



**Ministry of Higher Education**  
**Modern Academy for Engineering & Technology in Maadi**  
**Electronics & Communication Dept.**  
***The Graduation Project***  
***Of***

**Extracting Official Papers Using the**  
**Immediate Documents Machine**  
**( IDM )**

**By**

- |                           |                              |
|---------------------------|------------------------------|
| ▪ Omar Ashraf Wageh Mekky | ▪ Sara Sameeh Shokry         |
| ▪ Samuel Makram Mounir    | ▪ Nourhan Ahmed Taleb        |
| ▪ Doaa Abdel Mohsen Sayed | ▪ Hadeer Mohamed Abdel Razek |

**Supervised By**

**Dr. Eng. Eman Mohamed**  
**Eng. Mohamed Nabil**

**11 July 2016**



**Ministry of Higher Education  
Modern Academy for Engineering & Technology in Maadi  
Electronics & Communication Dept.  
*The Graduation Project*  
*Of*  
**Extracting Official Papers Using the  
Immediate Documents Machine  
( IDM )****

**By**

- |                           |                              |
|---------------------------|------------------------------|
| ▪ Omar Ashraf Wageh Mekky | ▪ Sara Sameeh Shokry         |
| ▪ Samuel Makram Mounir    | ▪ Nourhan Ahmed Taleb        |
| ▪ Doaa Abdel Mohsen Sayed | ▪ Hadeer Mohamed Abdel Razek |

***Discussion Committee***

**Supervised By**

**Prof. Dr. Magdy Tantawy  
Dr. Shouman Ibrahim Shouman  
Dr. Eman Mohammed**

**Signature: .....**

**Signature: .....**

**Signature: .....**

# *Index*

	<i>Page</i>
<b>Chapter1:</b> Introduction	1
1.1 Problem Description	1
1.2 Project Idea	1
1.3 Project Description	1
1.4 Project Organization	4
1.5 Conclusion	5
<b>Chapter 2:</b> Programming Languages and Platforms	6
2.1 Introduction	6
2.2 Project Database	6
2.2.1 Microsoft SQL server	6
2.2.1.1 Relational Database	7
2.2.1.2 Management System	7
2.3 Project Websites	8
2.3.1 ASP.NET	9
2.3.2 MVC	9
2.3.2.1 The MVC Programming Model	10
2.4 Microsoft Visual Studio	11
2.4.1 Programming language	12
2.4.2 Microsoft C#	12
2.4.3 HTML	13
2.5 Raspberry Pi Brief Overview	14
2.5.1 Raspberry Pi models	15
2.5.2 Raspberry-Pi 2 model B	16
2.6 Conclusion	17

	<i><b>Page</b></i>
<b>Chapter 3: Creating Requests Website</b>	18
3.1 Introduction	18
3.2 Overview	18
3.3 User interface	20
3.3.1 Website interface	20
3.4 User characteristics	24
3.4.1 Website Actors	24
3.4.2 Use Case	24
3.4.2.1 Use Case Details	25
3.4.2.2 Use case diagram	26
3.5 Website Administrator interface	27
3.6 Publishing and Hosting the Website	29
3.6.1 Steps to publish and Host the website	29
3.6.1.1 Step one	29
3.6.1.2 Step two	31
3.6.1.3 Step three	34
3.7 System Database	36
3.7.1 Database Interface	37
3.8 Conclusion	39
<b>Chapter 4: Receiving Pre-created Requests Website</b>	40
4.1 Introduction	40
4.2 Overview	40

	<i><b>Page</b></i>
4.3 User interface	43
4.3.1 Website interface	43
4.4 Conclusion	44
<b>Chapter 5: Windows Form Applications</b>	<b>45</b>
5.1 Introduction	45
5.2 Windows Form Applications	45
5.3 GSM Module	46
5.3.1 GSM Brief Overview	46
5.3.2 GSM Application Process	46
5.4 Printing Application	50
5.4.1 Printing Application Process	50
5.5 Conclusion	52
<b>Chapter 6: Universal Windows Application</b>	<b>53</b>
6.1 Introduction	53
6.2 Universal Windows Platform	53
6.2.1 Characteristics of UWP apps on Windows 10	54
6.2.2 UWP apps come to life on Windows	55
6.2.3 Monetize your app your way	56
6.2.4 UWP app features	56
6.2.5 Responsive design techniques	57
6.3 Requirements for UWP application	61
6.4 User Interface	61
6.5 Conclusion	63

	<i><b>Page</b></i>
<b>Chapter 7: Future Work</b>	64
7.1 Introduction	64
7.2 Network System	64
7.2.1 List of WAN types	64
7.2.2 Types of Routers	65
7.2.3 Advantage of Network System	65
7.3 GSM Message	66
7.4 Printer Racks	66
7.5 Touch Screen	67
7.5.1 Advantages of Touch Screen	67
7.6 Finger Print	68
7.7 ID Scanner	69
7.8 Data Input Devices	69
<b>Appendix</b>	70
<b>References</b>	

# ***LIST OF FIGURES***

<b>Figure number</b>	<b>Description</b>	<b>Page</b>
Figure 1-1	Project flowchart	2
Figure 1-2	Project Block diagram	3
Figure 2-1	Project database opened in MVC project	8
Figure 2-2	The MVC Programming Model	10
Figure 2-3	Raspberry-Pi kit	15
Figure 2-4	Raspberry-Pi Model A and Raspberry-Pi Model B	15
Figure 2-5	Raspberry-Pi Model A+ and Raspberry-Pi Model B+	16
Figure 2-6	Raspberry-Pi 2 Model B	17
Figure 3-1	Creating Requests Website Flowchart	19
Figure 3-2	Creating Requests Website homepage	20
Figure 3-3	Creating Requests Website Governmental establishments page	21
Figure 3-4	Creating Requests Website Request page	22
Figure 3-5	Creating Requests Website make more processes or logout page	23
Figure 3-6	Creating Requests Website use case diagram	26
Figure 3-7	Creating Requests Website Administrator interface	27
Figure 3-8	Add or edit the governmental establishments	28
Figure 3-9	Add or edit the type of official papers	28
Figure 3-10	Publish the website	29
Figure 3-11	Redirect to a page	30
Figure 3-12	Expand connections and find sites	31
Figure 3-13	Selecting Site Name and Physical Path	32
Figure 3-14	Directory Browsing	33

Figure 3-15	Enable page	33
Figure 3-16	Selecting hosts	34
Figure 3-17	Adding Website name	35
Figure 3-18	Program connecting database and website	36
Figure 3-19	Governmental Establishments table in project Database	37
Figure 3-20	Types of official papers table in project Database	37
Figure 3-21	Citizens table in project Database	38
Figure 3-22	Requests table in project Database	38
Figure 3-23	Tables connection in project Database	39
Figure 4-1	Receiving Pre-created Requests Website first page	41
Figure 4-2	Receiving Pre-created Requests Website flowchart	42
Figure 4-3	Receiving Pre-created Requests Website Homepage	43
Figure 4-4	Saved Requests page in the Receiving Pre-created Requests Website	45
Figure 4-5	toPrint and printed_or_not columns in Database	45
Figure 5-1	Windows form application for GSM section	47
Figure 5-2	Request page in Creating Requests website	48
Figure 5-3	"Msg_Sent_Or_Not" column in the database	48
Figure 5-4	GSM application Flowchart	49
Figure 5-5	User information and requests table in receiving the pre-created requests website	50
Figure 5-6	toPrint and printed_or_not columns in Database	51
Figure 5-7	Printing Application flowchart	52
Figure 6-1	Universal Windows Platform	53
Figure 6-2	Devices can run windows 10	54
Figure 6-3	Viewing distance and screen density	56
Figure 6-4	Universal Windows Application Reposition	57
Figure 6-5	Universal Windows Application Resize	58
Figure 6-6	Universal Windows Application Reflow	59



Figure 6-7	Universal Windows Application Reveal	59
Figure 6-8	Universal Windows Application Replace	60
Figure 6-9	Universal Windows Application Re-architect	60
Figure 6-10	Universal Windows Application Home page	61
Figure 6-11	Universal Windows Application Creating or Receiving Requests page	62

# ***LIST OF TABLES***

		<i><b>Page</b></i>
Table 3-1	Type of users interact with the “Creating Requests Website” and the role of each.	24

# ***ABBREVIATIONS***

<b>ASP</b>	Application Service Provider
<b>CLI</b>	Command Line User Interface
<b>GSM</b>	Global System for Mobile Communications
<b>HTML</b>	HyperText Markup Language
<b>MVC</b>	Model – View - Controller
<b>IDE</b>	Integrated Development Environment
<b>IDM</b>	Immediate Documents Machine
<b>IIS</b>	Internet Information Services
<b>SQL</b>	Structured Query Language
<b>UMP</b>	Universal Windows Platform

# ***DEDICATION***

## ***To Our Family & Our Doctors:***

**We really cannot deny how much you inspired and supported us, and how helpful you were to us.**

**Thanks a lot and may GOD give you the good health, bless your days & fill your hearts with peace and happiness.**

# ***ACKNOWLEDGEMENT***

First we would like to greatly thank our supervisor **Dr. Eman Mohamed** for her efforts in making us able to work as a team and for her support with knowledge that helped us improve our work and her continuous inspection of our work.

We also extend our thanks to Engineer **Eng. Mohamed Nabil** for following up on us in all stages of our project and for his support throughout by offering his ideas and suggestions.

Finally, we can't forget thank our families that always supported us in our life and gave us endless help.

## ***ABSTRACT***

Our project aims to computerize the process of extraction the official papers in the governmental establishments to make the procedures easier for the citizens.

This idea will minimize the load of the governmental establishments and save time and effort.

For achieving our aim, we created two websites, one for creating requests of the required official papers, and the other is for receiving pre-created requests through the IDM machine which will be outdoors or in the places specifies for it so that the user will be able to print the required official paper he/ she requested immediately.

For the printing process we created a windows form application connected to the citizens' database to print the requests and a GSM module is used to send a thankful SMS for using the service to the user mobile number.

A Raspberry-Pi 2 Model B is used to runs the Universal applications on the IDM machine to browse both websites for the user to receive the pre-created website or to create a one using the website on the machine.

# **Chapter 1**

## **Introduction**

### **1.1 Problem Description**

The problem of overcrowding faced by citizens and employees in the governmental establishments to finish or extract an official papers has been overwhelming in the last few years; due to increase in demand, some due to human factors and others due to slowness of the process, in parallel with the increasing in the population, causing irritation, wasting of time and effort and corruption in a few cases.

### **1.2 Project Idea**

Our project is to computerize the process of extracting the official papers in the governmental establishments to make the procedures easier for the citizens, minimize the load of the governmental establishments and save time and effort.

### **1.3 Project Description**

The project is composed of two section, a software section and a hardware section.

In the software section a database holding citizens' information is been created and then connected to two websites, one the user can use for creating requests to the official paper he/she needed, and the other is for receiving pre-created requests to be printed afterwards, all the user need is to enter the National ID and a saved Password in the database to access the website for making or printing a request..

Also, two windows form applications are created, one responsible for the printing process and the other is responsible for sending an SMS to the user after creating

the request telling him/her whether the request is received and the nearby IDM (Immediate Documents Machine) to print it.

A universal windows application starts on the IDM machine holding the two websites have been created where the user can print a pre-created request or he/ she can make a new one.

The process the citizen goes through is described in the flowchart in Figure 1-1

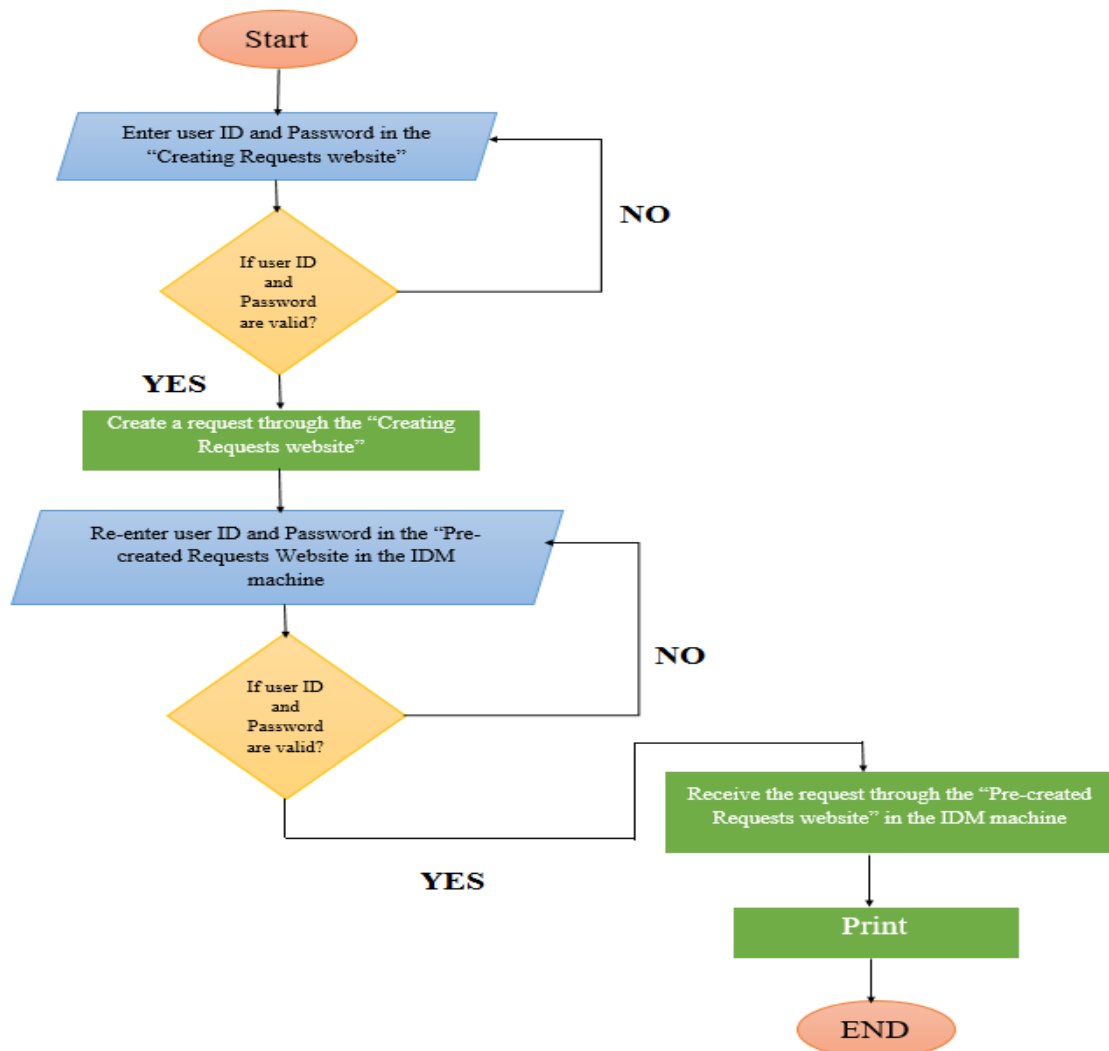


Figure 1-1



In the hardware section which is related to the IDM machine, a Raspberry-Pi kit is used to run the Universal applications on the (IDM) to browse both websites, it is also connected to LCD screen for displaying, a keyboard with mouse pad for the user to input his/ her data.

A GSM module is used for the process of sending an SMS to the user. Also a one rack printer is used to print the pre-requested official papers.

The Router is used to transfer the hosts file to the Raspberry-Pi Kit to be able to browse both websites in Web-views through IIS

The project block diagram is included in Figure 1-2.

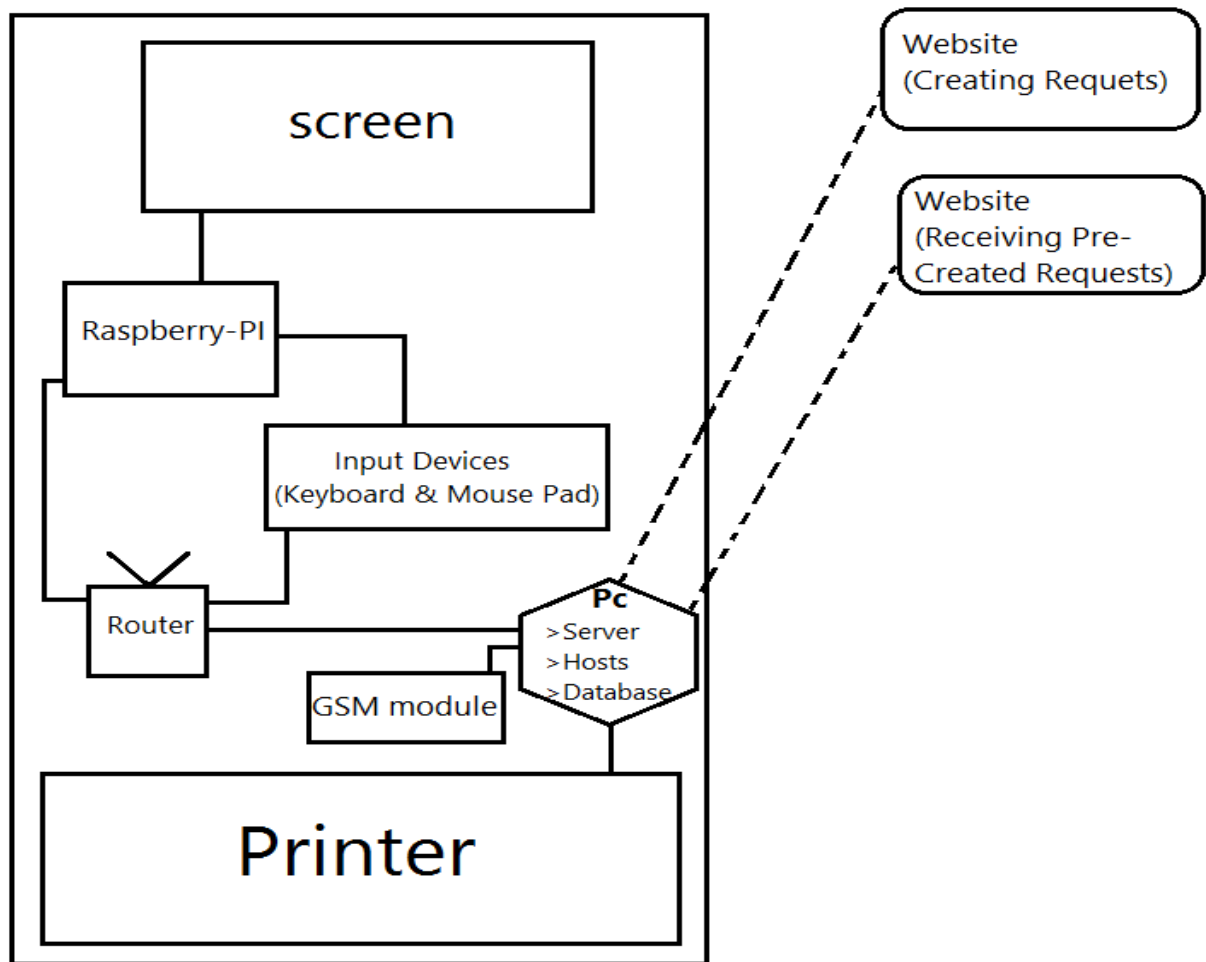


Figure 1-2

## **1.4 Project Organization**

Chapter 2 .. contains the complete information about the programming languages and the platforms used for creating the project, it also will contains an overview about the Raspberry-Pi module used for controlling the hardware parts related to the IDM machine.

