

DESIGN AND IMPLEMENTATION OF A CONNECTIVITY TOOL APPLICATION FOR
DIGITAL TRANSFORMATION IN INSTITUTIONS IN THE DEVELOPING AND UNDER-
DEVELOPED COUNTRIES.

A FINAL CAPSTONE PROJECT SUBMITTED TO THE
DEPARTMENT OF COMPUTER SCIENCE
HOFSTRA UNIVERSITY

BY

OLUWATOBI KAREEM

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE
IN THE DEMATTEIS SCHOOL OF ENGINEERING AND APPLIED SCIENCES

SUPERVISED BY: PROF. OREN SEGAL

Title of Project: Design and Implementation of a Connectivity Tool Application for Digital Transformation in Institutions in the Developing and Under-Developed Countries.

Submitted by: Oluwatobi Kareem

Approval of the Fred DeMatteis School of Engineering and Applied Science – Graduate Department of Computer Science.

Prof. Krishnan Pillaipakkamnatt
Chair, Department of Computer Science

Approval of Supervisor.

Prof. Oren Segal
Project Supervisor

STATEMENT OF NON-PLAGIARISM

I hereby confirm that this project is my own work based on my personal study and/or research. It is not copied from any other person's work except for quotations from published or unpublished sources which are clearly cited and referenced as such. I am conscious, and I have confirmed that I read and understood the University's policy and regulations on plagiarism.

Name: Oluwatobi Kareem

Signature:

Date: May 2, 2018

ABSTRACT

TobChat is a connectivity tool application that connects to a network in an institution to provide access, connections and resources to other applications that are connected to the same network without the need for the internet access. The scope of this project solves the inefficient internet connectivity problem in schools in the developing and the under-developed countries. Deploying this project will radically change educational experience in institutions by transforming the way educational information is been disseminated.

Keywords: Networking Technology, Local Area Network, File System, Mobile Application development, Web Application development.

OLUWATOBI KAREEM

M.Sc. Computer Science

SUPERVISOR: PROF. OREN SEGAL

86 Pages

13,146 words

MAY 2018

ACKNOWLEDGEMENT

First and foremost. I am grateful to God, the author and finisher of my faith, for I have sought him for Wisdom, Knowledge and Understanding during this project.

I would like to thank my father, for so expertly guiding all aspects of my career with unprecedented fervor, energy and personal care. For his resolute dedication to my success, and my mother for prayers and motivational speech, I am eternally grateful.

To Prof. Krishnan Pillaipakkamnatt, Dr. Edward Currie, Prof. Oren Segal, Prof. Steven Lindo, Daksha Shah and Oladoyin Olajide. The brains that combine to make my brain appear much larger than it is. Your contributions helped me during this project. Thank you for your helpful advises.

To my friends and colleagues who may have felt neglected at any time during coding and writing this project, I hope you find the output of this work justified my absence. Every day, you've been in my heart.

To the indefatigable Akinkintan Awelewa, thanks for the many four a.m. calls that praises my efforts and pushes me to sleep sometimes.

TABLE OF CONTENTS

STATEMENT OF NON-PLAGIARISM	iii
ABSTRACT.....	iv
ACKNOWLEDGEMENT	v
TABLE OF FIGURES.....	x
LIST OF ABBREVIATIONS.....	xii
CHAPTER 1 - INTRODUCTION.....	1
1.1 Problem Statement	2
1.2 The Purpose of the Research.....	3
CHAPTER 2 – LITERATURE REVIEW	5
2.1 Networking.....	5
2.1.1 Time Sharing – the first online communities - 1960	6
2.1.2 Multiplexers: Getting more users into the same line – 1968	6
2.1.4 Inventing the Internet – 1973.....	7
2.1.5 Linking the Ethernet and Local Networks	8
2.2 Application Software.....	8
2.2.1 Blackboard, 1997	10
2.2.2 Skype, 2003.....	10
2.2.3 Facebook, 2004.....	11

2.2.4	Gmail, 2004.....	12
2.2.5	GroupMe, 2010	12
2.2.6	GitHub, 2008.....	13
2.3	Database	13
2.4	Discussion	14
CHAPTER 3 – METHODOLOGY		17
3.1	Sql Database File System.....	17
3.1.1	Connecting to SQL Server	18
3.1.2	Creating the Database	19
3.1.3	The System Administrator Roles	21
3.1.4	Database Description	21
3.1.5	Database Logical Structure	22
3.2	Raspberry Pi 3 Raspbian	24
3.2.1	Setting Up the Raspberry Pi as an access point for a LAN	25
3.2.1.1	The Steps	25
3.3	Node.js.....	29
3.3.1	Remote Connection to SQL Server.....	30
3.4	ASP.NET MVC.....	32
3.4.1	Models.....	32
3.4.2	Views	32
3.4.3	Controllers.....	32

3.4.4	MVC (TobChatWeb) Folder Structure	33
	App_Data:	34
	App_Start:	34
	Content:	35
	Controllers:	35
	fonts:	35
	Models:	35
	Scripts:	36
	Views:	36
	Global.asax:	37
	Packages.config:	37
	Web.config:	38
3.5	Xamarin.Forms.....	38
3.5. 1	Xamarin.Forms (FinalCap) Folder Structure.	39
	Dependencies:	41
	Model:	41
	Services:	41
	View:	41
	ViewModel:	41
	App.Xaml / App.Xaml.cs	41
	FinalCap.Android:	41

FinalCap.iOS:	42
FinalCap.UWP (Universal Windows):	43
3.5.2 Testing the Mobile Applications.....	44
CHAPTER 4 – RESULTS AND ANALYSIS.....	48
4.1 Technologies Chosen to Design and Develop the System.....	48
4.1.1 Microsoft SQL Server 2014 Management Studio	48
4.1.2 Raspberry Pi for LAN Network.....	61
The Test!.....	62
4.1.3 Node.js Server.....	62
4.1.4 ASP.NET MVC Web Development	63
4.1.5 Xamarin. Forms Mobile Applications	65
4.2 Result Analysis.....	67
4.3 System Requirement for Mobile Application Use	68
4.3.1 Android Mobile Devices.....	68
4.3.2 iOS Mobile Devices.....	68
4.3.3 Windows Phone Devices	68
CHAPTER 5 - CONCLUSION	69
5.1 Recommendations	69
5.2 Limitations	70
References.....	71

TABLE OF FIGURES

Figure 2.1 A User on the Dartmouth Time-Sharing System. (Timeline of Computer History, 2018)	6
Figure 2.2 The Problem Analysis. (Pfleeger, 1991)	9
Figure 2.3 The Solution Synthesis. (Pfleeger, 1991)	9
Figure 2.4 How a database management system and a database server relate.	14
Figure 3.1 SQL Server Management Studio (SSMS).....	18
Figure 3.2 SSMS connects to Database Server.....	19
Figure 3.4 Use case for the admin roles.....	21
Figure 3.5 An E-R diagram for the database.	22
Table 3.1 Student table	23
Table 3.2 Course table	23
Table 3.3 Teacher table.....	24
Table 3.4 Department table.....	24
Figure 3.6 A Raspberry pi.....	24
Figure 3.8 The MVC architecture illustration.....	33
Figure 3.9 TobChatWeb MVC folder structure.....	34
Figure 3.10 The TobChatWeb MVC fonts folder.....	35
Figure 3.11 The TobChatWeb MVC Models folder.....	36
Figure 3.12 The TobChatWeb MVC Views folder	37
Figure 3.16 FinalCap Xamarin.Forms Project folder structure	40
Figure 3.17 FinalCap.Android MainActivity class.....	42
Figure 3.18 FinalCap.iOS AppDelegate class	43
Figure 3.20 FinalCap Android Mobile Application Project	45

Figure 3.21 FinalCap iOS Mobile Application Project	46
Figure 3.22 FinalCap Windows Phone Mobile Application Project	47
Figure 4.1 The main interface of the SQL Server Management Studio	49
Figure 4.4 Result displaying Student record after insertion	58
Figure 4.5 Using the raspberry pi to create a host access point.	62
Figure 4.6 Node.js remote connection to file system (Course data table)	63
Figure 4.7 TobChatWeb Application Home Page	64
Figure 4.8 TobChatWeb Create New Student Web Page for Admin	64
Figure 4.9 Sign in screen on Android device emulator	65
Figure 4.10 List of Courses screen on iOS device simulator.....	66
Figure 4.11 Course History screen on Windows Phone device emulator.....	67

LIST OF ABBREVIATIONS

ARPA.....	Advanced Research Projects Agency
CMS	Content Management System
DBMS	Database Management Studio
DHCP	Dynamic Host Configuration Protocol
DSL	Digital Subscriber Line
IEEE.....	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
ISP	Internet Service Provider
LAN	Local Area Network
MVC	Model View Controller
NPL.....	Natural Language Processing
OS	Operating System
SDK.....	Software Development Kit
SQL.....	Structured Query Language
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UI	User Interface
URL.....	Uniform Resource Locator

CHAPTER 1 - INTRODUCTION

The internet which is a ‘global communication of networks consisting of thousands of other networks that are interconnected by fiber optic cables’, has become part of the daily life of our society. The use of the internet has become very popular and eLearning is reshaping the way education is being acquired. In essence, the purpose of the internet is making lives easier by connecting to one another and sharing important documents, files and other resources. The internet has its origins in the 1960s and have gone through transformation and somewhat revolution. People have been adding to the Internet ever since it was invented. However, no single individual, group or company owns or control all the hardware that connects to the internet. Anyone can use the internet provided they have a compactible device and connection. While this technology is made a free and neutral resource for individuals globally, it has fallen short in providing an efficient accessibility where data can be transferred, and where connectivity is required in some regions of the world.

