Storage Management Application

ZEHUA YU

B.Sc.(Honour) inSoftware Development

Number of Words:

April 15,2018

Final Year Project

Supervisor: Kevin O'Brien

CONTENT

Chapter 1

1. Abstract

In the project, I exert the UWP, Xamarin form and Azure Web technology, majority code is based on C#. The SQLite and Azure Cloud are used for database technology.

With an increasing number of people go to high speed the module of life. order online has been a kind of popular pattern, in order to adapt the issue that The fast pace of consumption patterns, more and more shops need a new way to manage orders and details of products. For example, some restaurants have co-operated with a takeaway website, like Just-Eat and so on. Therefore, my project can receive the order and manage the information in storage. Firstly, I code the mobile phone app, which is a client, which can show all of the menus on this app. By the Azure Web service [1], this app can post the order to cloud databases. My main Process, which is based on C# in Visual Studio, also receives the order from the cloud database. In another hand, the Main process owns another function to let users easily control all business details. For users, the user can also directly on the phone app to read all the valuable information. To choose their favourite goods. Technology-based on Xamarin Form 3.0[2], which is a kind of crossing-platform technology, the APP can be run on IOS, Windows and Android system. So coder can easily update the application.

Github URL: https://github.com/Zehuayu/Storage_Management_System

Chapter 2

Introduction

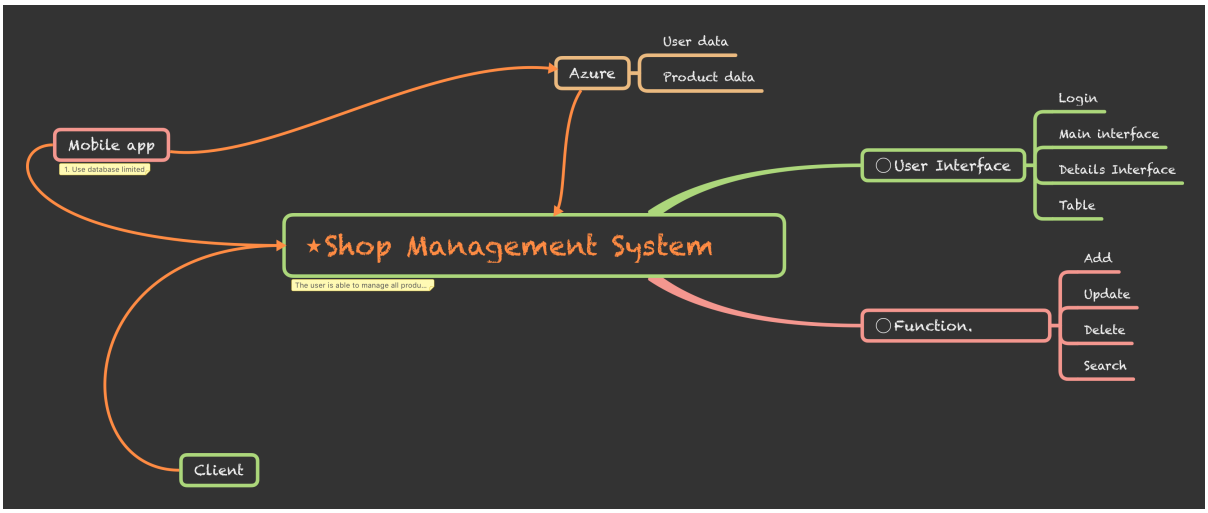
2.1 Idea

This idea is from a particular time when I did my part-time job, we need to manage all products in storage by my memory and notebook, but each stuff has the different pattern to note all list of products. So every stuff has to waste time in order to check lots of details. Depend on my mind, a process who has a uniformly operational command can greatly improve the work efficiency. At the same time, our shop also owns an app to let customers could easily buy something online. the shop can decrease the cost of communication because stuff has to take a call to communicate with customers. Fortunately, our teacher taught the cloud technology in this semester, I would like to exert the cloud technology and UWP to code a management system for the final project.

2.2 The functional Design

Because software must have a certain interaction ability, so at first I want to use web technology transfer data to each other between the two programs, but to build a cloud server. For an individual project work is great, because I must finish the client and the main program two large apps.Until beginning to understand Azure cloud technology, which could be able to connect two programs to the cloud to share data. This can be done in certain aspects the process interaction. In order to easily connect to Azure Cloud, the Xamarin Studio[4] becomes the good choice in the part of client. Because Xamarin Studio has majority tools to develop the mobile app and convenient to be interaction between the Azure and mobile app. while, the main process is base on the Visual Studio 2017. In the framework of UWP[3], this process can run in different device include windows tablet ,computer and others about Win10 OS devices. In the one word, them from the same company, which name is Microsoft, So in the suitability between the software and software can achieve the best, in some need to solve the problem of compatibility repeatedly I can spend the least amount of time to solve it.And all in the same language, the function I also can be used repeatedly

2.3 The User Interface Design



The UI(User Interface) mainly follow the first presentation, include the Login and change data function. while there are the list show and typing page to input data to Azure. Two process are mainly attention to function, so I try to make the page look simple in terms of the UI.And highlight features of integrity.

2.4 Which Shop Is Good for This Process

Because this program module has order service online, so the program is suitable for the shop who has a delivery service and collection service, the stuff can stock up in advance according to the point of network goods situation and concentrated delivery processing, because the main process has login interface, so user could get a guarantee of safety. So most of the stores and some restaurants who have a delivery ability can use this program management order information and management of data warehouse.And the low-cost of service in the Azure and also can be in network services as far as possible to reduce costs of the shop and restaurant to achieve maximum profit.

Chapter 3

Methodology

3.1How to Develop My Project

The whole plan of my project owns the process of development

1. Idea: Making sure what application I would like to do, and making the outline depend on this idea
2. Choose the Technology: this step is that which environment can adapt mine whole system of app
3. Making the diagram who shows some functionality details
4. User interface development
5. Background process development
6. Testing and fix up bugs founded
7. Write the document and make the introduction
8. Package and export and re-testing.
9. Submit this project

3.1.1 Development Method

The VS 2017 and Xamarin Studio[4] is my main tools to development, I was coding the management application on VS 2017 and mobile phone app is base on the Xamarin Studio. Firstly, I made a brief survey on shop because I worked in this shop, what is important function for shop owner and what is good way for customer.Depend on this result, I analysed the all of requirement then made the outline about the Functionality and the requirement of technology. After that, I made a plan in the GitHub[5]. Let oneself program is sequentially completed, with supervisor to discuss the details of the plan and complete some function according to the requirement of the he is put forward.

3.1.2 Testing of Process

The process the test has three step:

1. Finishing the the UI design of computer application, let me could enter any functionality page by the button in the interface. It show that interface is available.
2. Testing the main functionality, input and output can be run.
3. Computer Process whether can be interact between Azure and computer process
4. Testing all of functionality in the mobile app
5. Testing the interaction both two process (mobile app and computer app).

3.2.1 Hard Environment

Laptop configuration: CPU I5 2.2GHz

RAM 8GB

Storage 256GB

Operation System: MacOS and Windows

3.2.2 Software Environment

Development Tool: Visual Studio (C#)

Xamarin Studio(C#)

Running Environment: Visual Box

Genymotion

Visual Studio

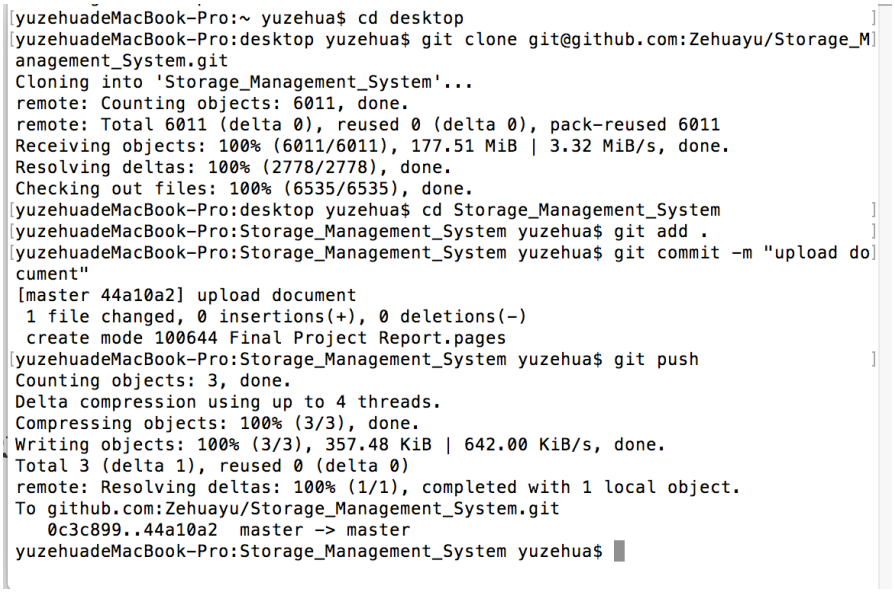
Database: Sqlite and Azure Cloud Database

3.3 Github and Code Storage

When I finished this phase code, I will put all the code is uploaded to Github website.I will write Commit command to mark what is code doing. Then write some document to describe its process.

Github: Github Is a web hosting service by git version control. It is mainly used for computer code. It provides all the features of distributed version control and Git source code management (SCM) as well as adding its own features. It provides several collaborative features such as access control and bug tracking, feature requests, task management, and wikis for each project. GitHub provides

private warehouses and program free accounts are commonly used open source software projects. As of April 2017, GitHub reported nearly 20 million users and 20 million libraries, making it becomes largest source code in all of world.



