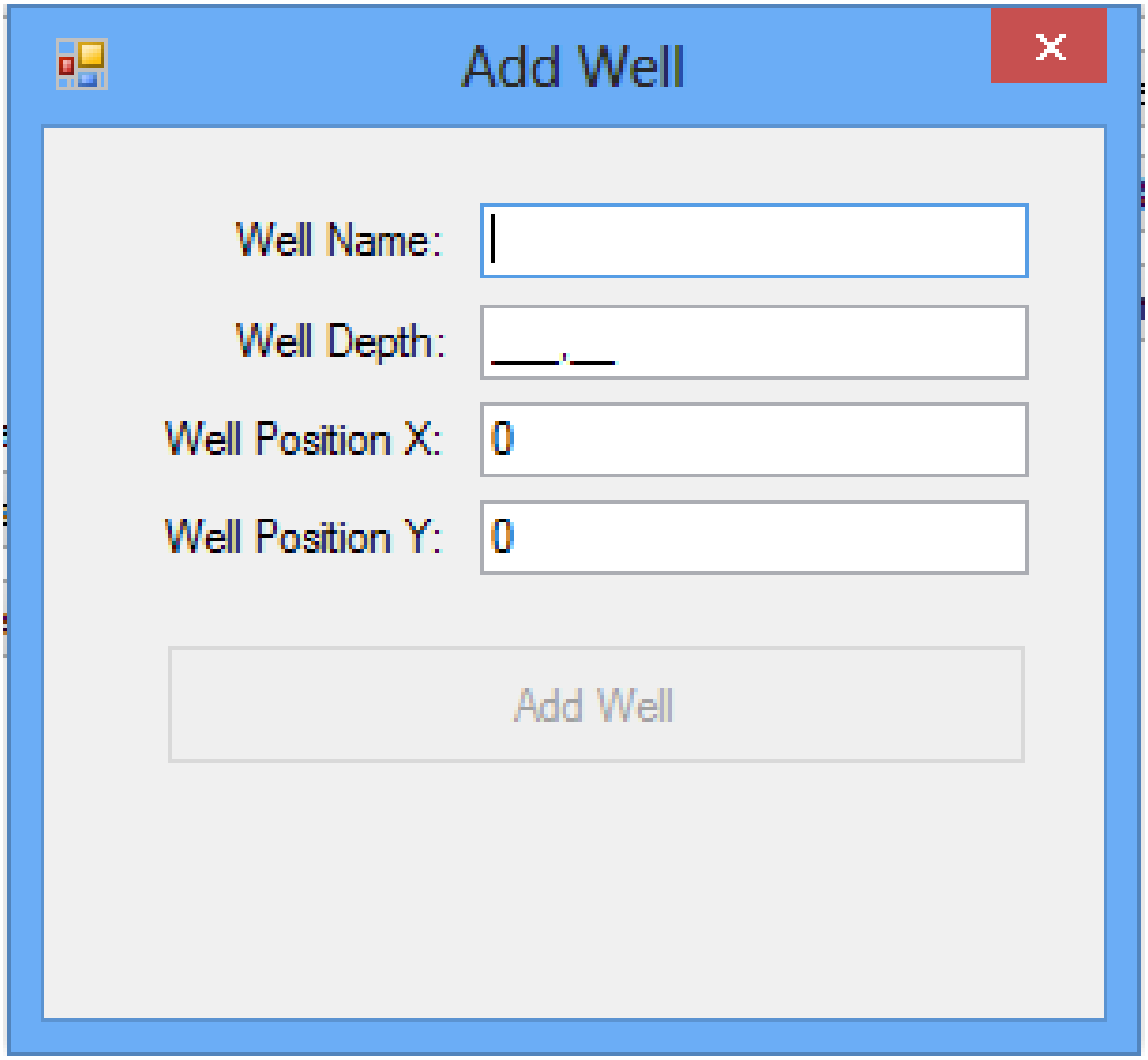

Figure 3.23: Add well sub window

In remove wells sub window, there are one data grid view, one label, and one button. In order to remove a well, select a well from database and click on remove well button.

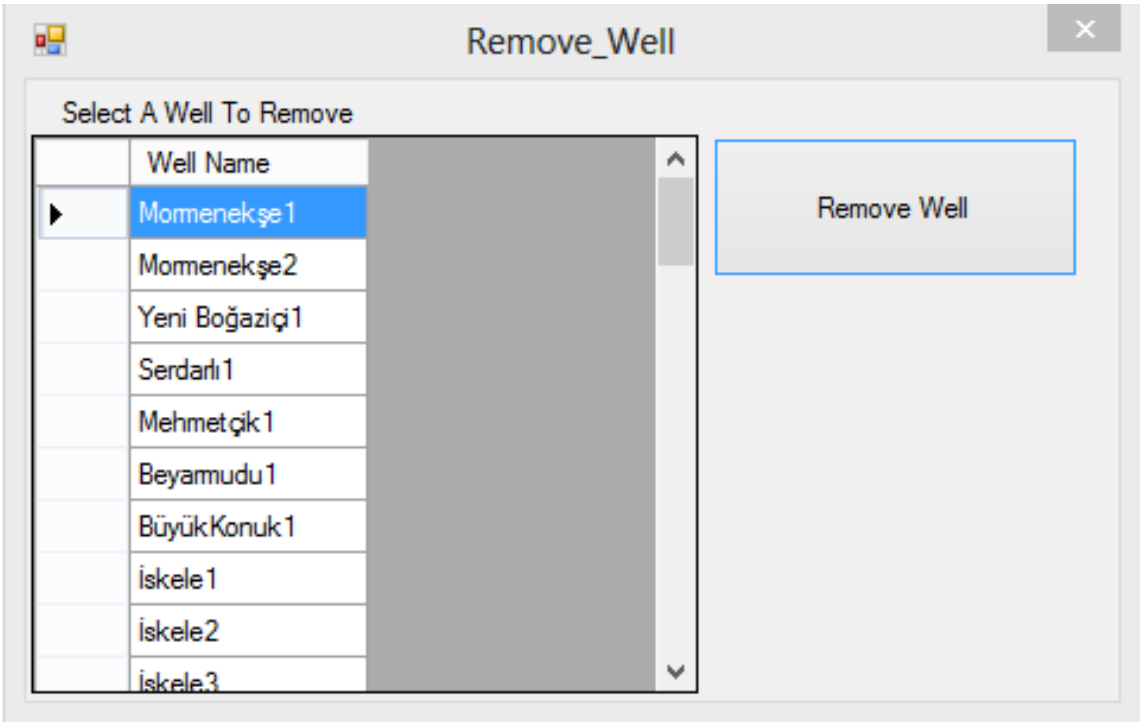

Figure 3.24: Remove well sub window

In update well depth sub window, there are two textboxes, one combo box, one text box, one masked text box, and one button. In order to update a well depth, select a well, type a different well depth and click on update well depth button.

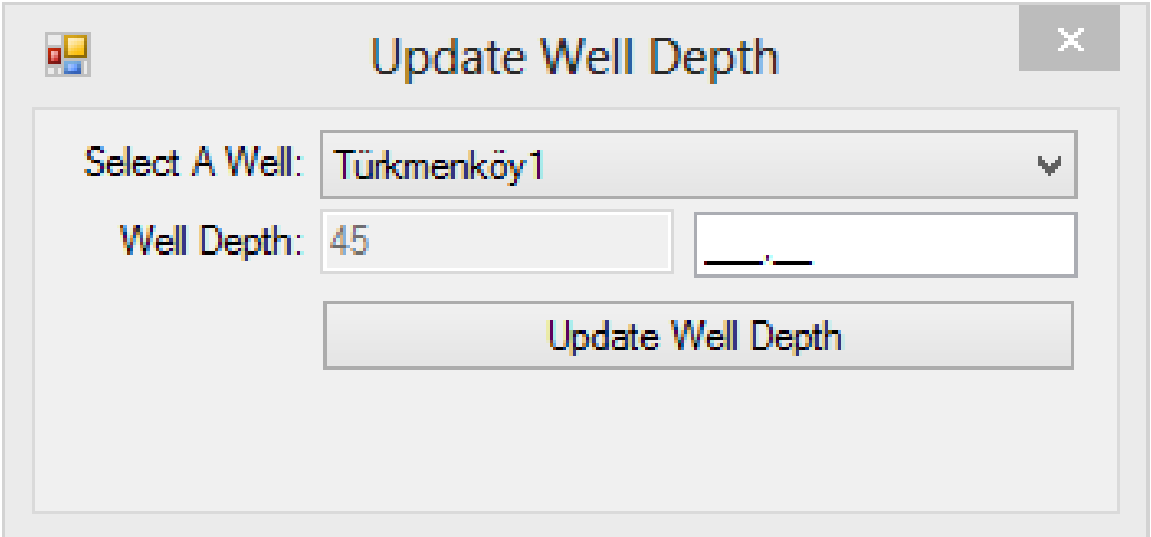

Figure 3.25: Update well depth sub window

In add pesticides sub window, there are two labels, one masked text box, one text box, and one button. In order to add a pesticide, type all text areas with pesticide information and click on add pesticide button.

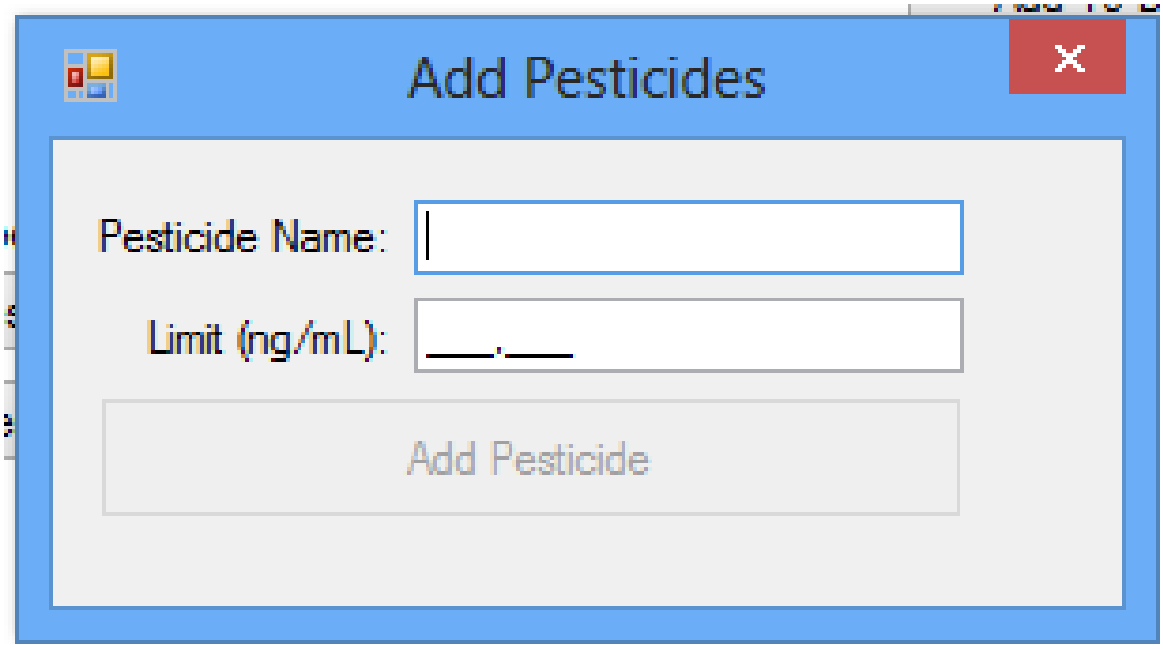

Figure 3.26: Add pesticides sub window

In remove pesticides sub window, there are one label, one data grid view, and one button. In order to remove a pesticide, select a pesticide to remove, and click on remove pesticide button.