To use the internet, an individual or institution must have a connection that is provided by the ISP – Internet Service Providers. The ISP company connects their customers to these global networks of networks known as the internet using a data transmission technology appropriate for delivering Internet Protocol datagrams such as dial-up, DSL, cable modem or dedicated high-speed interconnects. Often, the ISP companies charge their customers and are responsible for the speed and the efficiency of the internet. Since these costs can sometimes get too expensive, some schools are not able to fully integrate and serve their students with the internet access. They are boycotted resulting in the unavailability of connectivity in schools through the internet. And because of this problem of limited or no internet access for communication and transfer, the course documents are not sufficiently disseminated among students, eLearning is lagging in their institutions and the benefits internet has to offer is not accessible in these educational institutions.

The Internet connectivity in an educational institution plays a critical role in education and learning. Huge amount of information is passed through for learning enhancement and other educational practices, which demands more sophisticated and efficient medium of communication and connectivity. If we look at a university

setup in the developed countries, we can see that an efficient internet connection is inevitable in their activities. In the developing and underdeveloped countries, it is often a typical problem of little or no internet connectivity for educational enhancement. This typical problem of inefficient internet access leads to a lack of help for students and teachers to perform the necessary activities that can be done online.

1.1 Problem Statement

All institutions aim to provide the best and efficient way of schooling within the constraint of their available finances. This is also the case for schools mostly in the developing and under-developed nations. However, circumstances and resources vary markedly between countries and all these usually impacts the implementation of the efficient way of learning in schools. Because of the pivotal role that the ISP companies play on internet connectivity, the charges for this service often gets too unbearable for institutions and hereby causes the lack of internet connectivity between students and their teachers in many schools of the developing and under-developed countries. Although, sharing course document handouts among students and verbal method of communicating announcements is what is mostly used in these institutions. However, several problems can be itemized because of using this traditional method of schooling:

- The concept of Blackboard for educational purposes to view course materials, submit an assignment, create a forum between teachers and students and other benefits that the blackboard offers is not made available in the institution.
- Information is not sufficiently disseminated among students and most importantly, the students that are not available at the time the information was passed across gets little or no content of the information that was passed.
- Schools that can afford the internet services run them at higher costs.
- According to personal experience, when this internet service is available, they often run so slow.
- Distributing handouts is not an effective way of passing course documents because it is susceptible to misplacement.

1.2 The Purpose of the Research

For this project, the main purpose is to provide a solution that enhances education activities and solves the problem of inadequate internet efficiency for educational development in places that offer little or no internet access for connectivity and for communication. This solution can be considered a short term, but it depends on how fast and rapid the schools in developing and under-developed countries can catch up in terms of resources, development and technicality. Here, I propose a way to solve this problem by creating a network in a local area that every device can connect to, using a software application – mobile, desktop, web and these devices that are connected on the LAN can communicate and share resources of any file type without the need for internet. The medium of communication would be through a Wireless Networking Protocol using an access point router. Devices that are connected on the LAN would be able to drop resources on a file system and can retrieve these resources on the file system with proper authorizations. This solution for a no-internet network creation would solve most of the problems faced with connectivity and communication in educational institutions and thus, will transform the sharing of course documents manually into a computer based as a digital transformation hereby enhancing an effective way of efficiency, distribution, connectivity and communication.

The benefit of this project is to achieve the following objectives:

- Digital Transformation in educational institutions – Making all course documents in one central repository instead of having to distribute papers and handouts.
- Creation of a mobile application software as a client for Content Management System (CMS) – This CMS models a typical blackboard.
- Server creation for request and response purposes – This server fetches resources from the file system into the client devices.
- Local Area Network that models a peer-to-peer connectivity – Since the internet is not in the scenario of what the solution proposes, its only right to be creating a local area network having no doubt that internet is the most effective medium of communication in a wide area network.

- Reduced cost of internet bandwidth that will be purchased in an institution – Since most internal communication would be done on this local network, file transfer through the internet which may take a certain amount of bandwidth will be restricted to the network that is created.
- All these goals lead to a solution to the problem in most ramification and these forms the objective of the project.

CHAPTER 2 – LITERATURE REVIEW

For this project, the literature review of the scientific references founded by many studies and research to be considered are in Networking, Application software and Database as described below:

2.1 Networking

Networks are a collection of computers that are connected with wires and wireless signals. It can also be defined as a:

- Group of interconnected things.
- Number of interconnected computers.

Networking allows computer users to share data and interact with each other, even though they are using two different computers. “When network first came into being, computers could only communicate with computers that were produced from the same manufacturer. For instance, a company can run an IBM solution or a DECnet solution but not run both together. In the late 1970, the Open System Interconnection reference model was created to break this barrier.” (Lammle, 2007)

Networking is useful because it allows a group of people to access a pool of resources. One of the most useful resources a network makes available is people. Collaboration has become expected in a workplace, schools and other institutions because of networks. People are now able to talk, chat or send an electronic mail with someone else who is literally miles away. But networking does have some disadvantages, there is an initial cost to set up a new network – such as cost of purchasing networking equipment, labor cost for trained network personnel. Other disadvantage can occur because of the cost of maintenance and threats like viruses and hacking. However, with proper maintenance and security measures, these disadvantages can be mitigated

Networks and internetworking have grown exponentially over the last 20 years with constant evolution throughout the years. Networks are made up of different protocols and standards. It is helpful to know that Networking Protocols are the guidelines that define the communication between two or more devices while the

Networking Standards make sure that different networking products from different manufacturers are interoperable using some set of rules. The evolution of Networks and internetworking can be described below:

2.1.1 Time Sharing – the first online communities - 1960

According to a Wikipedia page, “In computing, time-sharing is the sharing of a computer resource among many users by means of multiprogramming and multi-tasking at the same time” (Wikipedia, 2017). By the early 1960s, many people can share a single computer, using terminals to log in over phone lines. Even though the computers were unable to connect to each other at the time, this idea is described as the first common multi-user systems, with many people online at the same time. It then gradually develops into many features of later networks, from file sharing to e-mail and chat.



Figure 2.1 A User on the Dartmouth Time-Sharing System. (Timeline of Computer History, 2018)

2.1.2 Multiplexers: Getting more users into the same line – 1968

The telegraph is typically a system for transmitting messages from a distance along a wire in the early years. People have had the utmost urge to have many more connections on a single wire. Then the concept of Frequency Division Multiplexing came up which allows 15 terminals share the same line. Along that same line of thought in 1968, comes a new generation of time-division multiplexers that radically expand the computer terminals that can share the same line from the initial 15 now until 45. This innovation advances helped more people go online.

2.1.3 Networks come online – 1969

In the late 1969, at this time created by the United States Defense Advanced Research Projects Agency (ARPA), the ARPAnet makes a debut as a large-scale, general-purpose computer network that connects multiple computers together. ARPAnet was a ground for experimental networks and satellite radios and the need to connect diverse systems led ARPA to begin its internet programs, which developed techniques for interconnecting networks, using this technique to connect other research networks forming the basis for today's internet, a worldwide network of networks. (Abbate, 1996--1988)

2.1.4 Inventing the Internet – 1973

Before 1957, Computers only worked on one task at a time and this is called Batch Processing. The concept of batch processing was ineffective as Engineers and Programmers started developing a need to work remotely. Then the idea of Time sharing came up – The concept of sharing the processing power of one computer with multiple users. So, the next challenge must be creating the 'networks of networks', a process called internetworking or internetting. The foundation of the internet begins with the Defense Advanced Research Project's ARPAnet for the concept of a military network, the National Physical Laboratory for a commercial network, and Cyclades for the scientific network. (Picolsigns, 2009).

In 1973, France's Cyclades and the Britain's NPL network were experimenting on a protocol that influenced the United States' development of ARPAnet TCP/IP internetworking protocol, which was first designed by Vint Cerf and Bob Kahn. Given that, the Cyclades had fewer nodes and the focus was laid on the communication with other networks. In this way, the term Internet was born. The collaboration of this led to the development of NCP – Network Control Protocol and later, it was replaced by a much efficient Transmission Control Protocol (TCP). The specific feature of the TCP is the verification of file transfer. To avoid congestion of these files that are transferred, the files that are sent are divided into smaller packets and this technology was

termed, Packet Switching. During communication and transfer, the computers serve as a transfer node, starting with electronic mail and adding file sharing, remote access and eventually the World Wide Web capabilities.

2.1.5 Linking the Ethernet and Local Networks

The IEEE is an institute of Electrical and Electronics Engineer that develops global standards for a range of technologies. The year 1973 marks the start of the standard that will eventually prevail: Ethernet (Timeline of Computer History, 2018). While the internet is comprised of multiple layers, among these layers is the Link layer where Ethernet is categorized. Ethernet uses both Data Link and Physical layer specifications. All network connected in a local area had to be connected to some type of cable. When these cables are connected in a network, they act as a medium that transfer resources from one computer to another. The Ethernet becomes a contention media access method that allows all hosts on a network to share the same bandwidth of a link.

