

## Zusammenfassung der Arbeit

## Abstract of Thesis

Fachbereich: **Electrical Engineering and Computer Science**

Department: :

Studiengang: **Information Technology**

University course: :

Thema: **Investigating and Testing communication methods**

Subject: **between the raspberry and mobile devices**

Zusammenfassung:

Abstract:

Xamarin.Forms is a C# based cross-platform programming tool which is integrated in Visual Studio 2017. This thesis is aimed to establish a real-time communication application between the raspberry pi 3 and Android or iOS devices or windows devices by using Xamarin.Forms. Through the application, the data can be transferred between the two devices so that the raspberry can realize a remote control of a hardware.

The thesis will start from introducing the basic knowledge of the tools (Visual Studio 2017, MySQL, Raspberry pi 3) and programming language (C#) which will be used in this thesis. Then give some information about the installation and configuration of the tools. The email and MySQL solution for the real-time communication will be presented in the thesis. The result will be discussed as well. Final part will be the summary and reflection about the whole process.

According to this mobile devices and raspberry development experience, the communication between the raspberry and other devices is reachable and possible. The limitation of the raspberry has not much influence on the programming.

Verfasser: **Qiang Ma**

Author:

Betreuender Professor/in: **Prof. Dr. Jörg Bayerlein**

Attending Professor:

WS / SS: **SS <2018>**

## Table of Contents

|   |    |
|---|----|
| Chapter 1 Introduction.....                             | 1  |
| 1.1 Topic description .....                             | 1  |
| 1.2 Motivation .....                                    | 1  |
| 1.1 Thesis structure.....                               | 1  |
| Chapter 2 Basic Knowledge .....                         | 3  |
| 2.1 Visual Studio .....                                 | 3  |
| 2.2 C# and Xamarin.Forms .....                          | 3  |
| 2.3 MySQL (Workbench).....                              | 4  |
| 2.4 Raspberry Pi 3 & IoT .....                          | 5  |
| Chapter 3 Installation and Configuration .....          | 6  |
| 3.1 Visual Studio 2017 and Xamarin.Forms .....          | 6  |
| 3.2 Raspberry Pi 3 & IoT .....                          | 7  |
| 3.3 Emulator .....                                      | 10 |
| 3.4 Workbench .....                                     | 12 |
| Chapter 4 Project Programming .....                     | 13 |
| 4.1 Xamarin Plugin.....                                 | 13 |
| 4.2 Simple Email Program .....                          | 16 |
| 4.2.1 Code Program.....                                 | 16 |
| 4.2.2 Android.....                                      | 18 |
| 4.2.3 Raspberry.....                                    | 20 |
| 4.2.4 iOS .....   | 22 |
| 4.3 MySQL.....  | 23 |
| 4.3.1 MySQL Database Programming Preparation .....      | 23 |
| 4.3.2 Android & iOS plugin Installation .....           | 23 |
| 4.4.3 Views and functions based on MySQL Database ..... | 26 |
| Chapter 5 Conclusion and outlook .....                  | 32 |
| 5.1 The project of Email and MySQL .....                | 32 |

|   |           |
|---|-----------|
| 5.2 Reflection .....                            | 32        |
| 5.2 Outlook.....                                | 33        |
| <b>Acknowledgement .....</b>                    | <b>34</b> |
| <b>Appendix A – List of figures.....</b>        | <b>35</b> |
| <b>Appendix B – List of tables.....</b>         | <b>37</b> |
| <b>Appendix C – List of code.....</b>           | <b>37</b> |
| <b>Appendix D – Contents of USB stick .....</b> | <b>37</b> |
| <b>Appendix E – Name of the component .....</b> | <b>38</b> |
| <b>Bibliography.....</b>                        | <b>41</b> |

## **Chapter 1 Introduction**

### **1.1 Topic description**

My topic is about to check the possibilities and limitations of the apps created by Xamarin.Forms on the raspberry PI 3 devices. The Xamarin.Forms is developed by the language C# in Microsoft development tool Visual Studio. The Xamarin.Forms offers the possibility to develop a cross-platform application which can run on Windows mobile phones (UWP), on Android and iOS phones as well. A more detail description of my task is to check the performance and response of the application running on the raspberry, which will be created first applied to UWP platform. I first install the internet of things (IoT) on the raspberry, because of the difference between IoT platform and windows, the responding time will be affected. My job is to find and investigate the solutions to develop a real-time communication between the raspberry and other platforms.

### **1.2 Motivation**

The communication between the Windows and mobile phones can be realized in many mature ways. Due to the limitation of the IoT platform, it is necessary to do the research how to change the parameters in a real-time communication. The professor wants to use the raspberry to make a remote control for the temperature control system and it needs to done as quickly as possible. It is very meaningful and useful for developers to learn how to use the Xamarin.Forms and create the apps that can run on the raspberry.

### **1.1 Thesis structure**

The first chapter is about to describe the detail information of the thesis task.

The second chapter presents some elementary knowledge about Xamarin, C#, and MySQL so that beginner can get a basic and quick understanding of the use of those tools.

Next, chapter 3 aims to presents the steps of installing the Visual Studio 2017, raspberry pi 3 and MySQL. The configuration of those tools will also be discussed in this chapter.

Chapter 4 will introduce the three main projects of the Email programs and database communication.

Chapter 5 will discuss the problem what I have met in the programming part.

Chapter 6 is the summary and reflection of this thesis task.

## Chapter 2 Basic Knowledge

This chapter is mainly talked about the basic knowledge of the tools which I used in the project. The Xamarin.Forms, Raspberry Pi 3 and MySQL are the most important parts. With the basic knowledge, the readers can have a better understanding of what the thesis is writing about.

### 2.1 Visual Studio

Visual studio is a strong powerful IDE (integrated development environment) tool developed by Microsoft. Visual Studio support almost 36 different programming languages and allow the editors and debuggers to support nearly any programming languages. Now the latest version is Visual Studio 2017, it was released on March 7, 2017 [1].

Microsoft has released Visual Studio .NET in February 2002. The .biggest change was the introduction of a managed code development using the .NET Framework.

Programs developed using .NET are not compiled to machine language (like C++ is, for example) but instead to a format called Microsoft Intermediate Language (MSIL) or Common Intermediate Language (CIL). When a CIL application executes, it is compiled while being executed into the appropriate machine language for the platform it is being executed on, thereby making code portable across several platforms. Programs compiled into CIL can be executed only on platforms which have an implementation of Common Language Infrastructure [2].

### 2.2 C# and Xamarin.Forms

C# is a multi-paradigm programming language encompassing strong typing, imperative, declarative, functional, generic, object-oriented (class-based), and component-oriented programming disciplines. It was developed by Microsoft within its .NET initiative and later approved as a standard by Ecma (ECMA-334) and ISO (ISO/IEC 23270:2006). C# is one of the programming languages designed for the Common Language Infrastructure [3].

C# is also a modern object-oriented programming language developed in 2000 by Anders Hejlsberg at Microsoft. C#. It was introduced by Microsoft as a Java competitor in 2000 and it derives the advantage of C++ and C as well. C# was designed for developing apps on the Microsoft platform and requires the .NET framework on Windows to work.

In recent years, mobiles are becoming an increasingly important part in the daily communication and business market. After a strong competition, the Apple families of iPhones and iPads, which run the iOS operating system and the Android operating system, developed by Google based on the Linux kernel, which runs on a variety of phones and tablets [4]. For a software developer, different platforms will create considerable difficulties in designing the user-interface, which has different conventions to program and different ways to invoke the methods. The different development environments and programming interface will also confuse those new software engineers. Moreover, the different language will also increase the hard work of the cross platform developers and they usually spend a lot of meaningless work on studying different grammars and try to remember those names and symbols.

So as to solve those problems, Microsoft has first developed Xamarin. It contributes to the implementation of the C# compiler and .NET Framework that could run on Linux. Then Xamarin Forms eventually was introduced, which allows the developers to realize the user-interface code at the same time for the iOS, Android, and Windows devices. With Xamarin Forms, the code can be shared among the iOS, Android and Windows devices.

In the latest version of Xamarin Forms, there exist three platforms---iOS for program running on the iPhone or iPad, Android for program running on the Android devices which has Android operating systems, UWP (Universal Windows Platform) for program running on the Windows Devices. I have to mention that the Windows Phone and Windows are deprecated in the newest version of Xamarin Forms (Visual Studio 2017).

## **2.3 MySQL (Workbench)**

MySQL Workbench is a unified visual tool for database architects, developers, and DBAs. MySQL workbench offers many functions such as server configuration and backup, user authorization for the data modeling and development of the SQL. MySQL workbench is available on different platforms such as Windows, Linux and Mac OS X.

MySQL has a high reputation among the world largest organizations including Facebook, Google. It can save a large amount of money and time to manage big data, high-complexity systems, and abstract information and do the most boring job efficiently and continuously.

## **2.4 Raspberry Pi 3 & IoT**

The raspberry Pi is a series of small signal-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote the teaching of basic computer science in schools and in developing countries. It has a processor, secure digital which is used to store the operating system and other program. On the board, there are four USB ports and HDMI for video output. It can also connect to Ethernet, Wi-Fi and Bluetooth.

IoT means the internet of things. It was an idea that a system of ubiquitous objects connecting the physical world to the internet.



## Chapter 3 Installation and Configuration

In this thesis, we will use Microsoft Visual Studio 2017 to accomplish the task, which is integrated with Xamarin.Forms. Workbench provides the MySQL service, which is easy to handle the installation problems of MySQL. A Mars Board is required to be installed so that we can see the user interface of the apps running on the raspberry. A mac with installed Xcode can be used as to run the apps on iPhone devices.

After a discussion with my professor Bayerlein and my classmate Mr. Gu and Mr. Yang, we decided to reduce the repetition of the steps of installation and configuration in this thesis. I will recommend you to see the detail steps of the installation and configuration in my classmate Mr. Gu's and Mr. Yang's thesis because our preparations of the installation and configuration have many similar aspects. As a result, I will simplify my description about the installation and configuration part.

### 3.1 Visual Studio 2017 and Xamarin.Forms

The Visual Studio 2017 was installed by the Visual Studio Installer, which can be downloaded from the Microsoft Visual Studio webpage [5]. There are three edition of

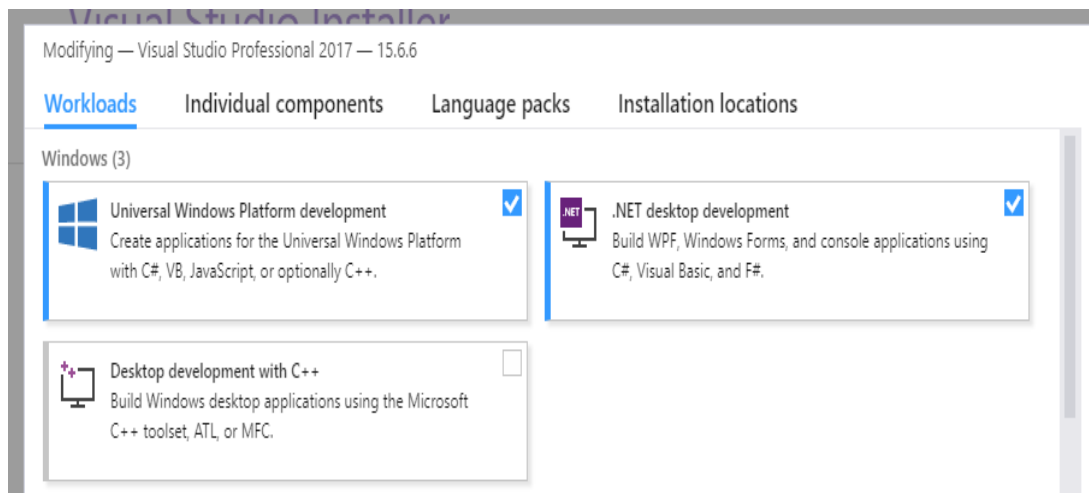


Figure 3-1-1 Workloads ---- Windows Configuration

Visual Studio will appear--- Community 2017, Professional 2017 and Enterprise 2017. Visual Studio Professional 2017 is very powerful and it can use some additional feature in MySQL database. Therefore, the Visual Studio Professional 2017 should be pressed. In the workload page, the “Universal Windows Platform development”, “.NET desktop development” (Figure 3-1-1) and “Mobile development with .NET” (Figure 3-1-2) should be selected.