Chapter 3 .. gives a scope description of everything included in the “Creating Request website” section. It also gives a detailed description of the requirements for the process of extracting the official papers through the website including the process of publishing the website through the IIS Host and the system database. The chapter also explains system constraints, and interface.

Chapter 4 .. gives a scope description of everything included in the “Receiving Pre-created Requests Website”. It gives a detailed description of the requirements for the process of extracting the official papers through the Receiving Pre-created Requests Website. Also, it will explain system constraints, and interface.

Chapter 5 .. gives a scope description and overview of the two Windows Form Applications created in the project which are responsible for the process of Printing the Pre-created requests and also sending an SMS for the User. It also contains brief overview about the GSM module used in the project which is responsible for sending the SMS.

Chapter 6 .. gives a scope description and overview of everything included in this Universal Application section. It also gives a detailed description of the requirements for the process of displaying the Interface and processing of the universal application.

Chapter 7 .. contains the future work which gives a description of the requirements and list of ideas, devices and sensors could be added to the project to make it a more powerful system.

## **1.5 Conclusion**

The project will be able to solve the problems faced by the citizens while extracting an official paper without irritation as the citizens can request the official paper through using the website specialized for this service and receive the request through the nearest IDM machine.

The citizens will always be able to help in developing and improving the quality of the service by sending their complains or ideas through the website.

The project can also be used in different areas like hospitals or universities by changing the requirements of the system to be suitable for helping the users to make the good use of the service the project can introduce in such fields.

The project can be developed to be a part of a network connecting the whole country as it will be explained in details in the Future Work Chapter.

# Chapter 2

## Programming Languages and Platforms

### 2.1 Introduction

This chapter gives a detailed description about the programming languages and the platforms used for creating the two websites and the project database including a brief history of each and reasons why using them.

### 2.2 Project Database

The project database is been created on Microsoft SQL Server, to hold information about the citizens' and the requests they made through the User Website and the Raspberry-Pi Website.

#### 2.2.1 Microsoft SQL server

SQL Server is a Microsoft product used to manage and store information. Technically, SQL Server is a “relational database management system” (RDMS). Broken apart, this term means two things. First, that data stored inside SQL Server will be housed in a “relational database”. Second, that SQL Server is an entire “management system”, not just a database. SQL itself stands for Structured Query Language. This is the language used to manage and administer the database server.

### **2.2.1.1 Relational Database**

Most of the products on the market today (SQL Server, Oracle, MySQL, and MS Access to name a few) are relational database products. This means that data is stored inside a structure called a “Table”, which uses Rows and Columns (like a spreadsheet).

Unlike a spreadsheet though, the data rows stored inside a Table is not in any particular order. In spreadsheet column, if we wanted to sort the data in the first column of a spreadsheet alphabetically, we would simply click the first column and then would click the Sort button. The Rows of data would change their order so they were now sorted the way we wished

In a Database Table, this never happens. The data would not be rearranged. If we wanted a sorted list like this, we would ask the database to present us with a display copy of the data sorted the way we wanted. This request to see the data is called a Query. So when we run a Query, we see our own personalized display copy of the data, the actual data items are not rearranged.

When discussing SQL Server, the term “Database”, can sometimes be thrown around loosely, meaning different things to different people. This happens because a database is a core, central component to SQL Server. Therefore, the term has become a slang shortcut way of meaning SQL Server as a whole. In actuality, SQL Server is RDBMS (Relational Database Management System). Its job is managing databases

### **2.2.1.2 Management System**

The second term in our SQL Server definition is “Management System”. This means that SQL Server is more than just an application to hold data; it also includes the tools needed to structure, manipulate, and manage that data. In addition, when you install SQL Server, there are options for including Report Writing tools, Data Import Export applications, Analysis tools, and Management Interfaces.

## 2.3 Project Websites

The project websites are created as two MVC projects ASP.NET web applications using Microsoft Visual Studio 2015 as a platform, one for making the requests and the other for calling these requests from the SQL server database which is connected easily to the Microsoft visual studio as a feature it introduces as in Figure 2-1, allowing requests to be printed afterwards.

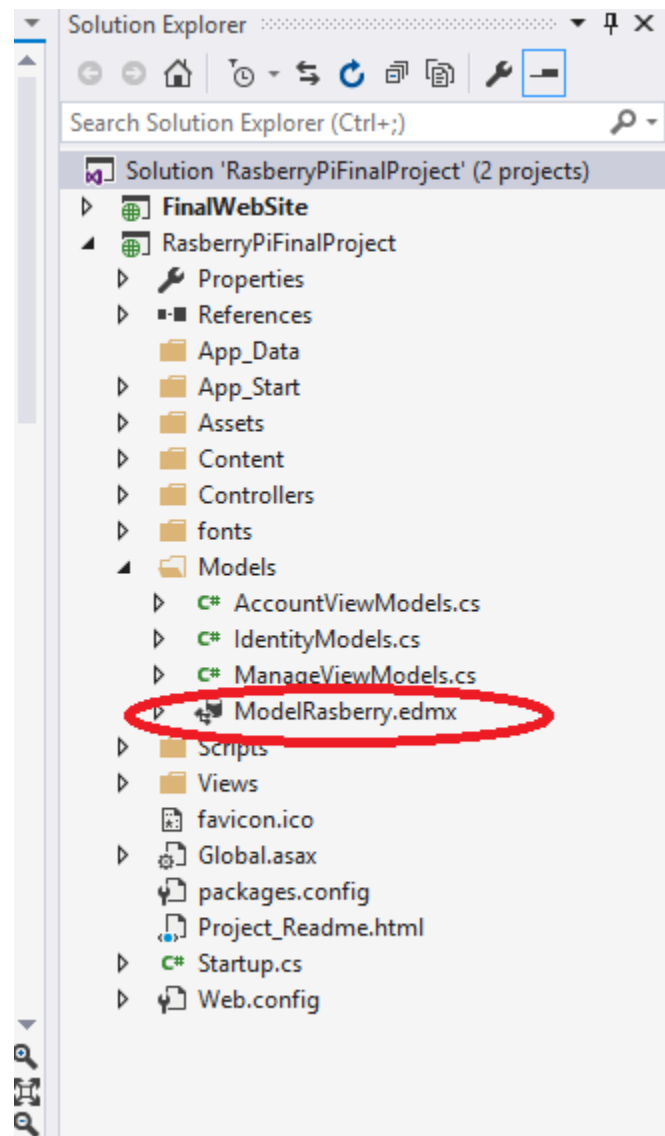


Figure 2-1

### **2.3.1 ASP.NET**

ASP.NET (originally called ASP+) is the next generation of Microsoft's Active Server Page (ASP), a feature of their Internet Information Server (IIS). Both ASP and ASP.NET allow a Web site builder to dynamically build Web pages by inserting queries to a relational database in the Web page.

ASP.NET is different than its predecessor in two major ways:

First, it supports code written in compiled languages such as Visual Basic, C++, C#. Second, it features server controls that can separate the code from the content, allowing WYSIWYG (what you see is what you get), which allows developers to see what the end result will look like while the interface or document is being created.

A WYSIWYG editor can be contrasted with more traditional editors that require the developer to enter descriptive codes (or markup). An HTML WYSIWYG editor, such as Microsoft's FrontPage conceals the markup and allows the Web page developer to think entirely in terms of how the content should appear.

Although ASP.NET is not backwards compatible with ASP, it is able to run side by side with ASP applications. ASP.NET files can be recognized by their .aspx extension.

### **2.3.2 MVC**

ASP.NET is a development framework for building web pages and web sites with HTML, CSS, JavaScript and server scripting.

ASP.NET supports three different development models: Web Pages, MVC (Model View Controller), and Web Forms.

### 2.3.2.1 The MVC Programming Model

MVC is one of three ASP.NET programming models. As shown in Figure 2-2, MVC is a framework for building web applications using a MVC (Model View Controller) design:

- The Model represents the application core (for instance a list of database records).
- The View displays the data (the database records).
- The Controller handles the input (to the database records).

The MVC model also provides full control over HTML, CSS, and JavaScript.



Figure 2-2

**The Model** is the part of the application that handles the logic for the application data.

Often model objects retrieve data (and store data) from a database.

**The View** is the parts of the application that handles the display of the data. Most often the views are created from the model data.

**The Controller** is the part of the application that handles user interaction. Typically controllers read data from a view, control user input, and send input data to the model.



The MVC separation helps you manage complex applications, because you can focus on one aspect a time. For example, you can focus on the view without depending on the business logic. It also makes it easier to test an application.

The MVC separation also simplifies group development. Different developers can work on the view, the controller logic, and the business logic in parallel.

The MVC programming model is a lighter alternative to traditional ASP.NET (Web Forms). It is a lightweight, highly testable framework, integrated with all existing ASP.NET features, such as Master Pages, Security, and Authentication.

One More thing about MVC is Bootstrap, the MVC project template has been updated to use Bootstrap to provide a sleek and responsive look and feel that you can easily customize.

## **2.4 Microsoft Visual Studio**

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs for Microsoft Windows, as well as web sites, web applications and web services. *It's* is a comprehensive collection of developer tools and services to help you create apps for the Microsoft platform and beyond.

Microsoft Visual Studio is an environment for creating Web services based on use of the Extensible Markup Language (XML) to provide a visual interface for identifying a program as a Web service, forms for building a user interface (including support for mobile device interfaces), features for integrating existing application data, and for debugging.

In our project we used Microsoft Visual Studio 2015; as it provides a Universal Windows Platform (UWP) app template for each language that lets you create a single project for all devices. When your work is finished, you can produce an app package and submit it to the Windows Store from within Visual Studio to get your app out to customers on any Windows 10 device.

As for developers, Windows 10 makes it easier to develop apps for the UWP with just one API set, one app package, and one store to reach all Windows 10 devices – PC, tablet, phone and more. It's easier to support a number of screen sizes, and also a variety of interaction models, whether it be touch, mouse & keyboard, a game controller, or a pen.

### **2.4.1 Programming language**

Microsoft Visual Studio supports several programming languages including Visual Basic, Visual C++, and Visual C#.

In our project we used C# for coding with HTML for designing the websites interfaces.

### **2.4.2 Microsoft C#**

C# (pronounced "C-sharp") is an object-oriented programming language from Microsoft that aims to combine the computing power of C++ with the programming ease of Visual Basic. C# is based on C++ and contains features similar to those of Java.

C# is very similar in syntax to Java, which is, however, perceived differently by different involved parties. While the people belonging to the team developing the Java language will claim that C# is just a clone of Java, the developers of C# itself will hotly deny it and claim that their language is much closer in syntax to C++. While it's very obvious why their opinions on the matter differ, we have to mention that there is more to a programming language than syntax alone – there are other underlying concepts that define the language and its role in the IT industry on the whole.

First and foremost, it's important to explain in a few words what the .NET initiative is and how C# fits in. .NET is a software framework that can be installed on computers running Microsoft Windows, so any software based on it will run on Windows computers only.

.NET introduces Common Language Infrastructure (CLI) – an open specification describing the executable code and runtime environment that form the core of the Microsoft .NET Framework. Just like it happens with Sun's Java Virtual Machine,

.NET utilizes the virtual machine model with intermediate byte-code. But .NET's byte-code has to be compiled before execution, whereas Java's can also be interpreted. Besides, as we mentioned above, .NET's applications can run only under Windows, while Java takes pride in its platform-independence.

C#, along with Visual Basic, is designed for CLI and has been part of .NET since the beginning.

C# can be used both for standalone programming and for web programming. The latter is commonly done under the ASP.NET web application framework – another Microsoft initiative considerably enhancing the ASP platform. With ASP.NET programmers can use any .NET language – including C# – to build a full-scale web application. ASP.NET web applications run on ASP.NET web servers.

C# is case sensitive language where all statements and expression must end with a semicolon (;). The program execution starts at the Main method. But unlike Java, program file name could be different from the class name.

### **2.4.3 HTML**

HTML is a computer language devised to allow website creation. It is the set of markup symbols or codes inserted in a file intended for display on a World Wide Web browser page. The markup tells the Web browser how to display a Web page's words and images for the user. Each individual markup code is referred to as an element (but many people also refer to it as a tag). Some elements come in pairs that indicate when some display effect is to begin and when it is to end.

It is constantly undergoing revision and evolution to meet the demands and requirements of the growing Internet audience under the direction of the “W3C”, the organization charged with designing and maintaining the language.

HTML refers to the word (Hypertext Markup Language):

Hypertext is the method by which you move around on the web — by clicking on special text called hyperlinks which bring you to the next page. The fact that it is hyper just means it is not linear — i.e. you can go to any place on the Internet whenever you want by clicking on links — there is no set order to do things in.

Markup is what HTML tags do to the text inside them. They mark it as a certain type of text (*italic* text, for example). HTML is a Language, as it has code-words and syntax like any other language.

HTML consists of a series of short codes typed into a text-file by the site author — these are the tags. The text is then saved as an html file, and viewed through a browser, like Internet Explorer or Google Chrome. This browser reads the file and translates the text into a visible form.

## **2.5 Raspberry Pi Brief Overview**

A Raspberry Pi is a credit-card sized computer originally designed for education, inspired by the 1981 BBC Micro. Creator Eben Upton's goal was to create a low-cost device that would improve programming skills and hardware understanding at the pre-university level. But thanks to its small size and accessible price, it was quickly adopted by tinkerers, makers, and electronics enthusiasts for projects that require more than a basic microcontroller (such as Arduino devices).

The Raspberry Pi is slower than a modern laptop or desktop but is still a complete Linux computer and can provide all the expected abilities that implies, at a low-power consumption level.

The Raspberry Pi is open hardware, with the exception of the primary chip on the Raspberry Pi, the “Broadcomm SoC” (System on a Chip), which runs many of the main components of the board—CPU, graphics, memory, the USB controller, etc. Many of the projects made with a Raspberry Pi are open and well-documented as well and are things that can be built and modified. The Raspberry-Pi kit is shown in Figure 2-3.

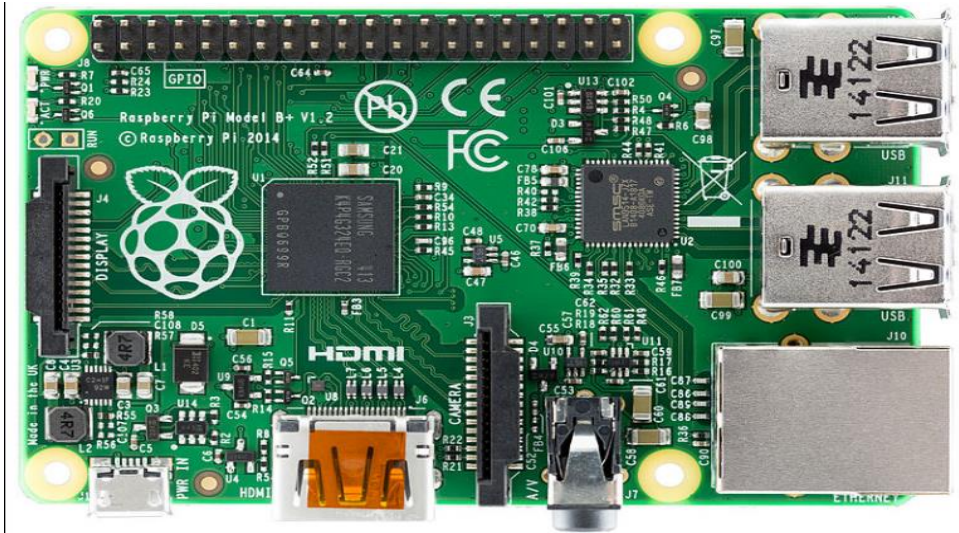


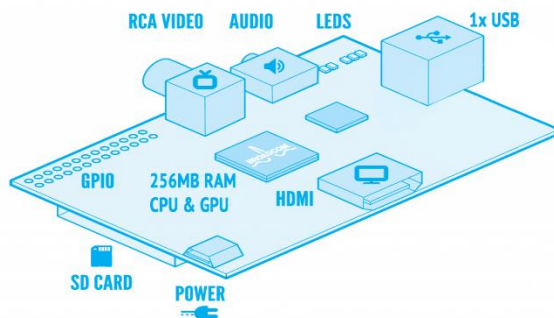
Figure 2-3

## 2.5.1 Raspberry Pi models

There are a two Raspberry-Pi models, the A and the B, named after the aforementioned BBC Micro, which was also released in a Model A and a Model B.

The Raspberry-Pi model A, comes with 256MB of RAM and one USB port. It is cheaper and uses less power than the B. The current Raspberry-Pi model B, comes with a second USB port, an Ethernet port for connection to a network, and 512MB of RAM, as shown in Figure 2-4

### RASPBERRY PI MODEL A



### RASPBERRY PI MODEL B

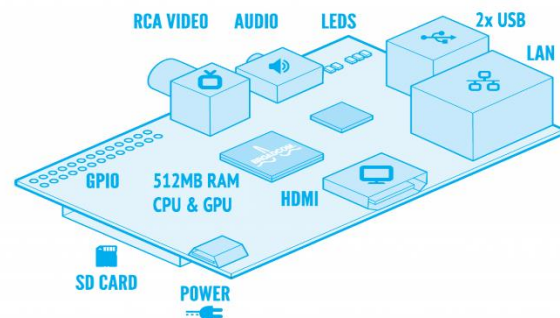


Figure 2-4

The Raspberry Pi A and B boards been upgraded to the A+ and B+ which are shown in Figure 2-5. These upgrades make minor improvements, such as an increased number of USB ports and improved power consumption.