Chapter 4

4 Technology Explanation

4.1 UWP

UWP(Universal Windows Platform) is an integral part of Microsoft's Windows operating system Details of the type of application programming interface included in the window, Window 10 Mobile, Windows 10 Internet of Things, Xbox, Windows Mixed reality Replace Windows Runtime Support Status Current Related Components Windows Store. The purpose of this software platform is to help develop generic applications running on Windows, Windows 10 Mobile, Xbox, Holographic Lenses do not need to be rewritten. It supports Windows application development using C++, c#, and VB. Net, XAML. Implementation in C++ API, support for C++, VB. Net, c#, f # and JavaScript. Designed as an extension to the Windows Runtime Platform for the first time in Windows Server 2012 and Windows 8, UWP allows developers to create applications that may run on multiple types of devices.

4.1.1 XAML

XAML is an extensible application markup language or XAML (pronounced "zammel"), an XML-based markup language developed by Microsoft. The language behind XAML is the visual representation of the application you are developing, Microsoft Expression Blend, just like the visual presentation Web page behind the HTML language. Creating an application in Expression Blend means writing XAML code by hand or visually in Expression Blend's design view.

In general, my UI design of computer application is mainly depend on the XAML language. Including the Main page, Goods page, Menu Page, Storage Page and Add page.
XAML Example

```
<Grid Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    <Button Content="ADD" Margin="49,371,0,213" FontSize="30" Height="56" Width="80" Click="SAP_Add"/>
```

```
    <TextBlock Text="Product Name" Margin="181,135,0,480" TextAlignment="Center" FontWeight="Bold"
HorizontalAlignment="Left" Width="114" />
    <TextBlock Text="Amount" Margin="181,185,0,430" TextAlignment="Center" FontWeight="Bold" HorizontalAlignment="Left"
Width="114" />

    <TextBox Name ="NameInput" Text="" BorderThickness="1"  HorizontalAlignment="Left"  Margin="18,129,0,479" Width="120"  /
>
    <TextBox Name ="AmountInput" Text="" BorderThickness="1" HorizontalAlignment="Left" Margin="18,176,0,432" Width="120"/>

  </Grid>
</Page>
```

This example shows how to code the text title and inputting box, and how to set the property of thing, such as weight, position and so on.

4.1.1.1 ListView

ListView and GridView both derive from the ListViewBase class, so they have the same functionality, but display data differently. In this article, when we talk about ListView, the info applies to both the ListView and GridView controls unless otherwise specified. We may refer to classes like ListView or ListViewItem, but the “List” prefix can be replaced with “Grid” for the corresponding grid equivalent (GridView or GridViewItem).

The ListView displays data stacked vertically in a single column. It's often used to show an ordered list of items, such as a list of emails or search results.
The ListView control is useful for displaying data in any repeating structure, similar to the DataList and Repeater controls. Unlike those controls, the ListView control supports edit, insert, and delete operations as well as sorting and paging. The paging functionality is provided for ListView by the new DataPager control.

4.1.1.2 GridView

A recurring task in software development is to display tabular data. ASP.NET provides a number of tools for showing tabular data in a grid, including the GridView control. With the GridView control, you can display, edit, and delete data from many different kinds of data sources, including databases, XML files, and business objects that expose data.

The GridView presents a collection of items in rows and columns that can scroll vertically. Data is stacked horizontally until it fills the columns, then continues

with the next row. It's often used when you need to show a rich visualisation of each item that takes more space, such as a photo gallery.

4.1.1.3 Layouts

The Xaml layouts system provides automatic sizing, layout panels, visual states, and even separate UI definitions to create a responsive UI. With a responsive layout, you can make your app look great on screen with different app windows size, resolutions, pixel densities, and orientations. You can also use XAML to reposition, resize, reflow, show/hide, replace, or re-architect your app's UI, as discussed in Responsive design techniques. Here, we discuss how to implement responsive layouts with XAML.

Layout properties has important function that Layout properties control the size and position of an element. To create a fluid layout, use automatic or proportional sizing for elements, and allow layout panels position their children as needed.

The Height and Width properties specify the size of an element. You can use fixed values measured in effective pixels, or you can use auto or proportional sizing.

Auto sizing resizes UI elements to fit their content or parent container. You can also use auto sizing with the rows and columns of a grid. To use auto sizing, set the Height and/or Width of UI elements to auto.
Example

```
<Grid>
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="Auto"/>
        <ColumnDefinition/>
        <ColumnDefinition Width="44"/>
        <ColumnDefinition Width="2*"/>
    </Grid.ColumnDefinitions>
    <TextBlock Text="Column 1 sizes to its conent." FontSize="24"/>
</Grid>
```

To position visual objects, you must put them in a panel or other container object. The XAML framework provides various panel classes, such as Canvas, Grid, RelativePanel and StackPanel which serve as containers and enable you to position and arrange the UI elements within them.

4.1.1.4 Templates

This section contains item templates that you can use with a ListView control. Use these templates to get the look of common app types.
To demonstrate data binding, these templates bind ListVitemItem to the example Recording class from the data binding overview data.
Currently, when a Data Template contains multiple controls (e.g. more than a single TextBlock), the default accessible name for screen readers comes from .ToString() on the item. As a convenience you can instead set the AutomationProperties.Name on the root element of the DataTemplate. For more on accessibility, see Accessibility overview.
Example

```
<ListView ItemsSource="{x:Bind ViewModel.Recordings}">
    <ListView.ItemTemplate>
        <DataTemplate x:Name="SingleLineDataTemplate"
x:DataType="local:Recording">
            <StackPanel Orientation="Horizontal" Height="44"
Padding="12" AutomationProperties.Name="{x:Bind CompositionName}">
                <Image Source="Placeholder.png" Height="16" Width="16"
VerticalAlignment="Center"/>
                <TextBlock Text="{x:Bind CompositionName}"
VerticalAlignment="Center" Style="{ThemeResource BaseTextBlockStyle}"
Foreground="{ThemeResource SystemControlPageTextBaseHighBrush}"
Margin="12,0,0,0"/>
            </StackPanel>
        </DataTemplate>
    </ListView.ItemTemplate>
</ListView>
```

4.1.2 Data Bind

Specifies the services, protocols, and devices to which this service is associated with a binding. The system builds the bind from the values in this entry and in the Bind entries of the associated protocols and devices.

Data binding is a way for your app's UI to display data, and optionally to stay in sync with that data. Data binding allows you to separate the concern of data

from the concern of UI, and that results in a simpler conceptual model as well as better readability, testability, and maintainability of your app.

You can use data binding to simply display values from a data source when the UI is first shown, but not to respond to changes in those values. This is called one-time binding, and it works well for data whose values don't change during run-time. Additionally, you can choose to "observe" the values and to update the UI when they change. This is called one-way binding, and it works well for read-only data. Ultimately, you can choose to both observe and update, so that changes that the user makes to values in the UI are automatically pushed back into the data source. This is called two-way binding, and it works well for read-write data.

1.You could use one-time binding to bind an Image to the current user's photo.

2. You could use one-way binding to bind a ListView to a collection of real-time news articles grouped by newspaper section.

3. You could use two-way binding to bind a TextBox to a customer's name in a form

There are two kinds of binding, and they're both typically declared in UI markup. You can choose to use either the {x:Bind} markup extension or the {Binding} markup extension. And you can even use a mixture of the two in the same app—even on the same UI element. {x:Bind} is new for Windows 10 and it has better performance. All the details described in this topic apply to both kinds of binding unless we explicitly say otherwise.

Example

Page code:

```
<ListView x:Name="MylistView"  BorderThickness="1"  ItemsSource="{x:Bind Recordings}" Margin="18,96,90,233">
        <ListView.ItemTemplate>
          <DataTemplate  x:DataType="data:GoodsInfo" >

            <StackPanel Orientation="Horizontal" BorderThickness="1" Height="80" Width="auto">
              <TextBlock Text="{Binding Name }" Width="70" TextAlignment="Center" ></TextBlock>
```

Background code:

```
public ObservableCollection<Recording> recordings = new ObservableCollection<Recording>();
    public ObservableCollection<Recording> Recordings { get { return this.recordings; } }

public class Recording
    {
```

```
    public int Id { get; set; }
    public string Name { get; set; }
}
```

4.1.3 SQLite

SQLite[6] is a popular choice as embedded database software for local/client storage in application software such as web browsers. It is arguably the most widely deployed database engine, as it is used today by several widespread browsers, operating systems, and embedded systems (such as mobile phones), among others. SQLite has bindings to many programming languages. Unlike client–server database management systems, the SQLite engine has no standalone processes with which the application program communicates. Instead, the SQLite library is linked in and thus becomes an integral part of the application program. The library can also be called dynamically. The application program uses SQLite's functionality through simple function calls, which reduce latency in database access: function calls within a single process are more efficient than inter-process communication. SQLite stores the entire database (definitions, tables, indices, and the data itself) as a single cross-platform file on a host machine. It implements this simple design by locking the entire database file during writing. SQLite read operations can be multitasked, though writes can only be performed sequentially.

Due to the server-less design, SQLite applications require less configuration than client-server databases. SQLite is called *zero-conf* because it does not require service management (such as startup scripts) or access control based on GRANT and passwords. Access control is handled by means of file system permissions given to the database file itself. Databases in client-server systems use file system permissions which give access to the database files only to the daemon process.

Another implication of the serverless design is that several processes may not be able to write to the database file. In server-based databases, several writers will all connect to the same daemon, which is able to handle its locks internally. SQLite on the other hand has to rely on file-system locks. It has less knowledge of the other processes that are accessing the database at the same time. Therefore, SQLite is not the preferred choice for write-intensive deployments. [8] However, for simple queries with little concurrency, SQLite performance profits from avoiding the overhead of passing its data to another process. SQLite uses PostgreSQL as a reference platform. “What would PostgreSQL do” is used to make sense of the SQL standard. One major deviation is that, with the exception of primary keys, SQLite does not enforce type checking; the type of a

value is dynamic and not strictly constrained by the schema (although the schema will trigger a conversion when storing, if such a conversion is potentially reversible). SQLite strives to follow Postel's Rule.
So, All login data and local data, such as goods information, is stored in SQlite database. It is a good tool in the process of development of UWP, its advantage is that sqlite is really small tool, never negatively impact on the stability and performance of application. how to add SQLite database into UWP application?