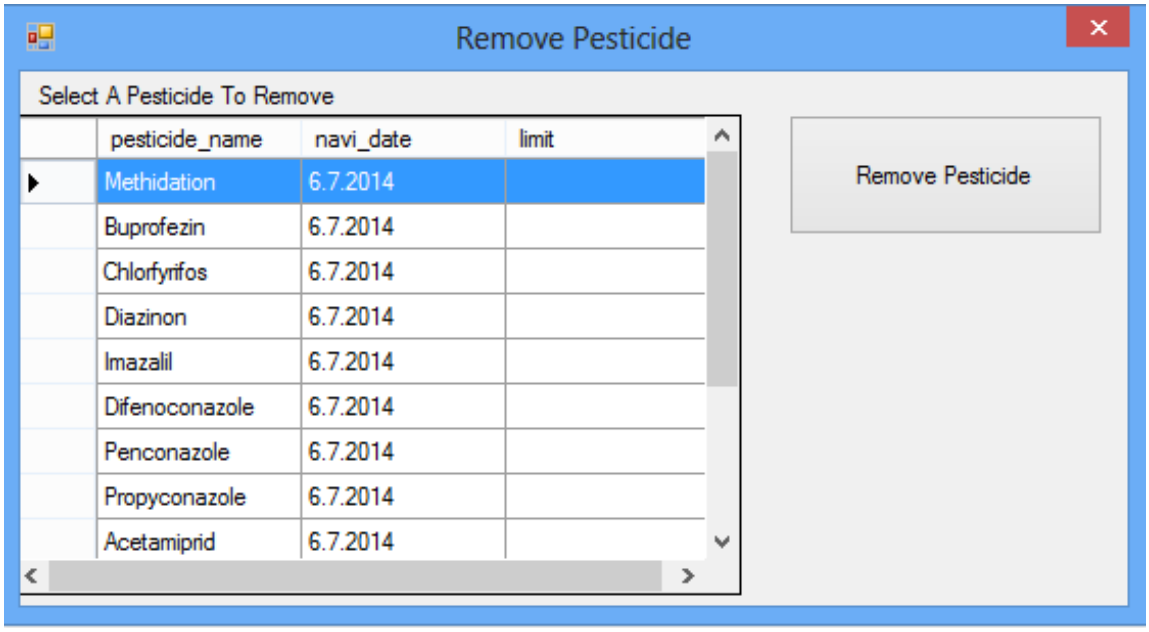

Figure 3.27: Remove pesticides sub window

In update pesticides sub window, there are three labels, one list box, two text boxes, one check box, and one button. In order to update a pesticide, select a pesticide from list box, correct the pesticide name if it is wrong, check the pesticide new list check box if there is a new limit and type new limit value and click on update pesticide button.

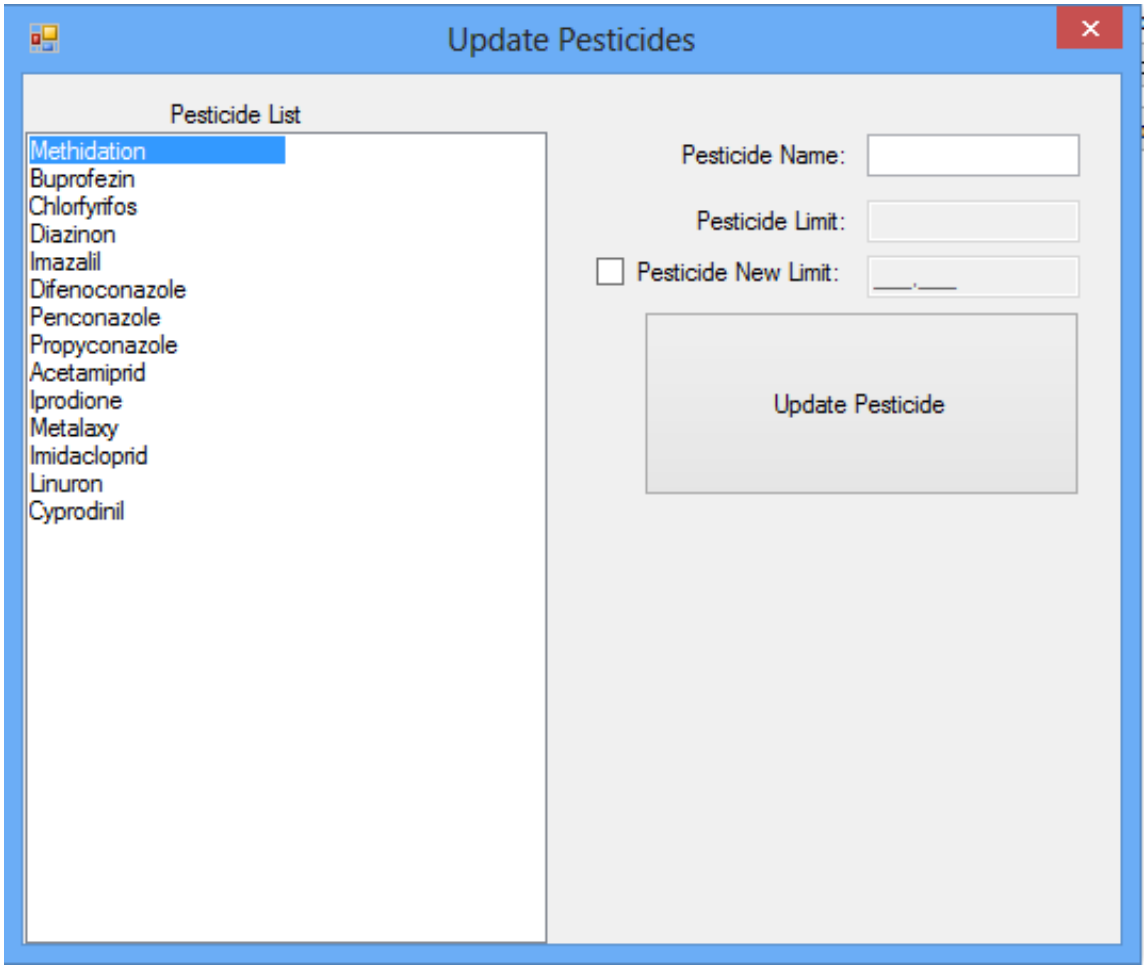

Figure 3.28: Update pesticides sub window

In add pesticides to black list sub window, there is one datagridview, one label, and one button.In order to add a pesticide in to black list, select a pesticide and click on add to black list button. Some pesticides are restricted to use in Northern Cyprus by Ministry of Agriculture and Natural Resources. Black list table can be updated periodically by checking the Ministry of Agriculture and Natural Resources official website. The restricted pesticides are located in link shown below:

- http://www.tdkb.gov.ct.tr/tr-tr/bilgilendirme/bitkisel%C3%BCretim/bitkikoruma.aspx

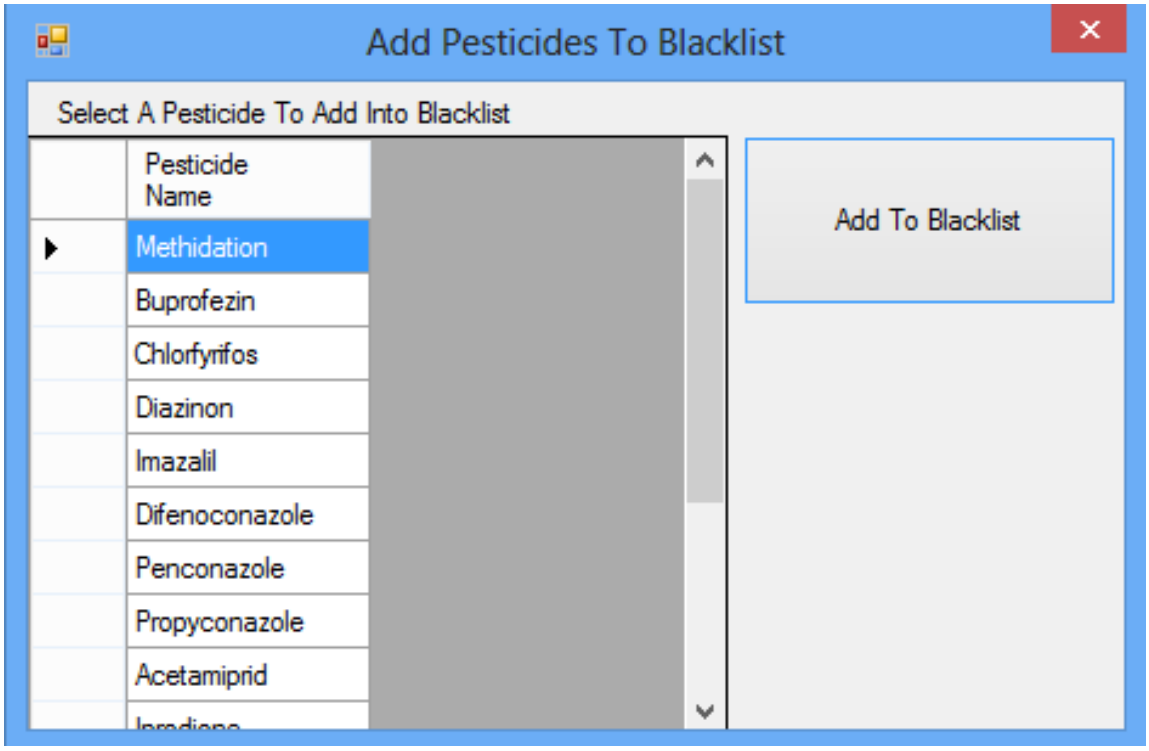

Figure 3.29: Add pesticides to blacklist sub window

In remove pesticide from black list sub window, there is one label, one datagridview, and one button. In order to remove pesiticides from black list, select a pesticide and click on remove button.

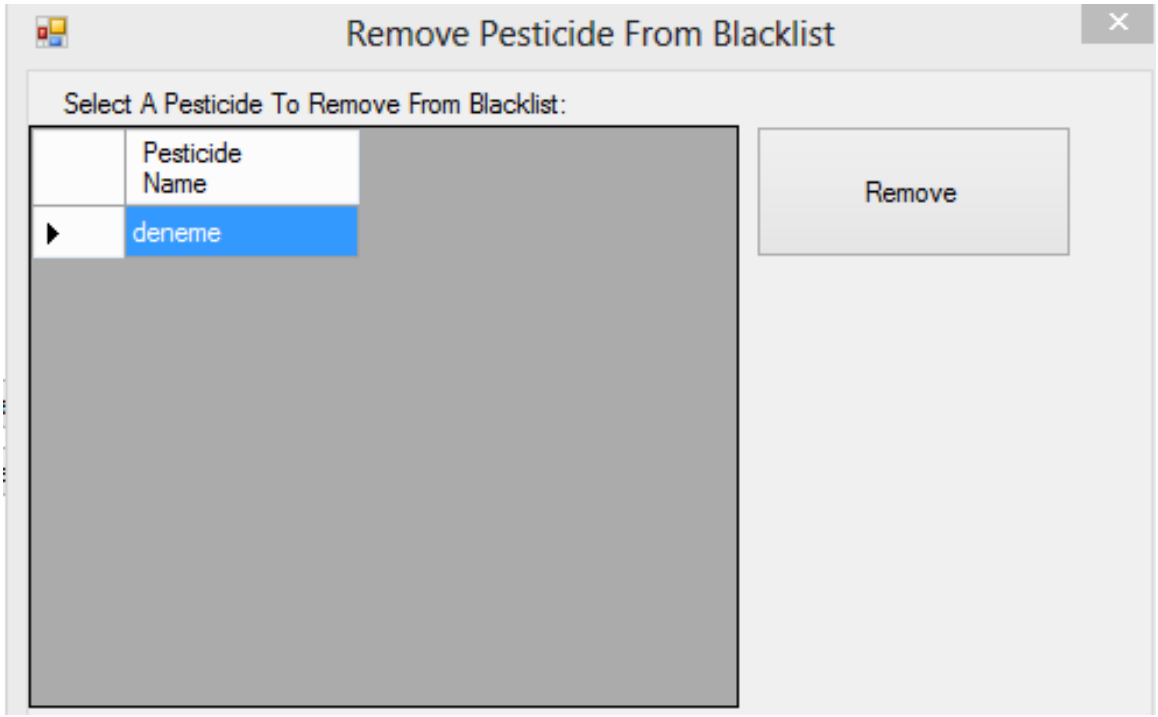

Figure 3.30: Remove pesticides from blacklist sub window

In add foods sub window, there is one label, one text box, and one button. In order to add a food, type a food name and click on add food button.