Fast forward to the year 1999, the IEEE 802.11b is a Wireless LAN for short range radio networking standard that came into place as “Wi-Fi”. The IEEE 802 family of standards made publicly available the 802.11 as a standard format for the Wireless Networking. “The wireless networking uses radio frequency to send information between devices that are able to pick up the frequency and translate the radio signals back into information that the device user can understand and use”. (edx, 2018). Radio frequencies come around in different types. The 5 GHz frequency has a less interference with a higher speed, but it has a shorter range than the 2.4 GHz is the common and popular frequency because it has a bigger range that is not mostly affected by obstacles. Wireless networking is measured in Mbps.

2.2 Application Software

An application software and a software application can be used interchangeably. A software application is a written set of software codes that serves as instructions for the computer to carry out the purpose of the written codes. This purpose is typically to solve problems. A way to solve problem is to begin analysis and break down of such problems into smaller pieces we can handle. A key note is to first understand the nature and concept of the problem as not all human problems require computational solution. However, if need be, we can go ahead to

use software technology for our solution. For this project, the analysis has shown that software application is desirable to solve this problem at hand.

According to Shari, the figures below describe the process of analyzing a problem and synthesizing a solution.

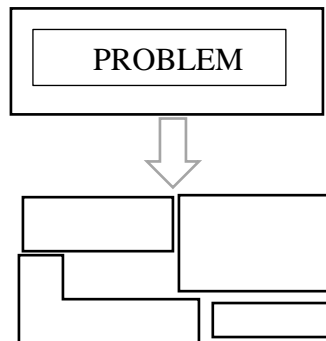


Figure 2.2 The Problem Analysis. (Pfleeger, 1991)

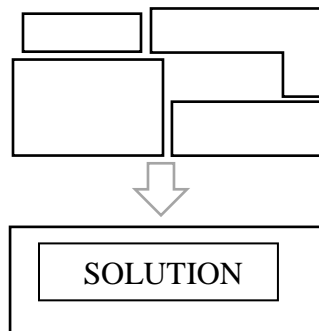


Figure 2.3 The Solution Synthesis. (Pfleeger, 1991)

Software application spans across multiple sections of modern technology in mobile phones, laptops and many more. Here, I categorize software application into two major types:

- System device application – Desktop, Mobile.
- Web design application – Website.

Software applications require a certain level of proficiency in a software programmer. Software programmers take designing and implementing a solution to a problem computationally and are responsible for quality software services. Just as product manufacturers aspire ways to make sure their products are of quality produce; software programmers too find ways to ensure their software is acceptable. The quality and acceptability of a software depends on how easy it is to use, learn and if in fact, it provides solution to the proposed problems.

There are different types of programming languages for software creation. Examples of this languages are Hypertext Markup language (HTML), JavaScript, ASP.NET, C#, Java and so on.

Literature review is a systematic summary of studies addressing a clear question, with an unbiased and valid method to identify, study and analyze data or other relevant topics (GET-IT, 2015). In the review of the relevant references founded many actual implementations of solution that solve problems of connectivity and digital transformation. The following is a summary of each research reference in terms of their goals and the solutions they provide.

2.2.1 Blackboard, 1997

Blackboard is a technology developed by the Blackboard Inc. This tool delivers a content management system allowing faculty to add resources for students to access online. The blackboard technology is mostly used to support effective learning process by providing an area to place information about courses in multiple learning styles and content formats – texts, images, pdf, audio, etc. Blackboard provides a software application that is web-based and may be installed on local servers. Its main purposes are to add online elements to courses that would have been traditionally delivered face-to-face hereby serving a digital transformation.

Conclusion

- According to Forbes.com, blackboard is used by more than 70 percent of the colleges and universities in the U.S.A. (Peter Bradford, 2007, pp. 35:301-314).
- Blackboard has proved ultimately useful in distance learning.
- A course content feature allows professors to post course documents on the platform.
- Students can chat in real time with other students in the classroom section.
- Students are also able to send emails.

2.2.2 Skype, 2003

Skype is a telecommunication software application that specializes in connecting many people through video calls, voice calls and chat. With Skype, individuals can hold a meeting, learn languages and do just about

anything that needs to be done together on their phones, tablets, computers or TV with skype on it. (Skype, 2003). Skype was created by Swede Niklas Zennstrom and Janus Friis, and it was originally a feature of hybrid peer-to-peer and client-server system (Tallinn, 2014), before it was acquired by Microsoft which transitioned the technology into a centralized Azure service. (Goodin, 2012)

Conclusion

- The use of peer-to-peer and client-server formed the basis and beginning of Skype.
- According to a Wikipedia entry in 2012, about 35 million number of concurrent users are on Skype (Caukin, 2012). The implication of this is that more and more people are connected to each other on daily basis because of this software application.
- According to reports, Skype has been improved to make Voice over IP (VoIP) a real-time communication option for schools. (Bransburg, 2007, p. 36)

2.2.3 Facebook, 2004

Facebook is a social media and social networking company based in California, United states of America. It was co-founded by Mark Zuckerberg in Harvard University. While in Harvard, Mark and his friend, Adam D'Angelo identified the need for students to upload lists of their friends and compare these friend lists with others. To them, it was a problem of connectivity that they were seeing to provide solutions to. In 2003, Mark's goal would be to create a software application that he named Facemash with a PHP programming language. To create Facemash, Mark broke into Harvard's web systems and copied student ID images. In 2004, Mark finally created a website that is called Facebook that lets users create profile for themselves and connect with their friends. (Telegraph, 2017)

Conclusion

- Facebook grew out to become a widely accepted solution to the problem of social connectivity.
- It has further become a marketing solution outreach to connect multiple people in business environments.
- A comprehensive database for individuals are stored that matches people from everywhere else.

- The Facebook system can store data that is necessary for people to keep in touch and connect with each other.

2.2.4 Gmail, 2004

Gmail is an email service that was developed by Google. A software application was the solution to this problem of transferring electronic mail from one individual or corporation to another. This solution typically mitigates the long time it takes to traditionally post a letter through the post office services. Even though Gmail is not the only email service available, it is a technology whose research is worthy of my study and whose innovation is worthy of emulation. Whenever possible, Gmail uses a transport layer security (TLS) to automatically encrypt emails that are sent and received on the web and on the devices. (Gmail, 2016).

Conclusion

- Gmail stands out as a wide service for digital transformation.
- Gmail becomes the first software application to reach 1 billion downloads on android device.
- According to the verge reports in 2016, Gmail has 1 billion monthly active users, and this shows the rate at which this software providing digital transformation is widely accepted. (Miller, 2016)

2.2.5 GroupMe, 2010

GroupMe is just another mobile group messaging app that is owned and controlled by Microsoft. It works similarly like a regular text message app on a mobile phone and even with a real-time communication. GroupMe solves the typical problem of people trying to connect to each other in groups. It even offers group calling as a conference call which does not push a time limit on you. Some of the features on the app includes ability to share photos, videos, location, create events and so on.

Conclusion

- By June 2012, GroupMe has also been widely accepted and over 550 million messages were delivered on GroupMe each month. (Shontell, 2011)

- GroupMe becomes a software that radically competes with top leading technology experts like Google with the launch of Google+.

2.2.6 GitHub, 2008

Started by Linus Trovalds, GitHub is a web-based version-control and a collaboration platform for software developers. Many people have alleged that GitHub is a social networking site for programmers. Here software codes and other programming materials are uploaded in a central repository and can be retrieved at anywhere on any platform. GitHub also allow multiple developers to collaborate on a project, update and track progress on the project. Three main terminologies serve as distinct features on GitHub – fork, pull request and merge. Forking is when a project is created based on another project that already exists. Pull Requests are the changes you’ve pushed to a GitHub repository. Merging is joining a project you modify to an existing project for purpose of update and changes.

Conclusion

- A software application written in Ruby on Rails.
- GitHub provides a light-weight workflow for developers.
- It allows developers to share their codes for others to view, modify and update
- GitHub can be used for any type of files.

2.3 Database

The collection of factual data can be described as a database. The database can further be described as a repository for a collection of computerized data files (Date, 2000, p. 2). Today, most universities and colleges have a centralized place they store student information. The departments that store these data are a very important section in colleges and it is in fact, a level of management of the college. Many routines and burdensome works need accuracy and little or no errors such as the student information, courses undertaken by each student in the school, degree level of the students and so on. A database server typically contains all the tables and data that are required to be saved.

A Database Management System is required to connect to the database server. In a typical development environment, the database can be installed on one centralized machine and usually developers can connect to that using a database management system that is installed on the respective machines. The figure below shows the relationship between a database and a database management system.