Step 1: Install the SQLite for Universal App Platform.

Step 2: Add this tool.

Step 3:Install the SQLite PCL.

Step 4:Coding the SQLite Class.

Step 5:Exert the function such as add, delete, update and so on.

4.1.4 Microsoft Azure

Microsoft Azure is a cloud computing service created by Microsoft for building, testing, deploying, and managing applications and services through a global network of Microsoft-managed data centers. It provides software as a service (SaaS), platform as a service (PaaS) and infrastructure as a service (IaaS) and supports many different programming languages, tools and frameworks, including both Microsoft-specific and third-party software and systems.
Microsoft Azure uses a specialised operating system, called Microsoft Azure, to run its "fabric layer":] a cluster hosted at Microsoft's data centres that manages computing and storage resources of the computers and provisions the resources (or a subset of them) to applications running on top of Microsoft Azure.
Microsoft Azure has been described as a "cloud layer" on top of a number of Windows Server systems, which use Windows Server 2008 and a customised version of Hyper-V, known as the Microsoft Azure Hypervisor to provide virtualisation of services.
Scaling and reliability are controlled by the Microsoft Azure Fabric Controller, which ensures the services and environment do not fail if one or more of the servers fails within the Microsoft data centre, and which also provides the management of the user's Web application such as memory allocation and load balancing.

Azure provides an API built on REST, HTTP, and XML that allows a developer to interact with the services provided by Microsoft Azure. Microsoft also provides a client-side managed class library that encapsulates the functions of interacting with the services. It also integrates with Microsoft Visual Studio, Git, and Eclipse.
In addition to interacting with services via API, users can manage Azure services using the Web-based Azure Portal, which reached General Availability in December 2015. The portal allows users to browse active resources, modify settings, launch new resources, and view basic monitoring data from active virtual machines and services. More advanced Azure management services are available.

4.2 Xamarin

Xamarin is a Microsoft-owned San Francisco, California-based software company founded in May 2011 by the engineers that created Mono, Mono for Android and MonoTouch, which are cross-platform implementations of the Common Language Infrastructure (CLI) and Common Language Specifications (often called Microsoft .NET).
With a C#-shared codebase, developers can use Xamarin tools to write native Android, iOS, and Windows apps with native user interfaces and share code across multiple platforms,
including Windows and macOS. According to Xamarin, over 1.4 million developers were using Xamarin's products in 120 countries around the world as of April 2017.
On February 24, 2016, Microsoft announced it had signed a definitive agreement to acquire Xamarin.

In 1999 Miguel de Icaza and Nat Friedman launched what would eventually be known as Ximian to support and develop software for de Icaza's nascent GNOME project. After Microsoft first announced their .NET Framework in June 2000, de Icaza began investigating whether a Linux version was feasible.] The Mono open source project was launched on July 19, 2001.
Ximian was bought by Novell on August 4, 2003, which was then acquired by Attach-mate in April 2011.
After the acquisition, Attach-mate announced hundreds of layoffs for the Novell workforce, including Mono developers, putting the future of Mono in question.

In December 2012, Xamarin released Xamarin.Mac,[19] a plugin for the existing MonoDevelop Integrated development environment (IDE), which allows developers to build C#-based applications for the Apple OS X operating system and package them for publishing via the Apple App Store.

In February 2013, Xamarin announced the release of Xamarin 2.0.[20] The release included two main components: **Xamarin Studio**, a re-branding of its open-source IDE Mono-develop and integration with Visual Studio, Microsoft's IDE for the .NET Framework, allowing Visual Studio to be used for creating applications for Android and iOS, as well as for Windows.

4.2.1 Xamarin Platform

Xamarin 2.0 was released in February 2013 Xamarin.Android and Xamarin.iOS that make it possible to do native Android, iOS and Windows development in C#, with either Visual Studio or Xamarin Studio. Developers re-use their existing C# code, and share significant code across device platforms. The product was used to make apps for several well-known companies including 3M, AT&T, HP, and Target. Xamarin integrates with Visual Studio, Microsoft's IDE for the .NET Framework, extending Visual Studio for Android and iOS development. Xamarin also released a component store to integrate backend systems, 3rd party libraries, cloud services and UI controls directly into mobile apps

4.2.2 Xamarin Form

Xamarin.Forms is a framework that allows developers to rapidly create cross platform user interfaces. It provides its own abstraction for the user interface that will be rendered using native controls on iOS, Android, Windows, or Windows Phone. This means that applications can share a large portion of their user interface code and still retain the native look and feel of the target platform. Xamarin.Forms allows for rapid prototyping of applications that can evolve over time to complex applications. Because Xamarin.Forms applications are native applications, they don't have the limitations of other toolkits such as browser sandboxing, limited APIs, or poor performance. Applications written using Xamarin.Forms are able to utilise any of the API’s or features of the underlying platform, such as (but not limited to) CoreMotion, PassKit, and StoreKit on iOS; NFC and Google Play Services on Android; and Tiles on Windows. In addition, it's possible to create applications that will have parts of their user interface created with Xamarin.Forms while other parts are created using the native UI toolkit.

Xamarin.Forms applications are architected in the same way as traditional cross-platform applications. The most common approach is to use Portable Libraries or Shared Projects to house the shared code, and create platform specific applications that will consume the shared code.

4.2.3 Multi-platform Development

Multi-platform[7] Development In computing, cross-platform software (also multi-platform software or platform-independent software) is computer software that is implemented on multiple computing platforms. Cross-platform software may be divided into two types; one requires individual building or compilation for each platform that it supports, and the other one can be directly run on any platform without special preparation, e.g., software written in an interpreted language or pre-compiled portable bytecode for which the interpreters or run-time packages are common or standard components of all platforms.

For example, a cross-platform application may run on Microsoft Windows on the x86 architecture, Linux on the x86 architecture and macOS on either the PowerPC or x86-based Apple Macintosh systems. Cross-platform programs may run on as many as all existing platforms, or on as few as two platforms. Cross-platform frameworks (such as Qt, Xamarin, Phonegap, or Ionic) exist to aid cross-platform development.

In the one word, the Xamarin is good choice because the developing tool has become a part of system in the Microsoft. Because of this reason, Xamarin would be gave more and more API and technology supported from Microsoft. Xamarin will be a popular development environment in the future because of good eco-system.

4.2.4 Xamarin Studio

At the time of its release in February 2013, Xamarin Studio was a standalone IDE for mobile app development on Windows and macOS as part of Xamarin 2.0 based on the open source project MonoDevelop. In addition to a debugger, Xamarin Studio includes code completion in C#, an Android UI builder for creating user interfaces without XML, and integration with Xcode Interface Builder for iOS app design.
On Windows Xamarin Studio is now deprecated and was replaced with Xamarin for Visual Studio. On macOS Xamarin Studio is still in development, but was rebranded 2016 as Visual Studio for Mac.[

4.2.5 Xamarin for Visual Studio

Xamarin claims to be the only IDE that allows for native Android, iOS and Windows app development within Microsoft Visual Studio. Xamarin supplies add-ins to Microsoft Visual Studio that allows developers to build Android, iOS, and Windows apps within the IDE using code completion and IntelliSense. Xamarin for Visual Studio also has extensions within Microsoft Visual Studio that provide support for the building, deploying, and debugging of apps on a simulator or a device. In late 2013, Xamarin and Microsoft announced a partnership that included further technical integration and customer programs to make it possible for their joint developer bases to build for all mobile platforms. In addition, Xamarin now includes support for Microsoft Portable Class Libraries and most C# 5.0 features such as async/await. CEO and co-founder of Xamarin, Nat Friedman, announced the alliance at the launch of Visual Studio 2013 in New York.
On March 31, 2016 Microsoft announced that they were merging all of Xamarin's software with every version of Microsoft Visual Studio including Visual Studio Community, and this added various Xamarin features to come pre-installed in Visual Studio such as an iOS emulator.

4.2.6 Data & Cloud Service

To function correctly, many mobile applications are dependent on the cloud, and so integrating web services into mobile applications is a common scenario. The Xamarin platform supports consuming different web service technologies, and

includes in-built and third-party support for consuming RESTful, ASMX, and Windows Communication Foundation (WCF) services.

Chapter 5

5 System Design

5.1 Language Choice

Before the development of project, the choice of language is extremely important step. Generally, every computer language owns the self-advantage. So, the applicable computer language can positively support the coder to develop a perfect process. Because of this reason, the choice of language normally becomes a part of the process of development.

5.1.1 Language choice

Before the choosing the language, majority coders have to understand what functionality is needed in the process. Then, I started to pick the coding language from Java, C sharp, JavaScript and so on. Firstly, let us to review the Java, I think the Java is perfect language in too many environments, it has lots of interface and package, in the another reason, Java language is the most popular language, because of behind reason, so internet, such as YouTube and so on, owns all kinds of resource and tutorial. But, in the desktop application aspect, Java is more difficult compare to C sharp. At the moment, C sharp is a kind of popular computer language also, it has lots of package and technology

supported. Visual Studio is also a good developing tools for me, the end of picking process, C sharp was mine developing language.

5.1.2 Object Oriented

Object-oriented programming is a programming paradigm based on the concept of "objects", which may contain data, in the form of fields, often known as *attributes;* and code, in the form of procedures, often known as *methods.* A feature of objects is that an object's procedures can access and often modify the data fields of the object with which they are associated (objects have a notion of "this" or "self"). In OOP, computer programs are designed by making them out of objects that interact with one another. There is significant diversity of OOP languages, but the most popular ones are class-based, meaning that objects are instances of classes, which typically also determine their type.