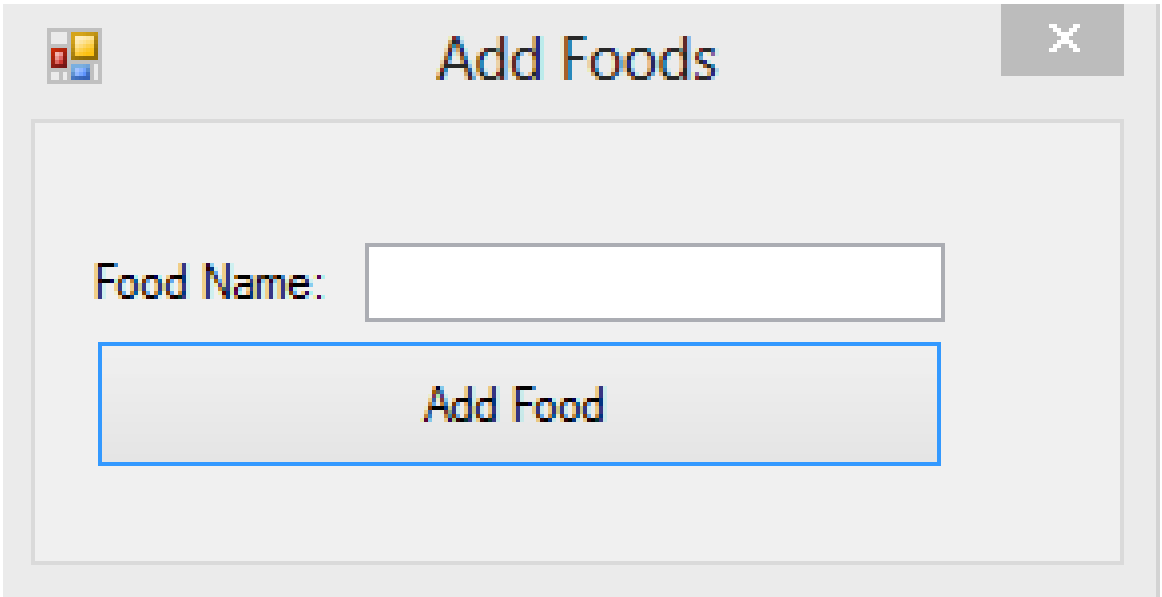

Figure 3.31: Add foods sub window

In remove foods sub window, there is one label, one data grid view, and one button. In order to remove a food, select a food from data grid view and click on remove food button.

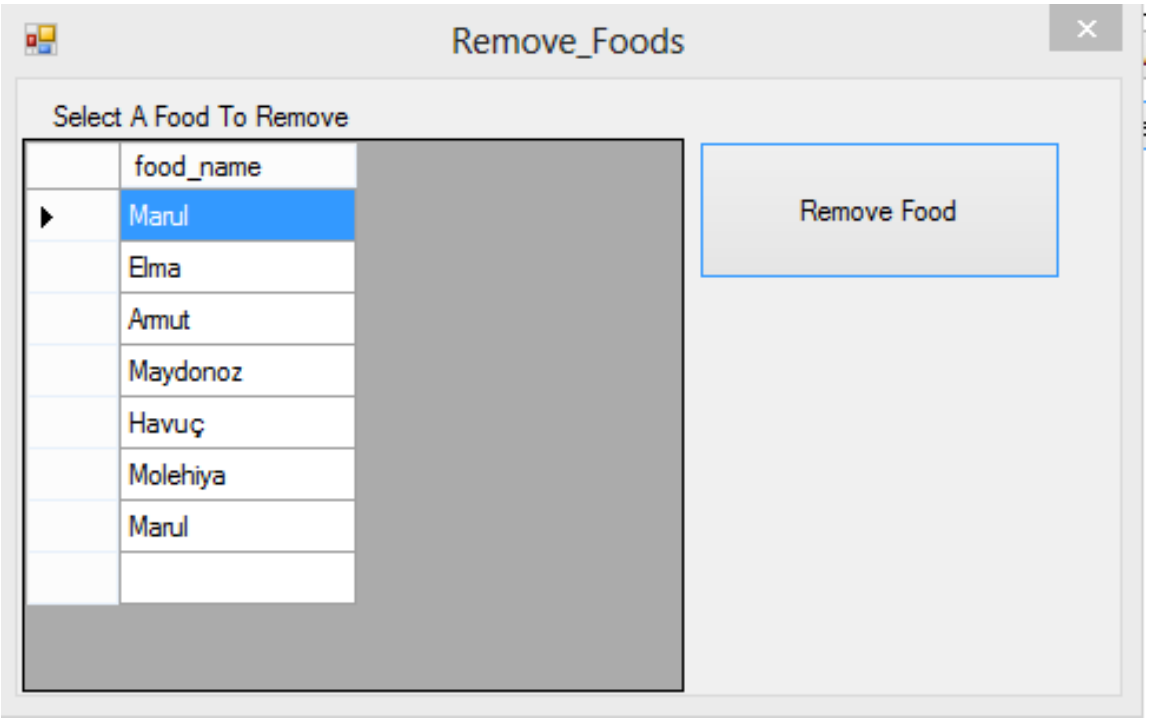

Figure 3.32: Remove foods sub window

In remove well results, there is one label, one data grid view, and one button. In order to remove a well result, select a well result from data grid view and click on remove button.

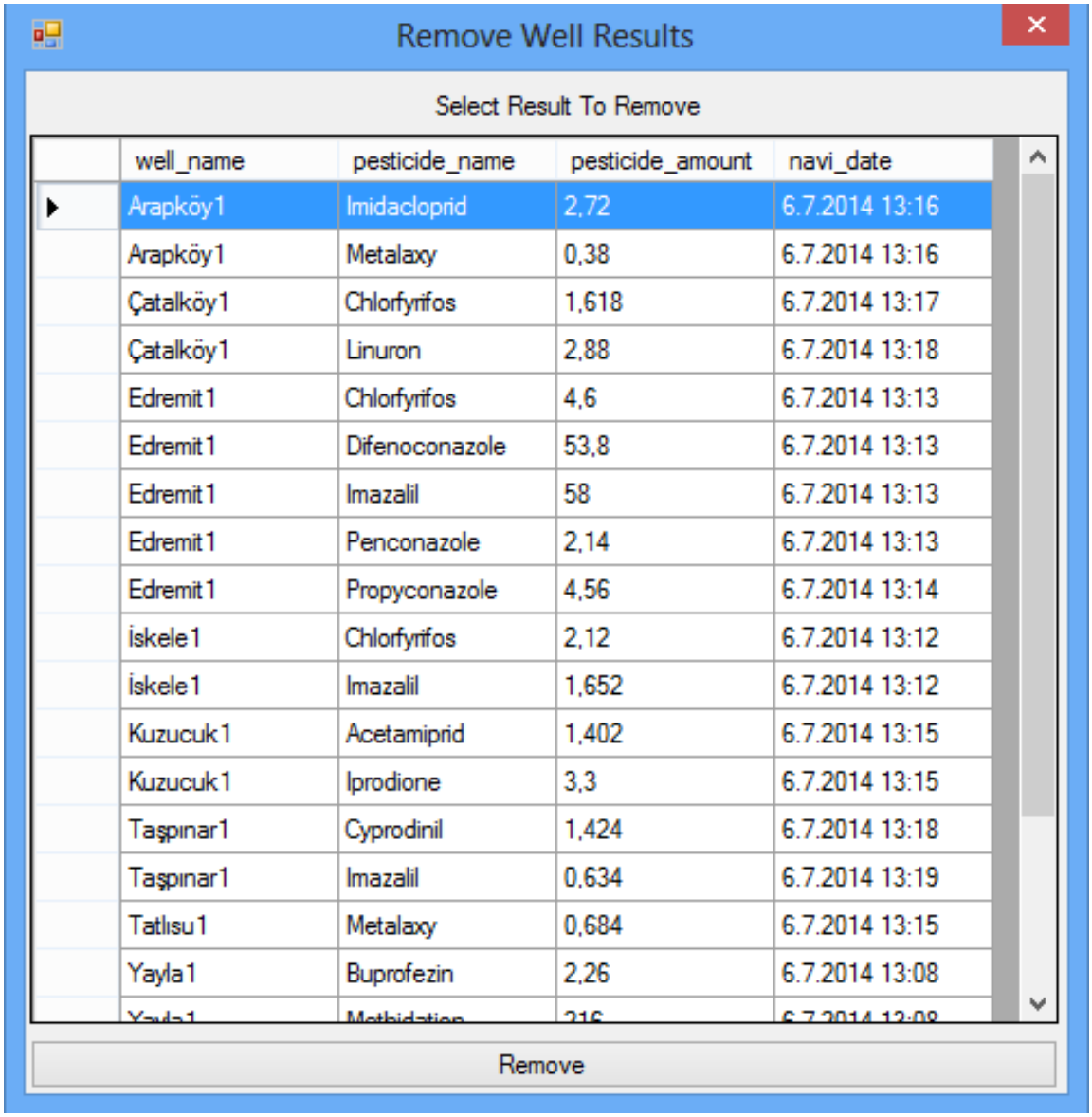

Figure 3.33: Remove well results sub window

In remove food test report, there are nine text box, two list box, seven text box, one check box, and one button. In order to remove a food test report, select a well then select a report and click on remove selected food test report button.

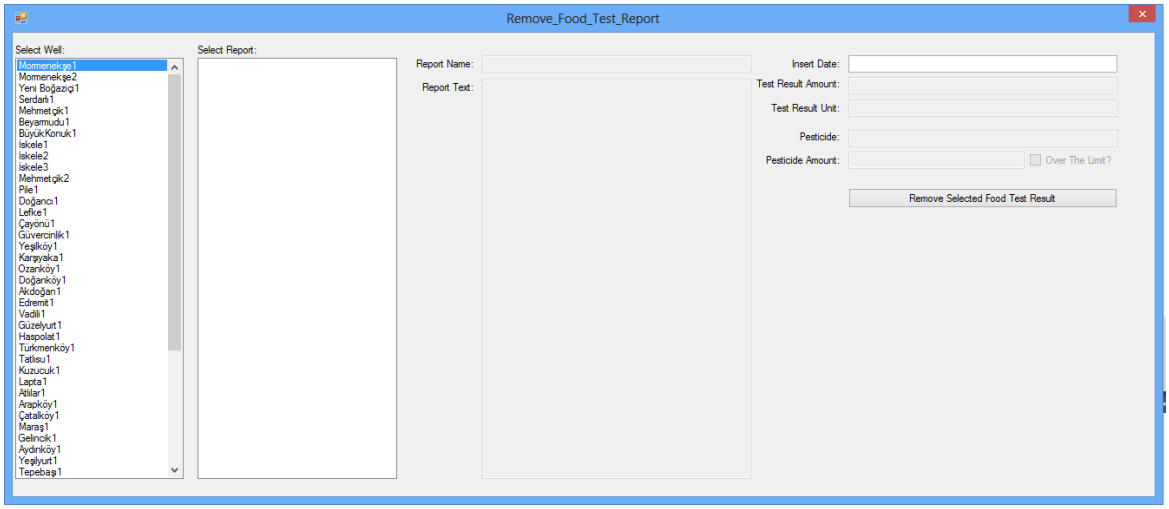

Figure 3.34: Remove food thest report sub window

CHAPTER FOUR

CONCLUSION

In this project, the main purpose is to develop a software which store historical water quality data in Northern Cyprus with the location of it is belong to. After the software development, users can store the location of wells, many types of pesticides, foods, analyses, and tests. Storing historical data helps users to make analyses about the orientation of quality of water. With this property, we can protect our waters and improve our life quality with many animals and plants.