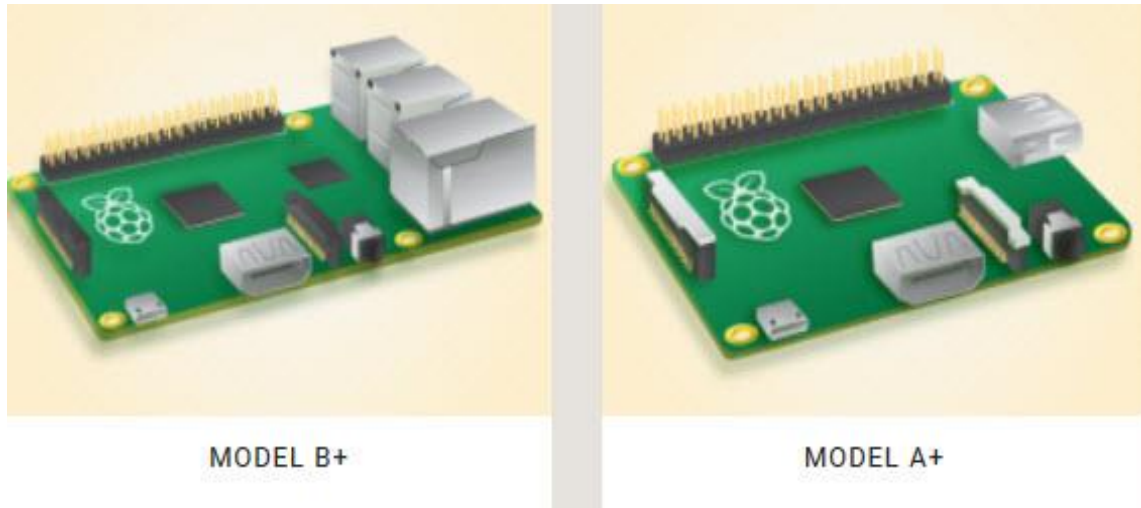


Figure 2-5

### 2.5.2 Raspberry-Pi 2 model B

The Raspberry Pi Foundation has the Raspberry-Pi model 2, which supersedes some of the previous boards, although the older boards will still be produced as long as there is a demand for them. It is generally backwards compatible with previous versions of the board.

For the Raspberry-Pi 2 Model B, which we used in our project, as shown in Figure 2-6, it is the second generation Raspberry Pi. It replaced the original Raspberry Pi 1 Model B+ in February 2015. Compared to the Raspberry Pi 1 it has a 900MHz quad-core ARM Cortex-A7 CPU, 1GB RAM and like the Raspberry-Pi 1 Model B+, it also has 4 USB ports, 40 GPIO pins, full HDMI port, Ethernet port, combined 3.5mm audio jack and composite video, camera interface (CSI), display interface (DSI), Micro SD card slot, VideoCore IV 3D graphics core.



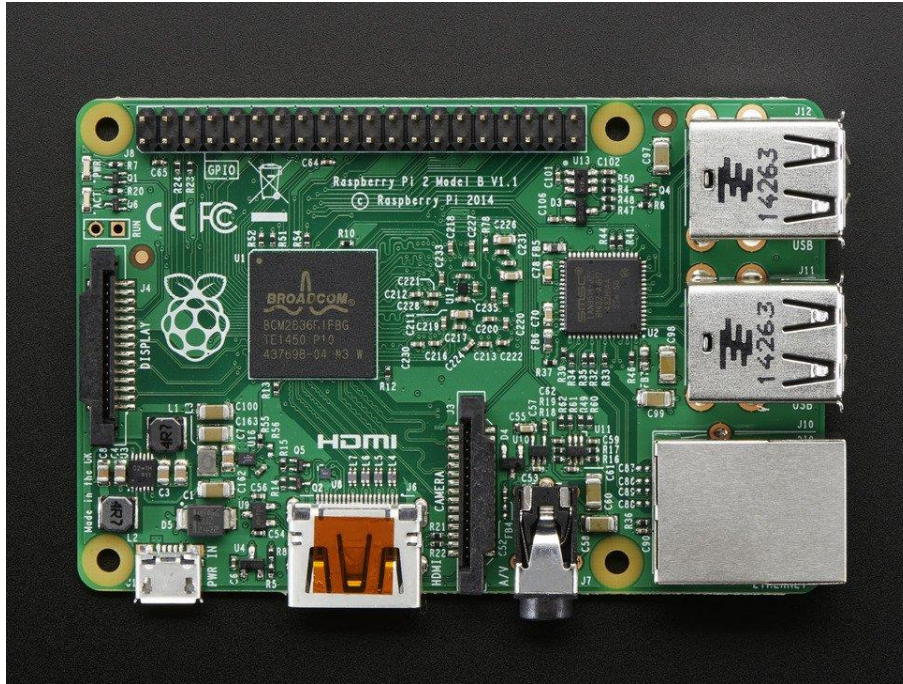


Figure 2-6

The reason for using Raspberry-Pi 2 Model B in our project is that it has an ARMv7 processor, it can run the full range of ARM GNU/Linux distributions, including Snappy Ubuntu Core, as well as Microsoft Windows 10.

The Raspberry Pi 2 has an identical form factor to the previous Raspberry-Pi 1 Model B+ and has complete compatibility with Raspberry Pi 1.

## 2.6 Conclusion

SQL Server is a relational database used for creating the project database. The project websites are created as two MVC projects ASP.NET web applications using Microsoft Visual Studio 2015 as a software. HTML Language is used for making the user interface.

For the Raspberry Pi module a Raspberry-Pi 2 Model B type is for its ability to run the full range of ARM GNU/Linux distributions, including Snappy Ubuntu Core, as well as Microsoft Windows 10.

The Next chapter will describe the process of making a request through the website.

# Chapter 3

## Creating Requests Website

### 3.1 Introduction

This chapter gives a scope description and overview of everything included in the “Creating Request website” section. It gives a detailed description of the requirements for the process of extracting the official papers through the website. Also, it will explain system constraints, and interface.

### 3.2 Overview

Fetching the official papers using the “Creating Request website” allow the users to make a request for the required official paper without irritation.

According to the website process cycle; the user enter the National ID and the password which have been saved in the database before.

The system checks if both the National ID and the password are valid, if they are valid the system continue the process if not it asks the user to re-enter them again.

The website then open the request page for the user where he/she can choose the “Governmental establishment” and the “required papers”.

Then the user can choose the number of copies.

The website will send a message to the user mobile to ensure the process accomplishment.



The process cycle through the website is explained in the flowchart in Figure 3-1

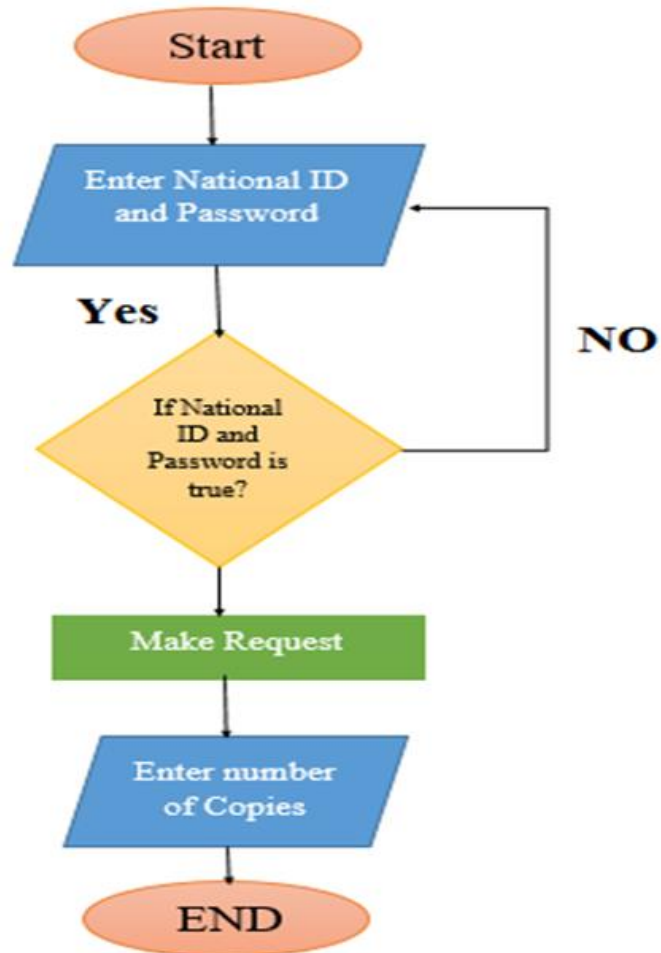


Figure 3-1

### 3.3 User interface

This section provides a detailed description of the user interface for the user website section according to the cycle of the process the user involves in.

#### 3.3.1 Website interface

First the user sees two text boxes in the Home page where he/she can enter National ID and Password as shown in Figure 3-2, then the website checks the user ID and password and allow access.



The image shows a dark-themed login page. At the top left, there is a green 'IDM' logo. At the top right, there is a 'Log in' link. In the center, there is a large white Arabic calligraphic logo that reads 'في خدمتك' (In your service). Below this logo, there is a green tagline that reads 'وفر وقتك وأمنك وأمنك ورقك' (Save your time, your safety, and your paper). Below the tagline, there are two white text input boxes. The first box is labeled 'الرقم القومي.' (National ID number.) and the second box is labeled 'كلمة السر.' (Password.). Below the second box, there is a blue button with the white Arabic text 'دخول' (Login).

Figure 3-2

A Page for the available governmental establishments and type of papers appears allow user to choose his required official paper to be printed as shown in Figure 3-3

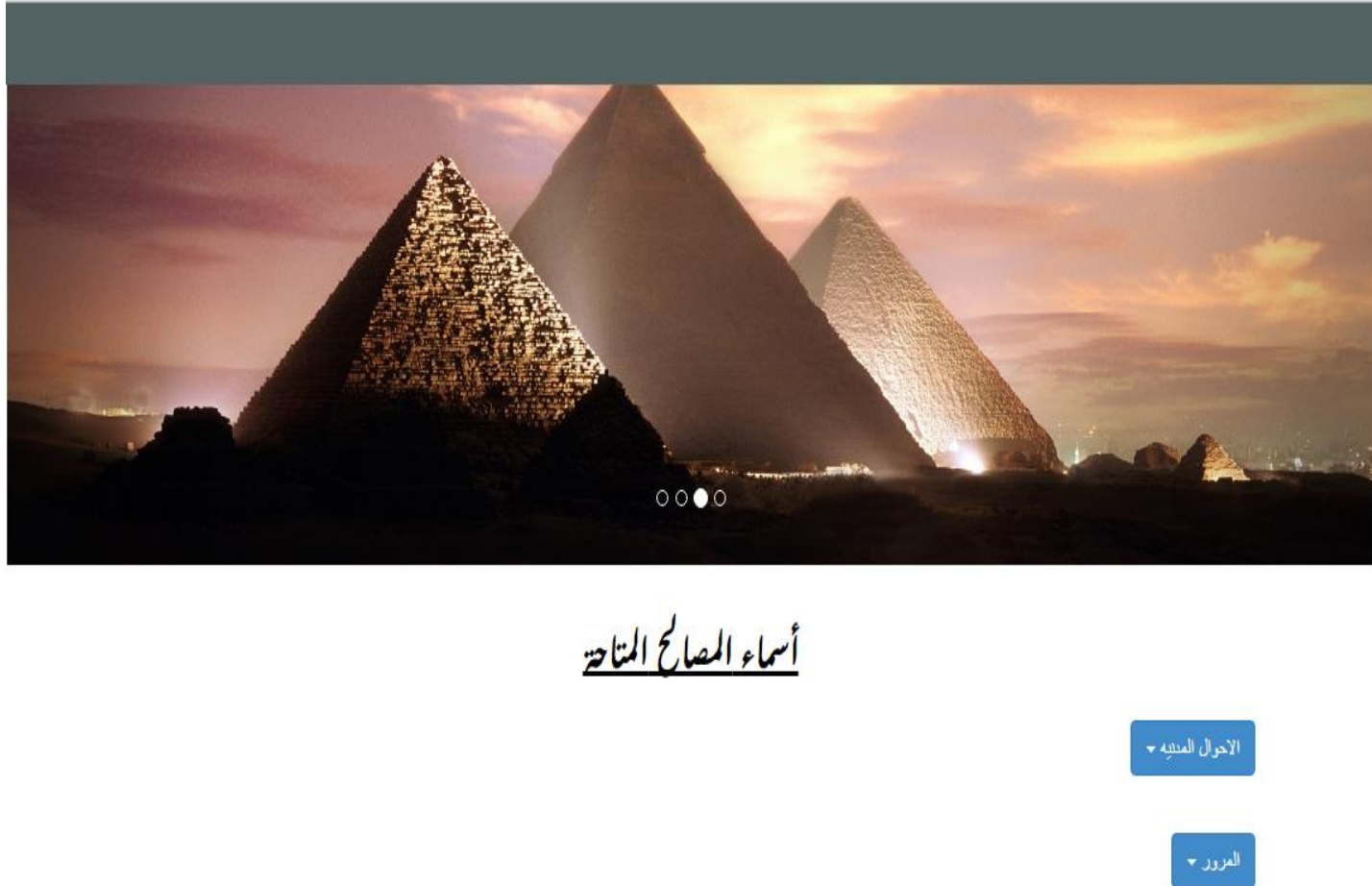


Figure 3-3

After that, the Request page allow the user to make the request and choose number of copies as shown in Figure 3-4.

The screenshot shows a web interface with a dark header bar containing the 'IDM' logo on the left and a 'Log In' link on the right. The main content area is white and contains a form with the following fields:

- الرقم القومي (National ID Number): 123456
- الاسم (Name): Omar Ashraf
- تاريخ الميلاد (Date of Birth): 10/9/1993
- محل الميلاد (Place of Birth): Saudi Arabia
- اسم الأم (Mother's Name): Mona Saleem
- الديانة (Religion): Muslim
- عدد النسخ (Number of Copies):
- رقم المحمول (Mobile Number):

At the bottom of the form is a blue button labeled 'تأكيد' (Confirm).

Figure 3-4

At the end, the website asks the user if he wants to make more processes or just log out as shown in Figure 3-5

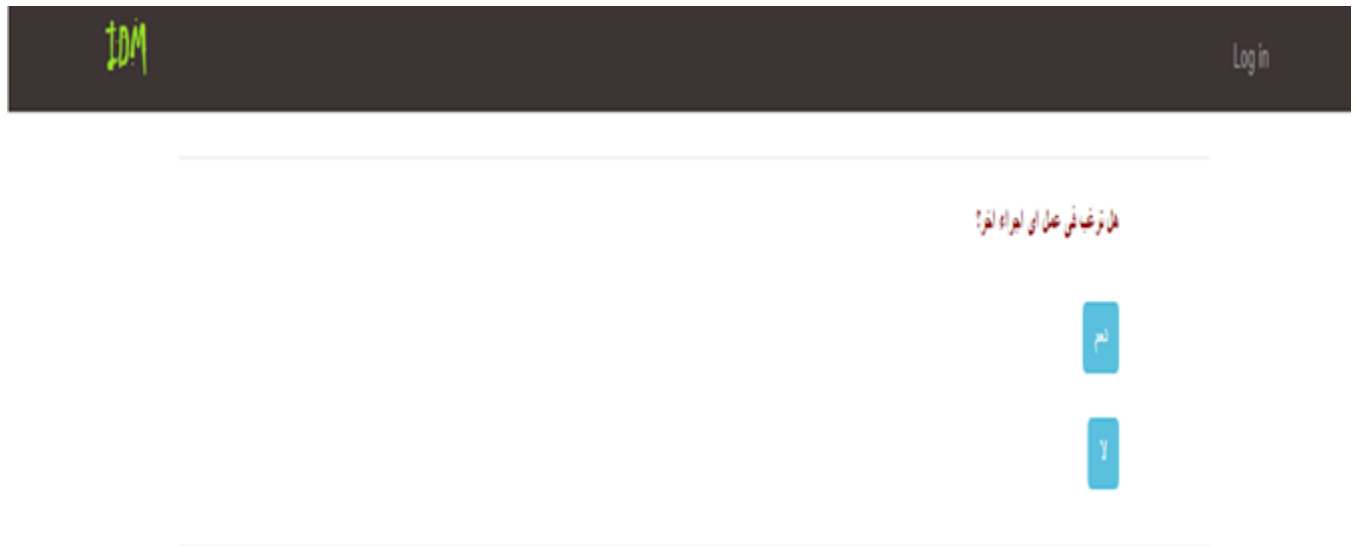


Figure 3-5

### 3.4 User characteristics

This section will describe the type of users that will interact with the website and what functionality is available for each type.

#### 3.4.1 Website Actors

There are three types of users that interact with the system: the citizen, the website administrator and the database administrator. Each of these three types of users has different use of the system so each of them has their own requirements.

The citizen interacts with the website for making a request for the required official paper. The database administrator deals with the citizens' database by checking users' data and making any new updates. Also the website administrator deals with the "Creating Request website" checking errors in requests and updating types of papers and establishments.

Type of users interact with the website and the role of each is detailed in Table 3-1

<b>Actors and devices</b>	<b>Role</b>
Citizen	- Someone who make the request.
Website Admin	- Someone who manages the website.
Database Admin	- Someone who manages the citizens' database connected to the website.
Device (mobile or computer)	- Device must be connected to the internet.
Internet	- Internet must be existed.

Table 3-1

### 3.4.2 Use Case

In this section the use case methodology is used to analyze system and to identify, clarify, and organize system requirements in a set of possible sequences of interactions or scenarios between the system and the users to reach the goal. It also describes user activities through the website.

#### 3.4.2.1 Use Case Details

**Use case name:** Making a request through the website

**Actor:** Citizen / website user

**Brief description:** This use case describes the process of entering the user National ID and Password successfully to request the type of official paper needed to be fetched.

**Goal:** The successful completion of making a request through the website.

**Precondition:**

- Used device must be connected to the Internet.

**Flow of events:**

- Enter National ID and password at home page.
- Choosing the governmental establishments and official papers.
- Making the request and choosing the number of copies.

**Post conditions:**

- Logging out to return to Home page.
- Make another request

**Exception:**

If the user enters invalid ID or password, and error message appears.