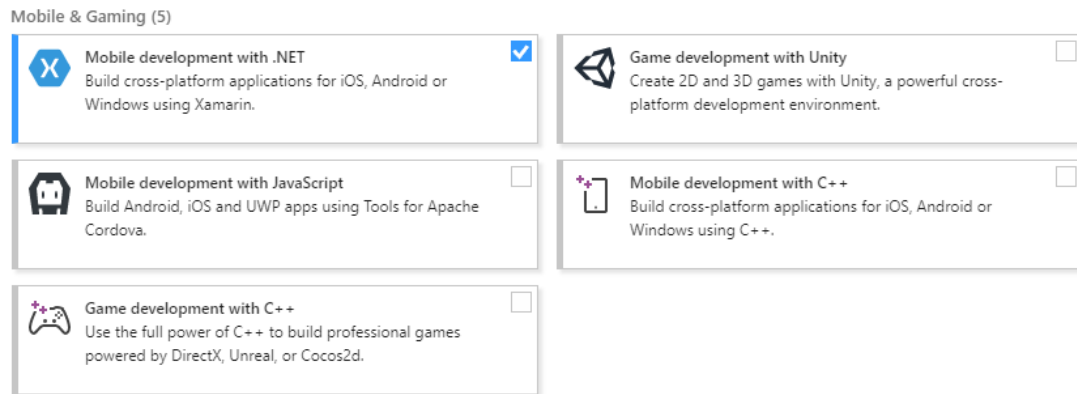


Figure 3-1-2 Workloads---Mobile &amp; Gaming Configuration

In the tab of the “Individual components”, some components are ticked when the workloads are selected. In the Emulators list, “Windows 10 Mobile Emulator” and “Visual Studio Emulator for Android” should be ticked (Figure 3-1-3). Emulators will help to run the program before running on a real mobile device.

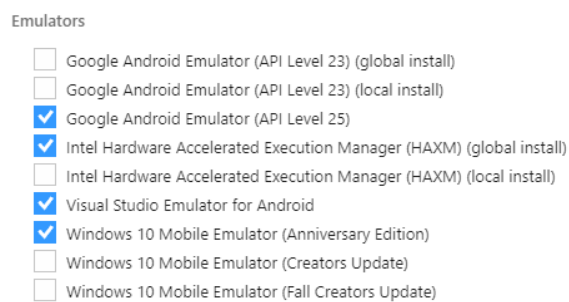


Figure 3-1-3 Individual components---Emulator Configuration

## 3.2 Raspberry Pi 3 & IoT

The raspberry pi 3 which was produced in China was given by professor Bayerlein. The detail information of the raspberry can be found on the waveshare electronics webpage [6]. There are some basic information and device structure to be shown on this page. The picture shows the front view pf the raspberry pi 3 model B (Figure 3-2-1).

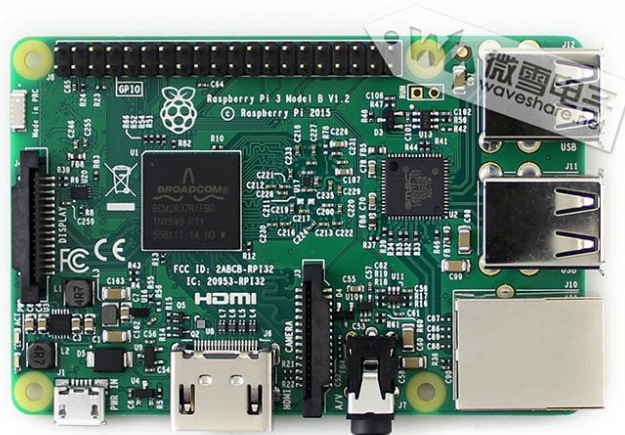


Figure 3-2-1 Raspberry Pi 3 Model B [7]

There is a video called “Waveshare 10.1 inch HDMI LCD (H)” on the YouTube to demonstrate how to install the monitor in a smooth order [8].

To install the IoT system on the SD card of the raspberry, the IoT dashboard should be downloaded. It is the best way to set up and connect the Windows 10 IoT Core devices from a PC. The view of the IoT dashboard is shown as figure 3-2-2. The installing step is presents on the website of Windows IoT dashboard [9].

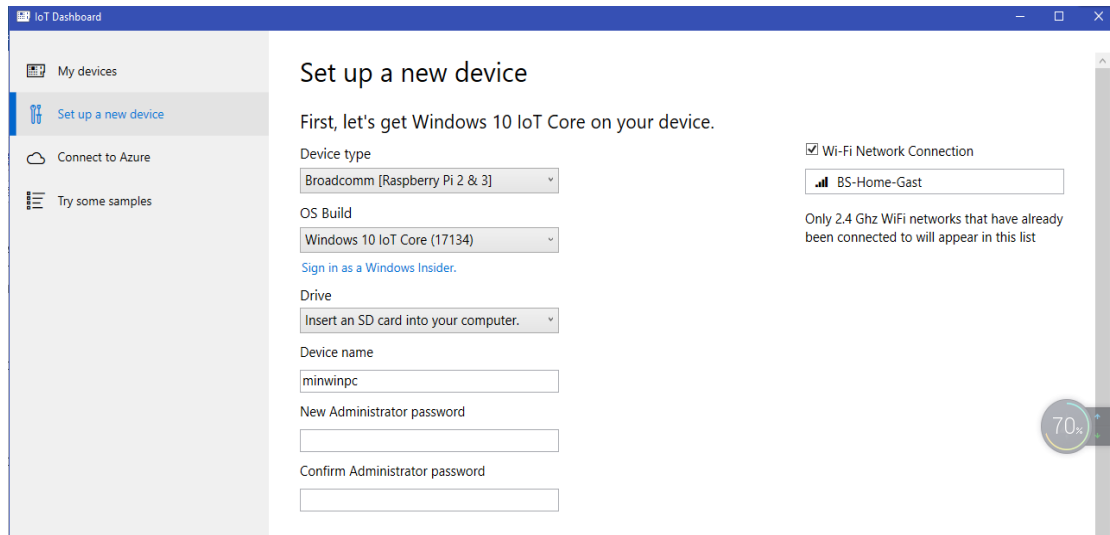


Figure 3-2-2 IoT Dashboard



Figure 3-2-3 IoT Device

After correctly installing the IoT to the raspberry and connecting the raspberry to the internet, the name of the device should be on the list of the “My devices” tab.

### 3.3 Emulator

The emulator of Android can be downloaded from the webpage [10] “Visual Studio Emulator for Android” to install or use the individual component already mentioned in the 3.1 Visual Studio part. (Figure 3-3-1)

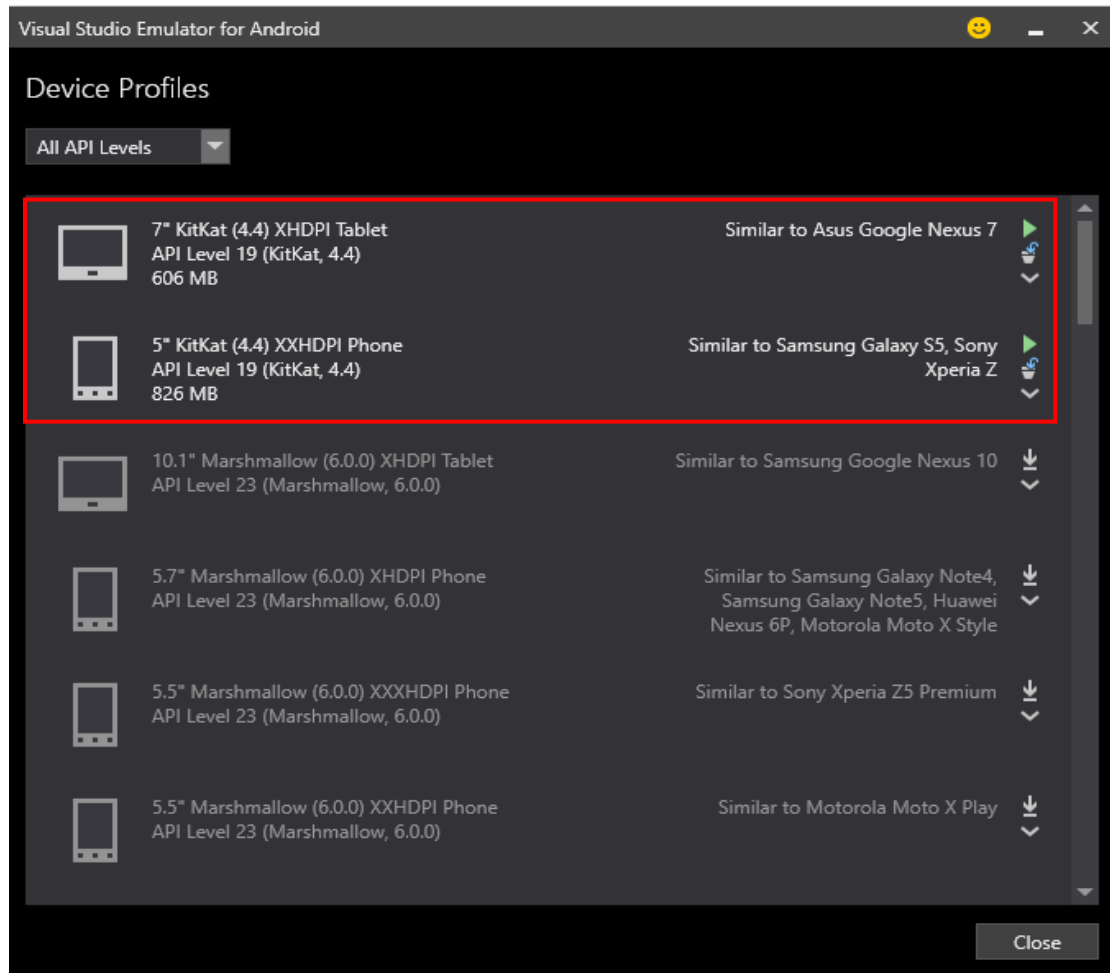


Figure 3-3-1 Visual Studio Android Emulator

In the Android SDK manager, there is a list of Android platforms. We choose the Android 8.1-Oreo platform and in the tab of tools, the Android SDK platform-tools should be ticked then click the apply changes button. (Figure 3-3-3)