The software is designed to be user friendly. Therefore, the software has verifications which appears after completing an action. The software helps to obtain accurate data.

The software can be used for a specific place or all around the world. If the database can be stored in a host, all users reach whole data via internet connection. This property increases the mobility of the software and usage capacity. Moreover, different goverments are enable to share the data between each other's and the historical data can be used all around the world.

After development of the software, the water analysis results are inserted into the database by using the software.

During the development of the software, the most difficult subject was designing a user friendy graphical user interface. The software has a complex database because of the fact that the database's tables are related to each other's. For example, during an insertion of a complete well information, you need to first insert the well in to the wells table to obtain the unique well_id as well as using the well_id the location of a well can be inserted in to the well positions. By using microsoft visual studio tools, the problems were solved in an effective way.

Connecting the database with any software contains very difficult actions and takes long time. In order to solve these problems, The LINQ was used.

This project sheds light on development of much cleverer and user friendly systems. New analysis and tests information are learned and the importance of a clean water are remembered again.

This project provides to store many data of water and food analysis. There are some advantages compared with traditional databases and software.

- The geographical location is not limited with only Cyprus. It can be used for all over the world.
- The software includes both google earth features and traditional databases.
- The software can be improved easily by the developers.

Hovewer, there are some disadvantages.

- Only developers can improve the software.
- The software needs internet connection to be mobile.

In this project, open source codes were used. Therefore, the new technology can be implemented easily. In Future, following properties can be included:

- Connecting a GPS (Global Positioning System) with GMAP software.
- Developing a new software for smartphones.

Connecting GPS is easliy to the current software. A serialport toolbox which is developed by Microsoft Visual Studio developers can be used in project to control gps system.

Another aproach is building a new software upon the current database. So the new development enviroment will be required. For example, Eclipse will be used to develop the new software. With the new android software gps or any other smartphone features will be used.

REFERENCES

Ajit, K., Parinar, S., Ajay, K., Gupta, R., Manoj, P., Archana, S., and Pandey, A. (2012). Physico-Chemical and Microbiological Analysis of Underground Water in and Around Gwalior City. *Research Journal of Recent Sciences, 1(6),* 62-65.

Dheeravath, S., Ramadevi, K., Saraswathi, Z., Maniklal, D., Bhagawan, D. (2013). RP-HPLC method development for simultaneous determination of the drugsRamipril and Amlodipine. *International Journal of Scientific Research,* https://www.worldwidejournals.com/international-journal-of-scientific-research-(IJSR)/file.php?val=February_2013_1359731148_c271d_123.pdf

Stiff, H. (1951). The Interpretation of Chemical Water Analysis by Means of Patterns. *Journal of Petroleum Technology, 3(10),* 1-4.

Madigan, M., Brock, D. (2009). Microorganisms and Microbiology, *Brock Biology of Microorganisms* 12(1), 77-78. San Francisco, CA: Pearson Benjamin Cummings.

Oymen, B. (2014). *Pesticide Residues In Ground Water of Northern Cyprus.* Nicosia, CYP.

Randall, C. (2013). *National Pestice Applicator Certification Core Manual*. Washington, DC: National Association of State Departments of Ariculture.

Randolp, N., Gardner, D., Minutillo, M., and Anderson, C. (2010). *Professional Visual Studio 2010.* New Jersey, NJ: Wiley Publishing.

Rattz, J. (2007). *Pro LINQ: Language Integrated Query in C# 2008.* New York, NY: Apress.

Reeve, R. (1994). *Environmental Analysis: Analytical Chemistry by Open Learning.* Chichester, UK: J. Wiley.

Richter, J. (2002). *Applied Microsoft .NET Framework.* Washington, DC: Microsoft Press.

Siegbahn, K. (1968). *Ray Spectroscopy*. New York, NY: North-Holland Publishing Company.

Vieria, R. (2008). *Beginning Microsoft SQL Server 2008 Programming.* Indianapolis, IN: Wiley Publishing.

Williams, M. (2002). *Microsoft Visual C#.* Washington, DC: Microsoft Press.

APPENDICES

APPENDIX A
CONNECTING DATABASE TO SOFTWARE

In Linq technology, data classes are generated automatically by a simple drag and drop operations. There is no need for an extra operation to create data classes. There is one data class file in this project. This file includes database tables and one connection string. Connection string is like a flight ticket which has a destination point and an identity.

A connection string example:

Data Source=mssql22.natro.com;Initial Catalog=DB140719101555;Persist Security Info=True;User ID=USR140719101555

Figure A.1 show the complete form of the project's data class file.

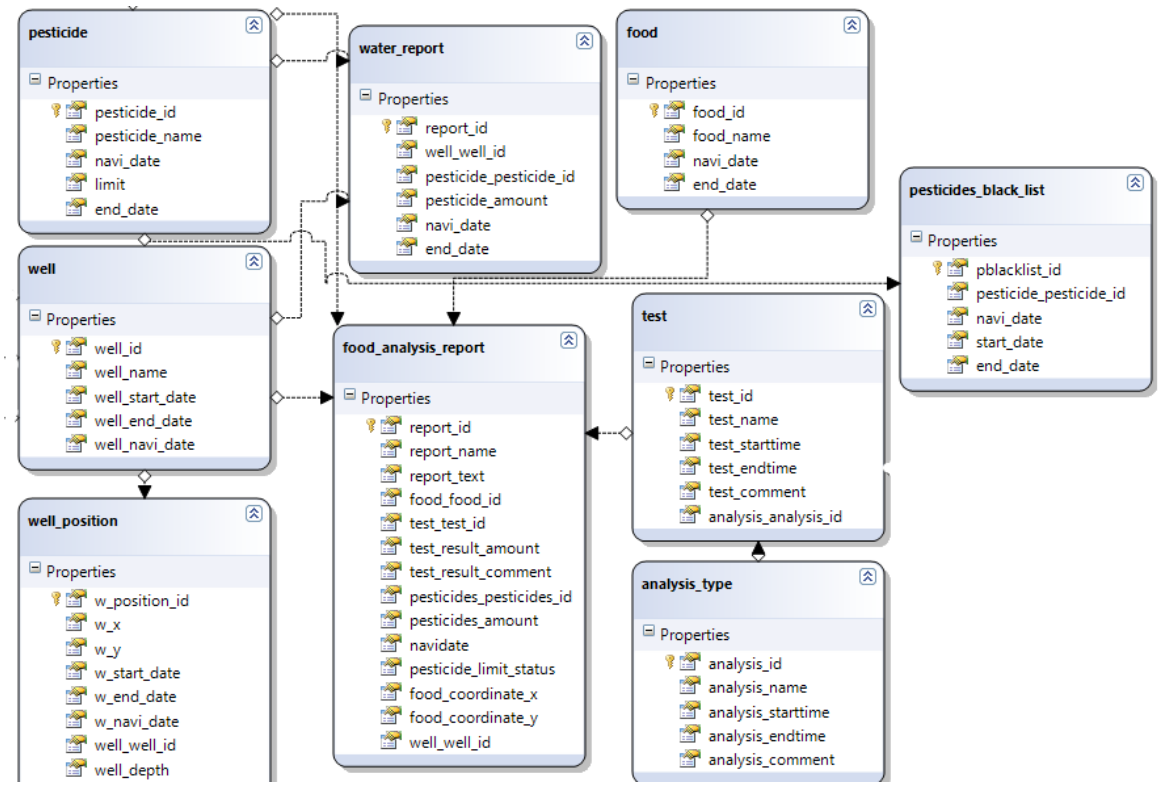

Figure A.1: Data class file (Dataclass.dbml)

APPENDIX B

CODES OF MAIN WINDOWS AND SUB WINDOWS

B.1 Codes of main window

The codes of main window maintain many actions which are select, update, delete and insert. The all codes are shown below.

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using GMap.NET.WindowsForms;

using GMap.NET.WindowsForms.Markers;

using GMap.NET;

using System.Net;

using System.Net.NetworkInformation;

namespace GMAP

{

    public partial class Form1 : Form
```

```
{
    public Form1()
    {
        InitializeComponent();
    }


    GMapOverlay markersoverlay1 = new GMapOverlay("markers");
    GMapOverlay markersoverlay2 = new GMapOverlay("markers");
    private void Form1_Load(object sender, EventArgs e)
    {
        // TODO: This line of code loads data into the 'dB140719101555DataSet.water_reports' table. You can move, or remove it, as needed
        Ping p = new Ping();
        PingReply pingstatus = p.Send(IPAddress.Parse("74.125.224.103"));
        if (pingstatus.Status == IPStatus.Success)
        {

        }
        else
        {
```

```
            MessageBox.Show("Verify Your Internet Connection And Try Again!");
            this.Close();
        }
        // TODO: This line of code loads data into the 'water_projectDataSet7.wells' table. You can move, or remove it, as needed.
        this.wellsTableAdapter2.Fill(this.water_projectDataSet7.wells);
        // TODO: This line of code loads data into the 'water_projectDataSet61.wells' table. You can move, or remove it, as needed.


        // TODO: This line of code loads data into the 'water_projectDataSet61.wells' table. You can move, or remove it, as needed.
        // Initialize map:
        gMapControl1.MapProvider = GMap.NET.MapProviders.GoogleHybridMapProvider.Instance;
        GMap.NET.GMaps.Instance.Mode = GMap.NET.AccessMode.ServerOnly;
        //gmap.SetCurrentPositionByKeywords("Maputo, Mozambique");
        //gMapControl1.SetPositionByKeywords("Maputo, Mozambique");
        //gmap.Position = new PointLatLng(-25.971684,32.589759);
        gMapControl1.Position = new GMap.NET.PointLatLng(35.232678, 33.144778);
        harita_isaretle();
    }
    DataClasses1DataContext dr = new DataClasses1DataContext();
```

```
well_position wp = new well_position();
void harita_isaretle()
{

    this.wellsTableAdapter2.Fill(this.water_projectDataSet7.wells);
    markersoverlay1.Clear();
    markersoverlay2.Clear();
    foreach (var item in this.water_projectDataSet7.wells)
    {
        if (item.well_end_date > DateTime.Now)
        {
          // wp = dr.well_positions.First(m => m.well_well_id == item.well_id);
           wp = dr.well_positions.First(x => x.well_well_id == item.well_id);
           if (wp.w_end_date > DateTime.Now)
           {

                GMarkerGoogle marker1 = new GMarkerGoogle(new PointLatLng(wp.w_y,
wp.w_x), GMarkerGoogleType.green);
                marker1.ToolTipText = item.well_name;
                markersoverlay1.Markers.Add(marker1);
```

```
            }}}


    gMapControl1.Overlays.Add(markersoverlay1);
    gMapControl1.Refresh();
    change = false;