### 3.4.2.2 Use case diagram

The use case diagram for the website including users and their activates through the website process is shown in Figure 3-6

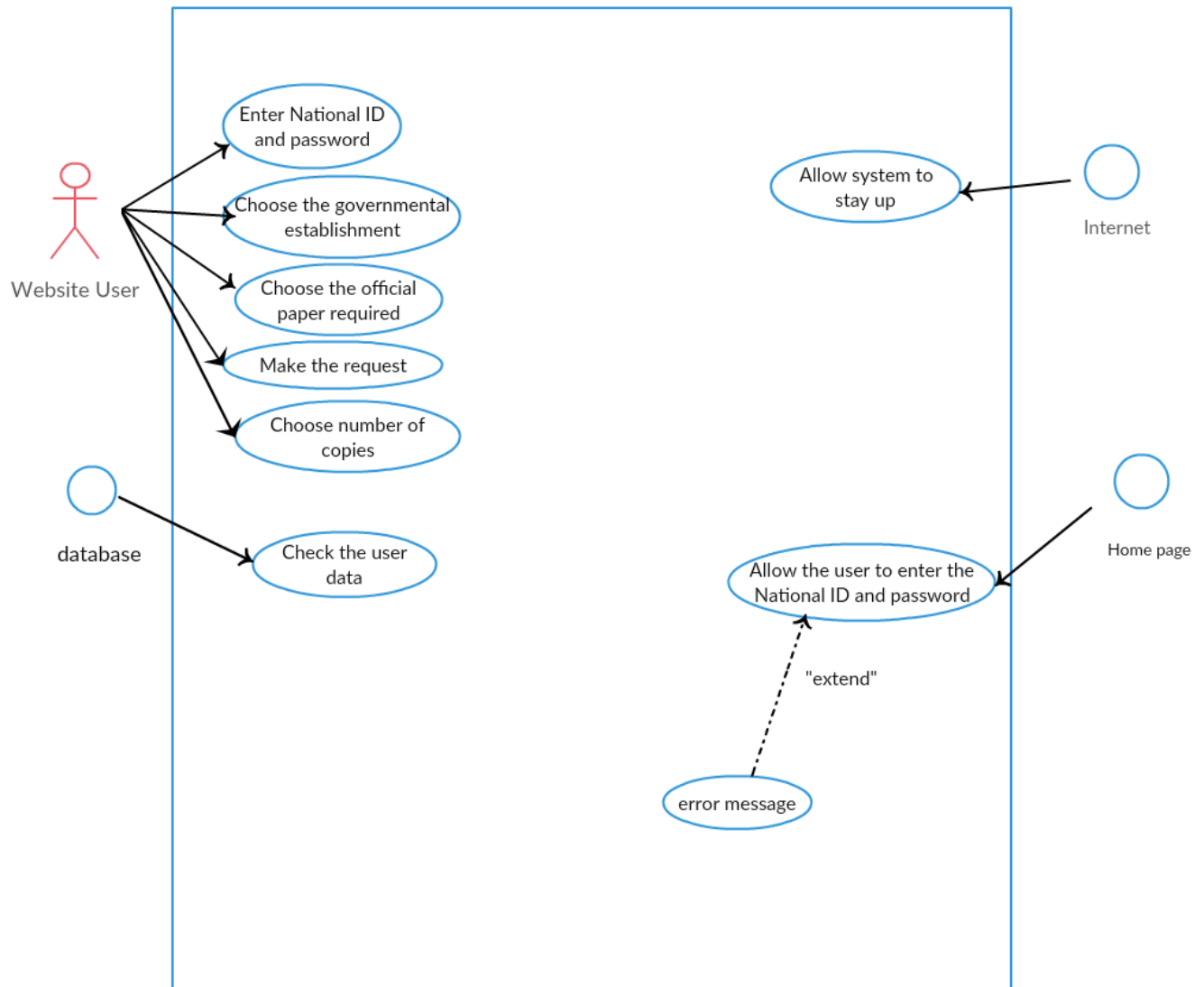


Figure 3-6



### 3.5 Website Administrator interface

The website administrator is one of the actors interact with the website. He is responsible for checking any errors in requests and updating types of papers and establishments available on the website.

The website administrator has a different interface than the users, as the users can access only to their interface while the administrator can access for both user and admin interfaces As shown in Figure 3-7, the website administrator can access to his interface through the same page as the user but by using National ID and password both are 000000 to introduce himself as an administrator to the website.



Figure 3-7

The website administrator can add new governmental establishments or new official papers through the following page in Figure 3-8 and Figure 3-9.

## Index

[Create New](#)

Msl7a_Name	
مصلحة الأحوال المدنية	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
المروور	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>

Figure 3-8

## Index

[Create New](#)

Mostanad_ID	Mostanad_Name	Msl7a_Name	
4	بطاقة	مصلحة الأحوال المدنية	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
5	شهادة الميلاد	مصلحة الأحوال المدنية	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
6	جند 2	المروور	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
7	جند 7	المروور	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>

Figure 3-9

## 3.6 Publishing and Hosting the Website

This section gives a detailed explanation of how to publish and host the website so as the website will be able to be viewed in a browser. The website will be hosted using the Internet Information Service Manager (IIS)

### 3.6.1 Steps to publish and Host the website

To publish and host the website the following three steps needed to be done.

#### 3.6.1.1 Step one

As shown in Figure 3-10, Open your website solution in Visual Studio then Open Solution Explorer, Right-Click on your Solution Name then Click on Publish.

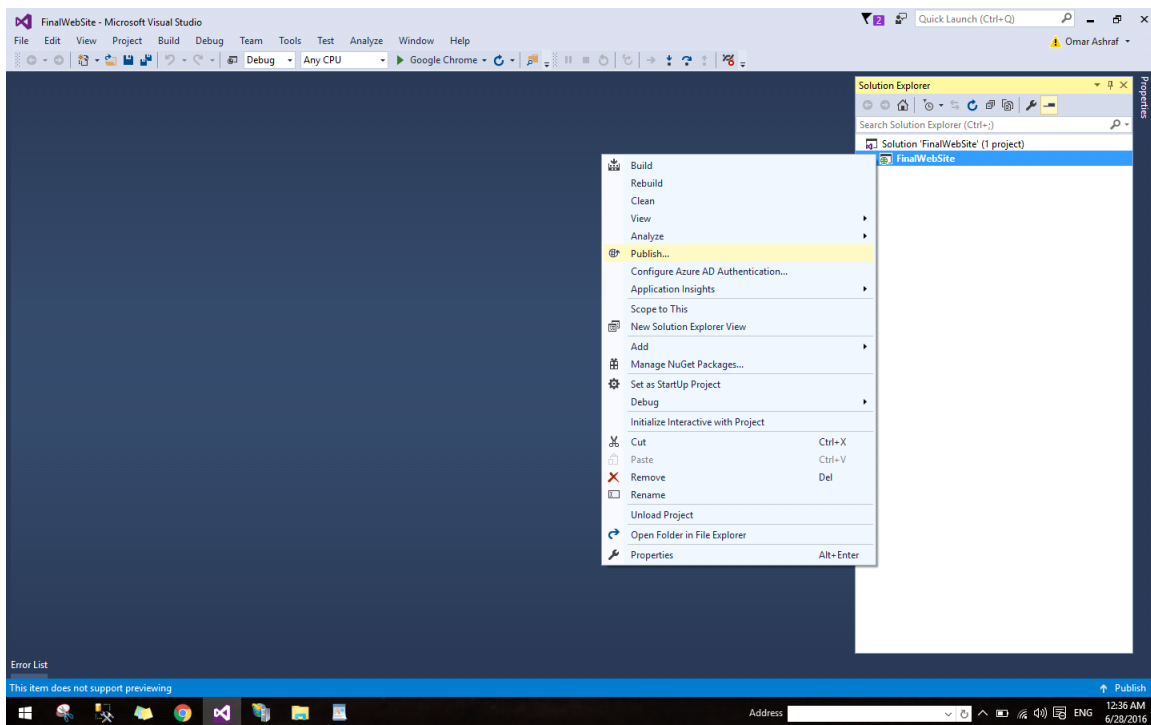


Figure 3-10

You will be redirected to a page as shown in Figure 3-11 where you can click on Custom, write “Local” and hit OK.

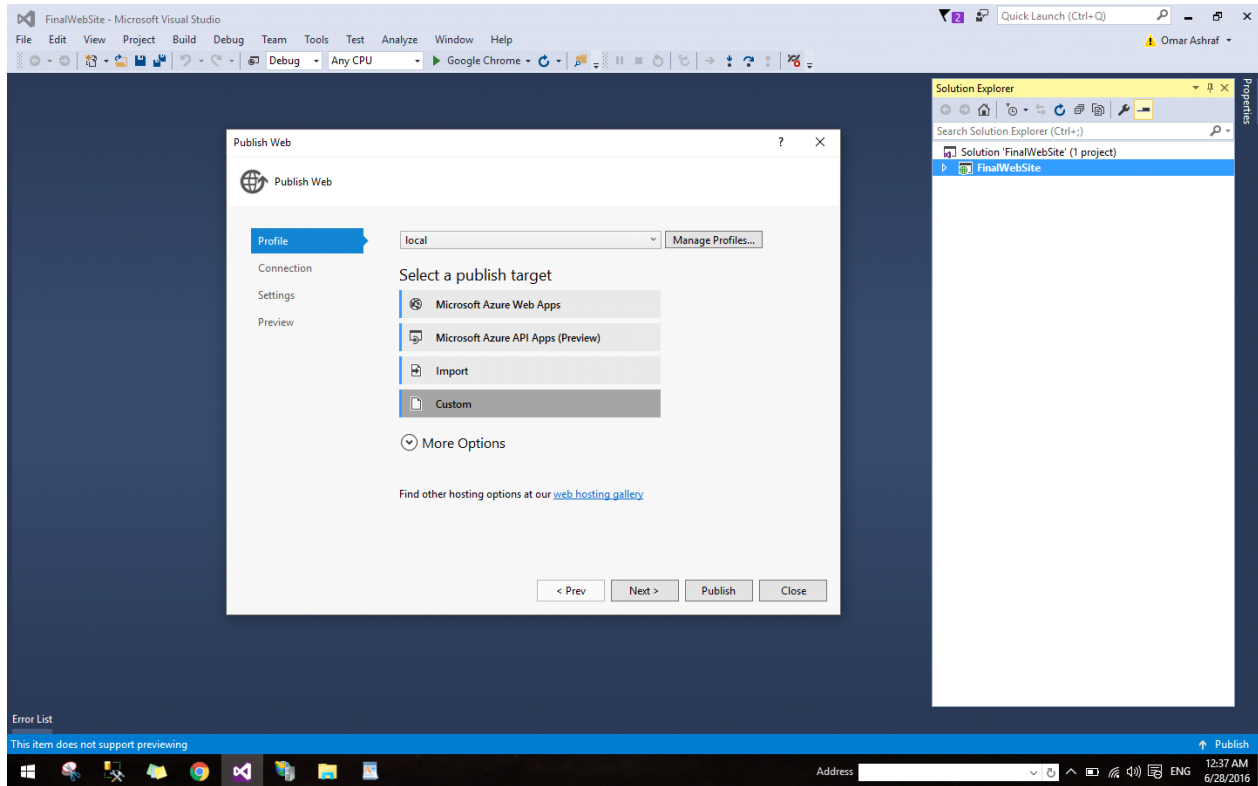


Figure 3-11

After that, you can click Next to set Publish Method to File System then Go to partition (C) and Create new folder, to set the Target Location to this new created folder, Click Next and Set Configuration to Release-Any CPU then Click Publish.

### 3.6.1.2 Step two

As shown in Figure 3-12, Open Start Menu and Search For IIS. On the left Side there is your connections expand your connections and you will find sites then Right-Click on Sites and choose Add Website

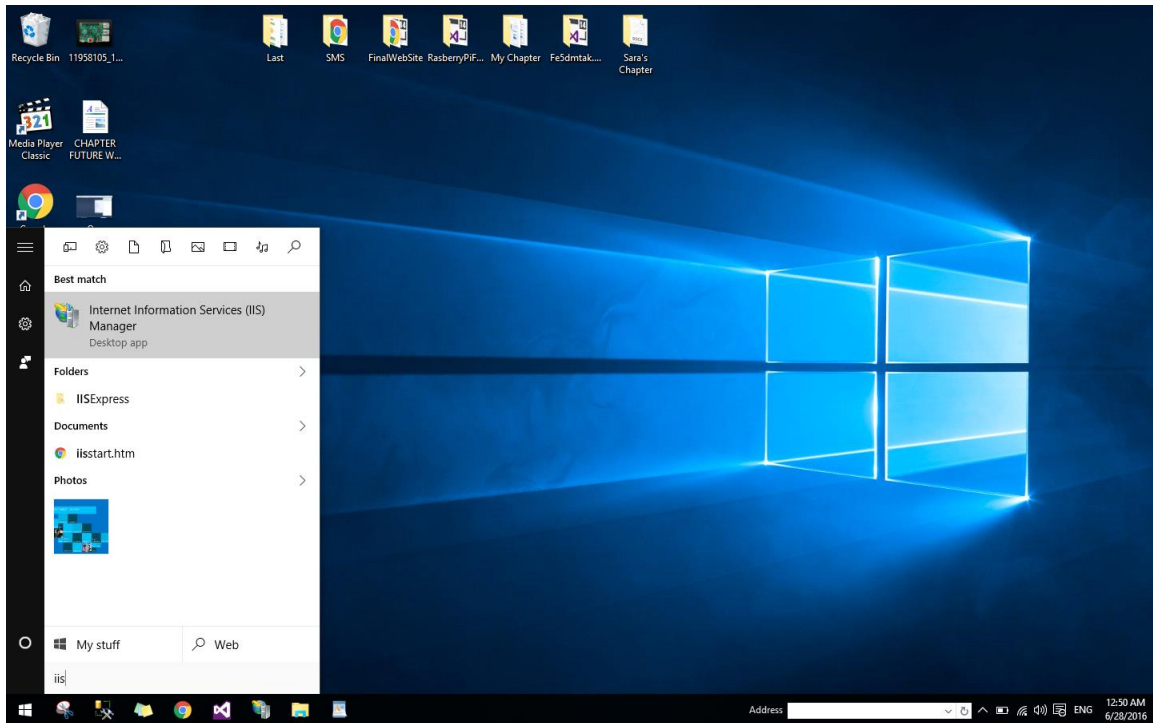


Figure 3-12

A Window will Open as shown in Figure 3-13, where you can Select Your Site Name, Select Your Physical Path ( The Folder You Created in partition (C) ), Type http (Default), IP address All Unassigned (Default), Port 80 (Default), Host name will be your Site address ex. [ www.example.com ] then you can check start Website immediately ( Default) and Hit OK.

The screenshot shows the 'Add Website' dialog box with the following fields and options:

- Site name:** An empty text box.
- Application pool:** A dropdown menu showing 'DefaultAppPool' and a 'Select...' button.
- Content Directory:**
  - Physical path:** An empty text box with a browse button (...).
  - Pass-through authentication:** A section containing 'Connect as...' and 'Test Settings...' buttons.
- Binding:**
  - Type:** A dropdown menu showing 'http'.
  - IP address:** A dropdown menu showing 'All Unassigned'.
  - Port:** A text box containing '80'.
  - Host name:** An empty text box.
  - Example:** 'www.contoso.com or marketing.contoso.com'.
- Start Website immediately:** A checked checkbox.
- Buttons:** 'OK' and 'Cancel' buttons at the bottom right.

Figure 3-13

Now you are back to connections in the left side of screen, expand it again and then expand sited you will be able to see your new site you have just created and click on your website, as shown in Figure 3-14 on the right side an icon called Directory Browsing, Click on Directory Browsing and a Page will Open, make Sure you Enable this from the right Side (Actions) as shown in Figure 3-15.

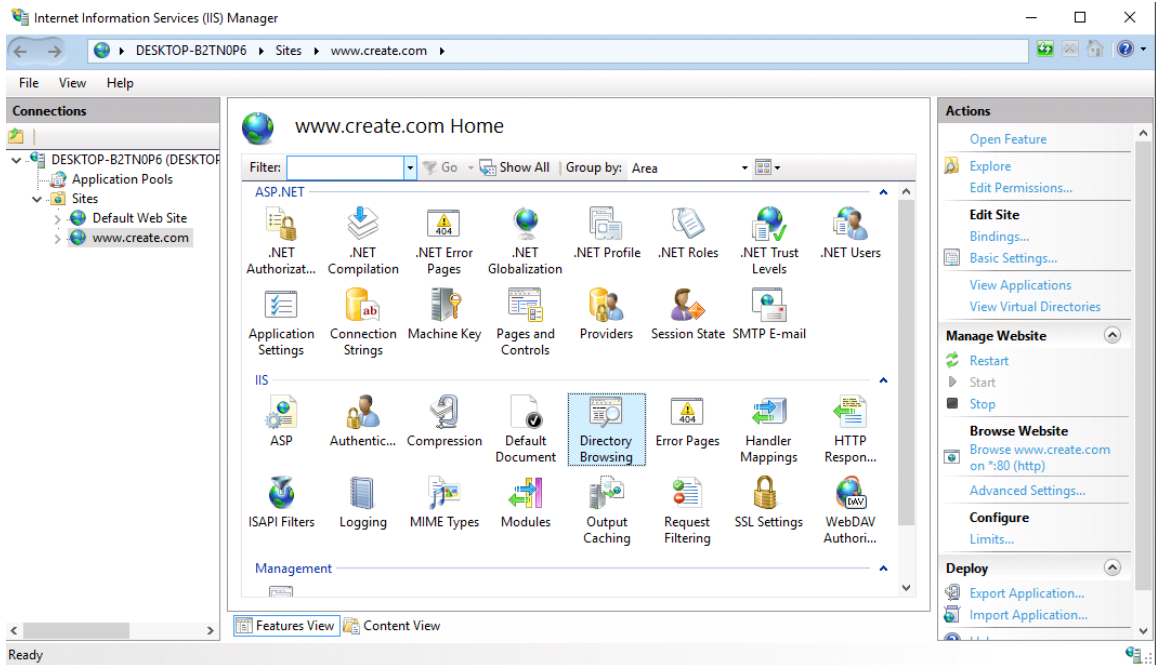


Figure 3-14

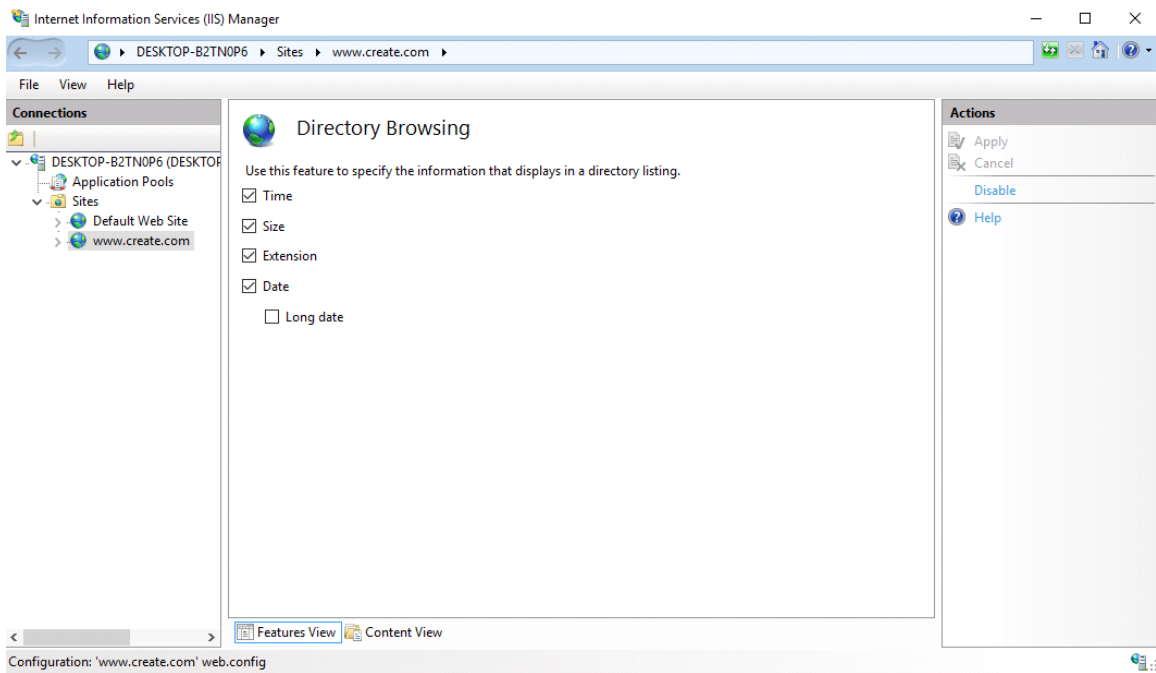


Figure 3-15

### 3.6.1.3 Step three

As shown in Figure 3-16, Open Notepad in Administrator Mode, Click on File then Open and Go to this Path “ C:\Windows\System32\drivers\etc” then Select hosts and Open it.