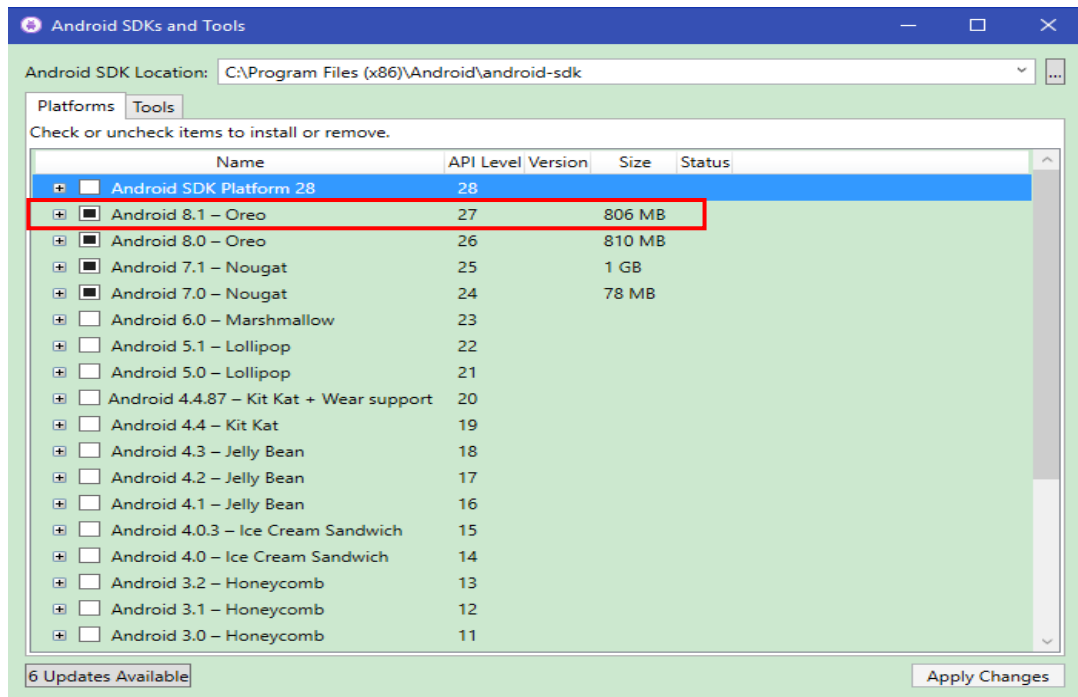


Figure 3-3-2 Android SDK Manager Platform

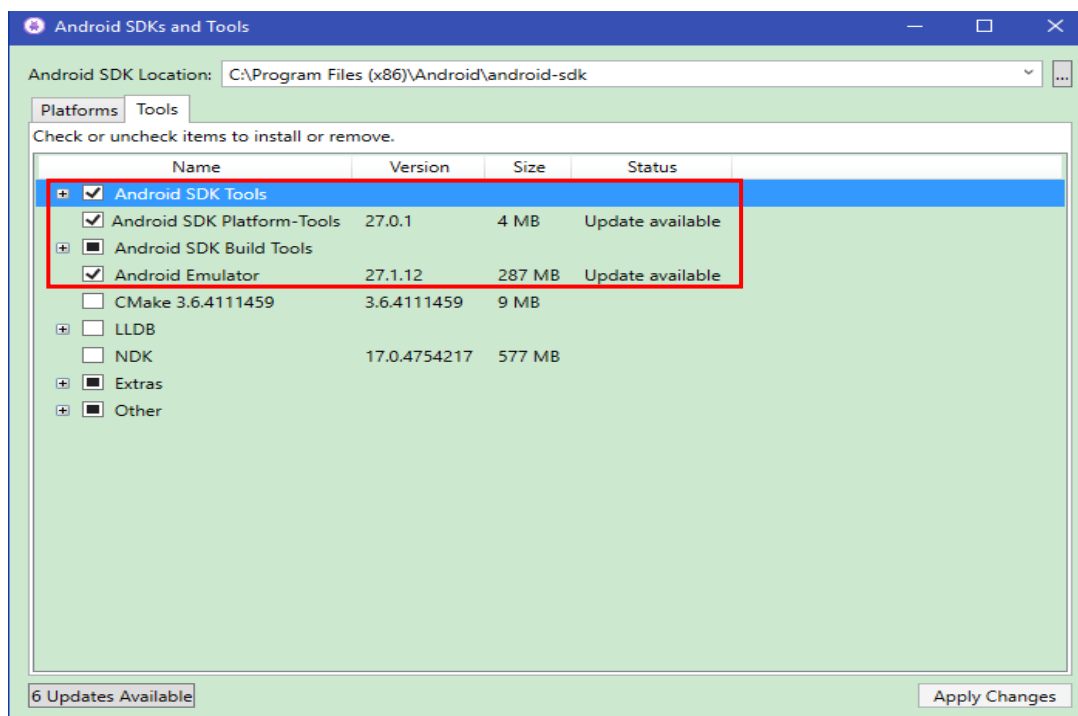


Figure 3-3-3 Android SDK Manager Tools

The emulator for iPhone devices will be presented detailed in my partner Mr. Yang's thesis for the reason that a mini mac was given to him. He is responsible for configuration the Mac and iPhone devices.

### 3.4 Workbench

Workbench can be downloaded from the webpage of the MySQL [11]. In the connection set up interface, the hostname and the username should be filled in. If you do not use the local database, you may also need the password to connect to other's database. (Figure 3-4-1)

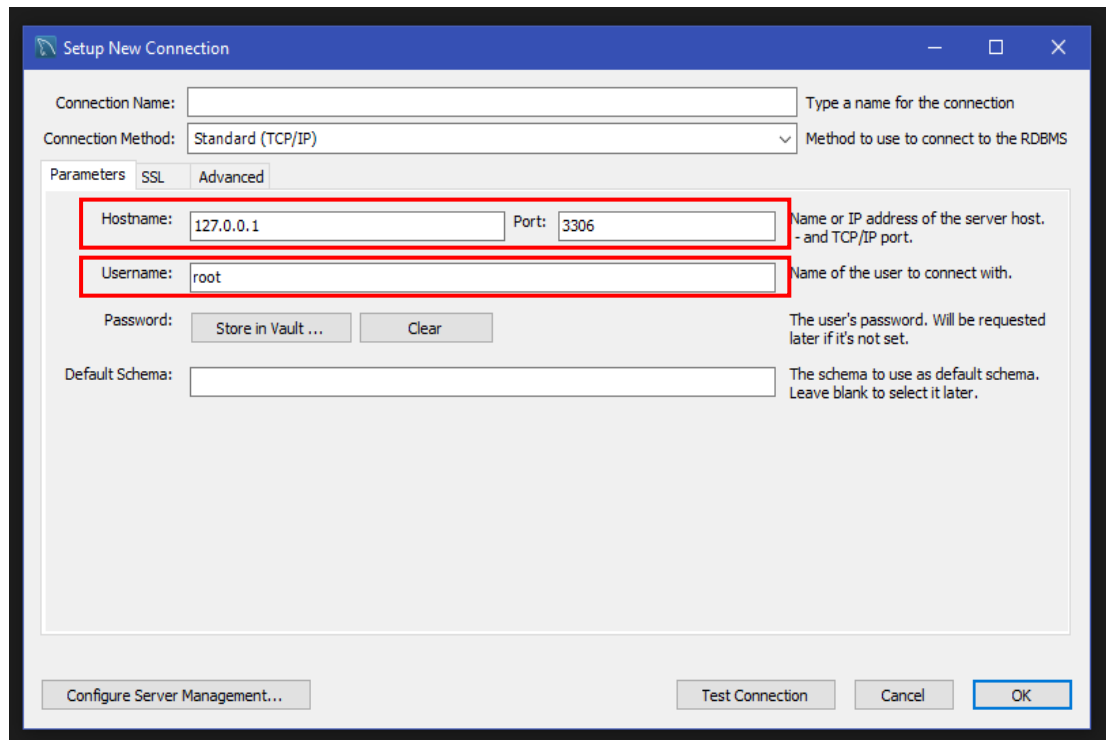


Figure 3-4-1 Workbench Connection Setting

## Chapter 4 Project Programming

This chapter will describe the detail project of Xamarin.Forms. The basic feature and guides of the Xamarin.Forms has been already introduced in the previous thesis of Mr. Liu [12]. In order to discover the possibilities and limitations of the raspberry, real-time communication was chosen to test. The project was divided into three parts--- first two parts are about the Email sending message, and the third part is about the MySQL transferring the message.

### 4.1 Xamarin Plugin

According to the project demands, an email program should be developed to receive and send simple messages like parameters or alarm to other devices with email-program. After several days of searching information on the internet, I found a Xamarin plugin on the NuGet for .NET applications. Although the Xamarin.Forms is relatively new to the software engineers, it still attracts numerous software engineers to make the contribution to it. The NuGet website is such a place contain the reusable code that other developers make available to your project. The message plugin can be downloaded from the website or on the GitHub [13]. The plugin can be easily installed by inputting the order “Install-Package Xam.Plugins.Messaging -Version 5.2.0” into the package manager console. (Figure 4-1-1).

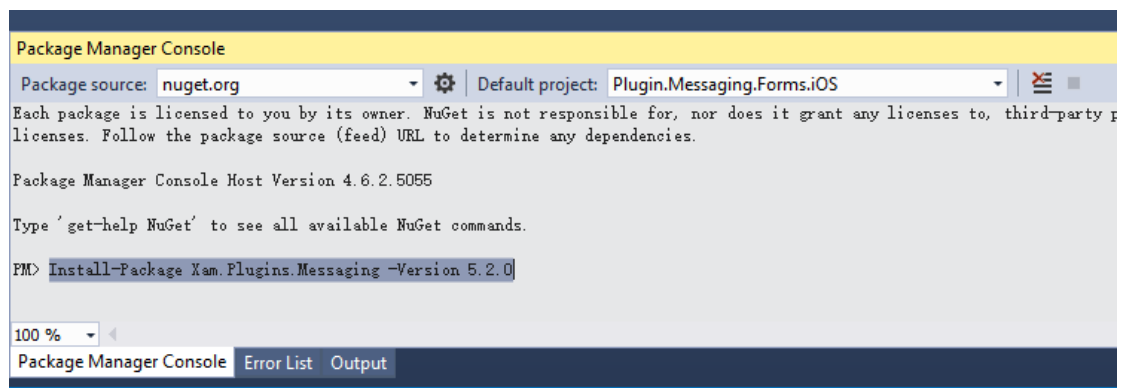


Figure 4-1-1 Package Manger Console

This Messaging plugin makes it possible to make a phone call, send SMS or send an email with the default email applications on the different platforms.

There is a sample of messaging plugin in the GitHub, from this sample we can have a look of how does this plugin work. We just need the function of sending the Email, so the focus will be put on the relative email file. In the figure 4-1-2, there are each four files in the Plugin.Messaging and Plugin.Messaging. Abstractions. In the Plugin.Messagin, emailAttachment.cs is used to realize the function of sending the



attachment with the email. The EmailMessage.cs is used to create the request to send the email. The EmailMessageBuilder.cs is used to build the pattern of the email. The MessagingPlugin.cs is responsible for implementation of the cross platform. For the Plugin.Messaging.Abstractions, it contains the interface of the email function, because different platforms need the function to be implemented separately. For example, you can see the EmailAttachment.cs both in the Plugin.Messaging.Android file and the Plugin.Messaging.iOSUnified file, which both realize the functions in the IEmailAttachment in the Plugin.Messaging.Abstractions.

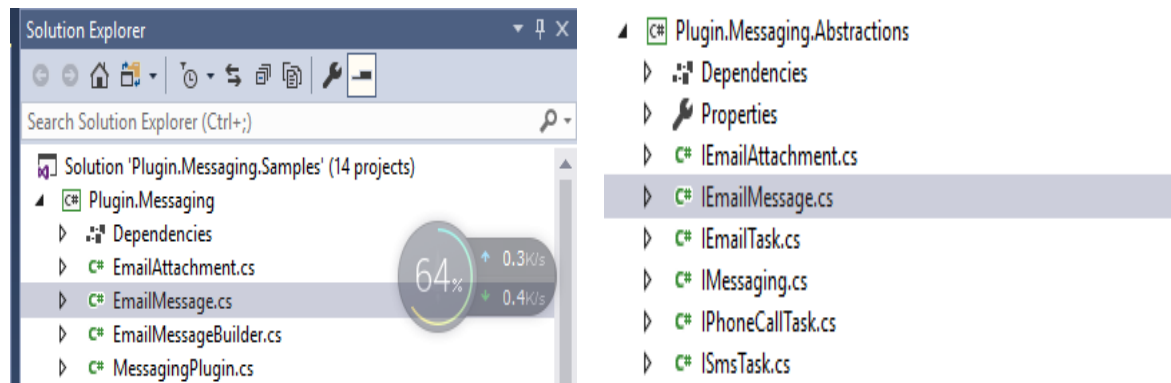


Figure 4-1-2 Messaging project

Example codes can be shown in the figure 4-1-3 and 4-1-4, the messaging plugin make used of IEmailTask abstraction to send an email. These abstractions are defined with the Plugin.Messaging.Abstractions PCL library. Platform specific implementations for those different abstractions are provided with a Plugin.Messaging library for the different platforms.

```
public interface IEmailTask
{
    bool CanSendEmail { get; }
    bool CanSendEmailAttachments { get; }
    bool CanSendEmailBodyAsHtml { get; }
    void SendEmail(IEmailMessage email);
    void SendEmail(string to, string subject, string message);
}
```

Figure 4-1-3 IEmailTask

```
var emailMessenger = CrossMessaging.Current.EmailMessenger;
if (emailMessenger.CanSendEmail)
{
    // Send simple e-mail to single receiver without attachments, bcc, cc etc.
    emailMessenger.SendEmail("to.plugins@xamarin.com", "Xamarin Messaging Plugin", "Well hello there from Xam.Messagi

    // Alternatively use EmailBuilder fluent interface to construct more complex e-mail with multiple recipients, bcc
    var email = new EmailMessageBuilder()
        .To("to.plugins@xamarin.com")
        .Cc("cc.plugins@xamarin.com")
        .Bcc(new[] { "bcc1.plugins@xamarin.com", "bcc2.plugins@xamarin.com" })
        .Subject("Xamarin Messaging Plugin")
        .Body("Well hello there from Xam.Messaging.Plugin")
        .Build();

    emailMessenger.SendEmail(email);
}
```

Figure 4-1-4 Usage of API Email

The iOS emulator is shown as the figure 4-1-5. The view of Android emulator is shown as figure 4-1-6. The figure 4-1-7 shows the view of the UWP program. The figure 4-1-8 presents the view of the program on the raspberry.