}


void harita_isaretle2()
{


    this.wellsTableAdapter2.Fill(this.water_projectDataSet7.wells);
    markersoverlay1.Clear();
    markersoverlay2.Clear();//bu
    foreach (var item in this.water_projectDataSet7.wells)
    {
        if (item.well_end_date > DateTime.Now)
        {
            wp = dr.well_positions.First(x => x.well_well_id == item.well_id);
```

```
            if (wp.w_end_date > DateTime.Now)
            {

                GMarkerGoogle marker2 = new GMarkerGoogle(new PointLatLng(wp.w_y,
wp.w_x), GMarkerGoogleType.green);
                marker2.ToolTipText = item.well_name;
                markersoverlay2.Markers.Add(marker2);

            }}}

        gMapControl1.Overlays.Add(markersoverlay2);
        gMapControl1.Refresh();
        change = true;

    }

    private void button1_Click(object sender, EventArgs e)
    {
        Add_Tests t = new Add_Tests();
        t.ShowDialog();
    }
```

```
private void button2_Click(object sender, EventArgs e)
{
    Remove_Tests rt = new Remove_Tests();
    rt.ShowDialog();
}

private void button4_Click(object sender, EventArgs e)
{
    Add_Analysis aa = new Add_Analysis();
    aa.ShowDialog();
}

private void button3_Click(object sender, EventArgs e)
{
    Remove_Analysis ra = new Remove_Analysis();
    ra.ShowDialog();

}
```

```
private void button6_Click(object sender, EventArgs e)
{
    Add_Well aw = new Add_Well();
    aw.ShowDialog();
}

private void button5_Click(object sender, EventArgs e)
{
    Remove_Well rw = new Remove_Well();
    rw.ShowDialog();
}

private void button8_Click(object sender, EventArgs e)
{
    Add_Pesticides ap = new Add_Pesticides();
    ap.ShowDialog();
}

private void button7_Click(object sender, EventArgs e)
{
```

```
    Remove_Pesticide rp = new Remove_Pesticide();
    rp.ShowDialog();
}

private void button10_Click(object sender, EventArgs e)
{
    Add_Foods af = new Add_Foods();
    af.ShowDialog();
}

private void button9_Click(object sender, EventArgs e)
{
    Remove_Foods rf = new Remove_Foods();
    rf.ShowDialog();
}

private void button12_Click(object sender, EventArgs e)
{
    Add_Extraction_Types aet = new Add_Extraction_Types();
    aet.ShowDialog();
```

```
}

private void gMapControl1_OnPositionChanged(PointLatLng point)
{

    label7.Text = Convert.ToString( gMapControl1.Position.Lat);
    label8.Text = Convert.ToString(gMapControl1.Position.Lng);

}
public double lat;
public double lng;
public bool fmap;
private void gMapControl1_DoubleClick(object sender, EventArgs e)
{

      lat = gMapControl1.Position.Lat;
      lng = gMapControl1.Position.Lng;
      fmap = true;
      Add_Well aw = new Add_Well();
```

```
aw.gfmap = fmap;

aw.glat = lat;

aw.glng = lng;

aw.ShowDialog();

degisiklikvarmi = aw.change;


if (change == true)

{

   if (degisiklikvarmi != false)

   {

      harita_isaretle();

      degisiklikvarmi = false;

   }}

else if (change == false)

{

   if (degisiklikvarmi != false)

   {

      harita_isaretle2();

      degisiklikvarmi = false;

   }}
```

```
private void tabControl1_Click(object sender, EventArgs e)
{
    gMapControl1.Refresh();
    harita_isaretle();

}
public int gi_well_id;
public int gi_well_p_id;
bool change;
public bool degisiklikvarmi;
private void gMapControl1_OnMarkerClick(GMapMarker item, MouseEventArgs e)
{

    DataClasses1DataContext df = new DataClasses1DataContext();
    well_position wpa = new well_position();
    well ww = new well();
    Options op = new Options();

        wpa = df.well_positions.First(a => a.w_x == item.Position.Lng);
```

```
gi_well_p_id = wpa.w_position_id;


ww = df.wells.First(c => c.well_id == wpa.well_well_id);
gi_well_id = ww.well_id;


op.gwell_p_id = gi_well_p_id;
op.gwell_id = gi_well_id;
   op.ShowDialog();
   gi_well_id = 0;
   gi_well_id = 0;
   degisiklikvarmi = op.action;
   if (change == true)
   {
      if (degisiklikvarmi!=false)
      {
         harita_isaretle();
         degisiklikvarmi = false;
      }
```

```
            }
            else if (change==false)
            {
               if (degisiklikvarmi!=false)
               {
                  // harita_isaretle2();
                   degisiklikvarmi = false;
               }
    }}

private void button14_Click(object sender, EventArgs e)
{
    Add_Pesticides_To_Blacklist apb = new Add_Pesticides_To_Blacklist();
    apb.ShowDialog();
}

private void button13_Click(object sender, EventArgs e)
{
    Remove_Pesticide_From_Blacklist rpb = new Remove_Pesticide_From_Blacklist();
```

```
            rpb.ShowDialog();
        }

        private void button15_Click(object sender, EventArgs e)
        {
            //Remove_Well_Test_Report_Results rpa = new Remove_Well_Test_Report_Results();
            Remove_Well_Results rpa = new Remove_Well_Results();
            rpa.ShowDialog();
        }

        private void button16_Click(object sender, EventArgs e)
        {
            Remove_Food_Test_Report rpf = new Remove_Food_Test_Report();
            rpf.ShowDialog();
        }

        private void button17_Click(object sender, EventArgs e)
        {
            Update_Well_Depth uwd = new Update_Well_Depth();
            uwd.ShowDialog();
```

```
}

private void button18_Click(object sender, EventArgs e)
{
    Update_Pesticides up = new Update_Pesticides();
    up.ShowDialog();
}

private void button18_Click_1(object sender, EventArgs e)
{
    Update_Pesticides up = new Update_Pesticides();
    up.ShowDialog();
}

private void button19_Click(object sender, EventArgs e)
{
    dataGridView2.DataSource = null;
    DataClasses1DataContext dt = new DataClasses1DataContext();

    var q1 = dt.water_reports.Join
```

```
            (dt.wells, wel => wel.well_well_id, wr => wr.well_id, (wel, wr) => new
            {
                wel.well_well_id,
                wr.well_name,
                wel.pesticide_pesticide_id,
                wel.pesticide_amount
            }
            )
            .Join(dt.pesticides, a => a.pesticide_pesticide_id, b => b.pesticide_id, (a, b) =>
new
            {
                a.well_well_id,
                a.well_name,
                b.pesticide_name,
                b.limit,
                a.pesticide_amount

            }).Where(x => x.pesticide_amount >= x.limit);

        dataGridView2.DataSource = q1;
    }
```

```
private void button20_Click(object sender, EventArgs e)
{
    dataGridView2.DataSource = null;
    DataClasses1DataContext dt = new DataClasses1DataContext();
    var q2 = dt.wells.Where(x => x.well_end_date < Convert.ToDateTime("31-12-2999"));
    dataGridView2.DataSource = q2;
}

private void button21_Click(object sender, EventArgs e)
{
    dataGridView2.DataSource = null;
    DataClasses1DataContext dt = new DataClasses1DataContext();
    var q3 = dt.pesticides_black_lists.Join
        (dt.pesticides, pbl => pbl.pesticide_pesticide_id, pes => pes.pesticide_id, (pbl, pes) => new
        {
            pes.pesticide_name,
            pbl.navi_date,
            pbl.start_date,
```

```
            pbl.end_date
        }).Where(x => x.end_date < Convert.ToDateTime("31-12-2999"));
    dataGridView2.DataSource = q3;


}


private void button22_Click(object sender, EventArgs e)
{
    dataGridView2.DataSource = null;
    DataClasses1DataContext dt = new DataClasses1DataContext();



    var q1 = dt.water_reports.Join
        (dt.wells, wel => wel.well_well_id, wr => wr.well_id, (wel, wr) => new
        {
            wel.well_well_id,
            wr.well_name,
            wel.pesticide_pesticide_id,
            wel.pesticide_amount
        }
```

```
            )
            .Join(dt.pesticides, a => a.pesticide_pesticide_id, b => b.pesticide_id, (a, b) =>
new
            {
                a.well_well_id,
                a.well_name,
                b.pesticide_name,
                b.limit,
                a.pesticide_amount

            }).Where(x => x.pesticide_amount < x.limit);

          dataGridView2.DataSource = q1;
      }
   }
}
```

B.2 Codes of sub windows

B.2.1 Add analysis sub window codes

```
using System;
using System.Collections.Generic;
```

```
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
   public partial class Add_Analysis : Form
   {
      public Add_Analysis()
      {
         InitializeComponent();
      }

      private void button1_Click(object sender, EventArgs e)
      {
         DataClasses1DataContext dt = new DataClasses1DataContext();
         analysis_type t = new analysis_type();
         t.analysis_name = textBox1.Text;
```

```
            t.analysis_comment = textBox2.Text;
            t.analysis_starttime = DateTime.Now;
            t.analysis_endtime = Convert.ToDateTime("31-12-2999");
            dt.analysis_types.InsertOnSubmit(t);
            dt.SubmitChanges();
            textBox1.Text = null;
            textBox2.Text = null;
            MessageBox.Show("Typed Analyse Is Successfully Added");

        }

        private void textBox1_TextChanged(object sender, EventArgs e)
        {
            if (textBox1.Text == null || textBox1.Text == " " || textBox1.Text == "  " ||
textBox1.Text == "")
            {
                button1.Enabled = false;
            }
            else
            {
                button1.Enabled = true;
```

```
        }
    }

    private void Add_Analysis_Load(object sender, EventArgs e)
    {}}}
```

B.2.2 Add food test report sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
    public partial class Add_Food_Test_Report : Form
    {
        public Add_Food_Test_Report()
```

```
        {
            InitializeComponent();
        }
        public int rf_well_id;
        private void Add_Food_Test_Report_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'water_projectDataSet16.foods' table. You can move, or remove it, as needed.
            this.foodsTableAdapter.Fill(this.water_projectDataSet16.foods);
            // TODO: This line of code loads data into the 'water_projectDataSet10.pesticides' table. You can move, or remove it, as needed.
            this.pesticidesTableAdapter.Fill(this.water_projectDataSet10.pesticides);
            // TODO: This line of code loads data into the 'water_projectDataSet9.tests' table. You can move, or remove it, as needed.
            this.testsTableAdapter.Fill(this.water_projectDataSet9.tests);
            // TODO: This line of code loads data into the 'water_projectDataSet8.analysis_types' table. You can move, or remove it, as needed.
            this.analysis_typesTableAdapter.Fill(this.water_projectDataSet8.analysis_types);
}

        private void button1_Click(object sender, EventArgs e)
        {
```

```
        DataClasses1DataContext gf = new DataClasses1DataContext();
        food_analysis_report far = new food_analysis_report();
        far.report_name = textBox1.Text;
        far.report_text = textBox2.Text;
        far.test_test_id = Convert.ToInt32(comboBox2.SelectedValue.ToString());
        far.well_well_id = rf_well_id;
        far.test_result_amount = Convert.ToDouble(maskedTextBox1.Text.Replace(" ",
""));
        far.test_result_comment = textBox3.Text;
        far.pesticides_pesticides_id =
Convert.ToInt32(comboBox3.SelectedValue.ToString());
        far.pesticides_amount = Convert.ToDouble(maskedTextBox2.Text.Replace(" ",
""));
        far.food_food_id = Convert.ToInt32(comboBox4.SelectedValue.ToString());
        if (checkBox1.Checked == true)
        {
           far.pesticide_limit_status = 1;
        }
        else
        {
           far.pesticide_limit_status = 0;
```

```
        }
        far.navidate = DateTime.Now;
        gf.food_analysis_reports.InsertOnSubmit(far);
        gf.SubmitChanges();

        MessageBox.Show("Food Analysis Report Successfully Added.");
    }

    private void comboBox1_SelectedIndexChanged_1(object sender, EventArgs e)
    {
        DataClasses1DataContext fd = new DataClasses1DataContext();
        var query = fd.tests.Where(x => x.analysis_analysis_id == Convert.ToInt32(comboBox1.SelectedValue.ToString()));
        comboBox2.DataSource = query;
        comboBox2.DisplayMember = "test_name";
        comboBox2.ValueMember = "test_id";
        // MessageBox.Show(rwell_id +"   " +comboBox2.SelectedValue.ToString());
    }

    private void maskedTextBox1_MaskInputRejected(object sender, MaskInputRejectedEventArgs e)
```

```
        {


        }}}
```