Many of the most widely used programming languages (such as C++, Object Pascal, Java, Python etc.) are multi-paradigm programming languages that support object-oriented programming to a greater or lesser degree, typically in combination with imperative, procedural programming. Significant object-oriented languages include Java, C++, C#, Python, PHP, Ruby, Perl, Object Pascal, Objective-C, Dart, Swift, Scala, Common Lisp, and Smalltalk.

the advantage of Object Oriented

- ***Code Reuse and Recycling***: Objects created for Object Oriented Programs can easily be reused in other programs.

- ***Encapsulation (part 1)***: Once an Object is created, knowledge of its implementation is not necessary for its use. In older programs, coders needed understand the details of a piece of code before using it (in this or another program).

- ***Encapsulation (part 2)***: Objects have the ability to hide certain parts of themselves from programmers. This prevents programmers from tampering with values they shouldn't. Additionally, the object controls how one interacts with it, preventing other kinds of errors.

- ***Design Benefits***: Large programs are very difficult to write. Object Oriented Programs force designers to go through an extensive planning phase, which makes for better designs with less flaws. In addition, once a program reaches a certain size, Object Oriented Programs are actually *easier* to program than non-Object Oriented ones.

- ***Software Maintenance:*** Programs are not disposable. Legacy code must be dealt with on a daily basis, either to be improved upon (for a new version of an exist piece of software) or made to work with newer computers and software. An Object Oriented Program is much easier to modify and maintain than a non-Object Oriented Program. So although a lot of work is spent before the program is written, less work is needed to maintain it over time.

5.1.3 Multi-Platform

With an increasing number of software company begin to the field of mobile phone and smart device, too much operation system has been established during last decades years. Multi-Platform will be a significant way to solve the problem that the same process cannot be adapted on different operation system.

Although the C sharp is main language in Visual Studio for Windows app, Microsoft begin to invest some Multi-platform tools, such as Momo-Studio. Therefore, C# also enter more field of development. For example, Android development, Java was the most popular developing tool for Android app, the IOS app is Swift and Xcode base on C-object.

UWP is only crossing the different device, such as Xbox, win tablet, win phone and so on, but all of them are from the Microsoft.

5.2 Why Do I Use The UWP

Windows 10 introduces the Universal Windows Platform (UWP), which provides a common app platform on every device that runs Windows 10. The UWP provides a guaranteed core API across devices. New adaptive controls and layout panels help you to tailor your UI across a broad range of device screen resolutions and sizes, and respond to multiple kinds of device input. A unified

app store makes your app available on Windows 10 devices such as PC, tablet, Xbox, HoloLens, Surface Hub, and Internet of Things (IoT) devices.
UWP is flexible. You can use languages such as C#, C++, Javascript, and VB. C++ desktop app that you want to modernise with UWP features and sell in the Microsoft store.

5.2.1 The advantage of UWP

- **There is a common API surface across all devices**
  The Universal Windows Platform (UWP) core APIs are the same for all Windows devices. If your app only uses the core APIs, it will run on any Windows 10 device no matter whether you are targeting a desktop PC, Xbox, Mixed Reality headset, and so on.
  A UWP app written in C++ /WinRT or C++ CX has access to the Win32 APIs that are part of the UWP. These Win32 APIs are implemented by all Windows 10 devices.

- **Extension SDKs expose the unique capabilities of specific device types**
  If you target the universal APIs, your app can run on all devices that run Windows 10. But if you want your UWP app to take advantage of device specific APIs, you can.
  An extension SDK lets you call specialised APIs for a device. For example, if your UWP app targets an IoT device, you can add the IoT extension SDK to your project to target features specific to IoT devices. You can write your app so that you expect it to run only on a particular type of device, and then limit its distribution from the Microsoft Store to just that type of device. Or, you can conditionally test for the presence of an API at runtime and adapt your app's behaviour accordingly.

- **There's one store for all devices.**
  You can submit your app to the store and make it available to all types of devices, or only those you choose. You submit and manage all your apps for Windows devices in one place.
  UWP apps integrate with Application Insights for detailed telemetry and analytics—a crucial tool for understanding your users and improving your apps.

- **Apps distributed from the Store provide a seamless install, uninstall, and upgrade experience**
  All UWP apps are distributed using a packaging system that protects the user, device, and system. Users never need to regret installing an app. UWP apps can be uninstalled without leaving anything behind. Apps can

be deployed and updated seamlessly. App packaging can be modularized so that you can download content and extensions on demand.

- **Apps support adaptive controls and input**
  UI elements respond to the size and DPI of the screen the app is running on by adjusting their layout and scale. And UWP apps 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 design UI that can adapt to the devices your app may run on.

5.2.2 How to use the Tools Of UWP

On the home page, coder could find out the official document, which shows all details of tools, include the old module. And document also gives the explanation with code example. developer can follow this code to understand the functionality of API and models. If coder meet some tough problems, also post email and ask for staff online to get response we want.

5.3 Why Do I Use The Xamarin

First reason, Xamarin is a part of System from Microsoft, I use C Sharp to develop a IOS or Android app on this development environment. Because the Xamarin Form 3.0 was born, coder can code a general process who can transfer to IOS app or Android app.

5.3.1 The advantage of Xamarin

The biggest advantage of Xamarin is Xamarin Form, which is a framework that allows developers to rapidly create cross platform user interfaces. It provides it's own abstraction for the user interface that will be rendered using native controls on iOS, Android, Windows, or Windows Phone. This means that applications can share a large portion of their user interface code and still retain the native look and feel of the target platform.

Xamarin.Forms allows for rapid prototyping of applications that can evolve over time to complex applications. Because Xamarin.Forms applications are native applications, they don't have the limitations of other toolkits such as

browser sandboxing, limited APIs, or poor performance. Applications written using Xamarin.Forms are able to utilise any of the API's or features of the underlying platform, such as (but not limited to) CoreMotion, PassKit, and StoreKit on iOS; NFC and Google Play Services on Android; and Tiles on Windows. In addition, it's possible to create applications that will have parts of their user interface created with Xamarin.Forms while other parts are created using the native UI toolkit.

Xamarin.Forms applications are architected in the same way as traditional cross-platform applications. The most common approach is to use Portable Libraries or Shared Projects to house the shared code, and create platform specific applications that will consume the shared code.
There are two techniques to create user interfaces in Xamarin.Forms. The first technique is to create UIs entirely with C# source code. The second technique is to use *Extensible Application Markup Language* (XAML), a declarative markup language that is used to describe user interfaces. For more information about XAML, see XAML Basics.

5.3.2 How to use the Tools in Xamarin

At the same way, The Xamarin home-page has the document for developer. They are able to reach any developing tools they want, and the document also give developers the example and explain.

However, there is a big different that is the website has three documents for developer, respectively the IOS app, Android and Xamarin Form document, than UWP document. If coder would like to only develop the Android app, they just look up the Android document as well. The Xamarin Form document shows the document of Crossing-platform app. Therefore, coder can share majority code between two operation system.

5.4 The Developing Process of Project

Because the project is full-year project, it experienced a long period from idea to testing.

On the meeting of project, supervisor gave me too many advises to enhance my project. Depend on this advises, I modified the details of project. adding more view module and new functionality.

5.4.1 Computer Application

The computer application is meant that is running on WIN 10 system. In general, UWP is new structure for windows app, Microsoft think the UWP will be able to become a kind of popular language for coder in the future. Not only are a computer app, but UWP apps also are crossing-platform app, which can be install in different device from Microsoft, such as XBOX, WIN tablet, WIN TV and so on.

Although this app could be install different device, I only want to use this app on the computer.

5.4.2 Mobile Application

A mobile application, most commonly referred to as an app, is a type of application software designed to run on a mobile device, such as a smartphone or tablet computer. Mobile applications frequently serve to provide users with similar services to those accessed on PCs. Apps are generally small, individual software units with limited function. This use of software has been popularised by Apple Inc. and its App Store, which sells thousands of applications for the iPhone, iPad and iPod Touch.

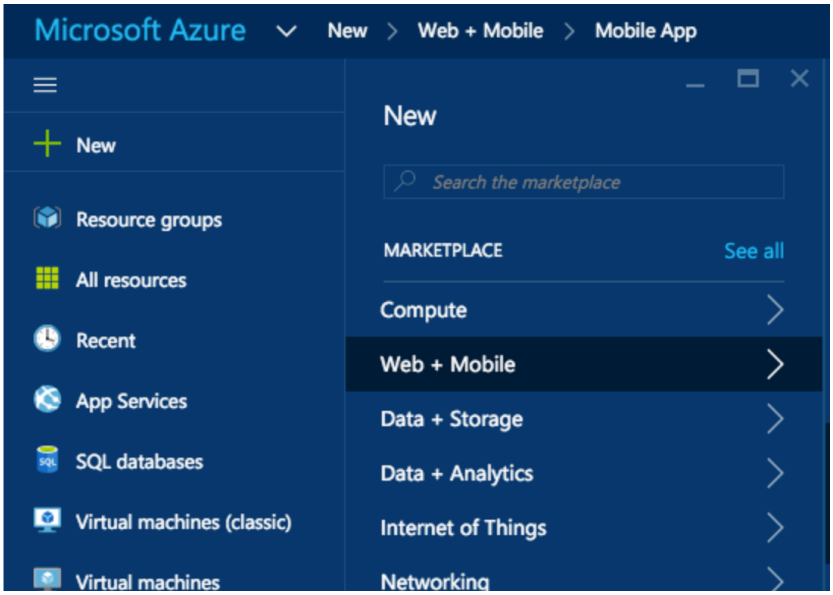
A mobile application also may be known as an app, Web app, online app, iPhone app or smartphone app.

Because my mobile app form Xamarin Studio, this app has two versions both Android and IOS.