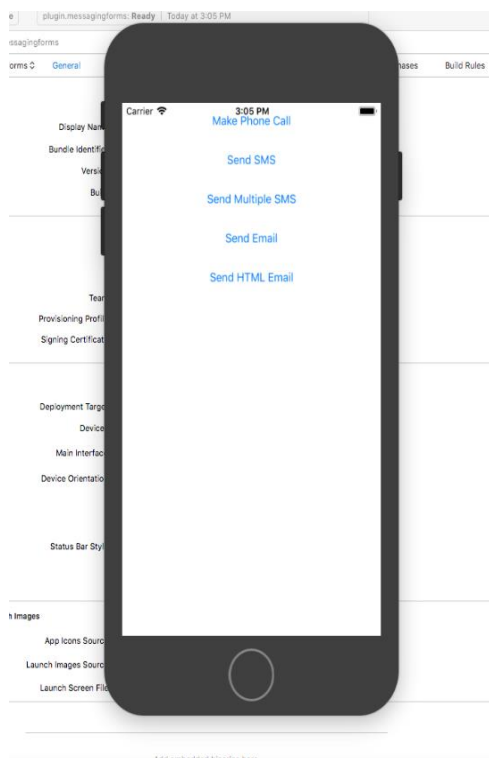


Figure 4-1-5 iOS View

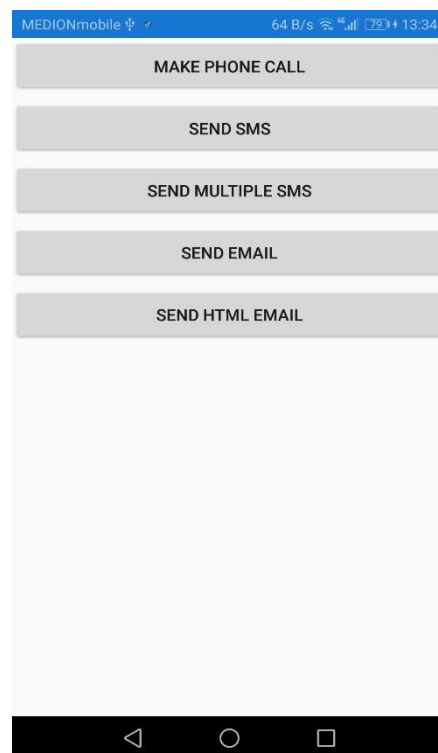


Figure 4-1-6 Android View

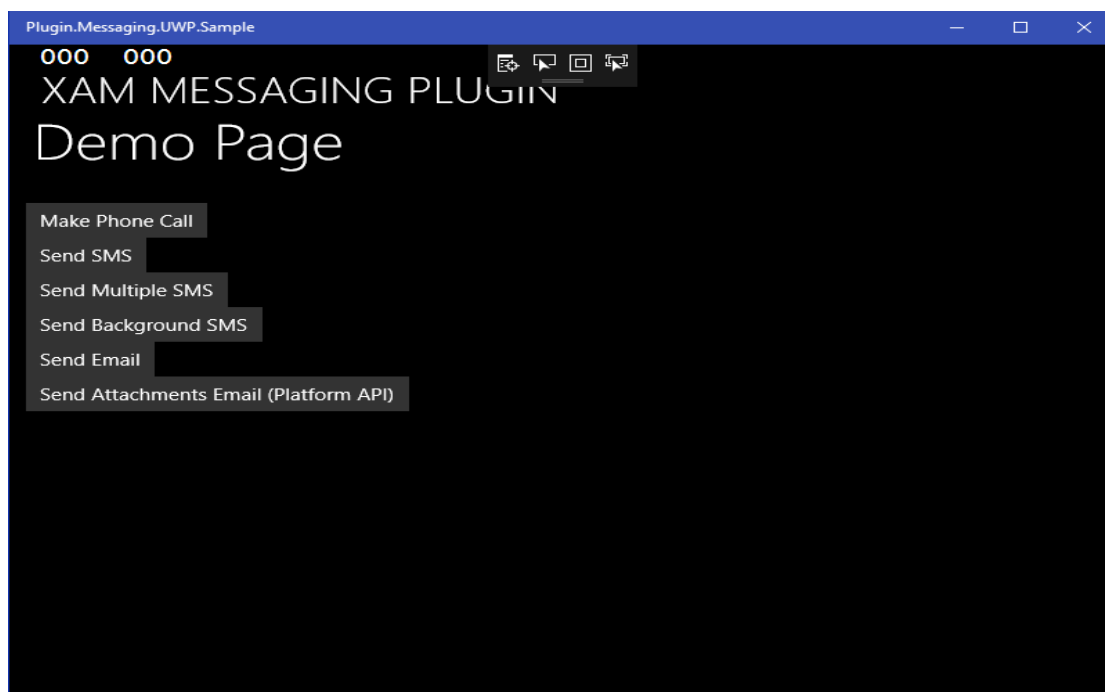


Figure 4-1-7 Raspberry View of Plugin

When you click the send email button, it will find the default email program installed on the device like the figure 4-1-8. This is a real Android mobile and it starts the

program of TypeApp installed on my device. Through this application, the windows desktop program can communicate with the real mobile devices by using the default email user. Then I found the problem, the mobile device can has a default email program, but the raspberry does not have a default program on it. Since the raspberry can only run one program at the same time, this solution can not accomplish the task of transferring the parameter. A new solution of the project should be considered.

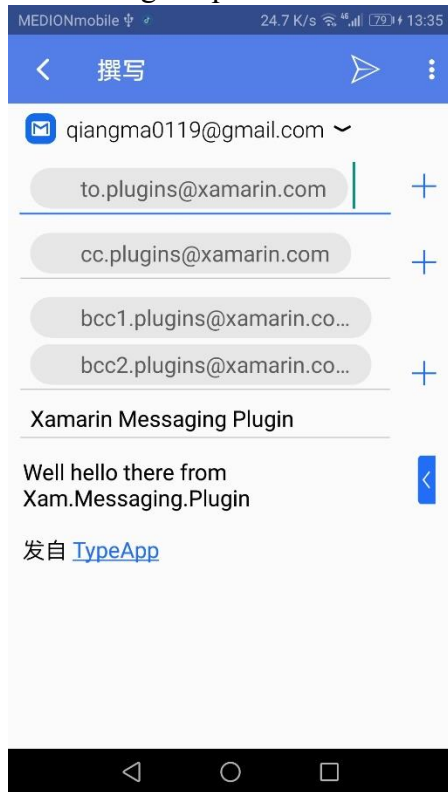


Figure 4-1-8 Android Mobile View



Figure 4-1-9 Letter from Xamarin Plugin

## 4.2 Simple Email Program

### 4.2.1 Code Program

Professor Bayerlein gave me a previous solution of the email program in Windows Forms written by himself and hoped me to rewrite the program into Xamarin.Forms. The former email program applied the System.Net.Mail namespace (using System.Net.Mail) to send electronic mail to a Simple Mail Transfer Protocol (SMTP) server for delivery. The biggest problem is that we are not sure whether the namespace of System.Net.Mail and the namespace of System.Net.Client both exists in the Xamarin.Forms, because windows form is much more sophisticated than the Xamarin.Forms. However, after several times to try, the conclusion is proved to be true. Some classes and descriptions of the System.Net.Mail is shown in the table of table 4-2-1-1.

Table 4-2-1-1 Class of System.Net.Mail

| Class         | Description  |
|---------------|--|
| AlternateView | Represents the format to view an email message                     |
| MailAddress   | Represents the address of an electronic mail sender or recipient   |
| MailMessage   | Represents an e-mail message that can be sent using the SmtpClient |
| SmtpClient    | Allows applications to send e-mail by using the SMTP               |

```

private void btnSendEmail_Click(object sender, EventArgs e)
{
    try
    {
        if (comboBoxReceiverAddr.Text != "")
        {
            MailMessage Email = new MailMessage();
            MailAddress Sender = new MailAddress(textBoxFrom.Text);
            Email.From = Sender;
        }
    }
}

```

Code 4-2-1-1 System.Net.Mail

As shown in the code block 4-2-1-1, the instance of MailMessage and MailAddress are created, then the message can be sent.

Except using System.Net.Mail namespace, another namespace is very important as well called System.Net.Sockets. It offers a managed implementation of the Windows Sockets interface of developers who need to tightly control access to the network. The table presents some classes and the descriptions of this namespace.

Table 4-2-1-2 Class of System.Net.Sockets

| Class           | Description   |
|-----------------|---|
| NetworkStream   | Provides the underlying stream of data network access   |
| Socket          | Implements the Berkeley sockets interface               |
| TcpClient       | Provides client connections for TCP network services    |
| SocketException | The exception that is thrown when a socket error occurs |

```

private bool Connect(string server, int port, string user, string password)
{
    POP3Server = new TcpClient();
    POP3Server.Connect(server, port);
    POP3Stream = POP3Server.GetStream(); // Create Stream Network
    StreamListener = new StreamReader(POP3Stream); // to create StreamReader from network stream

    if (POP3Server.Connected) // connection is successful
    {
        mailinfo = StreamListener.ReadLine();

        // Send Username
        CommandBuffer = AscEncoding.GetBytes("USER " + user + "\r\n");
        POP3Stream.Write(CommandBuffer, 0, CommandBuffer.Length);
        mailinfo = StreamListener.ReadLine();
    }
}

```

Code 4-2-1-2 System.Net.Sockets

The code block 4-2-1-2 presents the creating instance of TcpClient and get the stream from the server, which makes the communication between the server and client and converts the binary data into readable text.

To show the receiving message, a notice label was created in the receive page. The message should from the subject and the first letter should be exclamation mark (Code 4-2-1-3). The color of the message should be red (Code 4-2-1-4).

```
private void read(int mailNo)
{
    etField1.Text = ReadMailContent(mailNo);
    etBoxSubjectTemp.Text = strSubject;
    if (strSubject.Length > 0)
    {
        notice = new Label
        {
            Text = "!",
            FontSize = 50,
            TextColor = Color.FromHex("#ff0000"),
        };
    }
}
```

Code 4-2-1-4 Notice Label

## 4.2.2 Android

The user interface was designed into master-detail page. Master Detail page of the Xamarin.Forms is a page that managed two or more related pages of information---a master page that presents items, and a detail page that presents details about items on the master page. The figure 4-2-2-1 shows the master-detail of the email program on the Android emulator and the figure 4-2-2-2 shows the view of a real Android mobile.

In the figure 4-2-2-2, we can find that the application has three items called “Receive Email”, “Send Email” and “Setting”. In the setting page, the settings of the email can be found here. (Figure 4-2-2-3). In the sending page, there is an email structure which is waiting to be sent when receiving messages. (Figure 4-2-2-4)