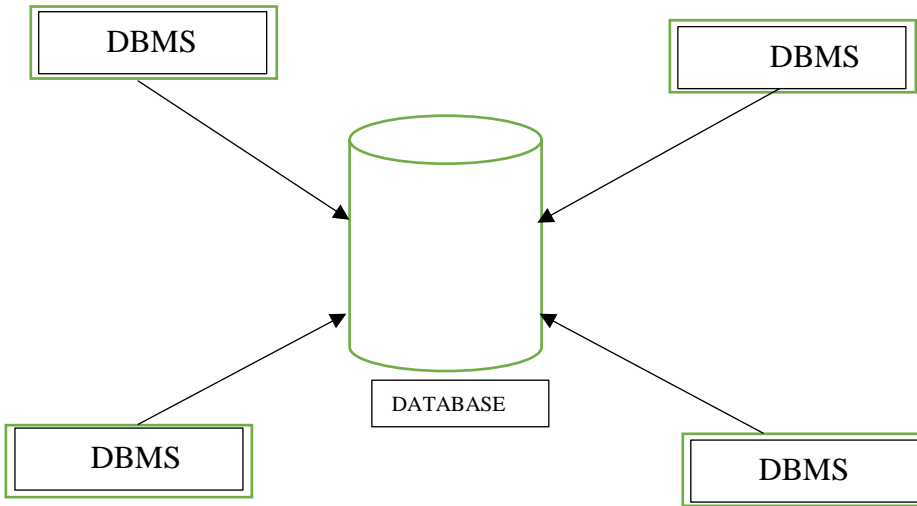


Figure 2.4 How a database management system and a database server relate.

2.4 Discussion

After critically looking at these technologies, one thing they all have in common is the internet connectivity. For these technologies to become optimal in a learning environment, they all require the internet to work. This is what has made a difference in my proposition. For this project, the main medium of communication would be done over a wireless networking protocol connected from a router and to other devices. No internet is required, and no Ethernet is required for this technology.

For the Networking review, the Timesharing system formed the basis of sharing a computer resource among many users. This technology that has developed into modern features of file sharing, to email and chat can be agreed as what came on to the design and implementation of networking connectivity. However, with the Multiplexers, it can be agreed that transmitting messages from a distance along a wire is a head start for modern telecommunication, but it can be criticized in the limited range this technology is able to cover. The Internet remains the largest community of online connections as the connections of different networks with little to no

limitations. The only shortcoming of the internet is its efficient availability in certain areas of under-developed and developing countries which has formed the scope of this research as to how connecting individuals can be made possible until the Internet finally arrive in these places.

In the Application Software review, it can be completely agreed upon that this software applications are a tool for efficient connectivity and a tool that can further make education much smoother for individuals. While the Blackboard is held in high regards for its innovation, the Blackboard has its limitations and according to a survey of 730 faculty staffs and students in Wisconsin who uses this blackboard found that course management systems are harder to learn to use than expected. (Carnevale, 2003, p. 49). Another limitation of the blackboard is that its options may be restricted to some operating systems. Even though the GitHub is regarded as the leading version control service and a social platform for developers, its technology requires more technical know-how and certain expertise to work on a master level.

Database File System is another common tool that these software applications have in common as they require a centralized place to store their information. Database utilizes authentication for administrators and the implication of this is to provide integrity to the information that is saved by ensuring the data is not manipulated by unauthorized individuals.

These reviews can be summarized as follows:

- The technology for connecting individuals in an institution require the internet connectivity.
- Each application software allows students to communicate either individually or in groups.
- File sharing are made possible with this networking and software tools.
- There is a centralized file system for storage of data.
- Communication is mostly done in a wide area network.

From the above, it is considered ideal the need for communicating and sharing information among individuals. The above systems solve a mutual problem of accessibility and connections, not only in educational institutions but in all ramifications of life. The system I am researching goes in a similar route and it provide a solution to this

imminent problem of inefficient connectivity in educational institutions too but using a different innovative strategy of communicating and sharing files without need for the internet access. The education systems in the under-developed and developing countries are in the process of development and with this solution, it raises a high level of education enhancement in their institutions.

CHAPTER 3 – METHODOLOGY

For the solution of connectivity and digital transformation to be effectively provided, a sophisticated piece of software must be used in conjunction and along with a hardware. The solution is designed in such a way that it carefully analyzes the problem faced in areas of low and inefficient internet connectivity and these analyses has shown that the desirable way to solve the problem at hand is by a successful design of a software application that will be user friendly meets with the requirements of beneficiary networking and hardware body. The application is designed to be modular and thus, required some packages to be installed at different ramifications of the development which will be detailed later in this chapter in their respective technologies. The solution is built upon different technologies and its methods will be described shortly but are itemized as:

- Sql Database File System
- Raspberry Pi 3 Raspbian
- Node. Js
- ASP.NET MVC
- Xamarin. Forms

3.1 Sql Database File System

At the heart of many organizations and institutions is the ability to collect, organize and manage data. This is precisely the focus of this sub-section. According to the Modern Database Management edition of Jeff Hoffer et al, over the past two decades, data has become a strategic asset for most institutions including education and libraries since database are used to store, manipulate and retrieve data in nearly every type of organization (Jeff, Ramesh, & Heikki, 2016). The understanding of database in financial, marketing and education can give clear understanding in answering questions almost as soon as they are asked. Database continue to become a more common part and central component of business and educational operations. For the rest of this section, it is helpful to describe the steps taken to accomplish the database part of this project. Different technologies can be used to build a database system. For this project, SQL server technology is used to construct the database.

3.1.1 Connecting to SQL Server

To connect to the database server to construct a database information, SQL Server Management Studio was installed on a windows x64 bit machine. The sql server would be automatically installed in the default `\Program Files\Microsoft SQL Server\mssql14.<instanceID>\` path of the windows which is configurable. It can be started automatically from the start menu. The Sql server management studio opens and as seen below.

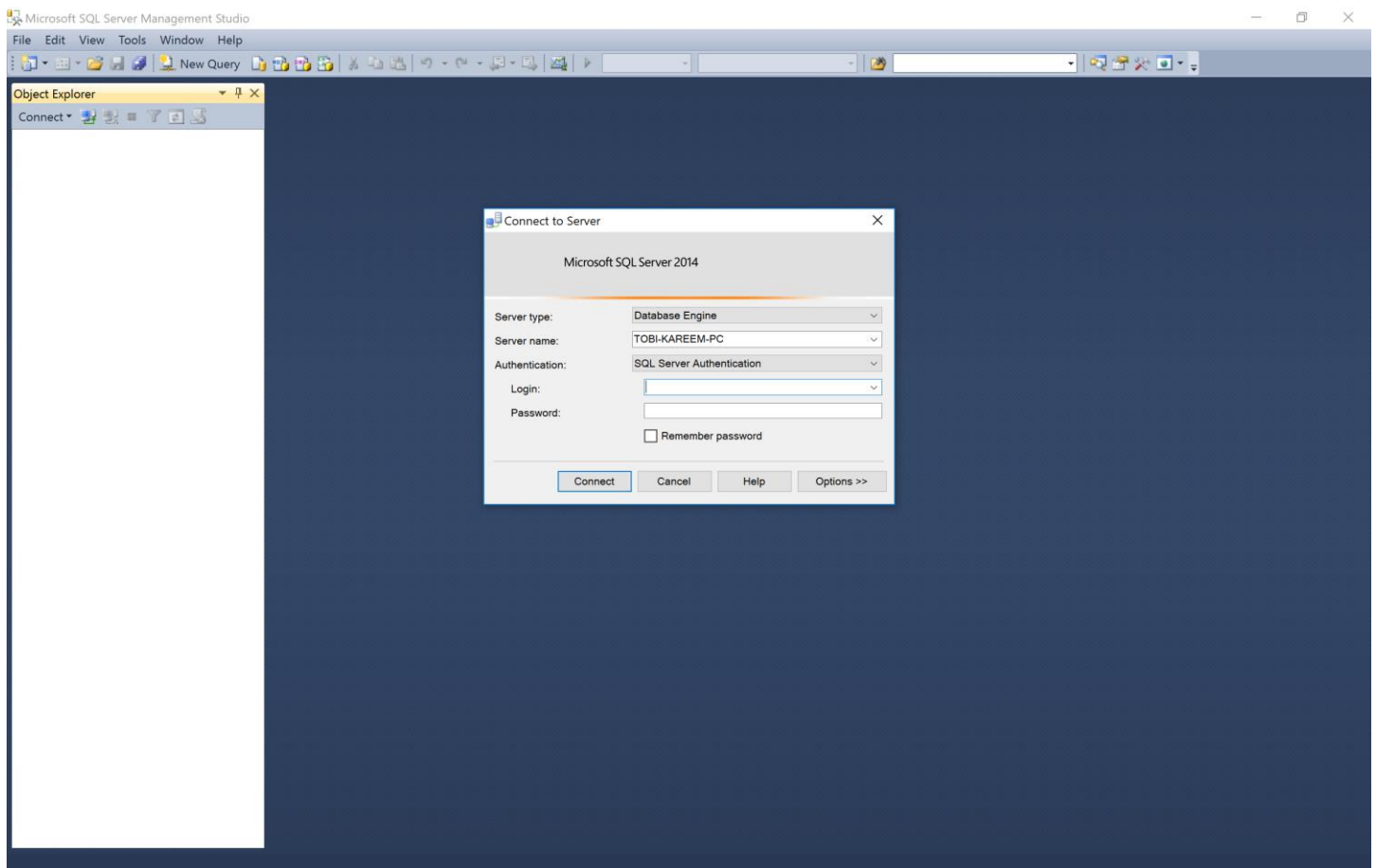


Figure 3.1 SQL Server Management Studio (SSMS)

On startup, the SSMS require certain authentication to connect to the database server. A database server typically contains all data and objects that are created to be stored on it. To connect to the database server, the client tool that is typically used is the Sql Server Management Studio, however, the SSMS is not the server itself but a client connection tool. This can be further illustrated below:

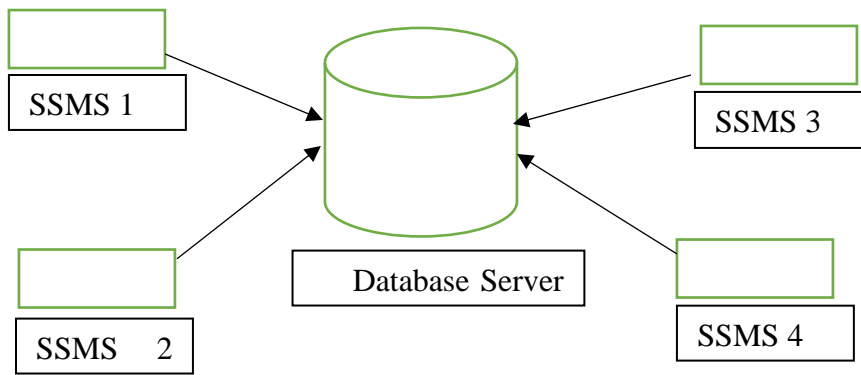


Figure 3.2 SSMS connects to Database Server

3.1.2 Creating the Database

Creating a database is done by writing database syntax known as queries. This database consists of tables that store records of the Students, Teachers, Courses and Departments. It was determined in the development of this application that the required information is needed to deliver the content management system for an educational institution. The information required for the database system are described as follows:

- Student's main information.
- Courses Information.
- Teacher's Information
- Department's information.

The student's main information consists of the student details, department id and password. The course information consists of course id, course name, teacher id and department id. The Teacher information consist of teacher's id, first name, last name, middle name, department id and password. Department information consist of department id, department name and the department administrator. This is further illustrated below:

STUDENT

- StudentID
- FirstName
- LastName
- MiddleName
- Password
- DepartmentID

COURSE

- CourseID
- CourseName
- TeacherID
- DepartmentID

TEACHER

- TeacherID
- FirstName
- LastName
- MiddleName
- Password
- DepartmentID

DEPARTMENT

- DepartmentID
- DepartmentName
- DepartmentAdmin

Figure 3.3 SSMS connects to Database Server

3.1.3 The System Administrator Roles

The role of the system administrator covers all right of creating, updating and deleting records in the database. The administrator is tasked with adding students, teachers, courses and departments' information into the database and performing other database operations on this information. The system administrator would be an employee of the institution and a use case for the role is described below:

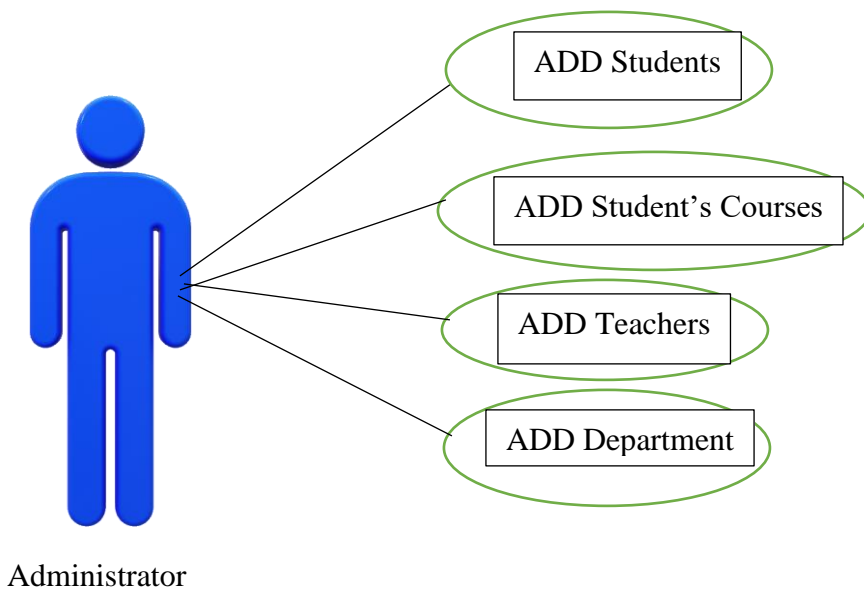


Figure 3.4 Use case for the admin roles.

3.1.4 Database Description

The description of the database covers the entities and their relationships. Entities and their relationships are a very important part of database architecture as they point us in the right direction and basis for the logical structure of our database. These entities are the records representing a noun in the database. The Entity-Relationship model is expressed in terms of entities in the institution, the relationships among these entities and the attributes or properties of both the entities and their relationships which is also referred to as an E-R diagram. A description of the relationship role can be in three parts namely:

1. One to One relationship
2. One to Many relationship

3. Many to One relationship.

The figure below shows the E-R diagram of the entities in our database and their relationships.

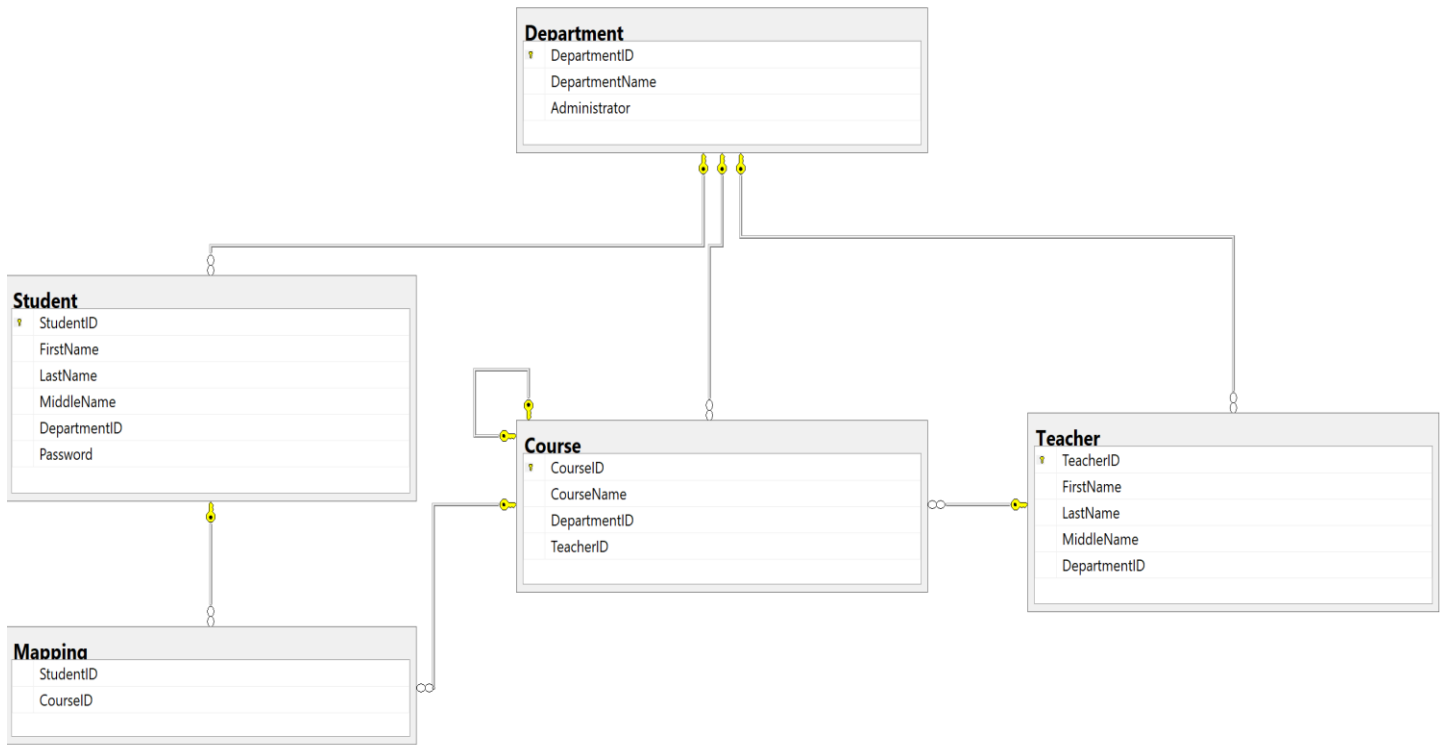


Figure 3.5 An E-R diagram for the database.

3.1.5 Database Logical Structure

During logical design, we transform the E-R model diagrams that were developed in the earlier section during the conceptual design into a relational database schema. This is a straightforward process as it only contains an additional well-defined set of rules. These rules describe the user input whether an alphabet is expected, or a numeric value is expected. The tables below describe the logical design of our database.