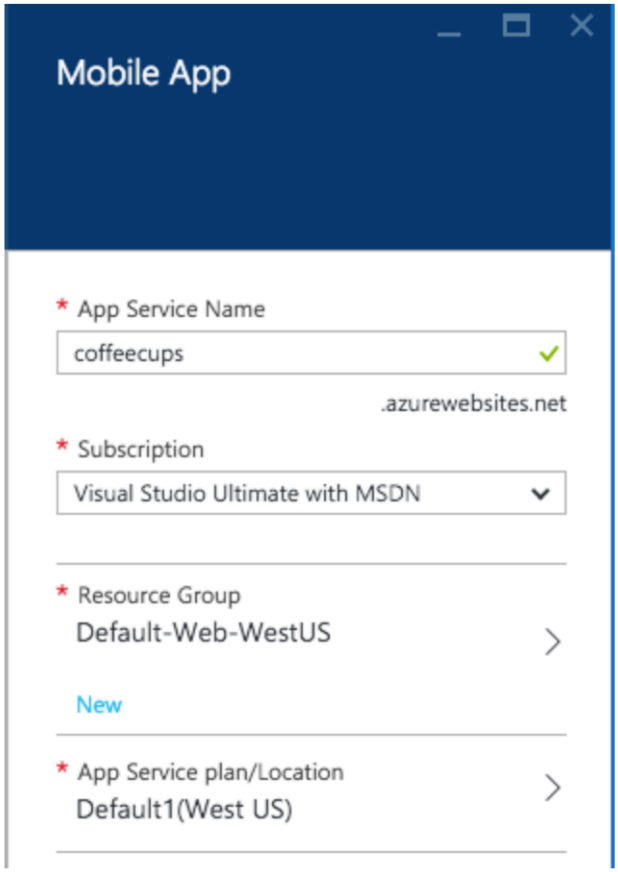
5.4.3 Azure Service Stage Established

1. Inside of the Azure portal, simply select **New -> Web + Mobile -> Mobile App**, which is the starting point to configure your Azure Mobile App backend.

2. When selecting Mobile App in Azure, you will need to configure the service name (this is the URL where your backend web app/ASP.NET website will live), configure your subscription, and set your resource group and plan. I call Seattle home, so I’ve selected the default West Euro locations:



3. Give Azure a few minutes to deploy your mobile app, and once deployed, the Azure portal will bring you directly to the configuration screen with the settings tab open. All of the settings we'll adjust can be found under the **Mobile** section.

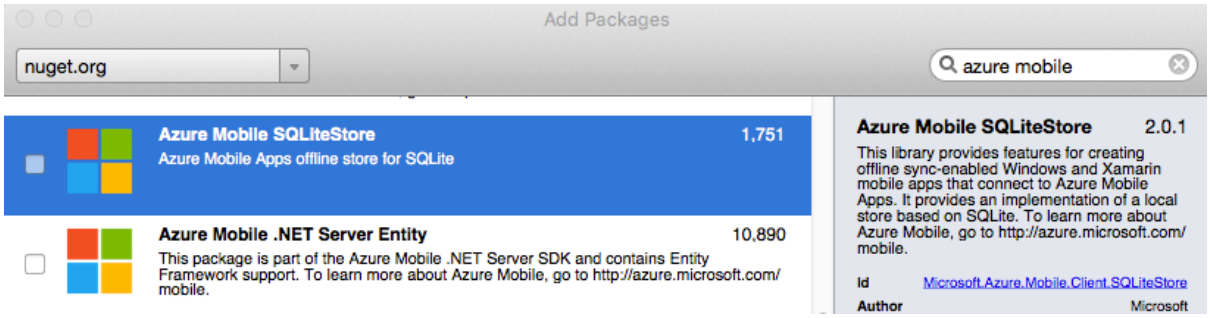


4. Adding a New Table Under Mobile settings is a new section called *Easy Tables*, which enable us to easily set up and control the data coming to and from the iOS and Android apps. Select the **Easy Tables** section, and we'll be prompted with a big blue warning asking us to configure *Easy Tables/ Easy APIs*:

5. After a few moments, the app will be fully initialised so we can add our first table of the database named **CupOfCoffee**. If you are adding Azure Mobile

Apps to an existing application, this table name should match the class name of the data you with to store. The beautiful part of *Easy Tables* is that it will automatically update and add the columns in the table dynamically based on the data we pass in. For this example, I'll simply allow full anonymous access to the table, however it is possible to add authentication with Facebook, Twitter, Google, Microsoft, and other OAuth login providers.

6. With our backend fully set up on Azure, it's now time to integrate the Azure Mobile SDK into your mobile apps. The first step is to add the Azure Mobile Apps NuGet package that's named Azure Mobile SQLiteStore. This package sits on top and includes the Mobile Services Client SDK to connect to our backend and adds full online/offline synchronisation and should be added to all application projects. For example CoffeeCups is written using Xamarin.Forms, so I added the NuGet to my PCL, iOS, Android, and Windows projects:



It is set up azure service below front. Then Initialise the Azure Mobile Client.

Add the Azure Mobile Client SDK initialisation code in the platform projects. For iOS, the following code must be added to the FinishedLaunching method of the AppDelegate class:

```
Microsoft.WindowsAzure.MobileServices.CurrentPlatform.Init();
SQLitePCL.CurrentPlatform.Init();
```

For Android, add the following to the OnCreate method of your MainActivity:

```
Microsoft.WindowsAzure.MobileServices.CurrentPlatform.Init();
```

The Data Model: It's now time to create the data model that we'll use locally to display information, but also save in our Azure Mobile App backend. It should

have the same name as the table that we created earlier. There are two string properties required to ensure the Mobile Apps SDK can assign a unique identifier and version in case any modifications are made to the data. Here is what the CupOfCoffee model looks like:

below code is just example from tutorial.

public class CupOfCoffee

```
{
    [Newtonsoft.Json.JsonProperty("Id")]
    public string Id { get; set; }

    [Microsoft.WindowsAzure.MobileServices.Version]
    public string AzureVersion { get; set; }

    public DateTime DateUtc { get; set; }
    public bool MadeAtHome{ get; set; }
}
```

then, use the Mobile App SDK to create a MobileServiceClient that will enable us to perform create, read, update, and delete (CRUD) operations that will be stored locally and synchronised with our backend. All of this logic will be housed in a single class, which for this app is called “AzureDataService” and has a MobileServiceClient and a single IMobileServiceSyncTable. Additionally, it has four methods to Initialize the service, get all coffees, add a coffee, and synchronize the data with the backend:

public class AzureDataService

```
{
    public MobileServiceClient MobileService { get; set; }
    IMobileServiceSyncTable coffeeTable;

    public async Task Initialize()
    {
    }

    public async Task<IEnumerable> GetCoffees()
    {
    }

    public async Task AddCoffee(bool madeAtHome)
    {
```

```
    }

    public async Task SyncCoffee()
    {
    }
}
```

Creating the Service and Table
Before we can get or add any of the data we must create the MobileServiceClient and the SyncTable. This is done by passing in the URL of the Azure Mobile App and specifying the file in which to store the local database:

```
public async Task Intialize()
{
   //Create our client
   MobileService = new MobileServiceClient("https://
coffeecups.azurewebsites.net");

   const string path = "syncstore.db";
   //setup our local sqlite store and intialize our table
   var store = new MobileServiceSQLiteStore(path);
   store.DefineTable();
    await MobileService.SyncContext.InitializeAsync(store, new
MobileServiceSyncHandler());

   //Get our sync table that will call out to azure
   coffeeTable = MobileService.GetSyncTable();
}
```

when Azure system is established, we can begin to connect Azure service from both two process.

5.4.4 Testing of Application

The testing of application is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test.Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software

implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test

- meets the requirements that guided its design and development
- responds correctly to all kinds of inputs
- performs its functions within an acceptable time
- is sufficiently usable
- can be installed and run in its intended environments, and
- achieves the general result its stakeholders desire

5.5 Database Design

An application who is really successful has good databases design, a databases need store what is information. Let process has too much functionality based on databases technology. Like SQLite, which is a small databases technology compare to others, and it is the one of local databases. Because of this reason, SQLite is good for Log-in and Log-up functionality on the user interface.

5.5.1 Why to Choose the SQLite

Zero-Configuration: SQLite does not need to be "installed" before it is used. There is no "setup" procedure. There is no server process that needs to be started, stopped, or configured. There is no need for an administrator to create a new database instance or assign access permissions to users. SQLite uses no configuration files. Nothing needs to be done to tell the system that SQLite is running. No actions are required to recover after a system crash or power failure. There is nothing to troubleshoot.

SQLite just works:Other more familiar database engines run great once you get them going. But doing the initial installation and configuration can be intimidatingly complex.

Serverless: Most SQL database engines are implemented as a separate server process. Programs that want to access the database communicate with the server using some kind of interprocess communication (typically TCP/IP) to send

requests to the server and to receive back results. SQLite does not work this way. With SQLite, the process that wants to access the database reads and writes directly from the database files on disk. There is no intermediary server process.

There are advantages and disadvantages to being serverless. The main advantage is that there is no separate server process to install, setup, configure, initialize, manage, and troubleshoot. This is one reason why SQLite is a "zero-configuration" database engine. Programs that use SQLite require no administrative support for setting up the database engine before they are run. Any program that is able to access the disk is able to use an SQLite database. On the other hand, a database engine that uses a server can provide better protection from bugs in the client application - stray pointers in a client cannot corrupt memory on the server. And because a server is a single persistent process, it is able control database access with more precision, allowing for finer grain locking and better concurrency.
Most SQL database engines are client/server based. Of those that are server-less, SQLite is the only one that this author knows of that allows multiple applications to access the same database at the same time.

Single Database File: An SQLite database is a single ordinary disk file that can be located anywhere in the directory hierarchy. If SQLite can read the disk file then it can read anything in the database. If the disk file and its directory are writable, then SQLite can change anything in the database. Database files can easily be copied onto a USB memory stick or emailed for sharing.