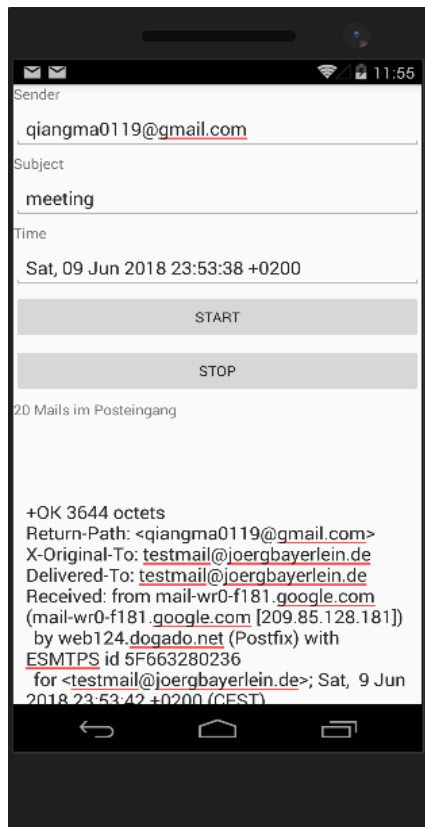


Figure 4-2-2-1 Android Emulator

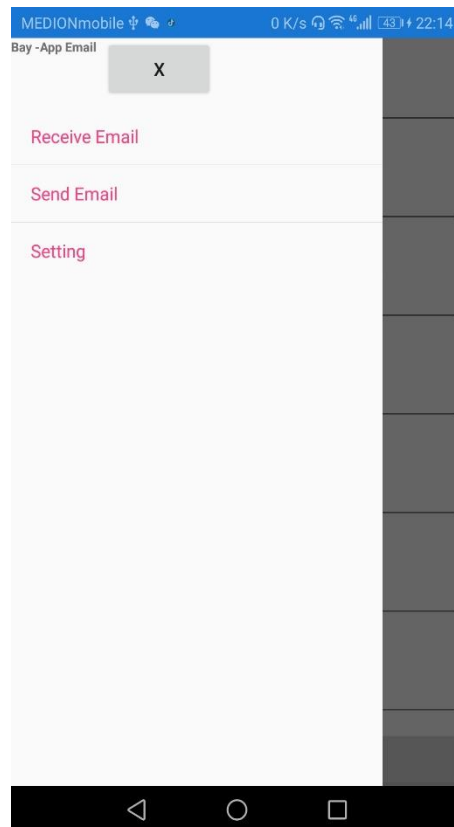


Figure 4-2-2-2 Real Android Mobile

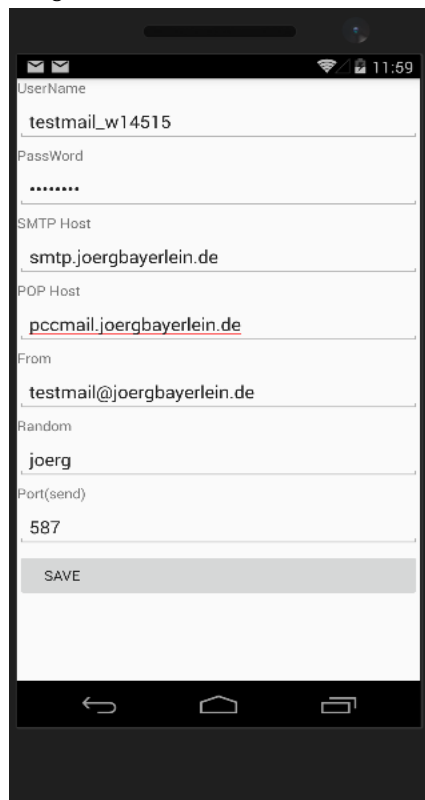


Figure 4-2-2-3 Setting Page

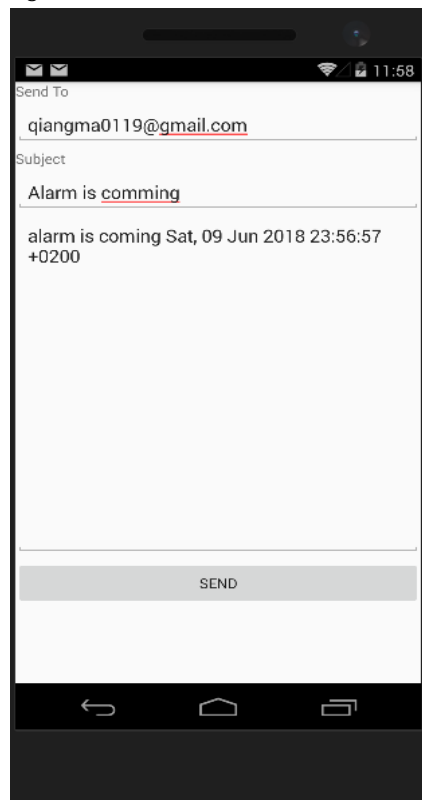


Figure 4-2-2-4 Send Page

### 4.2.3 Raspberry

After installing and debugging on the Android emulator and devices, we need to install the program on the raspberry. The solution platform should be switched to ARM (Advanced RISC Machine) and the emulator should be Remote Machine. The device can be found automatically or inputting the IP address of the device. (Figure 4-2-3-1)

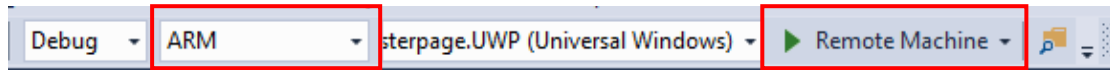


Figure 4-2-3-1 Remote Machine

After installing on the raspberry, the view of the application is similar to the Android device. Then we can start to test the email functionality of the application.

First we send an email to the raspberry, then wait for its response message and see whether the message will be shown on the raspberry.

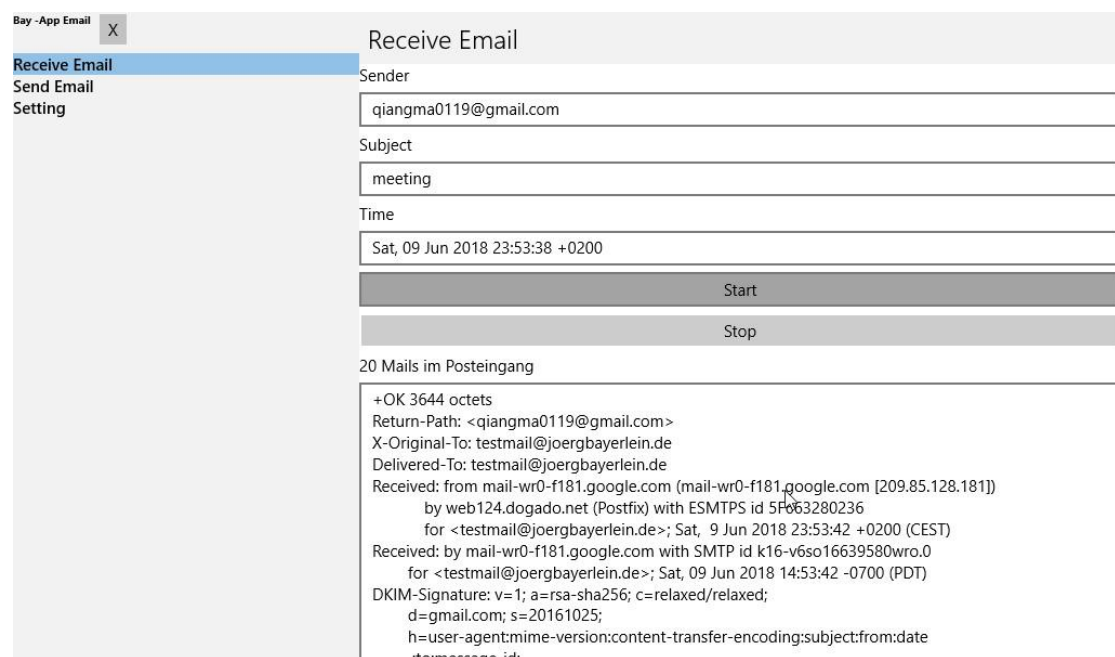


Figure 4-2-3-2 Raspberry View

I sent an email with the content of the subject “! dangerous”. Then the result of the red word dangerous was shown on the monitor (Figure 4-2-3-3) and sent back an email to to the sender.(Figure 4-2-3-4). With the confirmation of the email, the notice label will disappear.

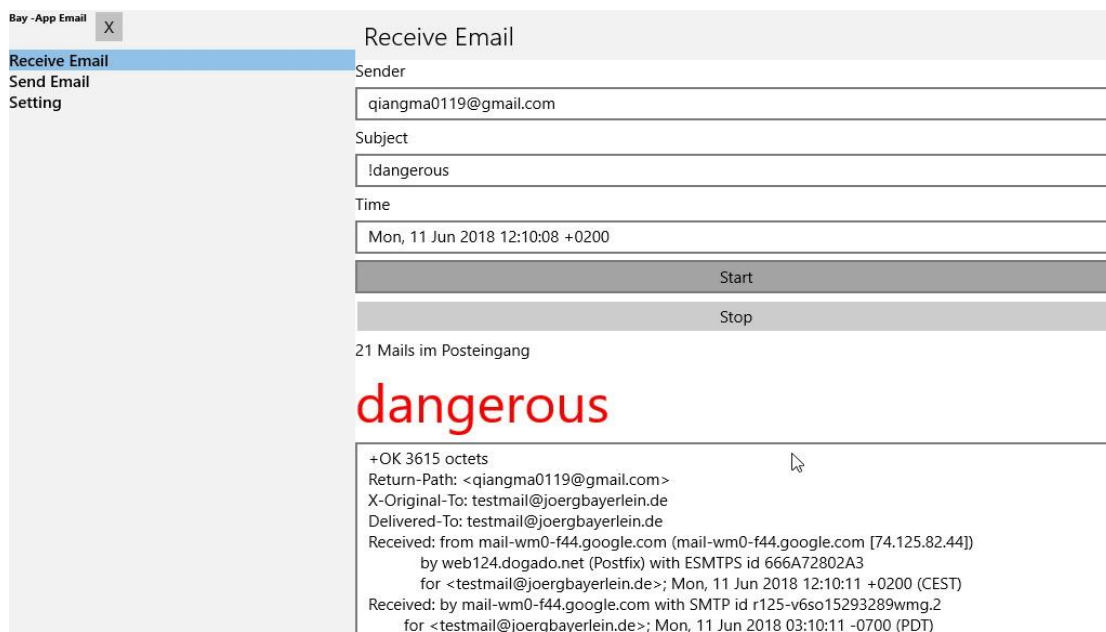


Figure 4-2-3-3 Parameter on Raspberry



Figure 4-2-3-4 Confirmation Email



#### 4.2.4 iOS

In the iOS system, the application has the same functionality (Figure 4-2-4-1 & Figure 4-2-4-2). We can also find that different operating systems has different ways to realize the layout. The menu has a little overlap of the status bar of the mobile.

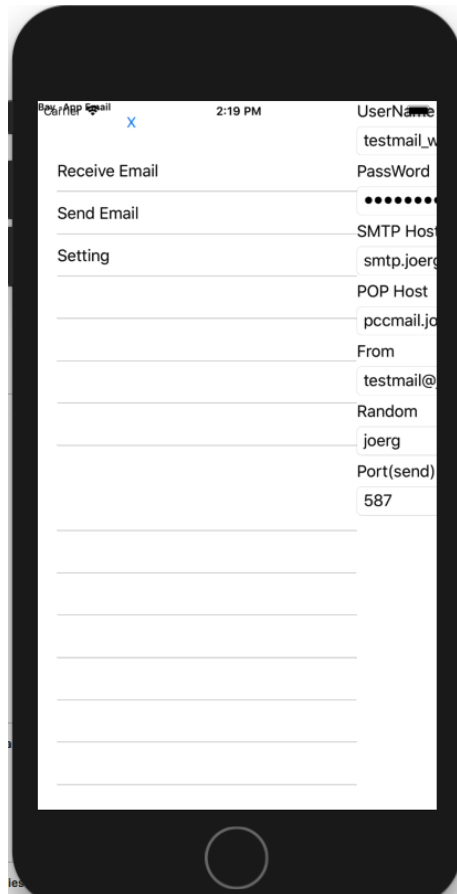


Figure 4-2-4-1 iOS Emulator

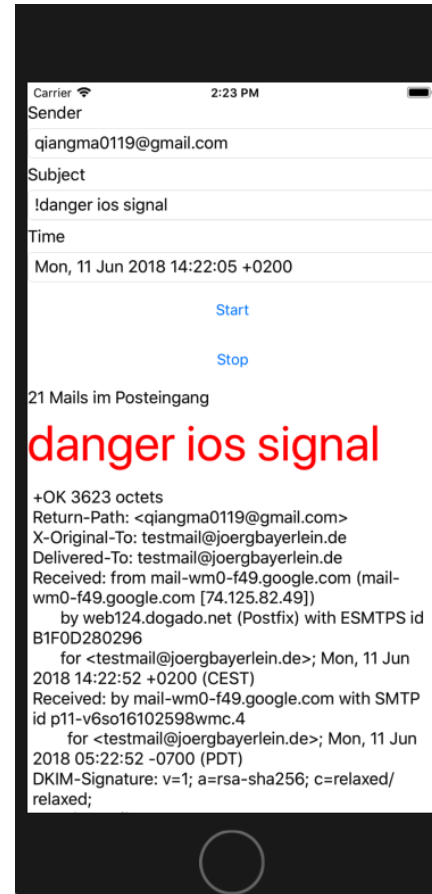


Figure 4-2-4-2 iOS Receive Parameter

## 4.3 MySQL

Last semester, the senior students had to make the connection between the MySQL database with the mobile devices by using Xamarin.Forms. Then we can have a deep look at the raspberry. The question is whether the raspberry can send and receive email from the database so that it can have a message exchange with the mobile device.

### 4.3.1 MySQL Database Programming Preparation

Professor Bayerlein gave me his database username and password, so that I could connect to his database (Figure 4-3-1-1).

The screenshot shows the MySQL Connection Wizard interface. At the top, the 'Connection Name' is 'raspi'. Below it, there are three tabs: 'Connection', 'Remote Management', and 'System Profile'. The 'Connection' tab is active. Under 'Connection Method', 'Standard (TCP/IP)' is selected. Below this, there are three sub-tabs: 'Parameters', 'SSL', and 'Advanced'. The 'Parameters' sub-tab is active. It contains the following fields: 'Hostname' (www.joergbayerlein.de), 'Port' (3306), 'Username' (NeuBay), 'Password' (with 'Store in Vault ...' and 'Clear' buttons), and 'Default Schema' (empty). To the right of these fields are descriptive labels: 'Name or IP address of the server host, - and TCP/IP port.', 'Name of the user to connect with.', 'The user's password. Will be requested later if it's not set.', and 'The schema to use as default schema. Leave blank to select it later.'