1. Student Table

Column Name	Data Types	Allow Nulls	Description
Student ID	int	No	Student identifier (PK)

FirstName	varchar (55)	No	First Name
LastName	varchar (55)	No	Last Name
MiddleName	varchar (55)	Yes	Middle Name
DepartmentID	varchar (55)	No	Department ID (FK)
Password	varchar (55)	No	Password

Table 3.1 Student table

2. Course Table

Column Name	Data Types	Allow Nulls	Description
CourseID	varchar (55)	No	Course Identifier (PK)
CourseName	varchar (55)	No	Course Name
DepartmentID	varchar (55)	No	Department Id (FK)
TeacherID	int	No	Teacher Id (FK)

Table 3.2 Course table

3. Teacher Table

Column Name	Data Types	Allow Nulls	Description
TeacherID	Int	No	Teacher Identifier (PK)
FirstName	varchar (55)	No	First Name
LastName	varchar (55)	No	Last Name
MiddleName	varchar (55)	Yes	Middle Name

DepartmentID	varchar (55)	No	Department ID (FK)
--------------	--------------	----	-----------------------

Table 3.3 Teacher table

4. Department Table

Column Name	Data Types	Allow Nulls	Description
DepartmentID	varchar (55)	No	Identifier (PK)
DepartmentName	varchar (100)	No	Name
Administrator	varchar (55)	Yes	Dean

Table 3.4 Department table

3.2 Raspberry Pi 3 Raspbian

The Raspberry Pi is a small widely used single board computer running on a Linux operating system (Rory, 2011). It has an HDMI interface on one end for a TV display, and a USB master socket on the other end for a keyboard, plus a mouse via an off-board hub if needed. The figure below shows what a typical raspberry pi looks like.



Figure 3.6 A Raspberry pi

The raspberry pi goes through many series of evolution in several versions with added features for functionality. The functionality of the device spans across many performances of computing and can be used for such things as a programming device, a hacking device and even a networking device. For this project, I have taken an advantage of the networking functionality of the raspberry pi to create a Local Area Network using the device to model an access point that is configurable to host DHCP servers, Dnsmasq and distribute IP addresses to every device that connects automatically. Below I describe the strategies that were used to make this happen.

3.2.1 Setting Up the Raspberry Pi as an access point for a LAN

The raspberry pi provides different functionalities and one of these functionalities is its used as a wireless access point router that can run a standalone network. To get this started, a fresh installation of Raspbian Operating System was made and booted into the device. Using *ifconfig -a* in the terminal, my raspberry pi model has three basic network interfaces namely *wlan0* for WLAN devices, *eth0* for basic ethernet and *lo* which presumably represents the localhost. Since I am using the model 3 that comes with an inbuilt wireless, I do not require a WIFI dongle. However, a WIFI dongle might be desired for a longer range.

3.2.1.1 The Steps

It is always helpful to update and upgrade the operating system. The apt-get commands in Linux is used to find new packages, install and upgrade new packages and to clean the system (Abhishek, 2017). Updating and upgrading with apt-get requires the super user privileges.

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

For software-wise, to use the access point functionality, an application package software is provided by hostapd which can be installed on the pi's OS with apt-get (Fredrik, 2013). This software will act as the host access point.

```
sudo apt-get install hostapd
```

Dnsmasq is a free software that provides Domain Name System (DNS), Dynamic Host Configuration Protocol (DHCP) server and a network boot (Kelley, n.d.). Since, we require IP assignments from a DHCP server. I have chosen to use the Dnsmasq by installing it as shown below:

```
sudo apt-get install dnsmasq
```

IP Tables manager can be installed with the command below and its use will be described later in this section.

```
sudo apt-get install iptables-persistent
```

My next stop was to configure a DHCP server provided by the dnsmasq. First, I must open the configuration file at

```
sudo nano /etc/dnsmasq.conf
```

I would like to configure the *wlan0* interface and provide an authoritative IP addresses in the range of 192.168.1.17 and 192.168.1.30 with a timeout of 24 hours using a subnet mask of 255.255.255.0 with the commands below:

```
interface=wlan0
```

```
dhcp-authoritative
```

```
dhcp-range=192.168.1.17,192.168.1.30,24h
```

```
dhcp-option=1, 255.255.255.0
```

```
dhcp-option=3, 192.168.1.1                      #dhcp-option =3 represents the gateway
```

```
dhcp-option=6, 192.168.1.1                      #dhcp-option=6 represents the dns
```

```
dhcp-host=c6:9d:ed:18:0a:4b, Tobi-Kareem-PC, 192.168.1.20    #this is a static ip address that I am  
#assigning #to a computer named "Tobi-Kareem-PC" that was preceded with the mac address.
```

On my raspberry pi, I shall be setting up a *wlan0* for a static IP Address 192.168.1.16 in the dhcpd configuration file. Since we are configuring a standalone network to act as a server, it is only right for the device to have a static IP address. The configuration file can be found in

```
sudo nano /etc/dhcpd.conf
```

At the top of the line put

```
interface wlan0
```

```
static ip_address=192.168.1.16/24
```

At this point, I am all set to configure my access point by creating a new the hostapd configuration file at

```
sudo nano /etc/hostapd/hostapd.conf
```

The information below is a configuration for the host access point. For this configuration, I am using channel 6 with a network name of *KareemNet* and password as describe below.

```
interface=wlan0
```

```
ssid=KareemNet
```

```
channel=6
```

```
hw_mode=g
```

```
wmm_enabled=0
```

```
macaddr_acl=0
```

```
wpa=2
```

```
wpa_passphrase=MYPASSWORD
```

```
wpa_key_mgmt=WPA-PSK
```

wpa_pairwise=TKIP CCMP

Since I am using the raspberry pi 3 that comes with an inbuilt wireless, I do not need to provide the *driver=nl80211* parameter.

At this point, we need to tell the system where to find the configuration file that was just created.

sudo nano /etc/default/hostapd

On the line where *#DAEMON_CONF* is, it should be replaced with:

DAEMON_CONF="/etc/hostapd/hostapd.conf"

The command above should also be added in the *DAEMON_CONF* location at

sudo nano /etc/init.d/hostapd

The next step is to add a routing system. This can be done by setting up a Network Address Translator (NAT).

The purpose of this is to allow multiple clients to connect to the WIFI and have all the data tunneled through the single IP. This process will also start IP forwarding on boot up. The file to edit can be found in

sudo nano /etc/sysctl.conf

We need to uncomment the line below or add this on a new line

net.ipv4.ip_forward=1

to activate the NAT routing system immediately, I ran the command below:

sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"

At this point, our ip_table configuration is next in command. The command below adds a masquerade for outbound traffic and creates a network translation between the ethernet port *eth0* and the wifi port *wlan0*

sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE


```
sudo iptables -A FORWARD -I eth0 -o wlan0 -m state --state RELATED,ESTABLISHED -j ACCEPT
```

```
sudo iptables -A FORWARD -I wlan0 -o eth0 -j ACCEPT
```

To save the iptables rule and always make the rule available upon every reboot:

```
sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"
```

To check the rules in the iptable

```
sudo iptables -t nat -S
```

```
sudo iptables -S
```

Another way to restore the rules on boot up is to edit the file

```
sudo nano /etc/rc.local
```

add the following command above the *exit 0*

```
iptables-restore < /etc/iptables.ipv4.nat
```

then reboot.

This works well and the *KareemNet* network that was created can now be visible on every device that searches for a wifi connection.

3.3 Node.js

Node.js is a JavaScript framework for asynchronous server communication. The node.js software needed to be install on a raspberry pi for this project. Node.js has set of built-in modules and one of the most important module to be considered in this project is the mssql module. To download the newest version of Node.js on the raspberry pi, I used the following command (w3schools.com, n.d.).

```
curl -sL https://deb.nodesource.com/setup\_8.x | sudo -E bash -
```

The command below installs the new downloaded node.js

```
sudo apt-get install -y nodejs
```

To confirm the successful installation and check the version of node, use

```
node -v
```

3.3.1 Remote Connection to SQL Server

To connect to the sql server, certain considerations must be taken into place. These are configuration settings and according to Marko Zivkovic (Zivkovic, 2016), the settings must be in place and they are so important that, without them, the connection to the remote SQL server would not be successful. These are:

- Allowing the sql server instance the protocol that is being requested
- Allowing the access through the firewall

To get the node.js server to connect to the remote database file server, a JavaScript packet manager – npm’s module called mssql was installed. With this, connection was made easy to the remote database. The command to install mssql from the packet manager is:

```
npm install mssql
```

Connect the raspberry pi hosting the node.js software to KareemNet. Open a new file from the terminal with a .js extension

```
var sql = require('mssql');    #this imports the mssql module
```

The next step is to define the remote database configuration for connection as described below:

```
var dbconfig = {
```

```
server : '169.254.142.131\\MSSQLSERVER',      #this is the remote location of my database server
```

```
user : 'username',
```

```
password : 'password',
```

```

database : 'TobChat',

port : 1433    #default port for sql

};

(async function() {

  Try {

    Let con = await sql.connect(dbconfig)

    Let result = await con.request().query('SELECT * FROM Student')

    Console.log(result);

  }

  Catch (err)

  {

    If (err)

      Console.log(err)

  }

})

()

Sql.on('error', err => {

  Console.log(err);

})

```

3.4 ASP.NET MVC

ASP.NET MVC incorporates the architectural design pattern known as the Model-View-Controller (MVC). According to the Microsoft documentation on ASP.NET MVC, the design pattern separates an application into three main components: the model, the view and the controller. (Microsoft, n.d.) ASP originally stands for Aspect Server Page and ASP.NET MVC is a web application framework which is also an open source framework. I have created this ASP.NET MVC project for the administrator's privileges as required. Having the permission, the administrator is tasked with the duty of adding and providing students and teacher's information including courses and departments into the file system through the MVC web application framework. The MVC framework includes the following components as described below:

3.4.1 Models

The Model is a collection of classes that stores and retrieve data in a database according to commands from the controller. A model object might retrieve information from a database, operate on it and write the updated information back to the database table.