Other SQL database engines tend to store data as a large collection of files. Often these files are in a standard location that only the database engine itself can access. This makes the data more secure, but also makes it harder to access. Some SQL database engines provide the option of writing directly to disk and bypassing the filesystem all together. This provides added performance, but at the cost of considerable setup and maintenance complexity.
Stable Cross-Platform Database File

The SQLite file format is cross-platform. A database file written on one machine can be copied to and used on a different machine with a different architecture. Big-endian or little-endian, 32-bit or 64-bit does not matter. All machines use the same file format. Furthermore, the developers have pledged to keep the file format stable and backwards compatible, so newer versions of SQLite can read and write older database files.
Most other SQL database engines require you to dump and restore the database when moving from one platform to another and often when upgrading to a newer version of the software.

5.5.2 Cloud Database[8]

Azure SQL Database is the intelligent, fully managed relational cloud database service that provides the broadest SQL Server engine compatibility, so you can migrate your SQL Server databases without changing your apps. Accelerate app development and make maintenance easy and productive using the SQL tools you love to use. Take advantage of built-in intelligence that learns app patterns and adapts to maximise performance, reliability, and data protection.

5.5.3 Another Functionality

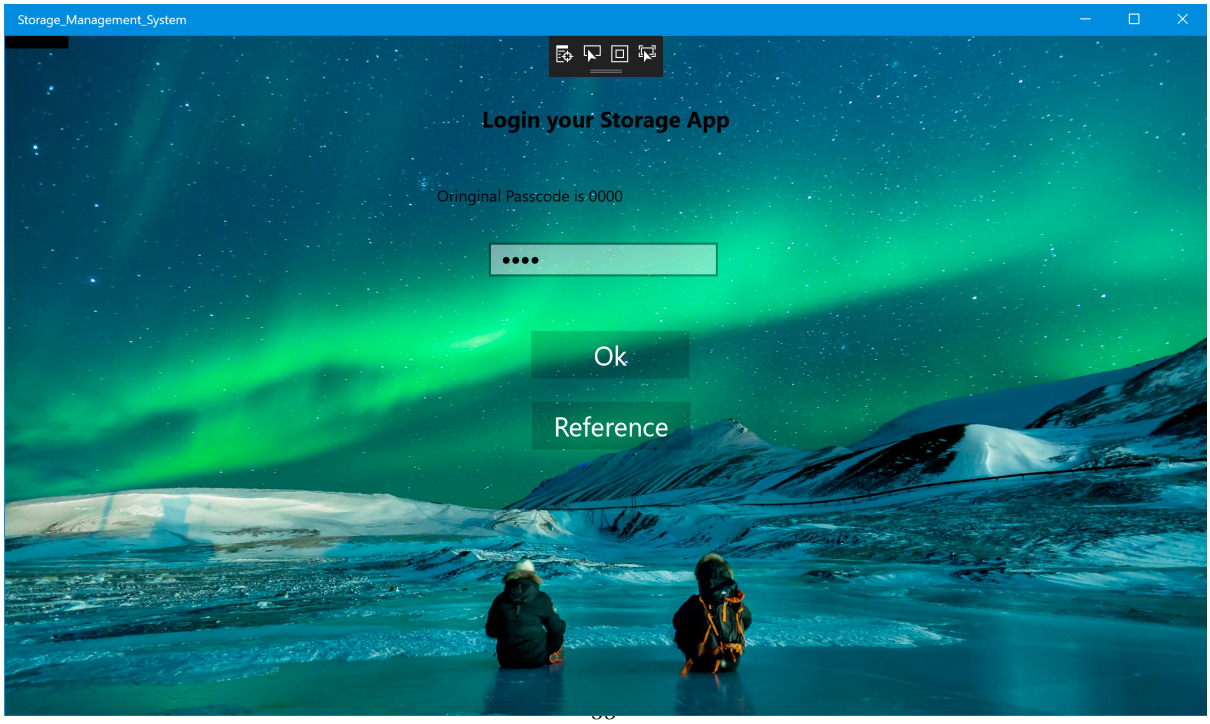
The Azure cloud databases service has lots of cloud service. Like web application service, this one brings a little web functionality.

5.6 Application Review

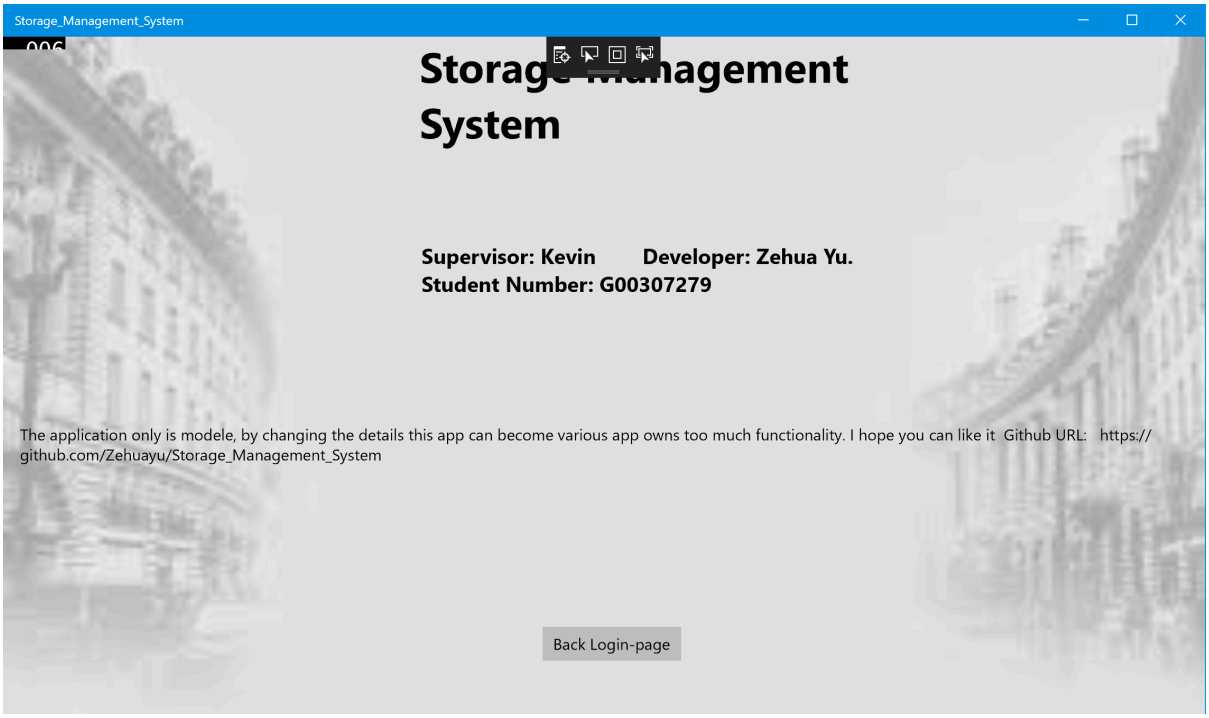
This part is that shows all of details, including the interface and some explanation with graph.

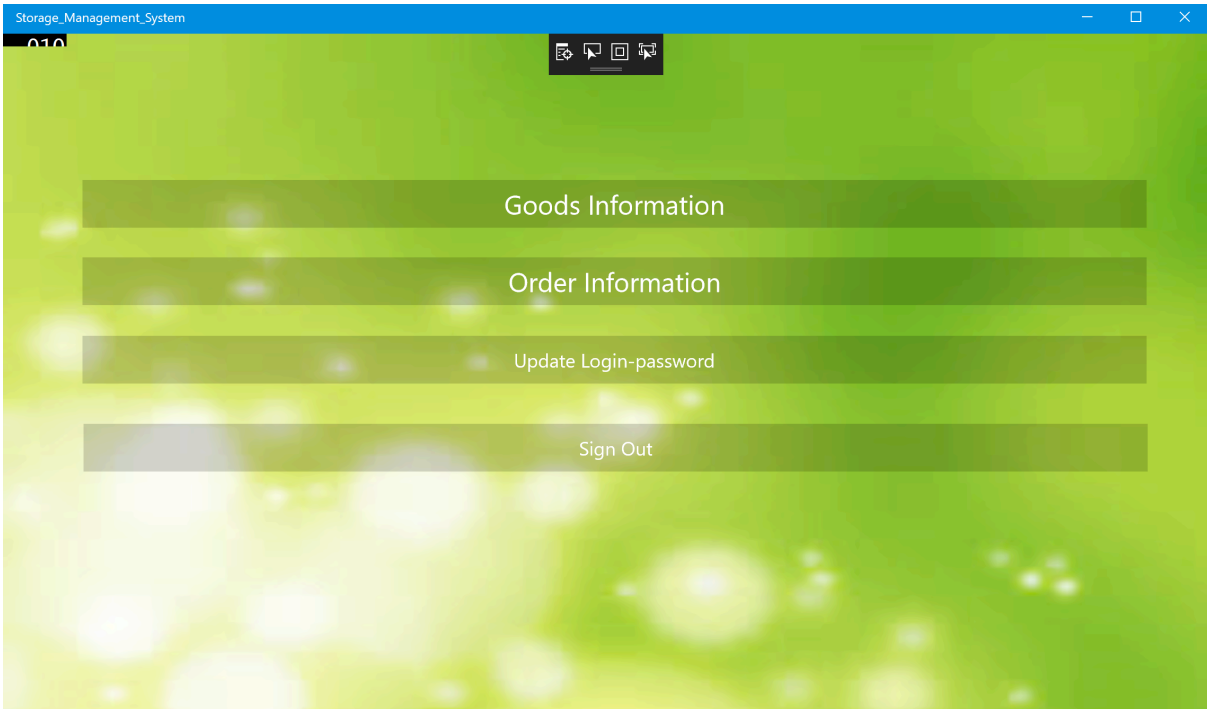
5.6.1 Computer App Review

This is the login page, there is note that original passcode is 0000, click OK to enter the Guide Page, if click Reference button to enter a sample page for introduction