Figure 4-3-1-1 Database of Bayerlein

There is one table which is used in my application. The structure of the “RaspiComm” is described in figure 4-3-1-2. The column nb (Number), the value and two text with different data type are stored in this table. Nb is the primary key and all the columns should not be empty.

| Column Name | Datatype | PK                                  | NN                                  | UQ                       | BIN                      | UN                       | ZF                       | AI                                  | Default |
|-------------|----------|-------------------------------------|-------------------------------------|--------------------------|--------------------------|--------------------------|--------------------------|-------------------------------------|---------|
| nb          | INT(11)  | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input checked="" type="checkbox"/> |         |
| value       | DOUBLE   | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>            |         |
| text1       | TEXT     | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>            |         |
| text2       | INT(11)  | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>            |         |
|             |          | <input type="checkbox"/>            | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>            |         |

Figure 4-3-1-2 RaspiComm Table

### 4.3.2 Android & iOS plugin Installation

The structure of the table is clear. Then next is to find a method for the connection between the Xamarin.Forms and database.

From the thesis of previous student, we know that Xamarin.Forms provides a MySQL plugin used to connect remote MySQL database. The plugin named “MySQL.Data.CF” used to be downloaded from the website [14], but now the Xamarin Components has been replaced by the NuGet packages which you can find the information on the Xamarin NuGet website [15]. In this project, I used the old version of the Xamarin component. There is also a way to download from the MySQL website [16]. However, this plugin is only active for Android and iOS platform. It is necessary to find a plugin for the windows platform. Through the experience of the previous students, the plugin named” Mysql.Data.RT” is suitable for the UWP platform. It can be downloaded from this website [17].

The following steps describe how to install the plugin.

Step 1; Download the MySQL plugin mentioned above. The user need a MySQL account to download the MySQL plugin and a Xamarin account to download the Xamarin plugin.

Step 2: Open the Android and iOS project tag in the solution explorer, then right click on the “Reference” tag then click the “Add Reference” in the pop up menu (Figure 4-3-2-1). A window named “Reference Manager” will pop up. (Figure 4-3-2-2)

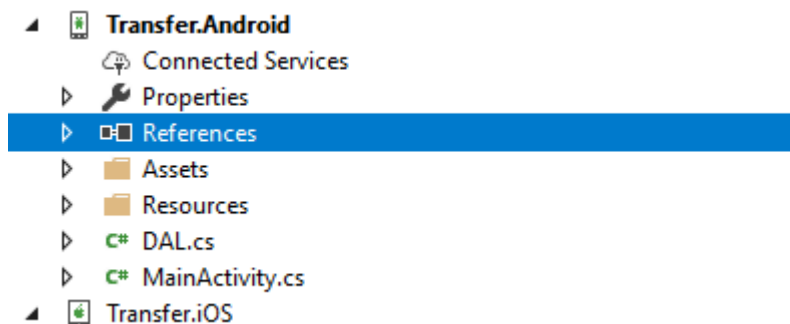


Figure 4-3-2-1 Transfer. Android

Step 3: In the Browse tag, if the MySQL plugin does not exist in the view, click the “Browse” button at the bottom of the window and find the plugin downloaded previously

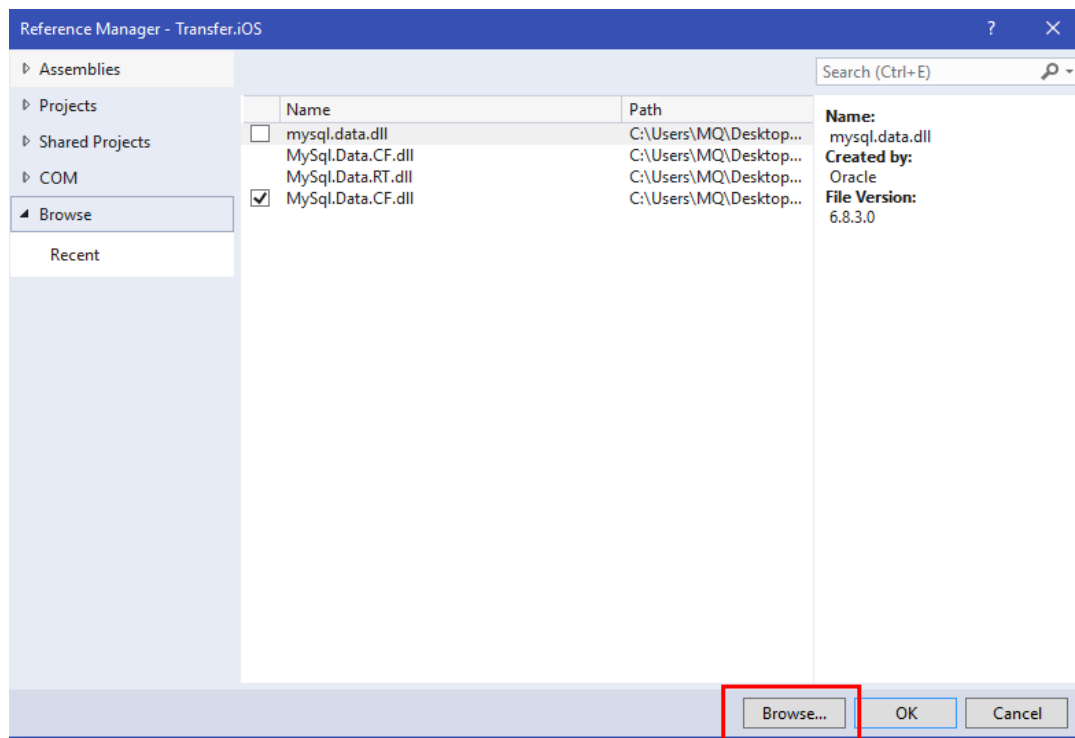


Figure 4-3-2-2 Browse of Reference Manager

Step 4: Then the plugin will appear in the view of “Reference Manager” window. Press the ok button. The plugin will be added into the reference. (Figure 4-3-2-3)

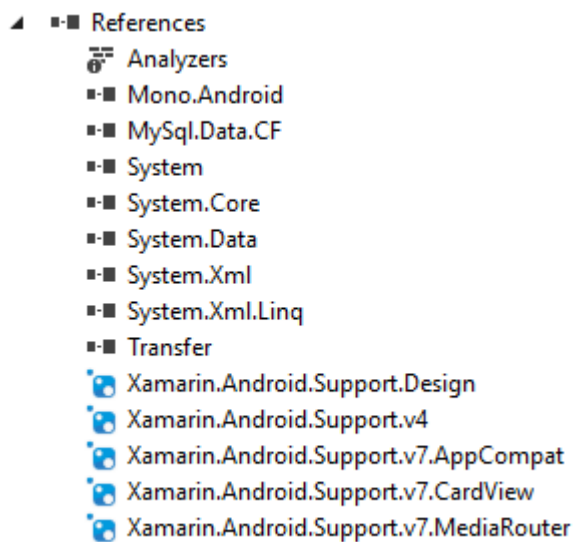


Figure 4-3-2-3 Android Reference

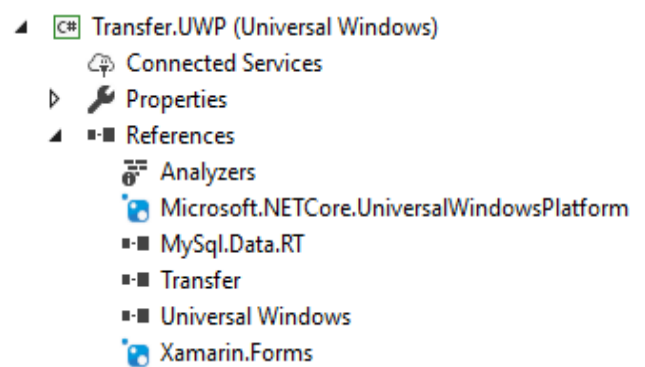


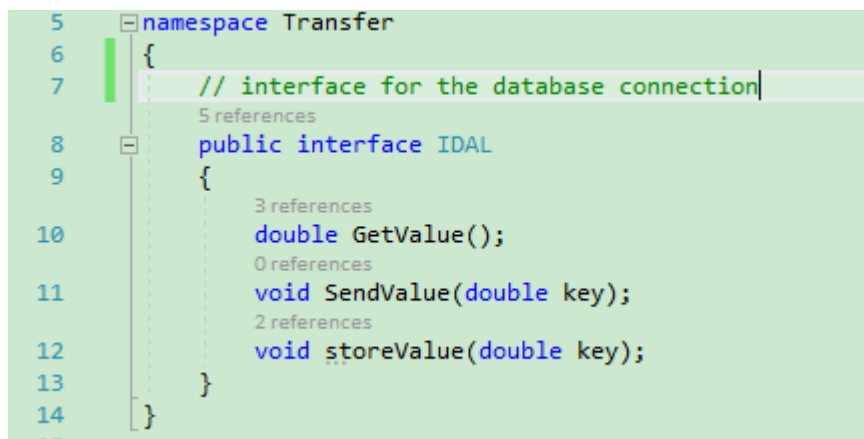
Figure 4-3-2-4 UWP Reference

For the UWP project, the steps are the same. Just change the plugin to “MySQL.Data.Rt” (Figure 4-3-2-4).

### 4.4.3 Views and functions based on MySQL Database

This section describes the functions used in the database and how to transfer the data realized in the C# code with the designed view.

First thing we need to know that different platforms in the Xamarin use different ways to realize the platform-specific function, so the code of connection to the database can not be written in the PCL project. Therefore, I used the “Dependency Service” provide by the Xamarin.Forms. So a public interface named “IDAL” was created for different platforms to call (Figure 4-4-3-1). A class named “DAL” is created in each platform project. The content of the class is almost the same except the namespace, assembly declarations and plugins they use (Figure 4-4-3-2).



```

5 namespace Transfer
6 {
7     // interface for the database connection
8     public interface IDAL
9     {
10         double GetValue();
11         void SendValue(double key);
12         void storeValue(double key);
13     }
14 }

```

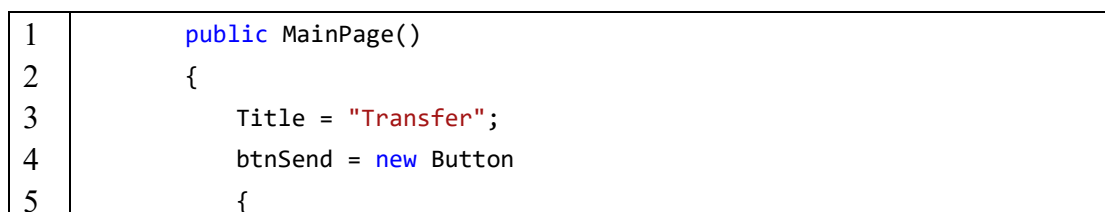
Figure 4-4-3-1 Interface IDAL

Table 4-4-3-2 Dependency Service

|   |  |
|---|--|
| 1 | [assembly: Dependency(typeof(Transfer.Droid.DAL))] |
| 2 | [assembly: Dependency(typeof(Transfer.iOS.DAL))]   |
| 3 | [assembly: Dependency(typeof(Transfer.UWP.DAL))]   |

The main project can be divide into two parts: design part and function part. Design part is responsible for creating the buttons, labels, entries and slider. The function part is responsible for the searching, getting and storing the data.

The MainPage.Xaml.cs file is used for design the layout. For example the button named “btnSend” is used to send the value stored in the Entry to the database (Code 4-4-3-3).



```

1 public MainPage()
2 {
3     Title = "Transfer";
4     btnSend = new Button
5     {

```

```

6      Text = "Send",
7      HorizontalOptions = LayoutOptions.Start,
8      TextColor = Color.Accent,
9      BorderWidth = 1,
10     BorderRadius = 10,
11     BorderColor = Color.Accent,
12 };
13
14     btnSend.Clicked += Button_SendClicked;
15
16
17     void Button_SendClicked(object sender, EventArgs args)
18     {
19
20         double num = Convert.ToDouble(etSend.Text);
21         key.storeValue(num);
22     }

```

Code 4-4-3-3 Send Button

The code block shows the definition of the button and the attached behavior with the click action. The behavior of send will take place when the button is clicked. The button also has many properties such as “Text”, “TextColor”, “BorderWidth” to define himself.