B.2.3 Add foods sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;


namespace GMAP
{
    public partial class Add_Foods : Form
    {
        public Add_Foods()
        {
```

```
        InitializeComponent();
    }
    private void Foods_Load(object sender, EventArgs e)
    {}
    private void button1_Click(object sender, EventArgs e)
    {
        DataClasses1DataContext dt = new DataClasses1DataContext();
        food f =new food();
        f.food_name = textBox1.Text;
        f.navi_date = DateTime.Now;
        f.end_date = Convert.ToDateTime("31-12-2999");
        dt.foods.InsertOnSubmit(f);
        dt.SubmitChanges();
        textBox1.Text = null;
        MessageBox.Show("Typed Food is Successfully Added");
    }
    private void textBox1_TextChanged(object sender, EventArgs e)
    {
        if (textBox1.Text == null || textBox1.Text == " " || textBox1.Text == "   " ||
textBox1.Text == "")
        {
```

```
            button1.Enabled = false;
        }
        else
        {
            button1.Enabled = true;
        }
    }
    private void label1_Click(object sender, EventArgs e)
    {


    } } }
```

B.2.4 Add pesticides sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
```

```
using System.Windows.Forms;

namespace GMAP
{
    public partial class Add_Pesticides : Form
    {
        public Add_Pesticides()
        {
            InitializeComponent();
        }
        private void button1_Click(object sender, EventArgs e)
        {
            DataClasses1DataContext dt = new DataClasses1DataContext();
            pesticide p = new pesticide();
            p.pesticide_name = textBox1.Text;
            if (is_changed == true)
            {
                p.limit = Convert.ToDouble(maskedTextBox1.Text.Replace(" ", ""));
            }
            p.navi_date = DateTime.UtcNow;
            p.end_date = Convert.ToDateTime("31-12-2999");
```

```
            dt.pesticides.InsertOnSubmit(p);
            dt.SubmitChanges();
            textBox1.Text = null;
            maskedTextBox1.Text = null;
            is_changed = false;
            MessageBox.Show("Typed Pesticide Is Successfully Added");
}

        private void textBox1_TextChanged(object sender, EventArgs e)
        {
            if (textBox1.Text == null || textBox1.Text == " " || textBox1.Text == "   " ||
textBox1.Text == "")
            {
                button1.Enabled = false;
            }
            else
            {
                button1.Enabled = true;
            }
        }
        private void Add_Pesticides_Load(object sender, EventArgs e)
```

```
    {}
    public bool is_changed;
    private void maskedTextBox1_TextChanged(object sender, EventArgs e)
    {
       is_changed = true;
    }}}
```

B.2.5 Add pesticides to black list sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
   public partial class Add_Pesticides_To_Blacklist : Form
   {
```

```
public Add_Pesticides_To_Blacklist()
{
    InitializeComponent();
}
private void label1_Click(object sender, EventArgs e)
{}
private void Add_Pesticides_To_Blacklist_Load(object sender, EventArgs e)
{
    // TODO: This line of code loads data into the 'water_projectDataSet11.pesticides' table. You can move, or remove it, as needed.
    this.pesticidesTableAdapter.Fill(this.water_projectDataSet11.pesticides);}
private void button1_Click(object sender, EventArgs e)
{
    DataClasses1DataContext dr = new DataClasses1DataContext();
    pesticide p = dr.pesticides.First(x => x.pesticide_id == int.Parse(dataGridView1.Rows[dataGridView1.CurrentRow.Index].Cells[0].Value.ToString()));
    pesticides_black_list pb = new pesticides_black_list();
    try
    {
        pb = dr.pesticides_black_lists.First(a => a.pesticide_pesticide_id == p.pesticide_id && a.end_date > DateTime.Now);
```

```
        }
        catch (Exception ex)
        {}
        if (pb.pesticide_pesticide_id == p.pesticide_id)
        {
            MessageBox.Show("Selected Pesticide Is Already Added To Blacklist, Select Another One...");
        }
        else
        {
            pesticides_black_list pbl = new pesticides_black_list();
            pbl.pesticide_pesticide_id = p.pesticide_id;
            pbl.start_date = DateTime.Now;
            pbl.end_date = Convert.ToDateTime("31-12-2999");
            pbl.navi_date = DateTime.Now;
            dr.pesticides_black_lists.InsertOnSubmit(pbl);
            dr.SubmitChanges();
            MessageBox.Show("Selected Pesticide Is Successfully Added To Blacklist");
        }
        }
```

```
        private void dataGridView1_SelectionChanged(object sender, EventArgs e)
        {
            button1.Enabled = true;
        }}}
```

B.2.6 Add tests sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
    public partial class Add_Tests : Form
    {
        public Add_Tests()
        {
```

```
            InitializeComponent();
        }
        private void Add_Tests_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'water_projectDataSet2.analysis_types' table. You can move, or remove it, as needed.
            this.analysis_typesTableAdapter.Fill(this.water_projectDataSet2.analysis_types);
        }
        private void button1_Click(object sender, EventArgs e)
        {
            if (textBox1.Text != null)
            {
                DataClasses1DataContext dc = new DataClasses1DataContext();
                test t = new test();
                t.test_name = textBox1.Text;
                t.test_comment = textBox2.Text;
                t.test_starttime = DateTime.Now;
                t.test_endtime = Convert.ToDateTime("31-12-2999");
                t.analysis_analysis_id = int.Parse(dataGridView1.Rows[dataGridView1.CurrentRow.Index].Cells[0].Value.ToString());
```

```
            dc.tests.InsertOnSubmit(t);
            dc.SubmitChanges();
            textBox1.Text = null;
            textBox2.Text = null;
            MessageBox.Show("Typed Test Is Successfully Added");
        }
        else
        {
            MessageBox.Show("You Should Write A Test Name! ");
        }}
    private void textBox1_TextChanged(object sender, EventArgs e)
    {
        if (textBox1.Text == null || textBox1.Text == " " || textBox1.Text == "   "
||textBox1.Text=="")
        {
            button1.Enabled = false;
        }
        else
        {
            button1.Enabled = true;
        }
```

```
    }}}
```

B.2.7 Add wells sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
   public partial class Add_Well : Form
   {
      public Add_Well()
      {
         InitializeComponent();
      }
      public bool change;
```

```
private void button1_Click(object sender, EventArgs e)
{
    DataClasses1DataContext dv = new DataClasses1DataContext();
    well w = new well();
    w.well_name = textBox1.Text;
    w.well_start_date = DateTime.Now;
    w.well_end_date = Convert.ToDateTime("31-12-2999");
    w.well_navi_date = DateTime.Now;
    dv.wells.InsertOnSubmit(w);
    dv.SubmitChanges();
    DataClasses1DataContext dxy = new DataClasses1DataContext();
    well_position wp = new well_position();
    wp.w_x = Convert.ToDouble(textBox2.Text);
    wp.w_y = Convert.ToDouble(textBox3.Text);
    well ww = dxy.wells.First(x => x.well_name == textBox1.Text);
    wp.well_well_id = ww.well_id;
    wp.w_navi_date = DateTime.Now;
    wp.w_start_date = DateTime.Now;

    wp.well_depth = Convert.ToDouble(maskedTextBox1.Text.Replace(" ",""));
```

```
    wp.w_end_date = Convert.ToDateTime("31-12-2999");
    dxy.well_positions.InsertOnSubmit(wp);
    dxy.SubmitChanges();
    textBox1.Text = null;
    textBox2.Text = null;
    textBox3.Text = null;
    maskedTextBox1.Text = null;
    MessageBox.Show("Typed Well is Successfully Added!");
    Form1 f = new Form1();
    change = true;
    f.degisiklikvarmi = chang
    if (gfmap==true)
    {
        gfmap = false;
        this.Close();
    }

}
private void textBox1_TextChanged(object sender, EventArgs e)
{
```

```
        if (textBox2.Text == null || textBox2.Text == " " || textBox2.Text == "  " ||
textBox2.Text == "" || textBox3.Text == null || textBox3.Text == " " || textBox3.Text == "
" || textBox3.Text == "" || textBox1.Text == null || textBox1.Text == " " || textBox1.Text ==
"  " || textBox1.Text == "")
        {
            button1.Enabled = false;
        }
        else
        {
            button1.Enabled = true;
        }
    }
    private void textBox2_TextChanged(object sender, EventArgs e)
    {
        if (textBox2.Text == null || textBox2.Text == " " || textBox2.Text == "  " ||
textBox2.Text == "" || textBox3.Text == null || textBox3.Text == " " || textBox3.Text == "
" || textBox3.Text == "" || textBox1.Text == null || textBox1.Text == " " || textBox1.Text ==
"  " || textBox1.Text == "")
        {
            button1.Enabled = false;
        }
        else
```

```
        {
            button1.Enabled = true;
        }}
    private void textBox3_TextChanged(object sender, EventArgs e)
    {
        if (textBox2.Text == null || textBox2.Text == " " || textBox2.Text == "  " || textBox2.Text == "" || textBox3.Text == null || textBox3.Text == " " || textBox3.Text == " " || textBox3.Text == "" || textBox1.Text == null || textBox1.Text == " " || textBox1.Text == "  " || textBox1.Text == "")
        {
            button1.Enabled = false;
        }
        else
        {
            button1.Enabled = true;
        }
    }
    public double glat;
    public double glng;
    public bool gfmap;
    private void Add_Well_Load(object sender, EventArgs e)
```

```
    {


        maskedTextBox1.ValidatingType = typeof(double);

            textBox2.Text = Convert.ToString(glng);

            textBox3.Text = Convert.ToString(glat);

        }}}
```

B.2.8 Add well results sub window codes

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace GMAP

{

   public partial class Add_Well_Results : Form

   {
```

```
public Add_Well_Results()
{
    InitializeComponent();
}
public int rwell_id;
private void Add_Well_Results_Load(object sender, EventArgs e)
{
    // TODO: This line of code loads data into the 'water_projectDataSet19.pesticides' table. You can move, or remove it, as needed.
    this.pesticidesTableAdapter.Fill(this.water_projectDataSet19.pesticides);
}
private void button1_Click(object sender, EventArgs e)
{
    DataClasses1DataContext dr = new DataClasses1DataContext();
    water_report wr = new water_report();
    wr.well_well_id = rwell_id;
    wr.pesticide_amount = Convert.ToDouble(maskedTextBox1.Text.Replace(" ", ""));
    wr.pesticide_pesticide_id = Convert.ToInt32(listBox1.SelectedValue.ToString());
    wr.navi_date = DateTime.Now;
    wr.end_date = Convert.ToDateTime("31-12-2999");
    dr.water_reports.InsertOnSubmit(wr);
```

```
            dr.SubmitChanges();
            MessageBox.Show("Well Report Successfully Added.");
            maskedTextBox1.Text = null;
    }}}
```

B.2.9 Options sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
    public partial class Options : Form
    {
        public Options()
        {
```