3.4.2 Views

The view component is the User Interface component, it displays data to the user using the model. Users can edit a view component based on the current state of the model object. In ASP.NET MVC application, the closest thing to a page is a View.

3.4.3 Controllers

The Controller is the basic central unit of the ASP.NET MVC application. The controller handles user interaction works with the model to return a view. The controller can be thought of as an intermediary between the model and the view. It takes data from the model and it passes this data to its respective view as requested.

Below is a pictorial representation of the interaction between the Model, View and Controller.

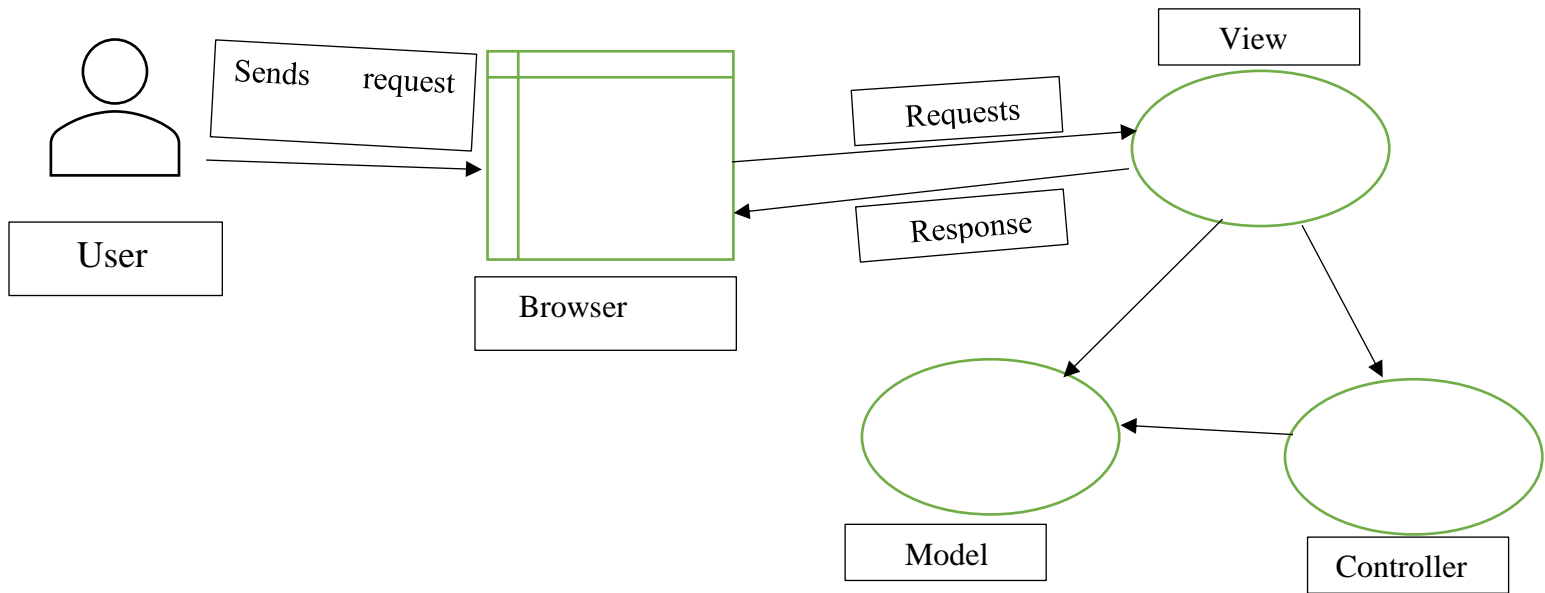


Figure 3.8 The MVC architecture illustration

For this project, I have created an ASP.NET MVC Empty project named “TobChatWeb”. This project resides in the same solution as the Xamarin.Forms project. The purpose of the TobChatWeb is to build a webpage that describes what the digital transformation is about and a platform for the administrator to perform administrative privileges like adding and updating students and teachers’ information. This information is required for login into the Mobile App. Below, I discuss the folder structure for the TobChatWeb.

3.4.4 MVC (TobChatWeb) Folder Structure

The project is written with the version 5 of the MVC framework. To create an MVC application from visual studio, click on a New Project in the Start page. Select Web from the left pane and choose ASP.NET Web Application (.NET Framework). Name the project and choose the MVC template. For this project, I selected Individual User Accounts since my application require user authentication. Visual Studio created the following folders as shown in the diagram below which will be described shortly.

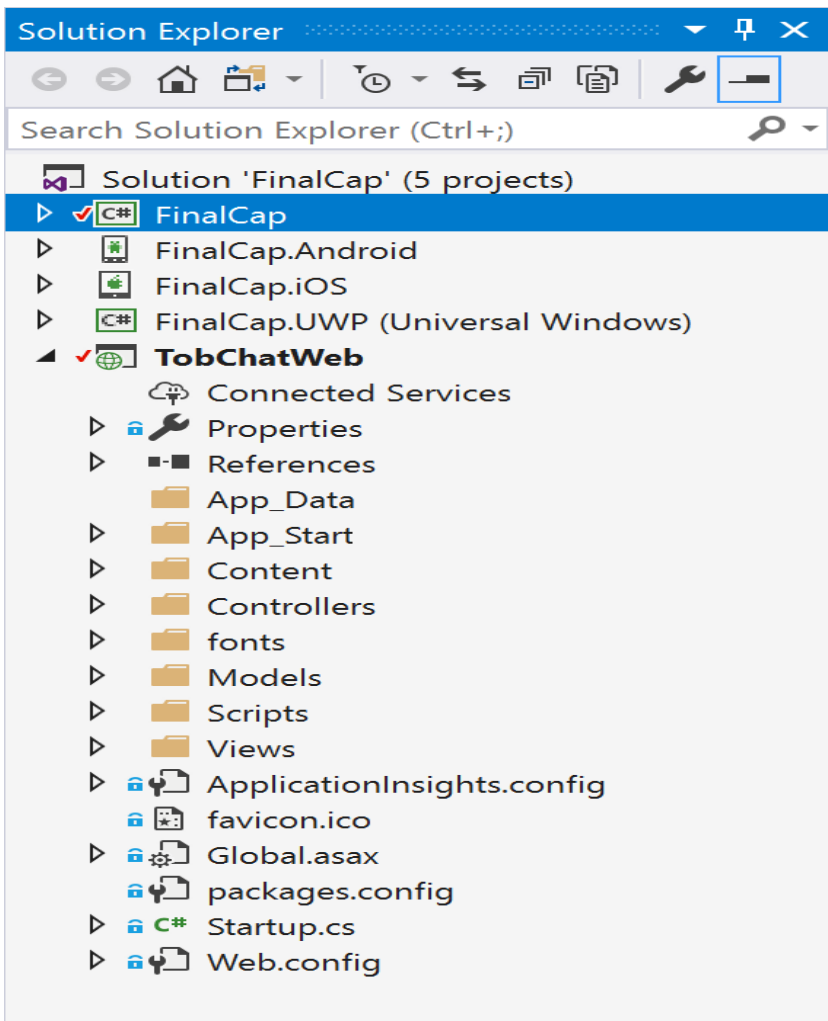


Figure 3.9 TobChatWeb MVC folder structure

App_Data:

This folder contains file-based data stores. This folder can also be used for storing database files.

App_Start:

This folder groups together ASP.NET MVC configuration files. It contains five c-sharp configuration files namely: **BundleConfig.cs**, used to improve request load time by registering the Styles and JavaScript to be bundled and minified. **FilterConfig.cs**, registers global filters. These filters are applied to all actions and controllers. **IdentityConfig.cs**, can be used to create user roles. **RouteConfig.cs**, is used to register various route patterns in the project. **Startup.Auth.cs** is a file used for third-party login providers like Facebook, Google+, et cetera.

Content:

This folder holds other files for the web application design for user interface. It holds the following static files: bootstrap.css, bootstrap.min.css, Site.css, It can also hold images and icon files.

Controllers:

In the Controllers folder as described above, it contains class files for the controllers. Here, user requests and responses are handled. An intermediary between the model and the view. This folder contains: AccountController.cs, CoursesController.cs, DepartmentsController.cs, HomeController.cs, ManageController.cs, StudentsController.cs, TeachersController.cs

fonts:

This folder contains custom font file for the TobChatWeb application.

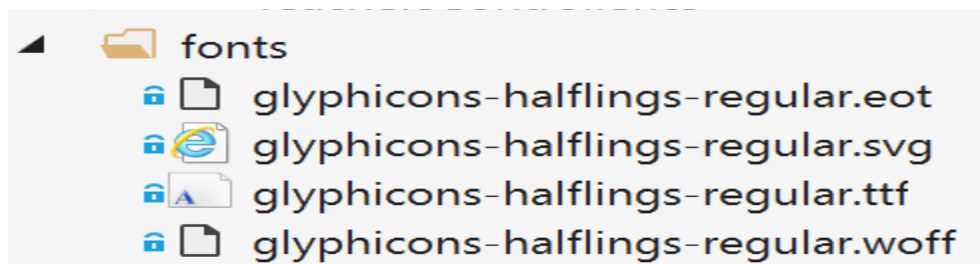


Figure 3.10 The TobChatWeb MVC fonts folder

Models:

The Models folder contains model class files for the application. The purpose of the model has been described above as a class that is used by the application to store and retrieve data from a database.