The button named “btnGet” is used to get the value from the database, the start and stop button is used to control the run of the timer. For the reason that the communication should be as fast as possible like real-time. The application should look for the value stored in the database per second, if the value has changed, the latest data should appear in the value field. This is realized by the method Device.StartTimer (Code 4-4-3-4)

```

1 Device.StartTimer(TimeSpan.FromMilliseconds(1000), () =>
2     {
3         if(isUse)
4         {
5             etGet.Text = key.GetValue().ToString();
6             // True = Repeat again, False = Stop the timer});
7         }
8         return true;
9     });

```

Code 4-4-3-5 Device StartTimer

The Device.StartTimer is the method provided by Xamarin.Forms itself, so it is used to use and the Boolean value isUse is to control the start and the off of the timer. The value 1000 means the interval of every check. If isUse is true, the entry of the text will show the value getting from the database.

There is also a slider in the view, when the slider moves, the value will change accordingly. It uses “ValueChanged” property of the slider attaching a behavior to it. (Code 4-4-3-6)

```

1      Slider MySlider = new Slider
2      {
3          Minimum = 0,
4          Maximum = 100,
5
6      };
7      MySlider.ValueChanged += (send, args) =>
8      {
9
10
11         double num = MySlider.Value / 20.0;
12         MyLabel.Text = num.ToString();
13         double sliderValue = Convert.ToDouble(MyLabel.Text);
14         key.storeValue(num);
15         etGet.Text = key.GetValue().ToString();
16     };

```

Code 4-4-3-6 Slider Change

Because the slider in Xamarin.Forms can change in a very little range. So I enlarge the value range of the slider and divide it by 20, and the range will change from 0-100 to 0-5. Assembling all these components, a simple view of the application of Windows platform will come out (Figure 4-4-3-7). The Android view and the iOS view also complete (Figure 4-4-3-8 & Figure 4-4-3-9)



Figure 4-4-3-7 Windows Transfer

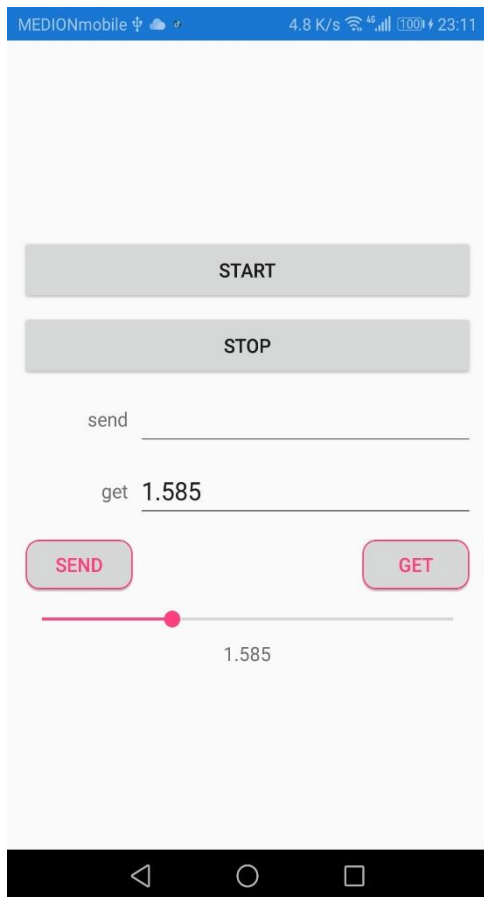


Figure 4-4-3-8 Android Transfer



Figure 4-4-3-9 iOS Transfer

When I change the slider on my mobile phone, the get entry on the raspberry will also change to the value accordingly. This is also a way to realize the communication by the database on the server.

The second part is the process of database function. There are three methods declaration in the interface “IDAL”. They are “GetValue”, “StoreValue”, “SendValue”. First thing is to establish the connection between the database and the application (Code 4-4-3-10).

```

1 System.Text.EncodingProvider ppp;
2 ppp = System.Text.CodePagesEncodingProvider.Instance;
3 Encoding.RegisterProvider(ppp);
4 string connsqstring = "Server='www.joergbayerlein.de';Port=3306;User
5 Id=NeuBay;Password=XXXXXX#;Database=w14515_locals2;charset=utf8";
6 MySqlConnection conn = new MySqlConnection(connsqstring);
7
8 conn.Open();
9
10
```

Code 4-4-3-10 Connection of database



The `connsqlstring` contains the online server, id of user, password, the name of database. Only if they are correct, the connection of the database will be successful. Then will be the realization of each function. Take the `GetValue()` function as a example (Code 4-4-3-11).

```

1  string sql = "SELECT value from RaspiComm where nb = (select max(nb)
2  from RaspiComm)";
3      MySqlCommand cmd = conn.CreateCommand();
4      cmd.CommandText = sql;
5      MySqlDataReader rd = cmd.ExecuteReader();
6      double result = 0;
7      while (rd.Read())
8      {
9          result = rd.GetDouble("value");
10     }
11     conn.Close();
12     return result;

```

Code 4-4-3-11 Get Function

Using the sql statement to select the latest data from the database, then command the statement and read from the data reader, finally store the value in the result.

The method “StoreValue” is similar to the structure of the method “GetValue” (Code 4-4-3-12).

```

1  string sql = "SELECT value from RaspiComm where nb = (select max(nb)
2  from RaspiComm)";
3      MySqlCommand cmd = conn.CreateCommand();
4      cmd.CommandText = sql;
5      MySqlDataReader rd = cmd.ExecuteReader();
6      double result = 0;
7      while (rd.Read())
8      {
9          result = rd.GetDouble("value");
10     }
11     conn.Close();
12     return result;

```

Code 4-4-3-12 Store Value Function

Through this method, we can have a control of the data and realize the function of transferring parameter. For example, I changed the value of the slider of the application installed on the Android mobiles to 1.685. Then after several seconds, the raspberry will refresh the get value to 1.685 (Figure 4-4-3-13). This also applies to the iOS mobiles.

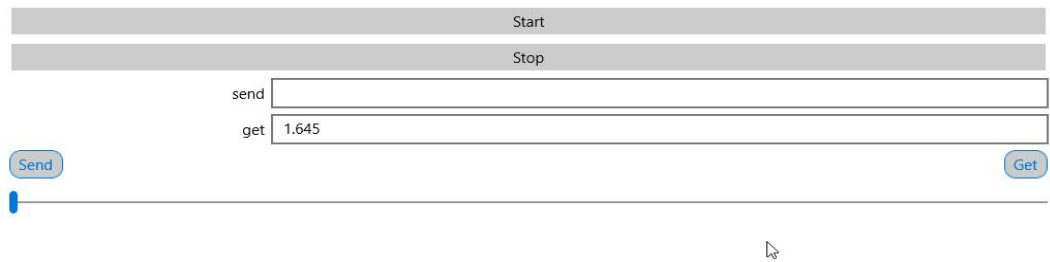


Figure 4-4-3-13 Raspberry Transfer

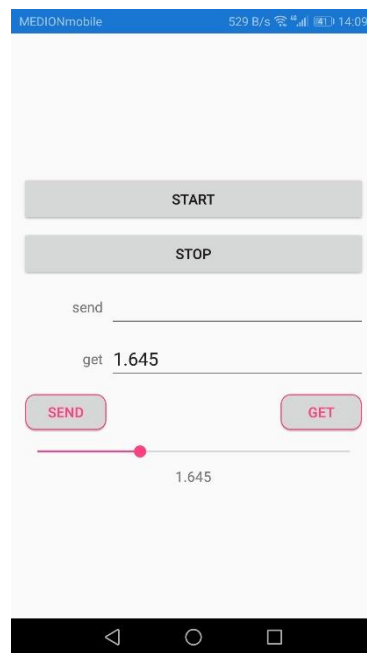


Figure 4-4-3-14 Android Transfer

## Chapter 5 Conclusion and outlook

### 5.1 The project of Email and MySQL

Creating an email program is a foundation for searching the way of communication between the raspberry and mobile devices. This method may not be new to a developer, but for the reader and the beginner, it is a good way to realize. Although it is not real-time communication, it can help the beginner to have a deep look of the structure and how the Xamarin.Forms work.

The email program can also be improved to send the attachment or pictures. The basic structure has completed, it is much easier to add external functions. The disadvantage is also obvious. The respond time is much longer than expected and this is the biggest problem for the email solution.

Using the online database to be an intermediate to transfer the parameter or message is also feasible. The database will have a faster response comparing to the email program. It can also save every data when the value changes and display the data in an arranged way and store as much value as possible. While the response speed can also not be satisfied. The application need to look for the change per second so that it may cause the network traffic.

The Xamarin.Forms has many obvious advantages, it has an easier way to arrange the code suitable for different platforms. But it also has some problems. The biggest problem will be the compatibility. Due to the reason that Xamarin.Forms is relative new to the developer, it updates frequently. Therefore, the application you program today may not be used tomorrow. It will cost some resource on maintaining the application. And the plugin library for the Xamarin still needs to be improved.

### 5.2 Reflection

A better way is to make the raspberry or mobile device set as a client. For the client, it can check the changes of the mobile device itself, then it can make the response very fast. As a result, it may need some programming on the server side. Professor Bayerlein gave me the suggestion about using Node.js to create webpage from his colleague Professor Hanemann. Due to the limitation of the time, I have done some research about the Node.js on the internet. The Node.js runs in a JavaScript runtime environment and process synchronously, which is the biggest difference from other server programming language.

To myself, in the whole process of writing thesis, I think I could prepared much better. The time management will run smoothly, the last several weeks are a little bit hurry. If

I can prepare the thesis as early as possible, the progress of the work will much larger.

## **5.2 Outlook**

As for the thesis, the topic of my thesis is about to find the limitation and possibilities of the Raspberry. Through the projects of email and database, the raspberry has the ability to send and receive emails and transfer data with the database. Although I have not found a real-time solution which is fast enough for the real-time communication, I prove that the real-time communication is reachable and it can work fine as the application on the windows form.

## Acknowledgement

Foremost, I would like to express my sincere gratitude to my supervisor Prof. Bayerlein for providing me with all the necessary facilities for the development, and continuous support and suggestion of my project.

I place on record, my sincere thank you to M.Sc. Hanesova and Professor Hanemann, for sharing expertise, and sincere and valuable guidance and encouragement extended to me.

I am also grateful to my partner Qiwen Gu and Linhui Yang, for the nice teamwork and great advice.

I also place on record, my sense of gratitude to one and all, who directly or indirectly, have lent their hand in this period.

## Appendix A – List of figures

|   |    |
|---|----|
| Figure 3-1-1 Workloads ---- Windows Configuration .....           | 6  |
| Figure 3-1-2 Workloads----Mobile & Gaming Configuration .....     | 7  |
| Figure 3-1-3 Individual components---Emulator Configuration ..... | 7  |
| Figure 3-2-1 Raspberry Pi 3 Model B [7] .....                     | 8  |
| Figure 3-2-2 IoT Dashboard .....                                  | 9  |
| Figure 3-2-3 IoT Device .....                                     | 9  |
| Figure 3-3-1 Visual Studio Android Emulator .....                 | 10 |
| Figure 3-3-2 Android SDK Manager Platform .....                   | 11 |
| Figure 3-3-3 Android SDK Manager Tools .....                      | 11 |
| Figure 3-4-1 Workbench Connection Setting .....                   | 12 |
| Figure 4-1-1 Package Manger Console.....                          | 13 |
| Figure 4-1-2 Messaging project .....                              | 14 |
| Figure 4-1-3 IEmailTask .....                                     | 14 |
| Figure 4-1-4 Usage of API Email .....                             | 14 |
| Figure 4-1-5 iOS View .....                                       | 15 |
| Figure 4-1-6 Android View .....                                   | 15 |
| Figure 4-1-7 Raspberry View of Plugin.....                        | 15 |
| Figure 4-1-8 Android Mobile View .....                            | 16 |
| Figure 4-1-9 Letter from Xamarin Plugin.....                      | 16 |
| Figure 4-2-2-1 Android Emulator .....                             | 19 |
| Figure 4-2-2-2 Real Android Mobile .....                          | 19 |

|  |    |
|--|----|
| Figure 4-2-2-3 Setting Page .....                | 19 |
| Figure 4-2-2-4 Send Page .....                   | 19 |
| Figure 4-2-3-1 Remote Machine.....               | 20 |
| Figure 4-2-3-2 Raspberry View .....              | 20 |
| Figure 4-2-3-3 Parameter on Raspberry .....      | 21 |
| Figure 4-2-3-4 Confirmation Email .....          | 21 |
| Figure 4-2-4-1 iOS Emulator .....                | 22 |
| Figure 4-2-4-2 iOS Receive Parameter.....        | 22 |
| Figure 4-3-1-1 Database of Bayerlein.....        | 23 |
| Figure 4-3-1-2 RaspiComm Table .....             | 23 |
| Figure 4-3-2-1 Transfer. Android.....            | 24 |
| Figure 4-3-2-2 Browse of reference manager ..... | 25 |
| Figure 4-3-2-3 Android Reference .....           | 25 |
| Figure 4-3-2-4 UWP Reference .....               | 25 |
| Figure 4-4-3-1 Interface IDAL .....              | 26 |
| Figure 4-4-3-13 Raspberry Transfer .....         | 31 |
| Figure 4-4-3-14 Android Transfer.....            | 31 |
| Figure 4-4-3-7 Windows Transfer.....             | 28 |
| Figure 4-4-3-9 iOS Transfer.....                 | 29 |
| Figure 4-4-3-8 Android Transfer.....             | 29 |