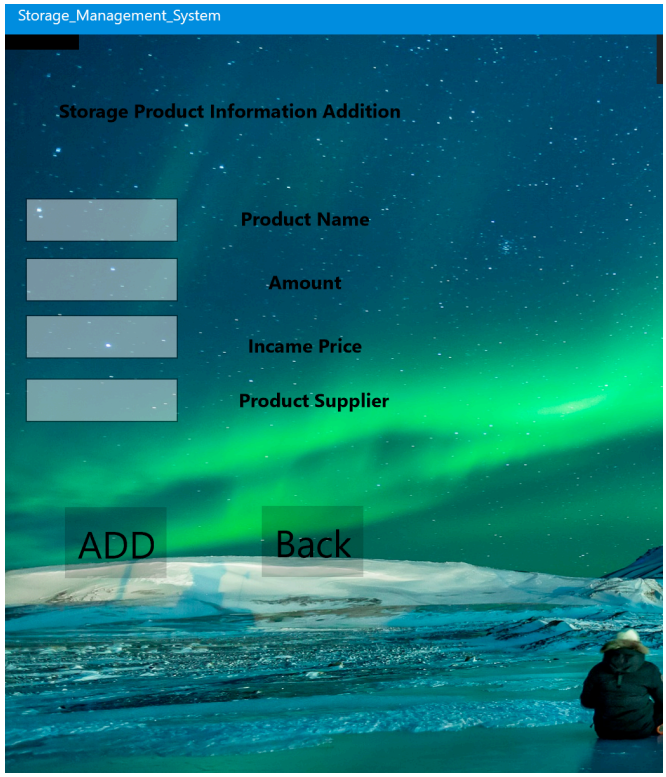
This is the reference page, the page shows my basic information and GitHub website.



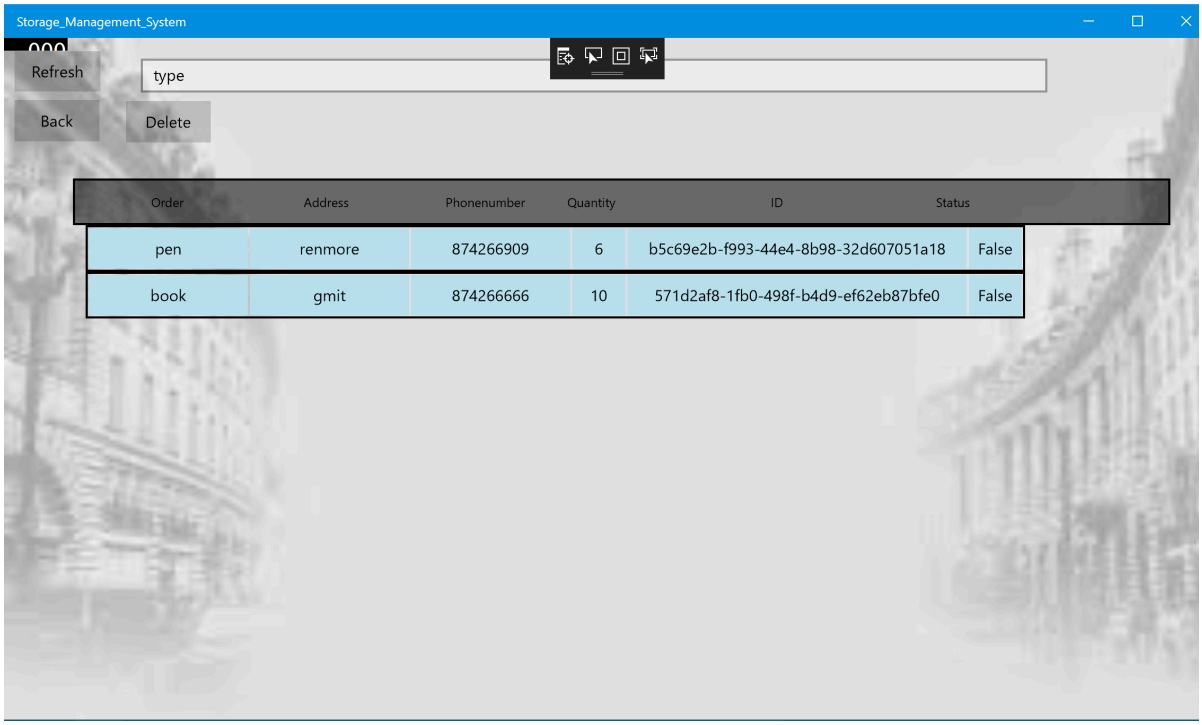


This is the menu page, the page give the four button
This is the page of goods management, the table is able to show the the goods name, quantity, price and sales price ， time and so on , user could delete some data by Delete button.

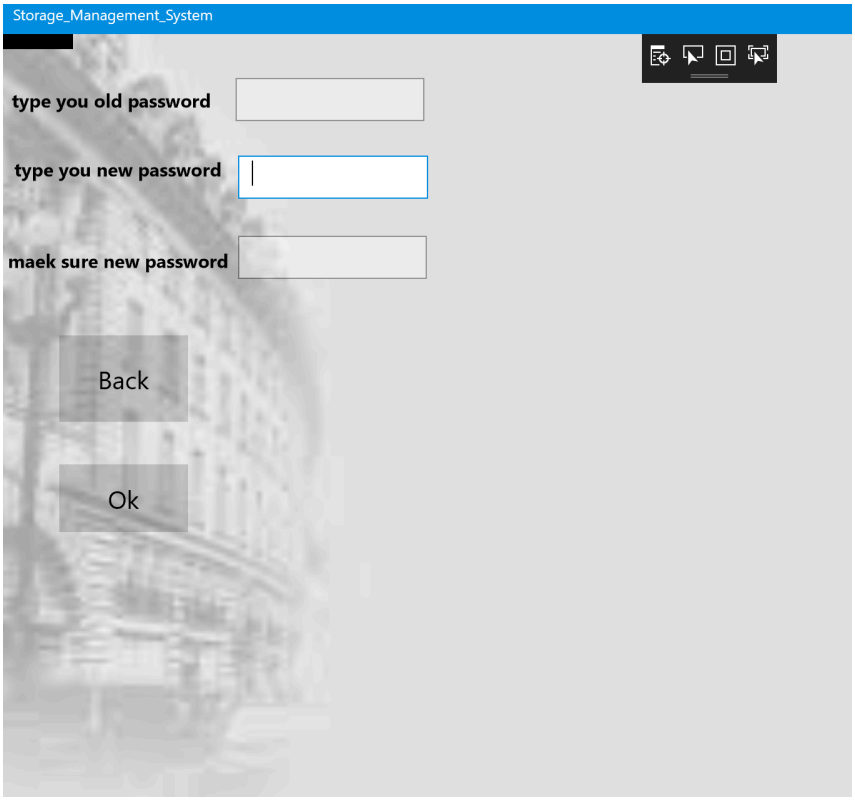
The left page is Adding page, user can insert the data to databases by this page.



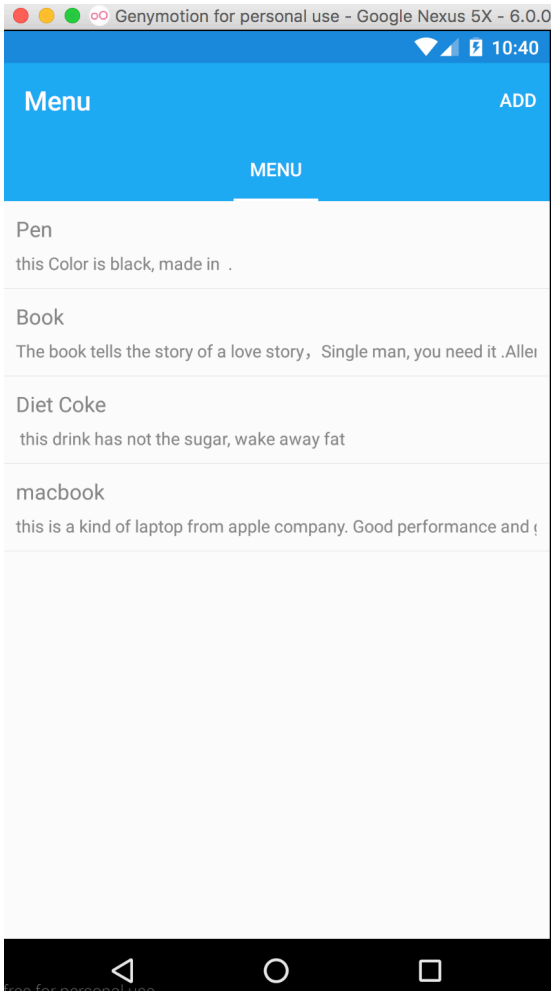
This is order page, the client post the order to this page, the computer app can receive the order. And user can Delete and Update button to change the data.



This page is mainly to provide change password functionality. User can change to private password for security.