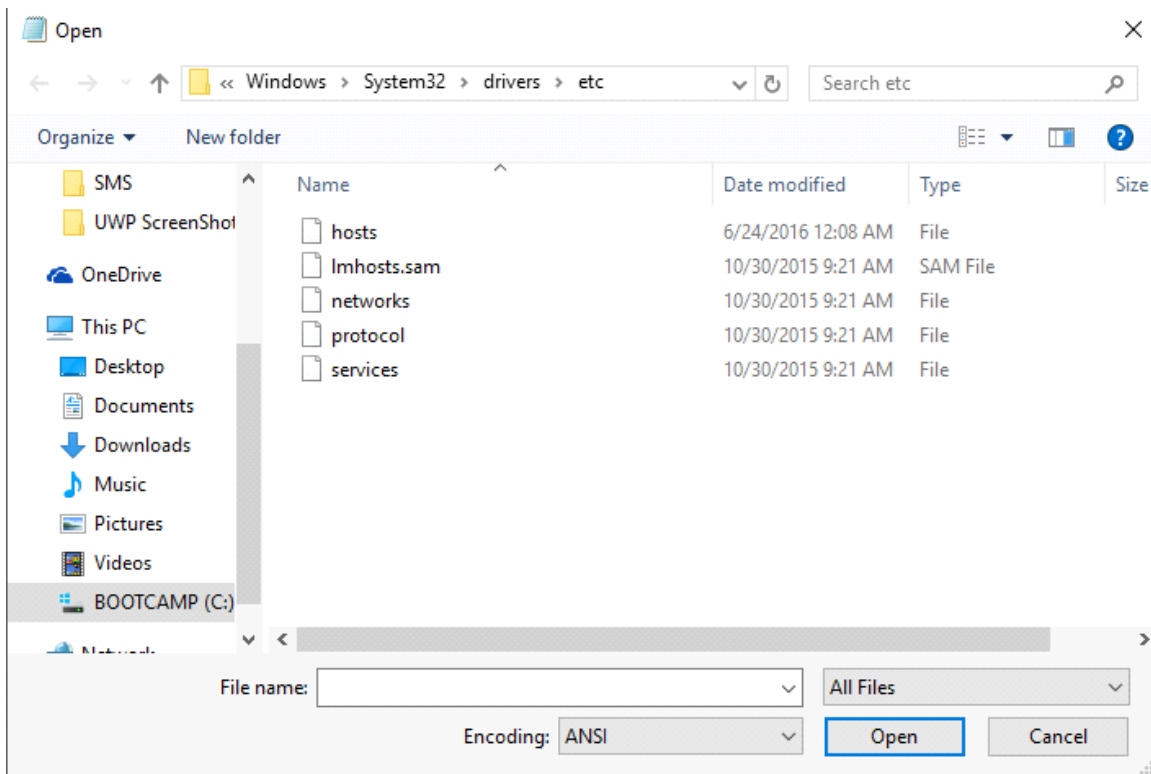
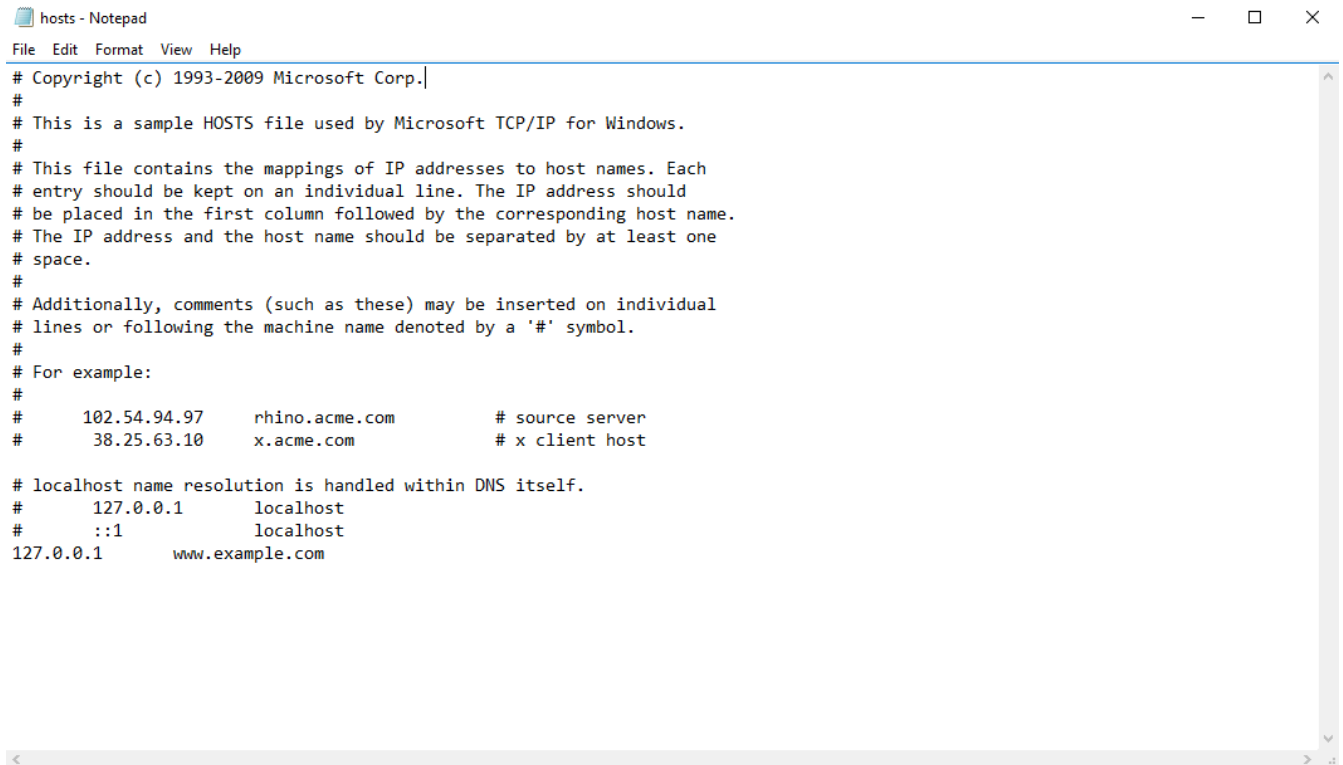


Figure 3-16



For the step in Figure 3-17, Write 127.0.0.1 www.example.com (This is the Host name you Created) then Save the file with the edited hosts, at last you will be able to Browse your website from your browser.



```
hosts - Notepad
File Edit Format View Help
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#       102.54.94.97     rhino.acme.com   # source server
#       38.25.63.10     x.acme.com      # x client host

# localhost name resolution is handled within DNS itself.
#       127.0.0.1       localhost
#       ::1             localhost
127.0.0.1       www.example.com
```

Figure 3-17

### 3.7 System Database

The “Creating Request website” section responsible for making the request is connected to a database section which is represented in the database system for the citizens that hold their details: Names, National ID, Nationality, Religion, Mother Name, and the user password, and phone number.

Figure 3-18 shows how the connection is done by using some kind of program between these two sections.

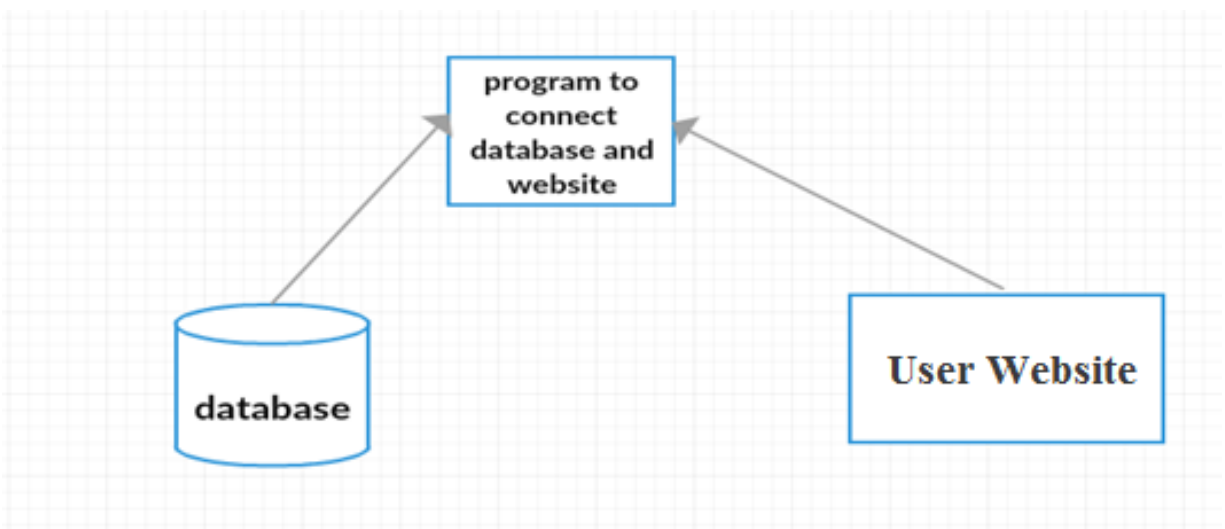
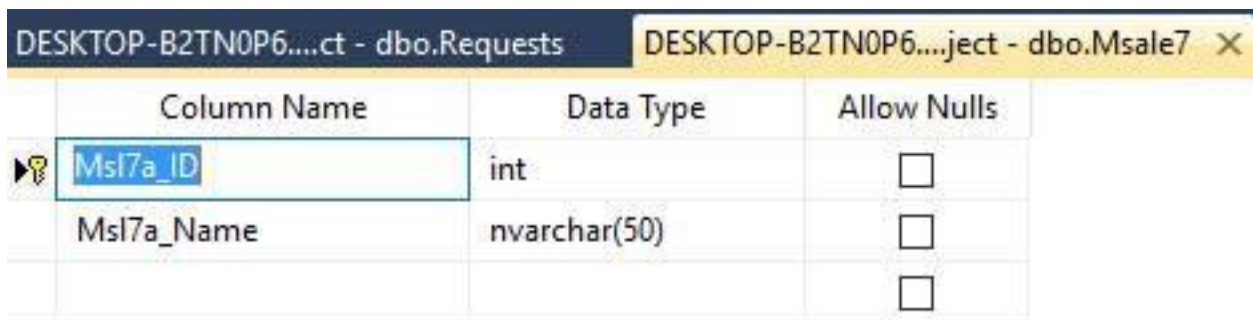


Figure 3-18

### 3.7.1 Database Interface

The database administrator is the one who is responsible for watching the database for errors, new entries and also updates. The database administrator deals with the database through this interface shown in the following figures.

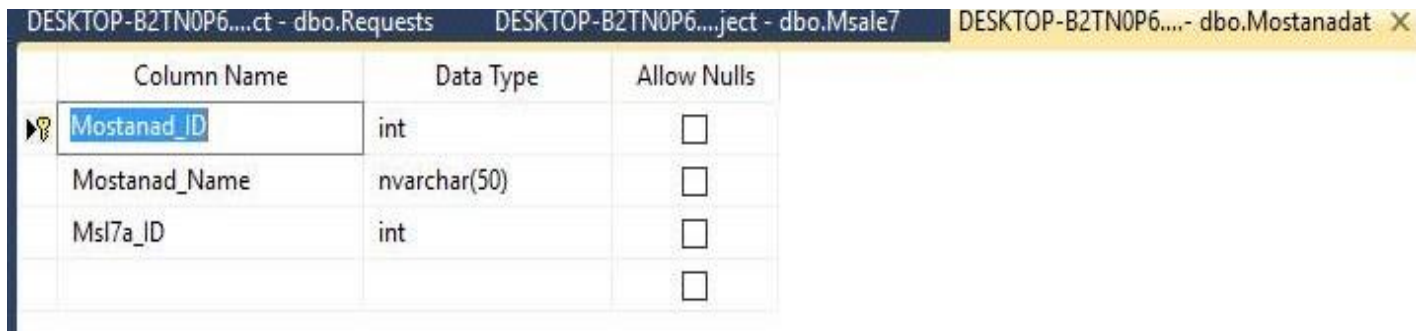
Figure 3-19 shows the possible type of governmental establishments that are available at the “Creating Request website”.



Column Name	Data Type	Allow Nulls
Msl7a_ID	int	<input type="checkbox"/>
Msl7a_Name	nvarchar(50)	<input type="checkbox"/>
		<input type="checkbox"/>

Figure 3-19

While Figure 3-20 shows the type of official papers related to each governmental establishments that are available at the “Creating Request website”.



Column Name	Data Type	Allow Nulls
Mostanad_ID	int	<input type="checkbox"/>
Mostanad_Name	nvarchar(50)	<input type="checkbox"/>
Msl7a_ID	int	<input type="checkbox"/>
		<input type="checkbox"/>

Figure 3-20

For Figure 3-21, it shows the table holding the Citizens' information that is been checked for validation by the website when the user enter username and password, or when website calls user data for making the request and choosing number of copies.

DESKTOP-B2TN0P6....ct - dbo.Requests			DESKTOP-B2TN0P6....ject - dbo.Msale7		
	Column Name	Data Type	Allow Nulls		
▶	National_ID	nvarchar(50)	<input type="checkbox"/>		
	Citizen_Name	nvarchar(50)	<input type="checkbox"/>		
	Password	nvarchar(50)	<input type="checkbox"/>		
	BirthDate	date	<input type="checkbox"/>		
	Relegion	nvarchar(50)	<input type="checkbox"/>		
	Mother_Name	nvarchar(50)	<input type="checkbox"/>		
	M7l_elmelad	nvarchar(50)	<input type="checkbox"/>		
	Nationality	nvarchar(50)	<input type="checkbox"/>		
			<input type="checkbox"/>		

Figure 3-21

The table in Figure 3-22, holds the citizens' requests as it contains the type of the governmental establishments, the type of official papers selected and the number of copies entered, even it holds data if the required papers been printed or not and whether the mobile message is been sent to the user mobile or not.

DESKTOP-B2TN0P6....ct - dbo.Requests			DESKTOP-B2TN0P6....ject -		
	Column Name	Data Type	Allow Nulls		
▶	Request_ID	int	<input type="checkbox"/>		
	Msl7a_ID	int	<input type="checkbox"/>		
	Mostanad_ID	int	<input type="checkbox"/>		
	No_of_copies	int	<input type="checkbox"/>		
	Citizen_ID	nvarchar(50)	<input type="checkbox"/>		
	Printed_or_not	bit	<input type="checkbox"/>		
	toPrint	bit	<input checked="" type="checkbox"/>		
	Phone_Number	nvarchar(50)	<input type="checkbox"/>		
	Msg_Sent_Or_Not	bit	<input checked="" type="checkbox"/>		
			<input type="checkbox"/>		

Figure 3-22

Figure 3-23, shows how the database tables are connected to each other.