```
    InitializeComponent();
}
public int gwell_id;
public int gwell_p_id;
public bool action2;
private void Options_Load(object sender, EventArgs e)
{

        radioButton4.Checked = true;
}
public bool action;
private void button1_Click(object sender, EventArgs e)
{
    Form1 f = new Form1();
    if (radioButton3.Checked==true)
    {
        DataClasses1DataContext df = new DataClasses1DataContext();
        well_position wp = df.well_positions.First(x => x.w_position_id == gwell_p_id);
        wp.w_end_date = DateTime.Now.Date;
        df.SubmitChanges();
```

```
    DataClasses1DataContext df2 = new DataClasses1DataContext();
    well w = df2.wells.First(y => y.well_id == gwell_id);
    w.well_end_date = DateTime.Now.Date;
    df2.SubmitChanges();
    MessageBox.Show("Selected Well Is Successfully Removed With It's Position");
   // this.Load += new System.EventHandler(this.Form1_Load);

    this.Close();
    action = true;
    f.degisiklikvarmi = action;
}
else if (radioButton2.Checked==true)
{
    action = false;
    f.degisiklikvarmi = action;
    Add_Food_Test_Report aft = new Add_Food_Test_Report();
    aft.rf_well_id = gwell_id;
    aft.ShowDialog();
}
else if (radioButton1.Checked==true)
```

```
{
    action = false;
   //eski pencre için
    //f.degisiklikvarmi = action;
    //Add_Well_Test_Report awt = new Add_Well_Test_Report();
    //awt.rwell_id = gwell_id;
    //awt.ShowDialog();
    //eski pencere için
    f.degisiklikvarmi = action;
    Add_Well_Results awr = new Add_Well_Results();
    awr.rwell_id = gwell_id;
    awr.ShowDialog();
}
else if (radioButton4.Checked == true)
{
    this.Close();
    action = false;
    f.degisiklikvarmi = action;
}}
private void Options_FormClosed(object sender, FormClosedEventArgs e)
```

```
    { }

    private void Options_FormClosing(object sender, FormClosingEventArgs e)

    {}}}
```

B.2.10 Remove well results sub window codes

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace GMAP

{

  public partial class Remove_Well_Results : Form

  {

    public Remove_Well_Results()

    {

      InitializeComponent();
```

```
    }
    private void Remove_Well_Results_Load(object sender, EventArgs e)
    {
        // TODO: This line of code loads data into the 'water_projectDataSet22.water_reports' table. You can move, or remove it, as needed.
        this.water_reportsTableAdapter.Fill(this.water_projectDataSet22.water_reports);

    }
    private void button1_Click(object sender, EventArgs e)
    {
        DataClasses1DataContext dm = new DataClasses1DataContext();
        water_report wr = dm.water_reports.First(x => x.report_id == int.Parse(dataGridView1.Rows[dataGridView1.CurrentRow.Index].Cells[0].Value.ToString()));
        //dm.water_reports.DeleteOnSubmit(wr);
        wr.end_date = DateTime.Now.Date;
        dm.SubmitChanges();
        this.water_reportsTableAdapter.Fill(this.water_projectDataSet22.water_reports);
        MessageBox.Show("Report Is Successfully Removed");
    }}}
```

B.2.11 Remove analysis sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
   public partial class Remove_Analysis : Form
   {
      public Remove_Analysis()
      {
         InitializeComponent();
      }
      private void Remove_Analysis_Load(object sender, EventArgs e)
      {
         // TODO: This line of code loads data into the 'water_projectDataSet2.analysis_types' table. You can move, or remove it, as needed.
```

```
            this.analysis_typesTableAdapter.Fill(this.water_projectDataSet2.analysis_types);
        }
        private void button1_Click(object sender, EventArgs e)
        {
            DataClasses1DataContext dt = new DataClasses1DataContext();
            analysis_type t = dt.analysis_types.First(x => x.analysis_id == int.Parse(dataGridView1.Rows[dataGridView1.CurrentRow.Index].Cells[0].Value.ToString()));
            t.analysis_endtime = DateTime.Now.Date;
            dt.SubmitChanges();
this.analysis_typesTableAdapter.Fill(this.water_projectDataSet2.analysis_types);
            MessageBox.Show("Selected Analyse Is Successfully Removed");
        }
        private void dataGridView1_SelectionChanged(object sender, EventArgs e)
        {
            button1.Enabled = true;
        }}}
```

B.2.12 Remove food test report sub window codes

```
using System;
using System.Collections.Generic;
```

```
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
    public partial class Remove_Food_Test_Report : Form
    {
        public Remove_Food_Test_Report()
        {
            InitializeComponent();
        }
        private void Remove_Food_Test_Report_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'water_projectDataSet17.wells' table.
You can move, or remove it, as needed.
            this.wellsTableAdapter.Fill(this.water_projectDataSet17.wells);
            }
```

```
private void listBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    try
    {
        listBox2.DataSource = null;
        listBox2.Items.Clear();
        DataClasses1DataContext dm = new DataClasses1DataContext();
        listBox2.DataSource = dm.food_analysis_reports.Where(x => x.well_well_id ==
Convert.ToInt32(listBox1.SelectedValue));
        listBox2.DisplayMember = "report_name";
        //listBox2.SelectedItem = "report_id";
        //listBox2.SelectedValue = "report_id";
        listBox2.ValueMember = "report_id";
    }
    catch (Exception ev)
    {}}
private void listBox2_SelectedIndexChanged(object sender, EventArgs e)
{
    try
    {
        listBox2.DisplayMember = "report_name";
```

```
            //listBox2.SelectedItem = "report_id";
            //  listBox2.SelectedValue = "report_id";
            listBox2.ValueMember = "report_id";
            //MessageBox.Show( listBox2.SelectedIndices[0].ToString());
            DataClasses1DataContext dma = new DataClasses1DataContext();
            food_analysis_report war = dma.food_analysis_reports.First(v => v.report_id ==
Convert.ToInt32(listBox2.SelectedValue));
            textBox1.Text = war.report_name;
            textBox2.Text = war.report_text;
            textBox3.Text = war.test_result_comment;
            pesticide p = dma.pesticides.First(c => c.pesticide_id ==
war.pesticides_pesticides_id);
            textBox4.Text = p.pesticide_name;
            textBox5.Text = Convert.ToString(war.test_result_amount);
            textBox6.Text = Convert.ToString(war.pesticides_amount);
            textBox7.Text = Convert.ToString(war.navidate);
            if (war.pesticide_limit_status == 1)
            {
               checkBox1.Checked = true;
            }
            else
```

```
            {
                checkBox1.Checked = false;
            }
        }
        catch (Exception en)
        {}}
    private void button1_Click(object sender, EventArgs e)
    {
        DataClasses1DataContext dms = new DataClasses1DataContext();
        food_analysis_report wam = dms.food_analysis_reports.First(z => z.report_id ==
Convert.ToInt32(listBox2.SelectedValue));
        dms.food_analysis_reports.DeleteOnSubmit(wam);
        dms.SubmitChanges();
        try
        {
            listBox2.DataSource = null;
            listBox2.Items.Clear();
            DataClasses1DataContext dm = new DataClasses1DataContext();

            listBox2.DataSource = dm.food_analysis_reports.Where(x => x.well_well_id ==
Convert.ToInt32(listBox1.SelectedValue));
```

```
            listBox2.DisplayMember = "report_name";
            //listBox2.SelectedItem = "report_id";
            //listBox2.SelectedValue = "report_id";
            listBox2.ValueMember = "report_id";
        }
        catch (Exception ev)
        {}
        textBox1.Text = null;
        textBox2.Text = null;
        textBox3.Text = null;
        textBox4.Text = null;
        textBox5.Text = null;
        textBox6.Text = null;
        textBox7.Text = null;
        MessageBox.Show("Report Is Successfully Removed");
}}}
```

B.2.13 Remove foods sub window codes

```
using System;
using System.Collections.Generic;
```

```
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
    public partial class Remove_Foods : Form
    {
        public Remove_Foods()
        {
            InitializeComponent();
        }
        private void Remove_Foods_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'water_projectDataSet.foods' table.
You can move, or remove it, as needed.
            this.foodsTableAdapter.Fill(this.water_projectDataSet.foods);
        }
        private void button1_Click(object sender, EventArgs e)
```

```
{
    DataClasses1DataContext dr = new DataClasses1DataContext();
    food f = dr.foods.First(x => x.food_id == int.Parse(dataGridView1.Rows[dataGridView1.CurrentRow.Index].Cells[0].Value.ToString()));

//int.Parse(dataGridView1.Rows[dataGridView1.CurrentRow.Index].Cells[0].Value.ToString())
    //
    f.end_date = DateTime.Now.Date;
    dr.SubmitChanges();
    this.foodsTableAdapter.Fill(this.water_projectDataSet.foods);
    MessageBox.Show("Selected Food Is Successfully Removed");
}
private void dataGridView1_SelectionChanged(object sender, EventArgs e)
{
    button1.Enabled = true;
}}}
```

B.2.14 Remove pesticide sub window codes

```
using System;
```

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
    public partial class Remove_Pesticide : Form
    {
        public Remove_Pesticide()
        {
            InitializeComponent();
        }
        private void Remove_Pesticide_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'water_projectDataSet20.pesticides' table. You can move, or remove it, as needed.
            this.pesticidesTableAdapter.Fill(this.water_projectDataSet20.pesticides);
```

```
        // TODO: This line of code loads data into the 'water_projectDataSet1.pesticides' table. You can move, or remove it, as needed.

    }

    private void button1_Click(object sender, EventArgs e)

    {

        DataClasses1DataContext dr = new DataClasses1DataContext();

        pesticide p = dr.pesticides.First(x => x.pesticide_id == int.Parse(dataGridView1.Rows[dataGridView1.CurrentRow.Index].Cells[0].Value.ToString()));

        p.end_date = DateTime.Now.Date;

        dr.SubmitChanges();

        this.pesticidesTableAdapter.Fill(this.water_projectDataSet20.pesticides);

        MessageBox.Show("Selected Pesticide Is Successfully Removed");

    }

    private void dataGridView1_SelectionChanged(object sender, EventArgs e)

    {

        button1.Enabled = true;

    }

    private void label1_Click(object sender, EventArgs e)

    {

    }
```

```
    private void dataGridView1_CellContentClick(object sender,
DataGridViewCellEventArgs e)
    {
    } }}
```

B.2.15 Remove pesticide from black list sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
  public partial class Remove_Pesticide_From_Blacklist : Form
  {
    public Remove_Pesticide_From_Blacklist()
    {
      InitializeComponent();
```

```
}

private void Remove_Pesticide_From_Blacklist_Load(object sender, EventArgs e)

{

    // TODO: This line of code loads data into the 'water_projectDataSet12.pesticides' table. You can move, or remove it, as needed.

    this.pesticidesTableAdapter.Fill(this.water_projectDataSet12.pesticides);

}

private void button1_Click(object sender, EventArgs e)

{

    DataClasses1DataContext dr = new DataClasses1DataContext();

    pesticides_black_list pbl = dr.pesticides_black_lists.First(x => x.pesticide_pesticide_id == int.Parse(dataGridView1.Rows[dataGridView1.CurrentRow.Index].Cells[0].Value.ToString()) && x.end_date>DateTime.Now );

        pbl.end_date = DateTime.Now.Date;

        dr.SubmitChanges();

        MessageBox.Show("Selected Pesticide Is Successfully Removed From Blacklist.");

        this.pesticidesTableAdapter.Fill(this.water_projectDataSet12.pesticides);

        //dr.pesticides_black_lists.First(a => a.pesticide_pesticide_id == p.pesticide_id && a.end_date > DateTime.Now);

      int toplam =  dr.pesticides_black_lists.Count(a => a.end_date > DateTime.Now);
```