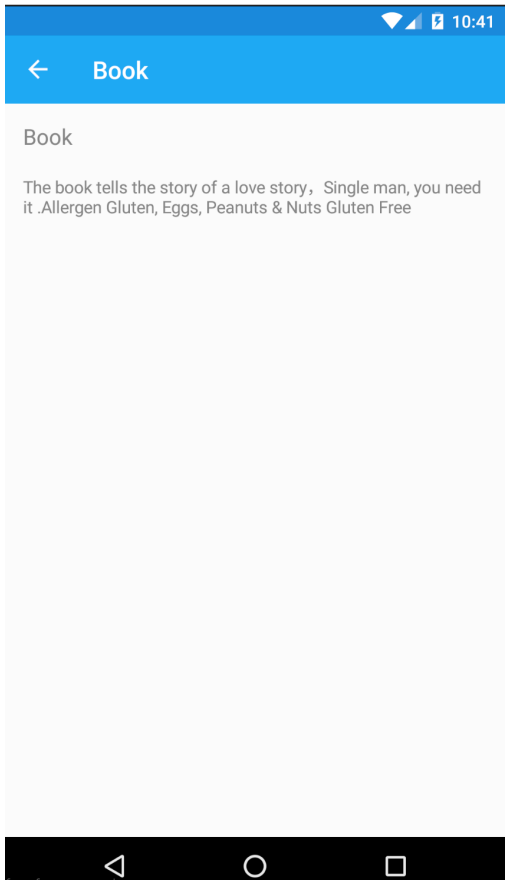


5.6.2 Mobile App Review

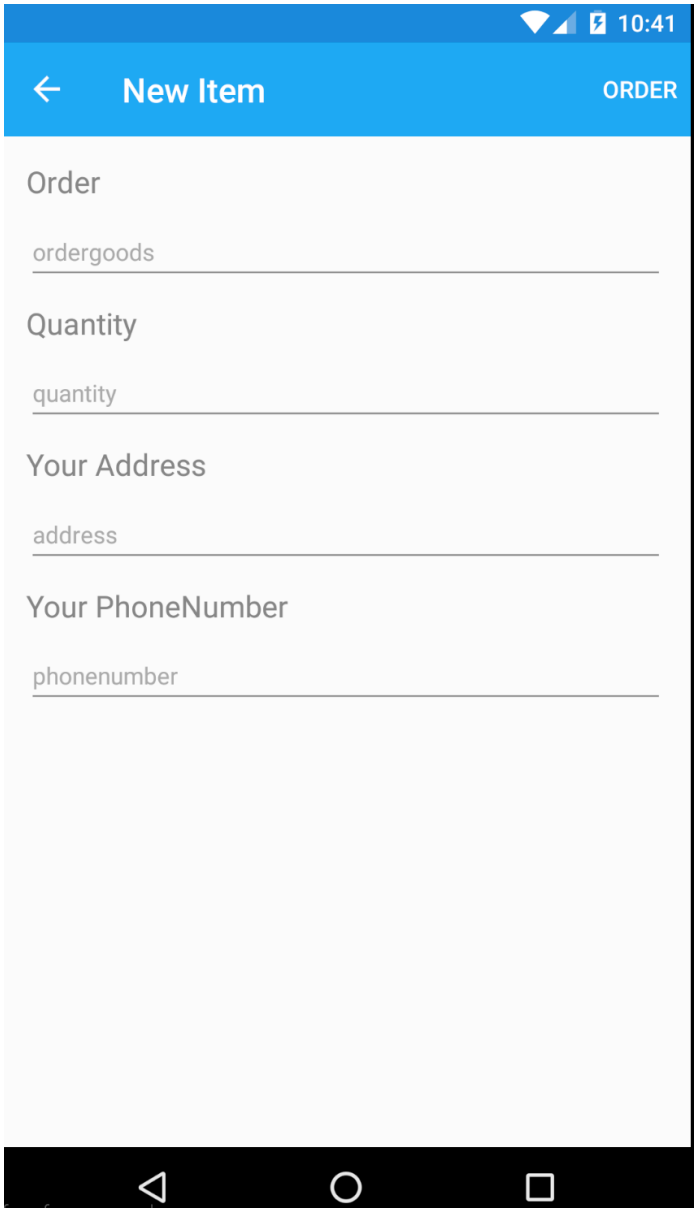


The right page is from my mobile app. this page is menu page that mainly shows the goods information.If user click the pen or someone else to entry the page of introduction.

This page is the page of introduction. The page mainly shows the introduction of goods, the storage owner can change the code



This is a order page, the user can type the information and click the order button, then this client can post the order to the Azure, my computer can catch this order and give the response.



Chapter 6

6. Software Evaluation

6.1 Advantages

My application system can have good interactive experience both the computer app and mobile app depend on the azure service.

Because the base of all devices' OS shares the same Windows 10 core, developers don't need to build different versions of their applications for different Windows 10 devices. A single application will run on Windows 10 PC, Windows 10 Tablets, Windows 10 Smartphone, Xbox console, Surface Hub, VR HoloLens… and even IoT

Therefore, the customer can easily use this app on too many kind of device. For example, customers use the tablet on Windows to manage all client information rather than computer. By changing some code, this model can adapt all kinds of shop and restaurant for the information of management.

6.2 Limitation

Because the equipment for UWP is only for Windows, so the UWP app cross-platform ability to compare restrictions, and Azure for only a year low performance of free services, if data volume is larger later, businesses have to

pay an additional Azure service. On the other hand, the function of the main program will continue to improve, for example, in terms of fees, you can also add a lot of convenient computing function in businesses improve work efficiency and management efficiency.

Mobile app using Xamarin is in the form frame, if the businessman have information need to change the single also need a programmer to upgrade and maintenance.This program is not simple to businesses can set up everything without the developers.

In term of security, I think the app needs to enhance the ability of security because my app is only a simple login interface, and a modified password function. If the user forgot the password, can't normal login, only by removing the database to restart the login function.If this software for further development.I will add in email and text message authentication function and functional point.

6.3 Future Development and Optimisation

This app has to satisfy the basic function of order, the client, if the next step to improve this app will be improved in three aspects.

1. Calculation functionality: In some businesses, such as supermarket or restaurant, shop by orders can be directly through the collection of the software database shows and calculate the total price.adding the VIP discount system, which can facilitate the employee's work and improve work efficiency.

2. Security System: the data of business can backup to the azure databases, when users lost the device, they also get back the important data from the Cloud service. On the another hand, improving the client app safety. Adding the client login page. Client user owns the private databases and plus the favourite functionality base on the local databases for mobile app.

3. Improving the Azure databases management. I am able to establish the better state for databases. The Azure service provide lots of service in order to coder can create personal service in the Azure.

6.4 Running for Customer

Get more customer survey, and modify the code in the future to meet the needs of more customers. And after outwit the conditions for merchants began trial operation. According to the feedback conditions, and constantly improve and optimise the code, let the system owns value and catches benefits for every customer.

Chapter 7

Conclusion

During the around two semesters, I started finishing the project by under five step. Each step gave me different experience about the development of application. I also learned too much skills and experience with supervisor, on some spacial field, the supervisor gave me supports. He clearly told me the requirement of project and talk about idea with me.

First step: Set up the idea of a program, and imagine me this program need to what kind of function, and consider what I need to take the technology to meet it.And technology comparison, choose the one of the most suitable for their own projects.And decided to use what surrounding technology, such as database technology and components.

Second step: The idea to make a complete plan, and to make some changes to the original technology.About my program there were several main page, I need a few of these procedures to me into my plan requirements, and mentor to discuss some technical details.

Third step: Complete two separate apps and test the basic functionality is can run according to the original thought.The focus is on the user interface design, completed two APP user interface design.And the function of the initial endows user interface components to test function is interactive.For the interface beautification

Forth step: To complete the main function and the main interface of the coding, and show a mentor to see all function.With mentor again after the discussion of some of the shortage of local changes, and strengthen the stability of the project and readability.Along with the code.For some resources and package and comment and mark address. On the another hand, testing the databases technology whether has a good interactive both process and databases.

Fifth step： Test all functions can run according to the requirement, delete function doesn't work and repair the bugs I know. In terms of the document and be completed, a further explanation of all functions and written documents.Ready for the rest of the materials and complete all other tasks that need to be done

During the period of development, I gradually understand that complete process of development. With the this process, I start to grasp the all model of development. And with the increase of development experience, had the new understanding for development system.For my later work experiences for the development and learning.

I believe that the development manager is the beginning of a number, the development of future application will become a target for a not so rare.

Appendix A

Appendix

All available source in Github Link:
https://github.com/Zehuayu/Storage_Management_System

Learning Resource:
https://docs.microsoft.com/en-us/windows/uwp/develop/
https://docs.microsoft.com/en-us/xamarin/xamarin-forms/get-started/

Tutorial
http://www.runoob.com/sqlite/sqlite-tutorial.html
https://docs.azure.cn/zh-cn/articles/training/azure-sql-database-user-manual-part-2
youtube tutorial
https://www.youtube.com/watch?v=0Rf1inu3jTs
https://www.youtube.com/watch?v=cGfNMDWYlUs&list=PLaoF-xhnnrRUNVx-JAfEy_kUrGGaKS7HL
https://www.youtube.com/watch?v=_ElhzkYYQpM
https://www.youtube.com/watch?v=IIo1udEHPNI&list=PLsrZV8shpwjN_z6f44P4gXO2L5CoBVFba

https://www.youtube.com/watch?v=6Jx7ZNZL9dk&list=PLCuRg51-gw5VqYchUekCqxUS9hEZkDf6l

Bibliography

[1] Azure MobileApp Service
https://azure.microsoft.com/en-us/services/app-service/web/
[2]Xamarin Form 3.0 *what is Xamarin Form 3.0*
https://blog.xamarin.com/glimpse-future-xamarin-forms-3-0/
[3] UWP *what is UWP*
https://docs.microsoft.com/en-us/windows/uwp/get-started/universal-application-platform-guide
[4] Xamarin Studio
https://developer.xamarin.com/releases/studio/xamarin.studio_6.3/xamarin.studio_6.3/
[5]GitHub
https://en.wikipedia.org/wiki/GitHub
[6]SQLite what is sqlite
https://www.sqlite.org/index.html
[7] Multiplatform *what is Multiplatform*
https://techterms.com/definition/multiplatform
[8]Cloud Database
https://en.wikipedia.org/wiki/Cloud_database