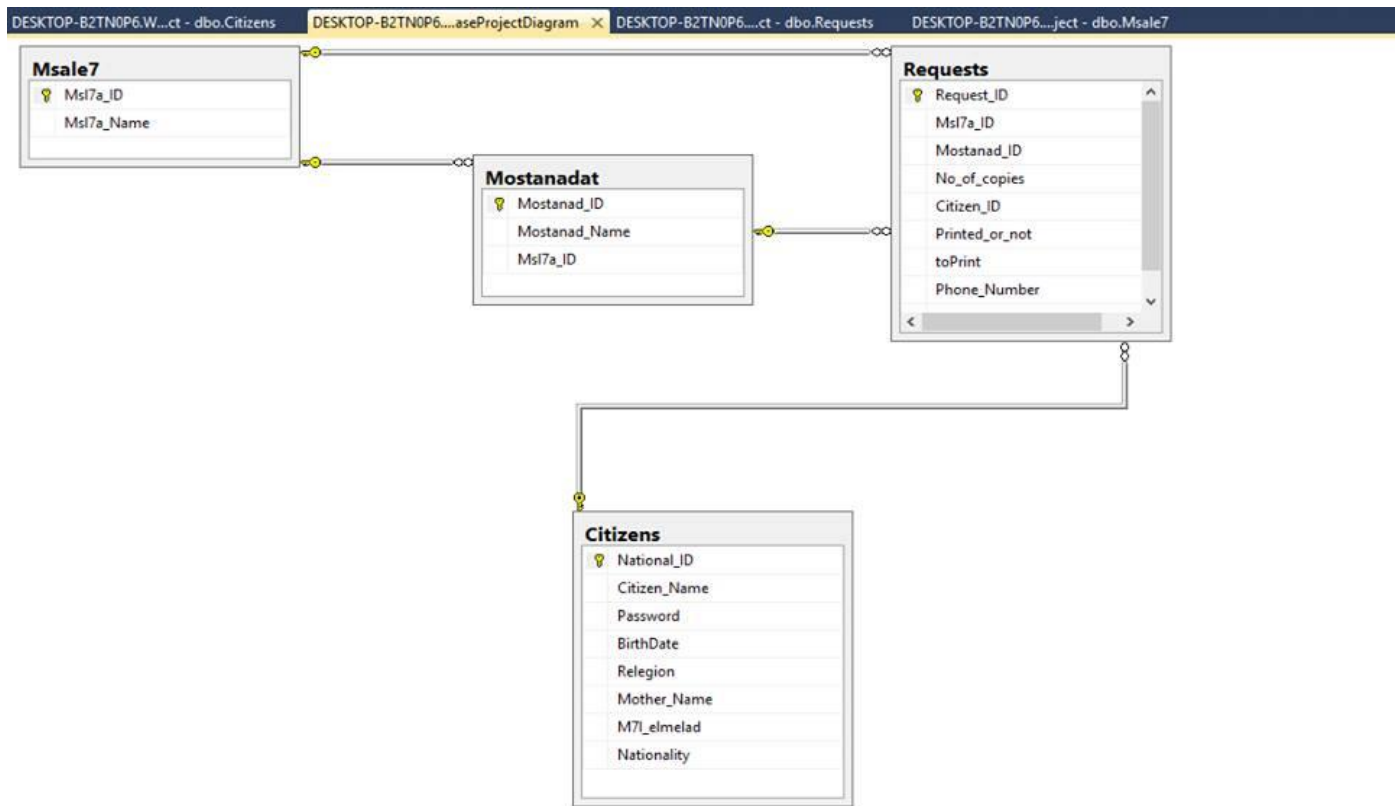


Figure 3-23

### 3.8 Conclusion

The user can make a request through the website by using the National ID and the password which are saved in the database. The admin has full view of the website also has the privilege to add, edit or delete the requests.

For the next chapter the user will know the steps for receiving the pre-created request.

# **Chapter 4**

## **Receiving Pre-created Requests Website**

### **4.1 Introduction**

This chapter gives a scope description and overview of everything included in the “Receiving Pre-created Requests Website”. It gives a detailed description of the requirements for the process of extracting the official papers through the Receiving Pre-created Requests Website. Also, it will explain system constraints, and interface.

### **4.2 Overview**

This section of the project deals with the Recieving the pre-created website for the users using the IDM machine outdoors or in the places specifies for it.

The website shown in Figure 4-1, allow the users to continue the process of extracting the official papers by selecting the required papers to be printed immediately while standing in front of the IDM machine as the machine is connected to a printed.



Figure 4-1

According to the process cycle through the “Receiving the pre-created websites, the user enter the National ID and the password which have been saved in the database before. The system checks if both the National ID and the password are valid, if they are valid the system continue the process if not it asks the user to re-enter them again. After that, a page containing details about the user and the requests he/she made before appears. Then the user can print the requests he/she requires.

The process cycle through the “Receiving the pre-created websites” is shown in the flowchart in Figure 4-2

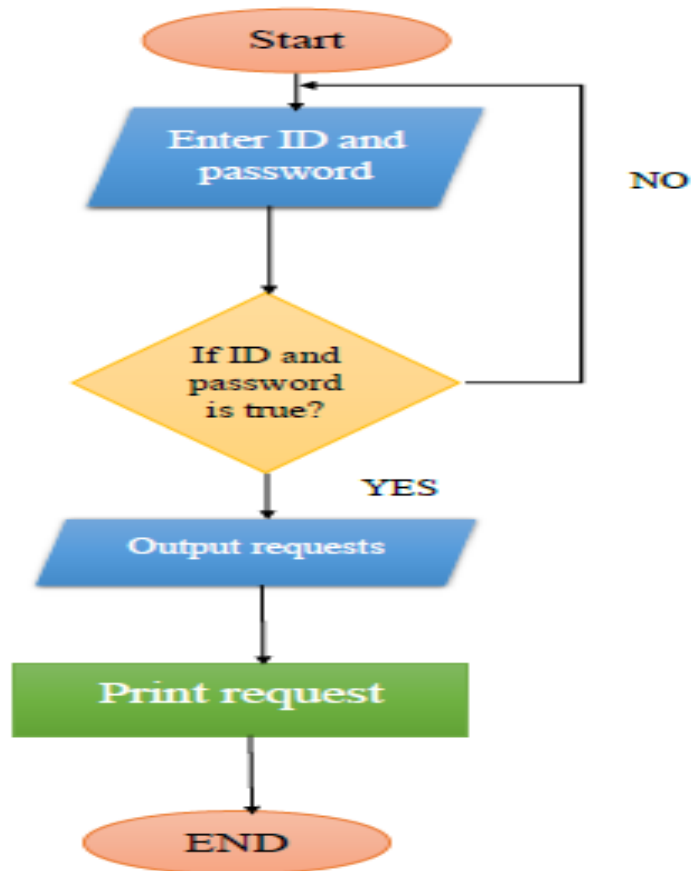


Figure 4-2



## 4.3 User interface

This section provides a detailed description of the user interface for the Raspberry-Pi website section according to the cycle of the process the user involves in.

### 4.3.1 Website interface

First the user sees two text boxes in the Home page where he/she can enter National ID and Password as shown in Figure 4-3, then the website checks the user ID and password and allow access.

Figure 4-3

After that, the Saved Requests page shown in Figure 4-4, appears containing users' information. It also shows details about the request the user made through the user website and been saved in the database as shown in Figure 4-5, with a print button and check box in front of every request so the user can check the paper he/she wants to print.

## المعلومات الشخصية

أسم العميل:

تاريخ الميلاد:

القبيلة:

أسم الأم:

محل الميلاد:

الجنسية:

أسم المستند	أسم المصلحة	عدد النسخ	طباعة
صورة بطاقة	مصلحة الأحوال المدنية	3	طباعة
استمارة جند 7	المرور	5	طباعة
استمارة جند 2	المرور	1	طباعة
استمارة جند 2	المرور	3	طباعة

Figure 4-4

Request_ID	Msl7a_ID	Mostanad_ID	No_of_copies	Citizen_ID	Printed_or_not	toPrint	Phone_Number	Msg_Sent_O...
1	1	1	3	123456	False	True	0123456789	NULL
2	2	4	2	123456	False	True	0123456789	NULL
3	2	3	1	123456	False	NULL	0123456789	NULL
4	2	3	3	123456	False	NULL	0123456789	NULL

Figure 4-5

## 4.4 Conclusion

Through the “Receiving Pre-created Requests Website” the user will be able to print his/her request and receive it immediately while standing in front of the IDM machine. The next Chapter will describe in details how the printing process is done.

# **Chapter 5**

## **Windows Form Applications**

### **5.1 Introduction**

This chapter gives a scope description and overview of everything included in the two Windows Form Applications created in the project which are responsible for the process of Printing the Pre-created requests and also sending an SMS Message for the User.

### **5.2 Windows Form Applications**

The Windows Form Applications is a Desktop Applications, so they has to be opened manually on the PC.

The Windows Form Applications should be connected to the Database of the Project.

The Applications contains only one Form which shows the Requests of the User to be printed.

The Applications will only work in the background and will not be visible for user to see it, its only purpose is to print the Requests through giving the order to the printer, and Sending SMS Message through giving the order to the GSM module.

## **5.3 GSM Module**

This section discusses the GSM module which is responsible for the process of sending an SMS to the user mobile phone after creating a request for an official paper extraction through the “Creating Request Website”. The SMS tells the user that the created request has been received and that he/she can visit any nearby IDM machine to receive the pre-created request.

### **5.3.1 GSM Brief Overview**

GSM is a Time Division Multiple access (TDMA) based wireless network technology developed in Europe that is used throughout most of the world. GSM phones make use of a SIM card to identify the user's account.

The use of the SIM card allows GSM network users to quickly move their phone number from one GSM phone to another by simply moving the SIM card. Currently GSM networks operate on the 850MHz, 900MHz, 1800MHz, and 1900MHz frequency bands.

Devices that support all four bands are called quad-band, with those that support 3 or 2 bands called tri-band and dual-band, respectively. GSM is Also known as: "Global System for Mobile Communications".

### **5.3.2 GSM Application Process**

The GSM Application as shown in Figure 5-1 is dealing with sending an SMS message to the user upon making his/ her request from the "Creating Requests Website" or from the IDM machine itself using the Universal Windows Application (UMP) which will be included in details in Chapter 6.

The SMS message will be "Your Request Has Been Sent To Us, You Can Visit any Nearby IDM to Receiver and Print your Request".

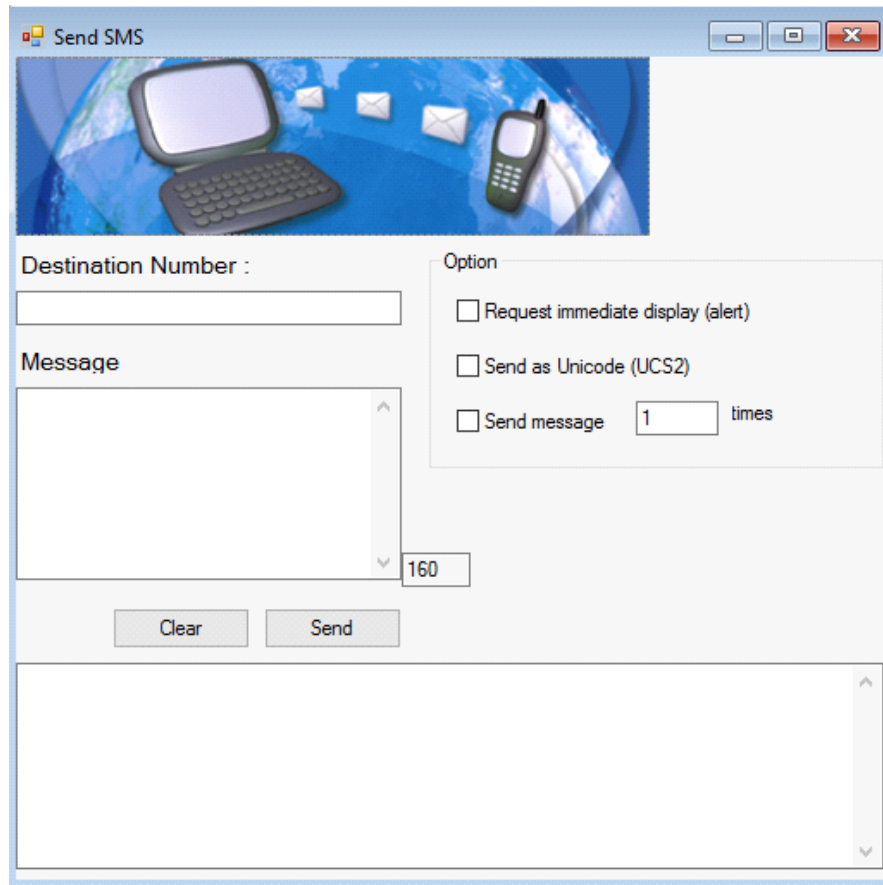
The image shows a Windows-style application window titled "Send SMS". The window has a standard title bar with minimize, maximize, and close buttons. Below the title bar is a decorative banner image featuring a laptop, a mobile phone, and several email icons on a blue globe background. The main content area is divided into several sections. On the left, there is a label "Destination Number :" followed by a single-line text input field. Below this is a label "Message" followed by a multi-line text area with a vertical scrollbar. To the right of the message area, there is a label "Option" followed by three checkboxes: "Request immediate display (alert)", "Send as Unicode (UCS2)", and "Send message". The "Send message" checkbox is checked, and next to it is a small text input field containing the number "1", followed by the word "times". Below the message text area, there are two buttons: "Clear" and "Send". At the bottom of the window, there is a large, empty text area with a vertical scrollbar. A small status bar at the bottom right of the message text area shows the number "160".

Figure 5-1

According to Figure 5-2, while the user is finishing the request through the "Creating Requests Website" he/she is asked to enter the number of copies and the Mobile number which are variable inputs in the database.

Figure 5-2

The GSM Application searches the database for the cell "Msg\_Sent\_Or\_Not" to be False which means message is not sent yet, as shown in Figure 5-3, when the cell is False the application sends the SMS message to the user mobile number which is pre-entered during making a request process.

After that, the cell "Msg\_Sent\_Or\_Not" in the database is set to True which means the GSM process is been accomplished.

Request_ID	Mel7a_ID	Mostanad_ID	No_of_copies	Citizen_ID	Printed_or_not	toPrint	Phone_Number	Msg_Sent_Or_Not
11	1	1	2	123456	<input type="checkbox"/>	<input type="checkbox"/>	01153214148	<input type="checkbox"/>
12	2	4	1	123456	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	01153214148	<input type="checkbox"/>
					<input type="checkbox"/>	<input type="checkbox"/>		<input type="checkbox"/>

Figure 5-3

The process through the GSM application is explained in the flowchart in Figure 5-4

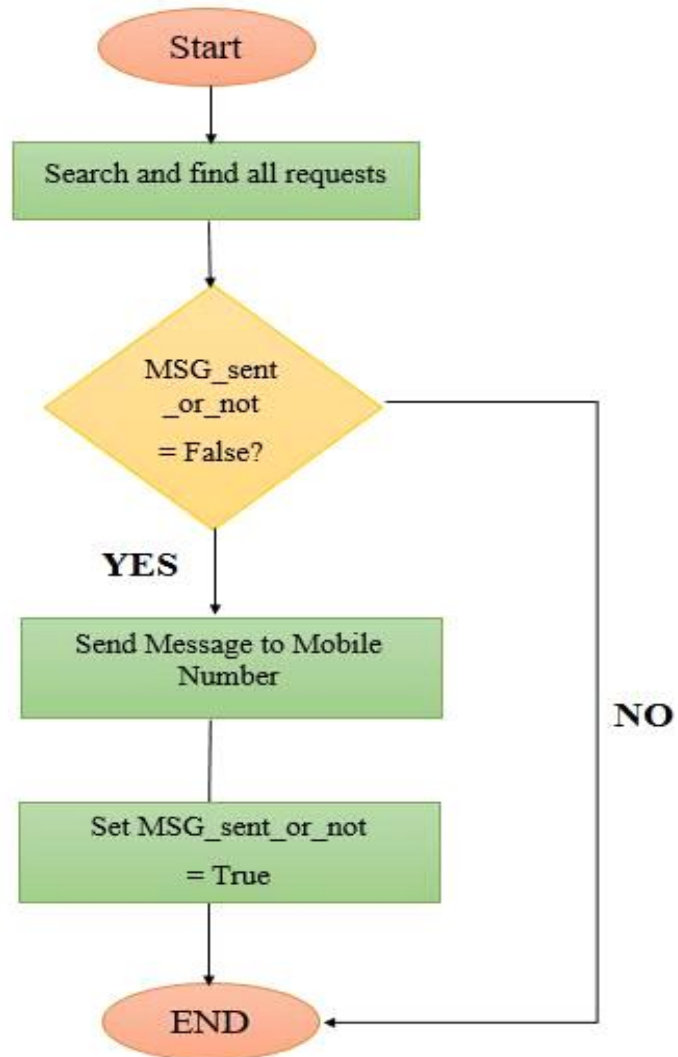


Figure 5-4

## 5.4 Printing Application

This section discusses the Printing Application which is responsible for the process of printing the pre-created requests received by the IDM machine.

### 5.4.1 GSM Application Process

According to Figure 5-5, when the user enters a valid National ID and password in the “receiving the pre-created requests website” a page containing detailed information about the user appears with a table of the requests he/she made. Before each request the user can select the request he/she wants to print.

IDM

Log in

المعلومات الشخصية

أسم العميل:

تاريخ الميلاد:

القياس:

أسم الأم:

محل الميلاد:

الجنسية:

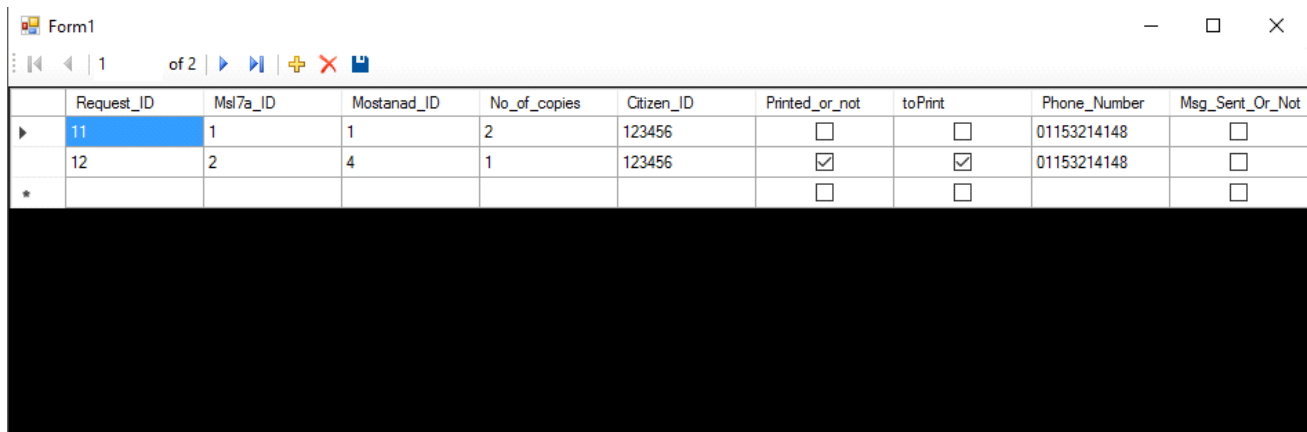
أسم المستند	أسم المصلحة	عدد الترخيص	طباعة
صورة بطاقة	مصلحة الأحوال المدنية	3	طباعة
استمارة جند 7	المرور	5	طباعة
استمارة جند 2	المرور	1	طباعة
استمارة جند 2	المرور	3	طباعة

Figure 5-5



Since the printing application is connected to the database, it displays all the requests of the users as shown in Figure 5-6, but only searches for the cell "toprint" to be True and cell "Printed\_or\_not" to be False which means that this request is ordered to be printed and it was not printed before.

If the cell "toprint" is set to True and cell "Printed\_or\_not" is False, it prints this request with the desired number of copies, then it sets the cell "Printed\_or\_not" to True in order not to be printed over again.