```
        if (toplam==0)
        {
            button1.Enabled = false;
        }
        else
        {
            button1.Enabled = true;
        }
       }
    private void dataGridView1_SelectionChanged(object sender, EventArgs e)
    {
    button1.Enabled = true;
    } }}
```

B.2.16 Remove tests sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
```

```
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
    public partial class Remove_Tests : Form
    {
        public Remove_Tests()
        {
            InitializeComponent();
        }
        private void Remove_Tests_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'water_projectDataSet3.tests' table.
You can move, or remove it, as needed.
            this.testsTableAdapter.Fill(this.water_projectDataSet3.tests);
        }
        private void button1_Click(object sender, EventArgs e)
        {
            DataClasses1DataContext dt = new DataClasses1DataContext();
```

```
        test t = dt.tests.First(x => x.test_id == int.Parse(dataGridView1.Rows[dataGridView1.CurrentRow.Index].Cells[0].Value.ToString()));
        t.test_endtime = DateTime.Now.Date;
        dt.SubmitChanges();
        this.testsTableAdapter.Fill(this.water_projectDataSet3.tests);
        MessageBox.Show("Selected Test Is Successfully Removed");
    }
    private void dataGridView1_SelectionChanged(object sender, EventArgs e)
    {
        button1.Enabled = true;
    }}}
```

B.2.17 Remove well sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
```

```
using System.Windows.Forms;

namespace GMAP
{
   public partial class Remove_Well : Form
   {
      public Remove_Well()
      {
         InitializeComponent();
      }
      private void Remove_Well_Load(object sender, EventArgs e)
      {
         // TODO: This line of code loads data into the 'water_projectDataSet4.wells' table. You can move, or remove it, as needed.
         this.wellsTableAdapter.Fill(this.water_projectDataSet4.wells);
      }
      private void button1_Click(object sender, EventArgs e)
      {
         DataClasses1DataContext dt = new DataClasses1DataContext();
         well_position wp = dt.well_positions.First(x => x.well_well_id == int.Parse(dataGridView1.Rows[dataGridView1.CurrentRow.Index].Cells[0].Value.ToString()));
```

```
            wp.w_end_date = DateTime.Now.Date;
            dt.SubmitChanges();
            DataClasses1DataContext dd = new DataClasses1DataContext();
            well w = dd.wells.First(y => y.well_id == int.Parse(dataGridView1.Rows[dataGridView1.CurrentRow.Index].Cells[0].Value.ToString()));
            w.well_end_date = DateTime.Now.Date;
            dd.SubmitChanges();
            this.wellsTableAdapter.Fill(this.water_projectDataSet4.wells);
            MessageBox.Show("Selected Well Is Successfully Removed With It's Position");
        }
        private void dataGridView1_SelectionChanged(object sender, EventArgs e)
        {
            button1.Enabled = true;
        }}}
```

B.2.18 Remove well test report results sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
```

```
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
   public partial class Remove_Well_Test_Report_Results : Form
   {
      public Remove_Well_Test_Report_Results()
      {
         InitializeComponent();
      }
      private void Remove_Well_Test_Report_Results_Load(object sender, EventArgs e)
      {
         // TODO: This line of code loads data into the 'water_projectDataSet13.wells' table.
You can move, or remove it, as needed.
         this.wellsTableAdapter.Fill(this.water_projectDataSet13.wells);

         listBox2.DisplayMember = "report_name";
        // listBox2.SelectedItem = "report_id";
         //listBox2.SelectedValue = "report_id";
```

```
            listBox2.ValueMember = "report_id";
            //water_analysis_report w = new water_analysis_report();
            //listBox2.SelectedValue = w.report_id;
            //listBox2.SelectedItem = w.report_id;
            //listBox2.DisplayMember = w.report_name;
        }
private void listBox1_SelectedIndexChanged(object sender, EventArgs e)
        {
            try
            {
                listBox2.DataSource = null;
                listBox2.Items.Clear();
                DataClasses1DataContext dm = new DataClasses1DataContext();

                listBox2.DataSource = dm.water_analysis_reports.Where(x => x.well_well_id ==
Convert.ToInt32(listBox1.SelectedValue));
                listBox2.DisplayMember = "report_name";
                //listBox2.SelectedItem = "report_id";
                //listBox2.SelectedValue = "report_id";
                listBox2.ValueMember = "report_id";
            }
```

```
        catch (Exception ev)
        {          }
    private void listBox2_SelectedIndexChanged(object sender, EventArgs e)
    {
        try
        {
            listBox2.DisplayMember = "report_name";
            //listBox2.SelectedItem = "report_id";
           //  listBox2.SelectedValue = "report_id";
            listBox2.ValueMember = "report_id";
            //MessageBox.Show( listBox2.SelectedIndices[0].ToString());
            DataClasses1DataContext dma = new DataClasses1DataContext();

            water_analysis_report war = dma.water_analysis_reports.First(v => v.report_id == Convert.ToInt32(listBox2.SelectedValue));

            textBox1.Text = war.report_name;
            textBox2.Text = war.report_text;
            textBox3.Text = war.test_result_comment;
            pesticide p = dma.pesticides.First(c => c.pesticide_id == war.pesticides_pesticides_id);
```

```
            textBox4.Text = p.pesticide_name;
            textBox5.Text = Convert.ToString(war.test_result_amount);
            textBox6.Text = Convert.ToString(war.pesticides_amount);
            if (war.pesticide_limit_status == 1)
            {
                checkBox1.Checked = true;
            }
            else
            {
                checkBox1.Checked = false;
            }
        }
        catch (Exception en)
        {
        }
    }
    private void button1_Click(object sender, EventArgs e)
    {
        DataClasses1DataContext dms = new DataClasses1DataContext();
                water_analysis_report wam = dms.water_analysis_reports.First(z =>
        z.report_id == Convert.ToInt32( listBox2.SelectedValue));
```

```
dms.water_analysis_reports.DeleteOnSubmit(wam);
dms.SubmitChanges();
try
{
   listBox2.DataSource = null;
   listBox2.Items.Clear();
   DataClasses1DataContext dm = new DataClasses1DataContext();
            listBox2.DataSource = dm.water_analysis_reports.Where(x => x.well_well_id == Convert.ToInt32(listBox1.SelectedValue));
   listBox2.DisplayMember = "report_name";
   //listBox2.SelectedItem = "report_id";
   //listBox2.SelectedValue = "report_id";
   listBox2.ValueMember = "report_id";
}
catch (Exception ev)
{
}
textBox1.Text = null;
textBox2.Text = null;
textBox3.Text = null;
textBox4.Text = null;
```

```
            textBox5.Text = null;
            textBox6.Text = null;
            MessageBox.Show("Report Is Successfully Removed");
        }}}
```

B.2.19 Update pesticides sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace GMAP
{
    public partial class Update_Pesticides : Form
    {
        public Update_Pesticides()
```

```
{
    InitializeComponent();
}

private void textBox1_TextChanged(object sender, EventArgs e)
{

}

private void listBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    DataClasses1DataContext dt = new DataClasses1DataContext();
    pesticide p = dt.pesticides.First(x=>x.pesticide_id==Convert.ToInt32(listBox1.SelectedValue.ToString()));
    textBox1.Text = p.pesticide_name;
    textBox2.Text = p.limit.ToString();

}

private void Update_Pesticides_Load(object sender, EventArgs e)
```

```
{
    // TODO: This line of code loads data into the 'water_projectDataSet21.pesticides' table. You can move, or remove it, as needed.
    this.pesticidesTableAdapter.Fill(this.water_projectDataSet21.pesticides);
    maskedTextBox1.Enabled = false;



}

private void button1_Click(object sender, EventArgs e)
{
    if (checkBox1.Checked==true)
    {
        DataClasses1DataContext dm = new DataClasses1DataContext();
        pesticide p = dm.pesticides.First(a=> a.pesticide_id== Convert.ToInt32(listBox1.SelectedValue.ToString()));
        p.limit = Convert.ToDouble(maskedTextBox1.Text.Replace(" ", ""));
        p.pesticide_name = textBox1.Text;
        p.end_date = Convert.ToDateTime("31-12-2999");
        dm.SubmitChanges();
        MessageBox.Show("Pesticide Is Successfully Updated...");
```

```
        maskedTextBox1.Text = null; textBox1.Text = null; textBox2.Text = null;
    }
    else
    {
        DataClasses1DataContext dx = new DataClasses1DataContext();
                pesticide pp = dx.pesticides.First(a => a.pesticide_id ==
    Convert.ToInt32(listBox1.SelectedValue.ToString()));
        pp.pesticide_name = textBox1.Text;
        pp.end_date = Convert.ToDateTime("31-12-2999");
        dx.SubmitChanges();
        MessageBox.Show("Pesticide Is Successfully Updated...");
        maskedTextBox1.Text = null; textBox1.Text = null; textBox2.Text = null;
    }
}
private void checkBox1_CheckedChanged(object sender, EventArgs e)
{
    if (checkBox1.Checked == true)
    {
        maskedTextBox1.Enabled = true;
    }
    else
```

```
        {
            maskedTextBox1.Enabled = false;
        }
    }}}
```

B.2.20 Update pesticides sub window codes

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace GMAP
{
    public partial class Update_Well_Depth : Form
    {
        public Update_Well_Depth()
        {
```

```
        InitializeComponent();
    }

    private void Update_Well_Depth_Load(object sender, EventArgs e)
    {
        // TODO: This line of code loads data into the 'water_projectDataSet18.wells' table. You can move, or remove it, as needed.
        this.wellsTableAdapter.Fill(this.water_projectDataSet18.wells);
    }

    private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
    {
        DataClasses1DataContext dc = new DataClasses1DataContext();

        well_position wp = dc.well_positions.First(x => x.well_well_id == Convert.ToInt32(comboBox1.SelectedValue.ToString()));
        textBox1.Text = wp.well_depth.ToString();
       s = maskedTextBox1.Text;

        // maskedTextBox1.Text = maskedTextBox1.Text.Replace(" ", "");
```

```
}
public string s;
private void button1_Click(object sender, EventArgs e)
{

   if (s==maskedTextBox1.Text)
   {
      MessageBox.Show(" New Well Depth Is Null, Type A Depth...");
   }
   else
   {
      DataClasses1DataContext dm = new DataClasses1DataContext();
      well_position wp1 = dm.well_positions.First(a => a.well_well_id == Convert.ToInt32(comboBox1.SelectedValue.ToString()));
      wp1.well_depth = Convert.ToDouble(maskedTextBox1.Text.Replace(" ", ""));
      dm.SubmitChanges();

      MessageBox.Show("Well Depth Is Successfully Updated...");
      maskedTextBox1.Text = null; textBox1.Text = null;
   }
}}}
```

APPENDIX C

REFERENCE CLASSES OF PROJECT

In current project, to use the google maps, some classes are included. They are shown in the Figure C.1

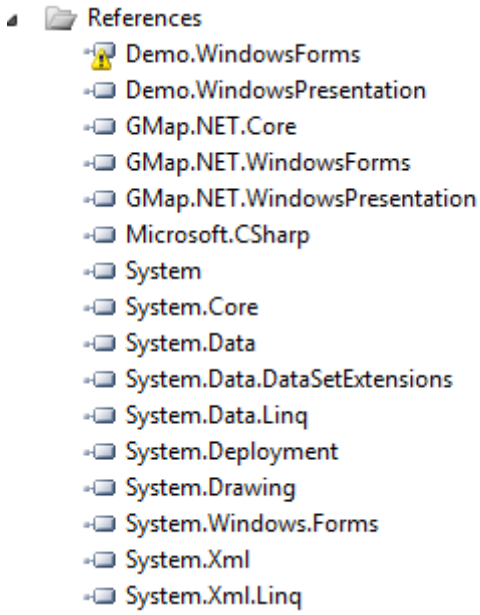

Figure C.1: Reference Class