## Appendix B – List of tables

|  |    |
|--|----|
| Table 4-2-1-1 Class of System.Net.Mail.....    | 17 |
| Table 4-2-1-2 Class of System.Net.Sockets..... | 17 |
| Table 4-4-3-2 Dependency Service.....          | 26 |

## Appendix C – List of code

|   |    |
|---|----|
| Code 4-2-1-1 System.Net.Mail .....        | 17 |
| Code 4-2-1-2 System.Net.Sockets .....     | 17 |
| Code 4-2-1-3 Read Mail .....              | 18 |
| Code 4-2-1-4 Notice Label .....           | 18 |
| Code 4-4-3-3 Send Button .....            | 27 |
| Code 4-4-3-5 Device StartTimer.....       | 27 |
| Code 4-4-3-6 Slider Change .....          | 28 |
| Code 4-4-3-10 Connection of database..... | 29 |
| Code 4-4-3-11 Get Function .....          | 30 |
| Code 4-4-3-12 Store Value Function .....  | 30 |

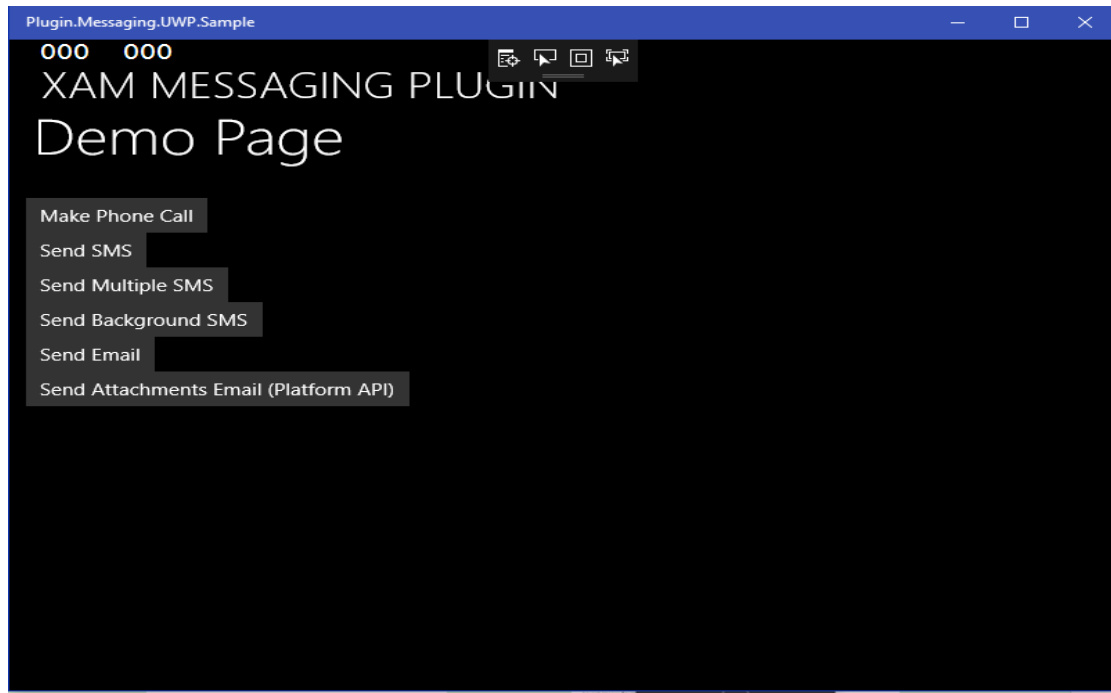
## Appendix D – Contents of USB stick

- Thesis paper (PDF file):"MaQiang\_Thesis.pdf"
- Thesis paper (Word file):"MaQiang\_Thesis.docx"
- Xamarin app of Xamarin Plugin : "Xamarin.Plugin.zip"
- Xamarin app of Email: "Masterpage.zip"
- Xamarin app of MySQL: "Transfer.zip"
- Database export: "mysql-workbench20180613.sql"
- Readme.txt



- Application extension files for MySQL: “MySql.Data.dll” and “MyAql.Data.rt.dll”
- Username and password:” Username.zip”

## Appendix E – Name of the component



Make Phone Call ----- MakePhoneCall()  
 Send Attachments Email (Platform) --- SendHtmlEmail()  
 Send SMS ----- SendSms()  
 Send Multiple SMS ----- SendMultipleSms()  
 Send Email ----- SendEmail()

Bay -App Email X

Receive Email

Send Email

Setting

label: sender

qiangma0119@gmail.com

etSender

Subject

!dangerous

etSub

Time

Mon, 11 Jun 2018 12:10:08 +0200

etTime

Start

Stop

21 Mails im Posteingang

dangerous

label: notice

etField1

+OK 3615 octets  
Return-Path: <qiangma0119@gmail.com>  
X-Original-To: testmail@joergbayerlein.de  
Delivered-To: testmail@joergbayerlein.de  
Received: from mail-wm0-f44.google.com (mail-wm0-f44.google.com [74.125.82.44])  
by web124.dogado.net (Postfix) with ESMTPS id 666A72802A3  
for <testmail@joergbayerlein.de>; Mon, 11 Jun 2018 12:10:11 +0200 (CEST)  
Received: by mail-wm0-f44.google.com with SMTP id r125-v6so15293289wmg.2  
for <testmail@joergbayerlein.de>; Mon, 11 Jun 2018 03:10:11 -0700 (PDT)

Receive Email ----- pt1  
Send Email ----- pt2  
Setting ----- pt3

label: userName

testmail\_w14515

Entry: etUser

label: passWord

.....

Entry: etPassword

label: smtpHost

smtp.joergbayerlein.de

Entry: etSmtip

label: popHost

pccmail.joergbayerlein.de

Entry: etPop

label: from

testmail@joergbayerlein.de

Entry: etFrom

label: random

joerg

Entry: etFromName

label: port

587

Entry: etPort

SAVE

Button: btnSavingSetting

label: sendTo

qiangma0119@gmail.com

Entry: etSendto

label: userName

Subject

Alarm is coming

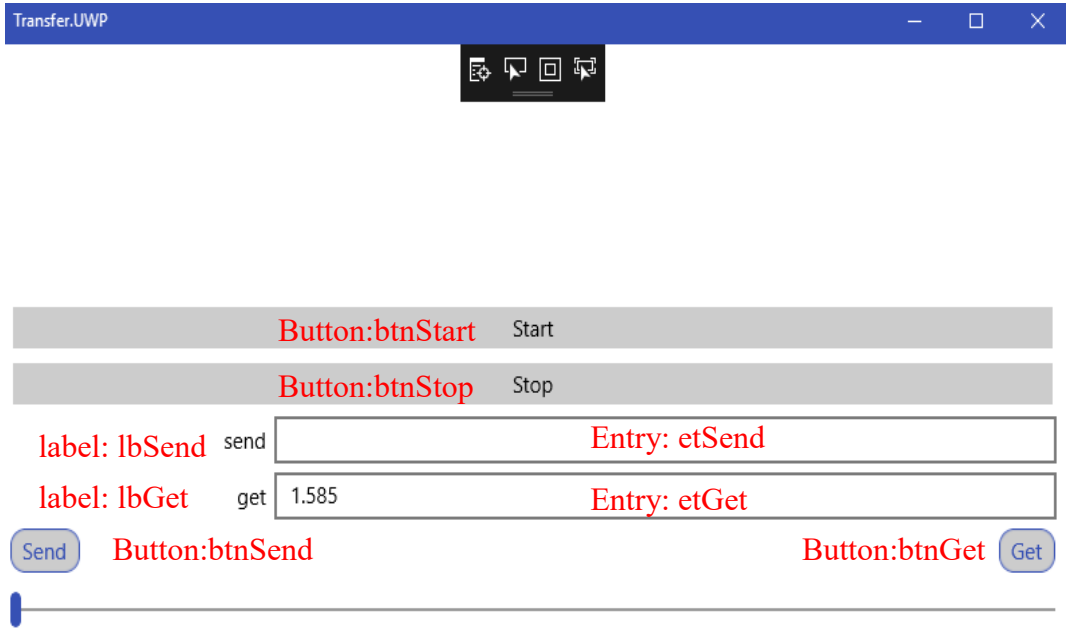
Entry: etSub2

alarm is coming Sat, 09 Jun 2018 23:56:57 +0200

Editor: etField2

SEND

button: btnSend



## Bibliography

- [1] Wikipedia, Visual Studio introduction. Retrieved from  
URL: [https://en.wikipedia.org/wiki/Microsoft\\_Visual\\_Studio](https://en.wikipedia.org/wiki/Microsoft_Visual_Studio)
- [2] Wikipedia, Visual Studio .NET. Retrieved from  
URL: [https://en.wikipedia.org/wiki/Microsoft\\_Visual\\_Studio](https://en.wikipedia.org/wiki/Microsoft_Visual_Studio)
- [3] Wikipedia, C#. Retrieved from  
URL: [https://en.wikipedia.org/wiki/C\\_Sharp\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/C_Sharp_(programming_language))  
[Accessed: 1<sup>st</sup> June 1, 2018]
- [4] Petzold, C., “How does Xamarin.Forms fit in? “in *Creating Mobile Apps with Xamarin.Forms*, Redmond: Microsoft Press, 2016, pp. 2-5
- [5] Microsoft Visual Studio. Retrieved from:  
URL: <https://www.visualstudio.com/>
- [6] Raspberry Pi 3, Waveshare electronics. Retrieved from  
URL: <http://www.waveshare.net/aspx/search.aspx?keywords=tag-RPi>
- [7] Waveshare electronic, *Front view of raspberry pi 3*, Retrieved from  
URL: <http://www.waveshare.net/photo/development-board/RPi3-B/RPi3-B-1.jpg>
- [8] YouTube (08.01.2017), *Waveshare 10.1 inch HDMI LCD (H)*. Retrieved from  
URL: <https://www.youtube.com/watch?v=zzDIFzCuEws>
- [9] Microsoft, Windows IoT Dashboard. Retrieved from  
URL:  
<https://docs.microsoft.com/en-us/windows/iot-core/connect-your-device/iotdashboard>
- [10] Microsoft, Visual Studio Emulator. Retrieved from  
URL: <https://www.visualstudio.com/vs/msft-android-emulator/>,
- [11] ORACLE, MySQL, Retrieved from  
URL: <https://www.mysql.com/products/workbench/>
- [12] Haocheng Liu, (2017), “*Thesis paper Haocheng\_Liu.pdf*”, (Unpublished master’s thesis). Fachhochschule luebeck, Luebeck, Germany.  
URL: <https://lernraum.fh-luebeck.de/mod/folder/view.php?id=77753>
- [13] Cjlotz & JamesMontemagno, Xam.Plugins.Messaging (5.2.0), Retrieved from  
URL: <https://www.nuget.org/packages/Xam.Plugins.Messaging>

- [14] Microsoft, Updating component references to NuGet, Retrieved from  
URL: <https://docs.microsoft.com/en-us/xamarin/cross-platform/troubleshooting/component-nuget?tabs=vswin>
- [15] nuget, MySql.Data (8.0.11), Retrieved from  
URL: <https://www.nuget.org/packages/MySql.Data/>
- [16] MySQL, Download Connector/Net Retrieved from  
URL: <https://dev.mysql.com/downloads/connector/net/>
- [17] Oracle, mysql.data.rt.dll 6.8.3.0, Retrieved from  
URL: <http://api.256file.com/mysql.data.rt.dll/m-download-420176.html>