	Request_ID	Msl7a_ID	Mostanad_ID	No_of_copies	Citizen_ID	Printed_or_not	toPrint	Phone_Number	Msg_Sent_Or_Not
▶	11	1	1	2	123456	<input type="checkbox"/>	<input type="checkbox"/>	01153214148	<input type="checkbox"/>
	12	2	4	1	123456	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	01153214148	<input type="checkbox"/>
*						<input type="checkbox"/>	<input type="checkbox"/>		<input type="checkbox"/>

Figure 5-6

The process through the Printing application is explained in the flowchart in Figure 5-7

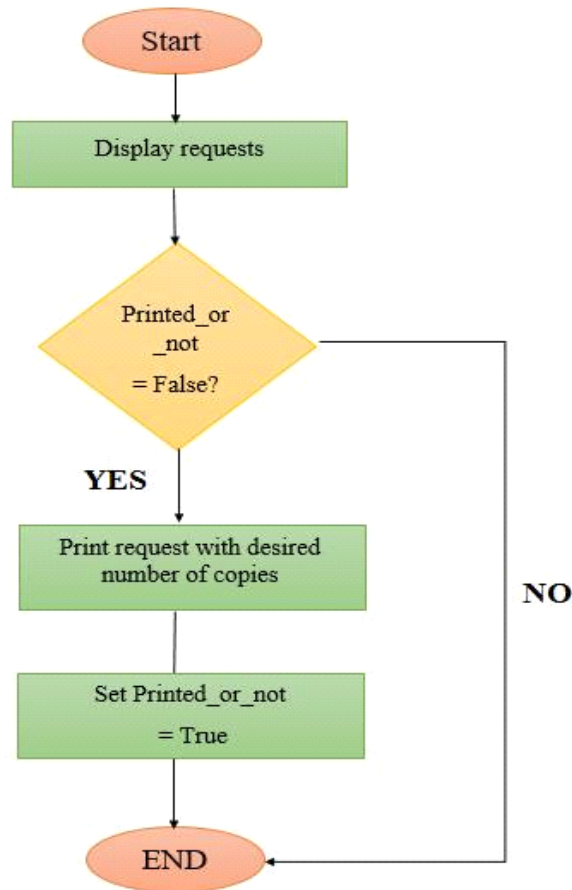


Figure 5-7

## 5.5 Conclusion

There are two windows applications in this project: one for the printing process and the other is for sending an SMS. For the printing process when the user click Print the cells toPrint and Msg\_sent\_or\_not in in Database become true. While for the GSM module sends an SMS message to the user upon making the request through the "Creating Requests Website".

The next chapter will discuss the Universal windows application that holds the two websites.

# Chapter 6

## Universal Windows Application (UWP)

### 6.1 Introduction

This chapter gives a scope description and overview of everything included in this Universal Application section. It also gives a detailed description of the requirements for the process of displaying the Interface and processing of the universal application.

### 6.2 Universal Windows Platform (UWP) app

A Universal Windows Platform (UWP) app as shown in Figure 6-1, is a Windows experience that is built upon the Universal Windows Platform (UWP), which was first introduced in Windows 8 as the Windows Runtime. At the core of UWP apps is the idea that users want their experiences to be mobile across ALL their devices, and they want to use whatever device is most convenient or productive for the task at hand.

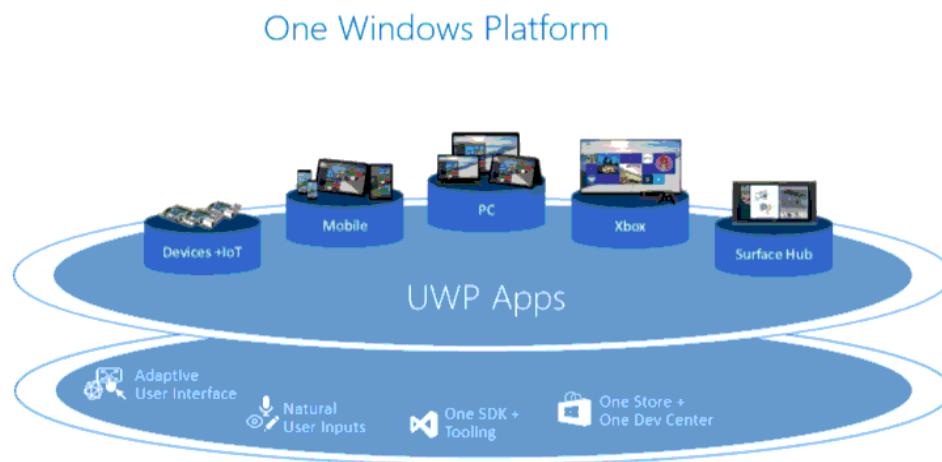


Figure 6-1

Windows 10 makes it easier to develop apps for the UWP with just one API set, one app package, and one store to reach all Windows 10 devices – PC, tablet, phone and more. It's easier to support a number of screen sizes, and also a variety of interaction models, whether it be touch, mouse & keyboard, a game controller, or a pen as shown in Figure 6-2.



Figure 6-2

### 6.2.1 Characteristics of UWP apps on Windows 10

You target device families, not an OS.

A device family identifies the APIs, system characteristics, and behaviors that you can expect across devices within the device family. It also determines the set of devices on which your app can be installed from the store.

Apps are packaged and distributed using the AppX packaging format.

All UWP apps are distributed as an AppX package. This provides a trustworthy installation mechanism and ensures that your apps can be deployed and updated seamlessly.

There's one store for all devices.

After you register as an app developer, you can submit your app to the store and make it available on all device families, or only those you choose. You submit and manage all your apps for Windows devices in one place.

There's a common API surface across device families.

The Universal Windows Platform (UWP) core APIs are the same for all Windows device families. If your app uses only the core APIs, it will run on any Windows 10 device.

Extension SDKs make your app light up on specialized devices. Extension SDKs add specialized APIs for each device family. If your app is intended for a particular device family, you can make it light up by using these APIs. You can still have one app package that runs on all devices by checking what device family your app is running on before calling an extension API.

### Adaptive Controls and input

UI elements use effective pixels (see Responsive design 101 for UWP apps), so they automatically adapt themselves based on the number of screen pixels available on the device. And they work well with multiple types of input such as keyboard, mouse, touch, pen, and Xbox One controllers. If you need to further tailor your UI to a specific screen size or device, new layout panels and tooling help you adapt your UI to the devices your app may run on.

## **6.2.2 UWP apps come to life on Windows**

On Windows, your app can deliver relevant, real-time info to your users and keep them coming back for more. In the modern app economy, your app has to be engaging to stay at the front of your users' lives. Windows provides you with lots of resources to help keep your users returning to your app:

Live tiles and the lock screen show contextually relevant and timely info at a glance.

Push notifications bring real-time, breaking alerts to your user's attention when they're needed.

The Action Center is a place where you can organize and display notifications and content that users need to take action on.

Background execution and triggers bring your app to life just when the user needs it.

Your app can use voice and Bluetooth LE devices to help users interact with the world around them.

### 6.2.3 Monetize your app your way

On Windows, you can choose how you'll monetize your app—across phones, tablets, PCs, and other devices. We give you a number of ways to make money with your app and the services it delivers. All you need to do is choose the one that works best for you:

A paid download is the simplest option. Just name your price.

Trials give you a great way to let users try your app before buying it, providing easier discoverability and conversion than the more traditional "freemium" options.

In-app purchase offers you the most flexibility for monetizing your app.

### 6.2.4 UWP app features

Effective pixels and scaling. UWP apps automatically adjust the size of controls, fonts, and other UI elements so that they are legible on all devices.

When your app runs on a device, the system uses an algorithm to normalize the way UI elements display on the screen. This scaling algorithm takes into account viewing distance and screen density (pixels per inch) to optimize for perceived size (rather than physical size). The scaling algorithm ensures that a 24 px font on Surface Hub 10 feet away is just as legible to the user as a 24 px font on 5' phone that's a few inches away as shown in Figure 6-3.

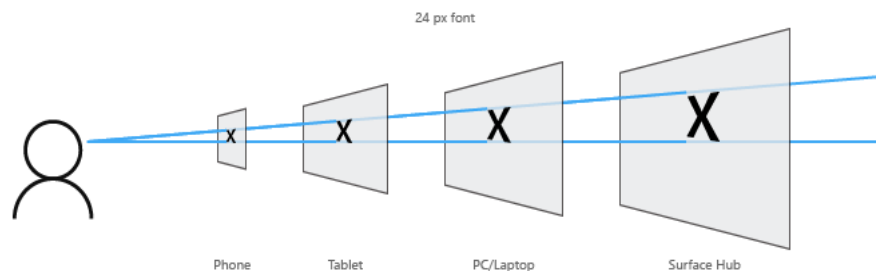


Figure 6-3

Because of how the scaling system works, when you design your UWP app, you're designing in effective pixels, not actual physical pixels. So you can ignore the pixel density and the actual screen resolution when designing. Instead, design for the effective resolution (the resolution in effective pixels) for a size class.

### 6.2.5 Responsive design techniques

When you optimize your app's UI for specific screen widths, we say that you're creating a responsive design. Here are six responsive design techniques you can use to customize your app's UI.

You can alter the location and position of app UI elements to get the most out of each device. In this example, the portrait view on phone or phablet necessitates a scrolling UI because only one full frame is visible at a time. When the app translates to a device that allows two full on-screen frames, whether in portrait or landscape orientation, frame B can occupy a dedicated space. If you're using a grid for positioning, you can stick to the same grid when UI elements are repositioned as shown in Figure 6-4.



Figure 6-4

You can optimize the frame size by adjusting the margins and size of UI elements. This could allow you, as the example here shows, to augment the reading experience on a larger screen by simply growing the content frame as shown in Figure 6-5.

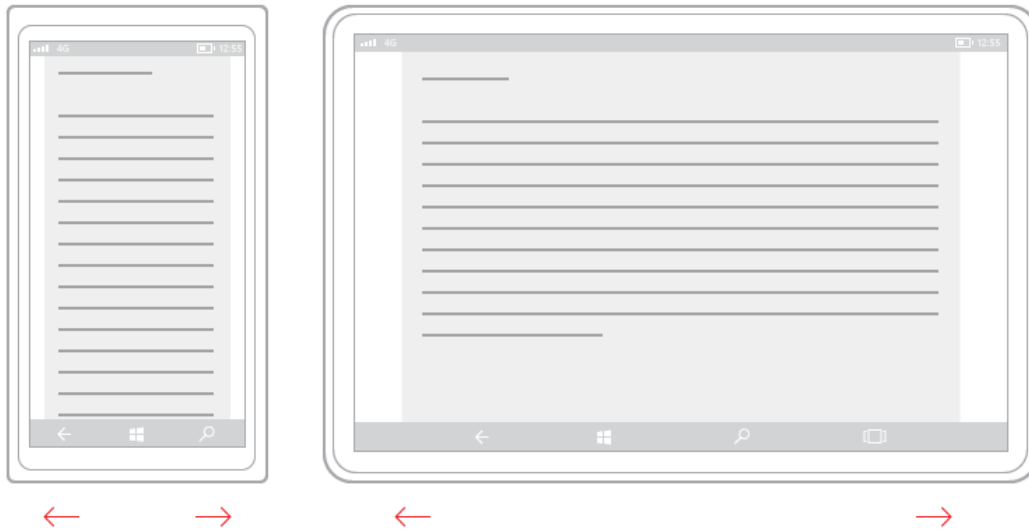


Figure 6-5

By changing the flow of UI elements based on device and orientation, your app can offer an optimal display of content. For instance, when going to a larger screen, it might make sense to switch larger containers, add columns, and generate list items in a different way. This example shows how a single column of vertically scrolling content on phone or phablet can be reflowed on a larger screen to display two columns of text as shown in Figure 6-6.





Figure 6-6

You can reveal UI based on screen real estate, or when the device supports additional functionality, specific situations, or preferred screen orientations.

In this example with tabs as in Figure 6-7, the middle tab with the camera icon might be specific to the app on phone or phablet and not be applicable on larger devices, which is why it's revealed in the device on the right. Another common example of revealing or hiding UI applies to media player controls, where the button set is reduced on smaller devices and expanded on larger devices. The media player on PC, for instance, can handle far more on-screen functionality than it can on a phone.



Figure 6-7

This technique lets you switch the user interface for a specific device size-class or orientation. In this example, the nav pane and its compact, transient UI works well for a smaller device, but on a larger device tabs might be a better choice as shown in Figure 6-8.



Figure 6-8

You can collapse or fork the architecture of your app to better target specific devices. In this example, going from the left device to the right device demonstrates the joining of pages as in Figure 6-9.

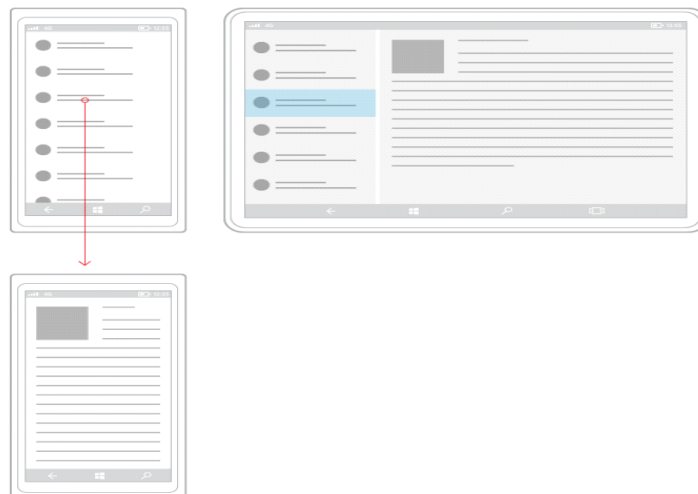


Figure 6-9

### 6.3 Requirements for UMP application

The Raspberry Pi should be connected to the Server through Router or an access point. The Application that runs on Raspberry Pi will be Started Automatically on Power on.

The Application contains a number of Pages to navigate through it freely just clicking on the desired page.

The Website allows the user to make a request or Reception of the pre-created request.

The Application allows the user to display the two websites in a web view.

### 6.4 User Interface

The user reads the description about the service and click a button to Start or click on about us button as shown in Figure 6-10.



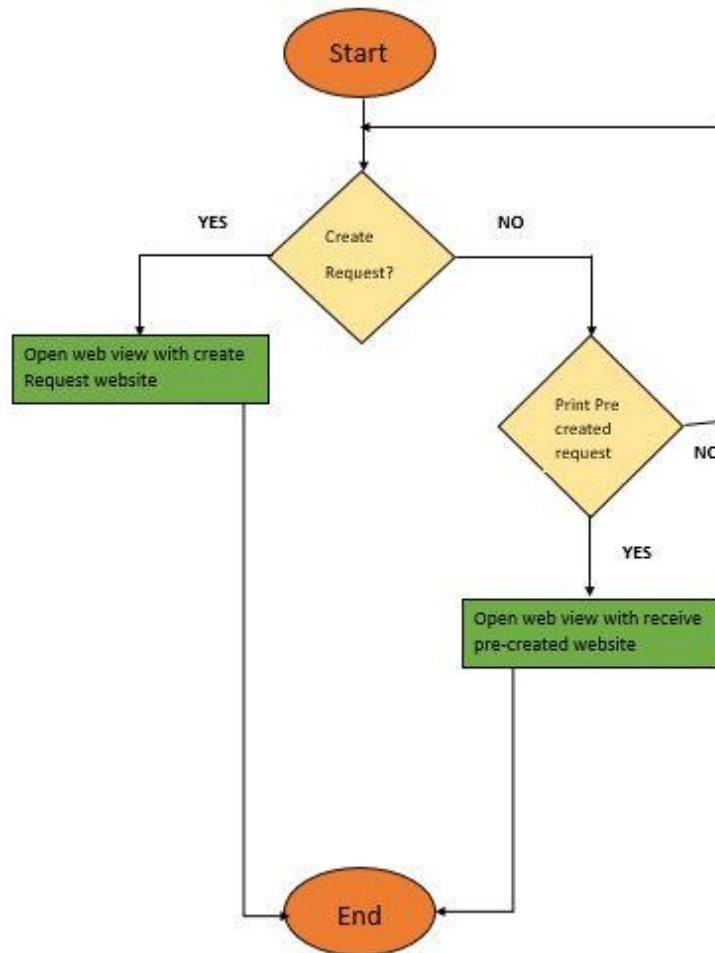
Figure 6-10

As in Figure 6-11, the user choose one of two services (to create a request or to receive a pre-created request to be printed eventually). After that, the user can open the website in the web view whether it is the “Creating requests website” or the “Receiving pre-created requests website” and Continue to use the Websites as been described in chapters 2 and 3.



Figure 6-11

The user process cycle through the Universal Application is shown in the flowchart in Figure 6-12.



## 6.5 Conclusion

A Universal Windows Application is a Windows experience that is built upon the Universal Windows Platform (UWP) that Windows 10 makes it easier to develop apps for the UWP with just one API (Application Programming Interface) set, one app package, and one store to reach all Windows 10 devices. In our project the Universal Windows Application is used as an umbrella to hold the two websites that starts on the Raspberry-Pi in the IDM machine.

# Chapter 7

## Future Work

### 7.1 Introduction:

This chapter gives a scope description and overview of the Future Work that should be done in this project. It also gives a description of the requirements and list of ideas, devices and sensors could be added to the project to make it more powerful System.

This Future Work is setting up more than one devices or sensors that allows the end user be more comfortable and ease to use our project system and make the user experience more reliable.

### 7.2 Network System

A network can be used in this Project as we can connect all governmental departments together by setting up a router on each ( IDM ) and connect between these routers by wide area network ( WAN ).

#### 7.2.1 List of WAN types

- Cable
- Dial-Up
- DSL
- Leased Line
- Microwave
- Fiber

## 7.2.2 Types of Routers

There is lots of router types to create this Network Sytem for the Project.

- Cisco
- Juniper
- HP
- 3com

## 7.2.3 Advantage of Network System

If the user checks in any of the IDMs (whether to request or receive a pre-created request) the database will be edited in all of the IDMs not only locally across the IDM in front of him (All IDMs are Connected).

We also Use (SWITCH) if we have many machines (IDM) in same place beside each other to connect with them to the router to be connected with all IDMs everywhere.

Figure 7-1 shows the Basic W AN Network Architecture.

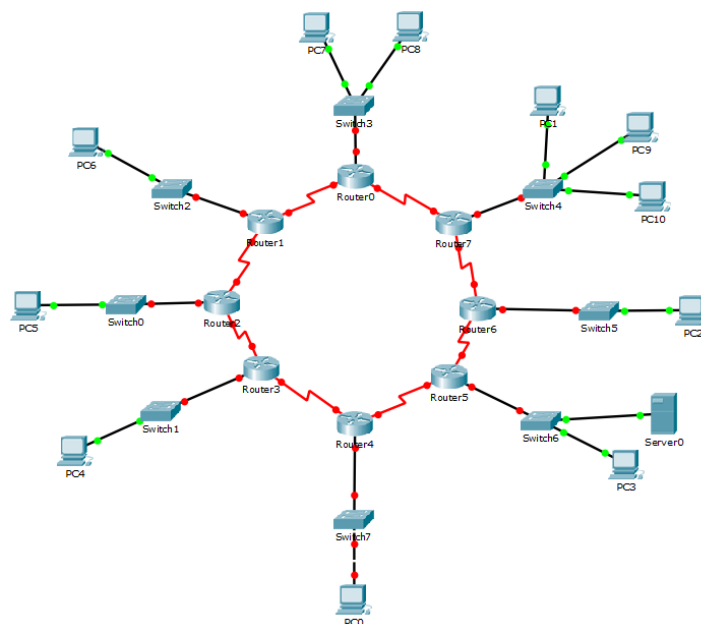


Figure 7-1

## **7.3 GSM Message**

The GSM Module will send the users an SMS message with the details of the document he requested, amount of fees must be paid and the possible ways to pay for using the service.

The users can pay for the service through the IDM which will be designed to have two Input tills one for paying in cash and the other for the credit card.

The users also can pay through "Fawry" Application that has been launched in cooperation with the Ministry of Planning that provides a number of governmental services demanded by citizens daily through smartphones.

"Fawry" Application speeds up services for citizens as it provides payment solutions for the three mobile operators in Egypt.

## **7.4 Printer Racks**

A Printer with several Racks can be used instead of our one rack printer, thus can a lot of governmental departments can be added , as each governmental department has its own kind of paper and specific paper size, so each kind of request or Documents will have its own rack.

When the user is ready to print his request the printer will choose that certain specific rack for the requested Document.

This System will be designed using embedded systems with very basic coding such if conditions and while loops.



## 7.5 Touch Screen

A touchscreen is an input device normally layered on the top of an information processing system. A user can give input or control the information processing system through simple or multi-touch gestures by touching the screen with a special stylus and/or one or more fingers as shown in Figure 7-2.



Figure 7-2

The Project will contain a Touch Screen instead of using LCD display, Keyboard & Mouse-Pad. The Touch Screen will replace all of these Device with only one Device for Simplicity and it will be more reliable.

### 7.5.1 Advantages of Touch Screen

The Touch Screen will provide the end users an opportunity to interact with the computer improving the human-computer interface and improving the user's experience as fumbling with the mouse and keyboard isn't always the easiest thing to do and touch screen technology can help, it saves time.

## 7.6 Finger Print

Fingerprint recognition or Fingerprint authentication refers to the automated method of verifying a match between two human fingerprints. Fingerprints are one of many forms of biometrics used to identify individuals and verify their identity.

The Project will contain a Finger print Sensor as shown in Figure 7-3, to take the finger print of the user while standing in front of the IDM Machine to make sure that this is the user who requested and not a person who took his password to print requests.



Figure 7-3

This sensor will require to edit the database and add a column for the finger print image for each User.

This technique is also achieved by using image Processing which will make correlation between the input finger print and the saved finger print in database, if the correlation exceed certain value it will allow access to the user for printing his own requests, if not it will deny him from entering the system to print any requests meaning that this is not the user for this data.

## 7.7 ID Scanner

ID Scanner device can be added to scans user ID for example the National ID, the Driving License ID or even the Student ID, stores it as an input in memory to Process the data read by the ID scanner.

The ID scanner also will capture data from Passports, Driver's Licenses, Barcodes, Magnetic Stripes, Business Cards, and Checks and electronically store the image of the full driver's license and/or cropped signature and face image.

The using if the ID Scanner will reduce data input errors and there will be no more manual data entry.

## 7.8 Data Input Devices

The website provides the service of extracting an official paper that has a constant copy in the local civil registry database such as the birth certificate, the website can be developed to provide more services which requires entering variable inputs or updating information in the local civil registry database such as the driving license and the passport.

For extracting such papers the Project's System needs a **Scanner** for scanning the documents needed and converts them into digital data. A **camera** for immediate passport photos can be added to the IDM. Also an **electronic pen** for the citizen's signature can be added.

# ***APPENDIX***

## **Creating Requests Website Code**

**This Code is to take the Citizen information to view it in the Requests Page**

```
public ActionResult PN()
{
    return Content(pn);
}

public ActionResult MSON()
{
    return Content(mson);
}

public ActionResult RQ()
{
    return Content(rq);
} // Rqm Qwmy
public ActionResult Name()
{
    return Content(nam);
}
```

```

    } // Name
    public ActionResult TM()

    {
        return Content(tm);

    } // tare5 MElad
    public ActionResult MM()

    {
        return Content(mm);

    } // M7l Melad
    public ActionResult MN()

    {
        return Content(mn);

    } // Mother Name
    public ActionResult RE()

    {
        return Content(re);

    } // Relegion
    public ActionResult Requestspage()

```

```

{
    MN();
    TM();
    MM();
    RE();
    RQ();
    Name();
    return View();
}

```

**This Code is to make the request for the selected document as each Government and each document have its own id**

```

public ActionResult msl7a1_1()
{
    msl7a_id = "1";
    mostand_id = "1";
    return RedirectToAction("Requestspage");
}

public ActionResult msl7a1_2()
{
    msl7a_id = "1";
    mostand_id = "2";
    return RedirectToAction("Requestspage");
}

```

```

public ActionResult msl7a2_3()
{
    msl7a_id = "2";
    mostand_id = "3";
    return RedirectToAction("Requestspage");
}

public ActionResult msl7a2_4()
{
    msl7a_id = "2";
    mostand_id = "4";
    return RedirectToAction("Requestspage");
}

```

**This Block of code is responsible for checking the citizen data entry for national id and his own password to check is it valid or not. If it is valid it redirect him to a page to select which government and which document he wants to select, if not it gives him an error message**

[HttpPost]

```

public ActionResult Create()
{
    nam = Request.Form["name"];
    string pass = Request.Form["password"];

    foreach (Citizen element in db.Citizens)
    {
        if ((element.National_ID == nam) && (element.Password == pass))
        {

```

```

    rq = element.National_ID;
    tm = element.BirthDate.ToShortDateString();
    mn = element.Mother_Name;
    mm = element.M7l_elmelad;
    nam = element.Citizen_Name;
    re = element.Relegion;

    return RedirectToAction("Msal7Page");

}
else if (("000000" == nam) && ("000000" == pass))
{
    Parameter = 1;
}
else if ((element.National_ID != nam) || (element.Password != pass))
{

    ermsg = "كلمة السر التي ادخلتها غير صحيحة /الرقم القومي ";
    erromsg();
}
}

```



**This block of code is to take the number of copies from the user and his phone number, and then all of his saved information in Database and make a request with it.**

[HttpPost]

```
public ActionResult save_requests()
```

```
{
```

```
    noc = Request.Form["numberofcopies"];
```

```
    pn = Request.Form["ThePhoneNumber"];
```

```
    Request request = new Request() {
```

```
        Citizen_ID = rq,
```

```
        Mostanad_ID = int.Parse(mostand_id.ToString()),
```

```
        Msl7a_ID = int.Parse(msl7a_id.ToString()),
```

```
        No_of_copies = int.Parse(noc.ToString()),
```

```
        Phone_Number = pn.ToString(),
```

```
    };
```

```
    if (ModelState.IsValid)
```

```
    {
```

```
        db.Requests.Add(request);
```

```
        db.SaveChanges();
```

```
        return RedirectToAction("QuestionPage");
```

```

    }
    return RedirectToAction("Requestspage");
}

```

## Receiving Pre-Created Requests Website Code

**This Website have the same exact code as in the Creating Requests Website except for this small Part.**

**This Block of code is for selecting an exact request from the pre-created requests and make toPrint cell = True which will make it ready to be printed, and then delete the to be printed request from the table in front of the User.**

```

public ActionResult Index()
{
    return View(db.Citizens.ToList());
}

public ActionResult lol(int id)
{
    Request r = db.Requests.Find(id);
    r.toPrint = true;
    db.SaveChanges();
    return Redirect("http://localhost:57104/Citizens/Details/" + r.Citizen_ID);
}

public ActionResult lol(int id)

```

```

{
    Request r = db.Requests.Find(id);

    r.toPrint = true;

    db.SaveChanges();

    return Redirect("http://localhost:57104/Home/Details/" + rq);
}

```

## Windows For Application (Printing Part)

**This Part is Responsible for Finding all the requests which is ready to be printed ( toprint = True )**

```

public partial class Form1 : Form
{
    Fe5dmtak.prn.WebsiteDatabaseProjectDataSet.RequestsRow curentrow;
    WebsiteDatabaseProjectEntities _context = new WebsiteDatabaseProjectEntities();

    public Form1()
    {
        InitializeComponent();
    }

    private void requestsBindingNavigatorSaveItem_Click(object sender, EventArgs e)
    {
        this.Validate();
        this.requestsBindingSource.EndEdit();
        this.tableAdapterManager.UpdateAll(this.websiteDatabaseProjectDataSet);
    }
}

```

```

    }

    private void Form1_Load(object sender, EventArgs e)
    {
        // TODO: This line of code loads data into the 'websiteDatabaseProjectDataSet.Requests'
        table. You can move, or remove it, as needed.

        this.requestsTableAdapter.Fill(this.websiteDatabaseProjectDataSet.Requests);

    }

    private void timer1_Tick(object sender, EventArgs e)
    {
        timer1.Enabled = false;

        this.websiteDatabaseProjectDataSet.Requests.Clear();

        this.requestsTableAdapter.FillByPrint(this.websiteDatabaseProjectDataSet.Requests);

        foreach (Fe5dmtak.prn.WebsiteDatabaseProjectDataSet.RequestsRow r in
this.websiteDatabaseProjectDataSet.Requests)
        {
            r.Printed_or_not = true;

            curentrow = r;

            //code elprinter hna search for it

            PrintDocument pd = new PrintDocument();

            pd.PrintPage += PrintPage;

            //printPreviewControl1.Document = pd;

            //printPreviewDialog1.Document = pd;

            pd.PrinterSettings.Copies = short.Parse(r.No_of_copies.ToString());

```

```

        pd.Print();
    }

    this.requestsTableAdapter.Update(this.websiteDatabaseProjectDataSet.Requests);

    timer1.Enabled = true;
}

private void PrintPage(object o, PrintPageEventArgs e)
{
    int reqID = curentrow.Request_ID;

    Request req = _context.Requests.Find(reqID);

```

**This Part of code is taking all the requests ready to be printed and Print it with the exact data of the user which is ordered and with the exact number of copies the user desired.**

```

        StringBuilder sb = new StringBuilder();

        if (req.Msl7a_ID == 1 && req.Mostanad_ID == 1)
        {
            sb.AppendLine("مصلحه الاحوال المدنيه");
            sb.AppendLine("");
            sb.AppendLine("شهاده ميلاد");
            sb.AppendLine("");
            sb.AppendLine("");
            sb.AppendLine("");
            sb.AppendLine("");

```

```

}

if (req.Msl7a_ID == 1 && req.Mostanad_ID == 2)
{
    sb.AppendLine("مصلحه الاحوال المدنيه");
    sb.AppendLine("");
    sb.AppendLine("صوره بطاقه");
    sb.AppendLine("");
    sb.AppendLine("");
    sb.AppendLine("");
    sb.AppendLine("");

}

if (req.Msl7a_ID == 2 && req.Mostanad_ID == 3)
{
    sb.AppendLine("المروور");
    sb.AppendLine("");
    sb.AppendLine("جند 2 استماره");
    sb.AppendLine("");
    sb.AppendLine("");
    sb.AppendLine("");
    sb.AppendLine("");

}

if (req.Msl7a_ID == 2 && req.Mostanad_ID == 4)
{
    sb.AppendLine("المروور");
    sb.AppendLine("");

```

```

sb.AppendLine("جند 7 استثماره");
sb.AppendLine("");
sb.AppendLine("");
sb.AppendLine("");
sb.AppendLine("");

}
else
{

};

sb.AppendLine(req.Citizen.Citizen_Name.ToString() + " : اسم المواطن");
sb.AppendLine("");
sb.AppendLine(req.Citizen_ID.ToString() + " : الرقم القومي");
sb.AppendLine("");
sb.AppendLine(req.Citizen.Mother_Name.ToString() + " : اسم الام");
sb.AppendLine("");
sb.AppendLine(req.Citizen.Relegion.ToString() + " : الديانه");
sb.AppendLine("");
sb.AppendLine(req.Citizen.Nationality.ToString() + " : الجنسيه");
sb.AppendLine("");
sb.AppendLine(req.Citizen.BirthDate.ToShortDateString() + " : تاريخ الميلاد");
sb.AppendLine("");
sb.AppendLine(req.Citizen.M7l_elmelad.ToString() + " : محل الميلاد");

```

```

e.Graphics.DrawString(sb.ToString(),

                                new System.Drawing.Font("Arial", 18),
                                Brushes.Black,
                                460,
                                80,

                                new StringFormat());
    }

}
}

```

## Windows For Application (SMS Part)

**This Part is Responsible for sending SMS to the Users through GSM Module.**

```

// Send an SMS message
SmsSubmitPdu pdu;
bool alert = chkAlert.Checked;
bool unicode = chkUnicode.Checked;

if (!alert && !unicode)
{
    // The straightforward version
    pdu = new SmsSubmitPdu(txt_message.Text,
txt_destination_numbers.Text, ""); // "" indicate SMSC No
}
else
{
    // The extended version with dcs
    byte dcs;
    if (!alert && unicode)

```



```

        dcs = DataCodingScheme.NoClass_16Bit;
    else if (alert && !unicode)
        dcs = DataCodingScheme.Class0_7Bit;
    else if (alert && unicode)
        dcs = DataCodingScheme.Class0_16Bit;
    else
        dcs = DataCodingScheme.NoClass_7Bit; // should
never occur here

```

```

        pdu = new SmsSubmitPdu(txt_message.Text,
txt_destination_numbers.Text, "", dcs);
    }

    // Send the same message multiple times if this is set
    int times = chkMultipleTimes.Checked ?
int.Parse(txtSendTimes.Text) : 1;

    // Send the message the specified number of times
    for (int i=0;i<times;i++)
    {
        CommSetting.comm.SendMessage(pdu);
        Output("Message {0} of {1} sent.", i+1, times);
        Output("");
    }
}
catch(Exception ex)
{
    MessageBox.Show(ex.Message);
}

Cursor.Current = Cursors.Default;
}

```

```

private void Output(string text)
{
    if (this.txtOutput.InvokeRequired)
    {
        SetTextCallback stc = new SetTextCallback(Output);
        this.Invoke(stc, new object[] { text });
    }
    else
    {
        txtOutput.AppendText(text);
        txtOutput.AppendText("\r\n");
    }
}

```

```

        }
    }

    private void Send_Load(object sender, System.EventArgs e)
    {
        chkMultipleTimes.Checked=true;
    }

    private void Output(string text, params object[] args)
    {
        string msg = string.Format(text, args);
        Output(msg);
    }

private void tmrSMS_Tick(object sender, EventArgs e)
{
    tmrSMS.Enabled = false;
    SMSEntities context = new SMSEntities();
    List<Request> RZ = (from Request r in context.Requests
    //intialaize a variable rz that reads requests r from request table
        where r.Msg_Sent_Or_Not != true
    //the condition is about the messages that are not sent yet
        select r).ToList();
    //select the the messages that are not sent yet from the list of requests
    foreach (Request R in RZ)
    //read all requests r in variable rz
    {
        txt_destination_numbers.Text = R.Phone_Number;
        txt_message.Text = "Dear " + R.Citizen.Citizen_Name + "\r\n";
        txt_message.Text += "You request " + "from " + R.Msale7.Msl7a_Name +
            "\r\n";
        txt_message.Text += "your password is " + R.Citizen.Password;
        R.Msg_Sent_Or_Not = true;
        btnSendMessage.PerformClick();
    //the message format which is automatically added from request table in database
    }
    context.SaveChanges();

    tmrSMS.Enabled = true;
}
}
}

```

## Universal Windows Application code

**This Part is to View both Websites in the WebViews created in the Universal Windows Application**

```
public WebViewShowRequests()
{
    this.InitializeComponent();
    Web2.Visibility = Visibility.Visible;
    Uri u = new Uri("http://192.168.1.105");
    Web2.Navigate(u);
}

public WebViewFillRequest()
{
    this.InitializeComponent();
    Web1.Visibility = Visibility.Visible;
    Uri x = new Uri("http://192.168.1.105:8080");
    Web1.Navigate(x);
}
```

**This Part is responsible for Navigating from Page to Page through buttons in the whole Application**

```
private void Button_Click(object sender, RoutedEventArgs e)
{
    Frame.Navigate(typeof(MainPage));
}
```

```
private void button_Click(object sender, RoutedEventArgs e)
```

```
{  
    Frame.Navigate(typeof(ChoosePage));  
}
```

```
private void image_Tapped_1(object sender, TappedRoutedEventArgs e)
```

```
{  
    Frame.Navigate(typeof(AboutPage));  
}
```

```
private void button_Click(object sender, RoutedEventArgs e)
```

```
{  
    Frame.Navigate(typeof(WebViewFillRequest));  
  
}
```

```
private void button_Copy_Click(object sender, RoutedEventArgs e)
```

```
{  
    Frame.Navigate(typeof(WebViewShowRequests));  
}
```

# ***REFERENCES***

- [1] Bruce Johnson, “Professional Visual Studio 2015”, 2015
- [2] Matthew MacDonald , “ASP.Net: The Complete Reference”, 2002
- [3] Adam Freeman “Pro ASP.NET MVC 5 “, 2013
- [4] Michel Mouly and Marie-Bernadette Pautet “The GSM System for Mobile Communications”, 1992
- [5] <https://opensource.com/resources/what-raspberry-pi>
- [6] <https://www.raspberrypi.org/products/raspberry-pi-2-model-b/>
- [7] <https://msdn.microsoft.com/en-us/windows/uwp/layout/design-and-ui-intro>
- [8] <https://msdn.microsoft.com/en-us/windows/uwp/get-started/whats-a-uwp>