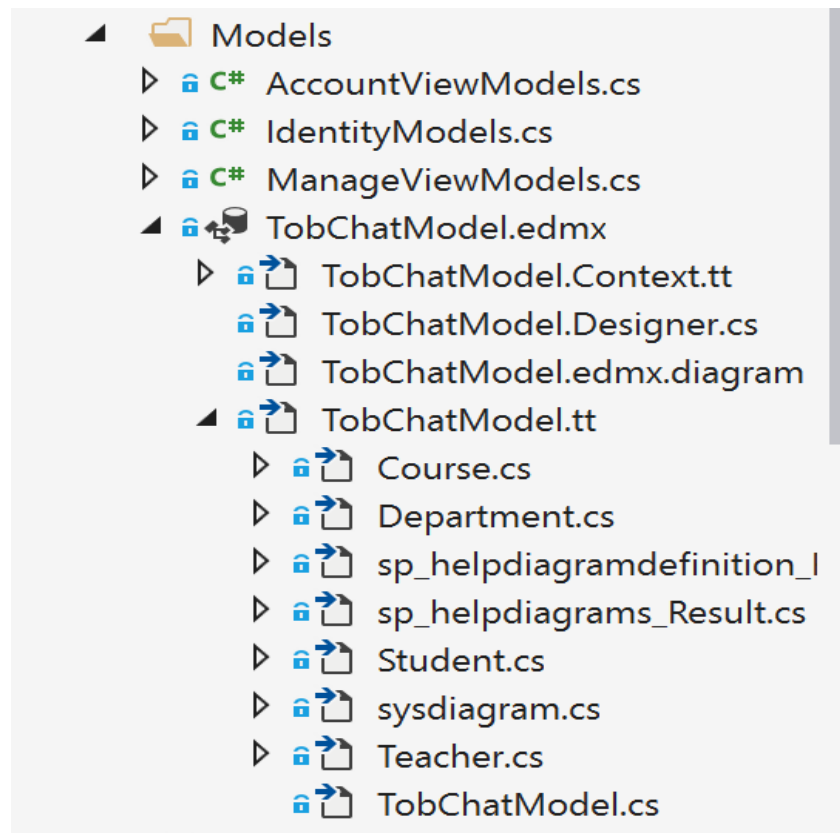


Figure 3.11 The TobChatWeb MVC Models folder

Scripts:

This folder contains JavaScript files for the TobChatWeb Application.

Views:

The Views folder contain mostly other folders for each controller file and .cshtml files in them for the User Interface. .cshtml file is a web design file that incorporates html and csharp. According to the Microsoft documentation on ASP.NET MVC, the benefit of using the Views help to establish a Separation of Concerns (SoC) within an MVC app by separating the user interface markup from other parts of the app. (Steve & Luke, 2017). It is also worthy to note that views that are specific to a controller are created in the Views/[ControllerName] folder.

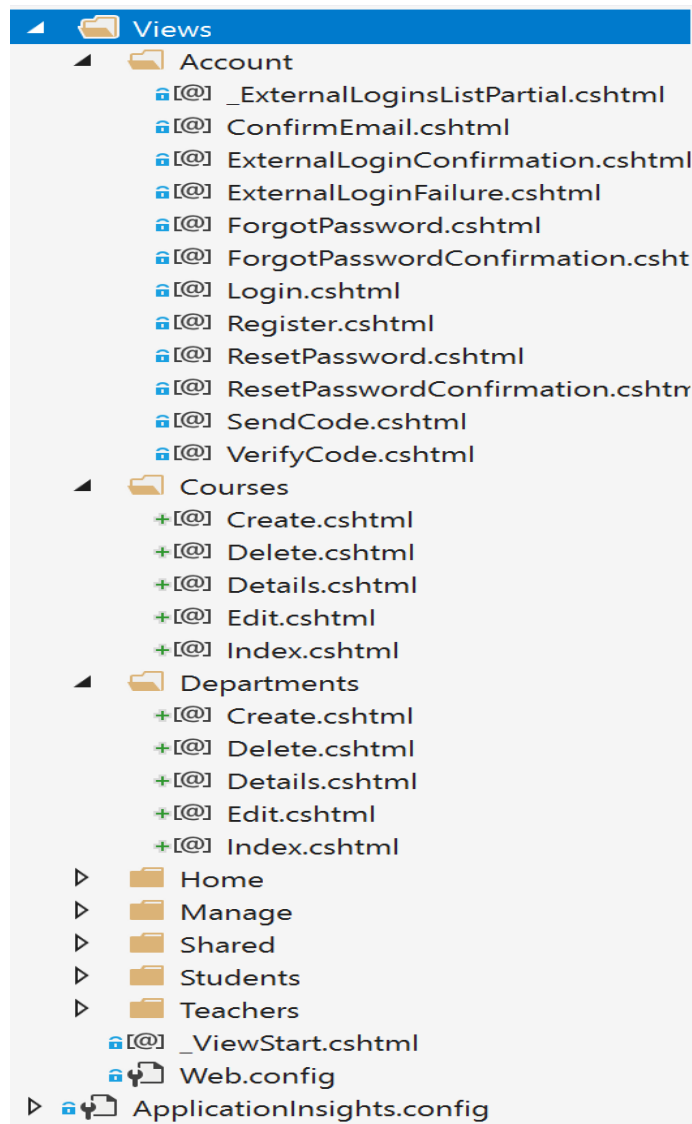


Figure 3.12 The TobChatWeb MVC Views folder

Global.asax:

Here, application level logic is coded in this file, developers can handle events or errors at the application level. The events and objects declared in the Global.asax file are applied to all resources in the web application. (Suprotim, 2008)

Packages.config:

This file keeps a record of the packages I have installed from NuGet and their versions.

Web.config:

This file holds the connection strings and other application level configurations.

3.5 Xamarin.Forms

According to the Xamarin developer documentation, “Xamarin.Forms is a cross-platform, natively-backed UI toolkit abstraction that allows developers to create user interfaces that can be shared across Android, iOS and Windows Phone” (Xamarin.Forms). This technology allows dotnet developers or any other developers that are interested, to build mobile applications using C# with XAML. Xamarin.Forms uses code sharing to share application logic and user interface, UI definition in a cross-platform fashion. With Xamarin.Forms, any developer can write their application code once, and then build a native application that uses the code to run on iOS, Android and Windows device. The business logic is written in C# and the UI can be defined in codes, or in XML-like XAML markup language.

For this project, I have taken advantage of this sophisticated technology to create a cross-platform mobile app that allows login for the students and the teachers. The mobile app authenticates students and teachers’ login from the TobChat file system and display the courses taken by the students or the courses that are taught by the teachers. The teacher can then upload course documents from the mobile application into the file system and the mobile app shows the history of files that are uploaded per course in the student and teacher’s account. The files can be downloaded from the mobile application. Students and Teachers can create forum on the mobile application and chat with each other. Individuals can go into the account tab and update their information. All these and other features are coded with the Xamarin.Forms technology using C# for the code behind and XAML for the user interface. The Xamarin.Forms also imbibe the concept of the Model-View-Controller as it was described earlier in the ASP.NET MVC section. However, it is popularly known as Model-View-ViewModel in Xamarin.Forms development. The ViewModel is serving as the controller in this case, the intermediary between the model objects that stores and/or retrieves the data from the database and the view that displays the User Interface. The ViewModel holds the business logic of the application. The Xamarin.Forms mobile application for

this project is tentatively, named as “FinalCap” and this is the client application that connects to the Local Network to fetch resources from the file system. This mobile application works in the institution that serves as a Local Area without the need for internet to enable connectivity and communication. Below, I describe the folder structure and ways to achieve the FinalCap mobile application client.

3.5.1 Xamarin.Forms (FinalCap) Folder Structure.

To create a Xamarin.Forms application, it can be done using the Visual Studio on a Windows PC or using the Visual Studio for Mac on a MacBook computer. However, to create a Xamarin.Forms project on a Windows PC that channels the three-major mobile devices (Android, iOS, Windows), the developer needs an additional MacBook with XCode installed and Visual Studio for Mac to be able to run the iOS project. On the other hand, developing a Xamarin project on a MacBook computer does not necessarily require a Windows PC but only Android and iOS devices can be channeled. Apple’s MacBook computers do not allow development for windows phone. The FinalCap mobile project was created using the Windows PC Visual Studio and also tested on a MacBook computer for the purpose of iOS mobile application development. Project description henceforth would be based on Visual Studio project development on Windows PC.

Create a new Project by clicking New Project in the Start page. Select Cross-Platform from the left pane and choose Mobile App (Xamarin.Forms). Name the project and choose the Blank App from the template. For this project, I selected .NET Standard since I needed a Portable Class Library. Visual Studio created four different folders: FinalCap, the main source of the project for code sharing, FinalCap.Android which runs Android specific codes, FinalCap.iOS, and this runs iOS specific codes, FinalCap.UWP(Universal Windows) runs Windows Phone specific codes. In the FinalCap folder, I created separate folders to model the MVVM architecture namely: Model, View and ViewModel. I also created a Services folder for code abstraction. These folders support data binding in my Xamarin.Forms application for easy testing and some level of security. Data binding allows code to run concurrently in different modes.

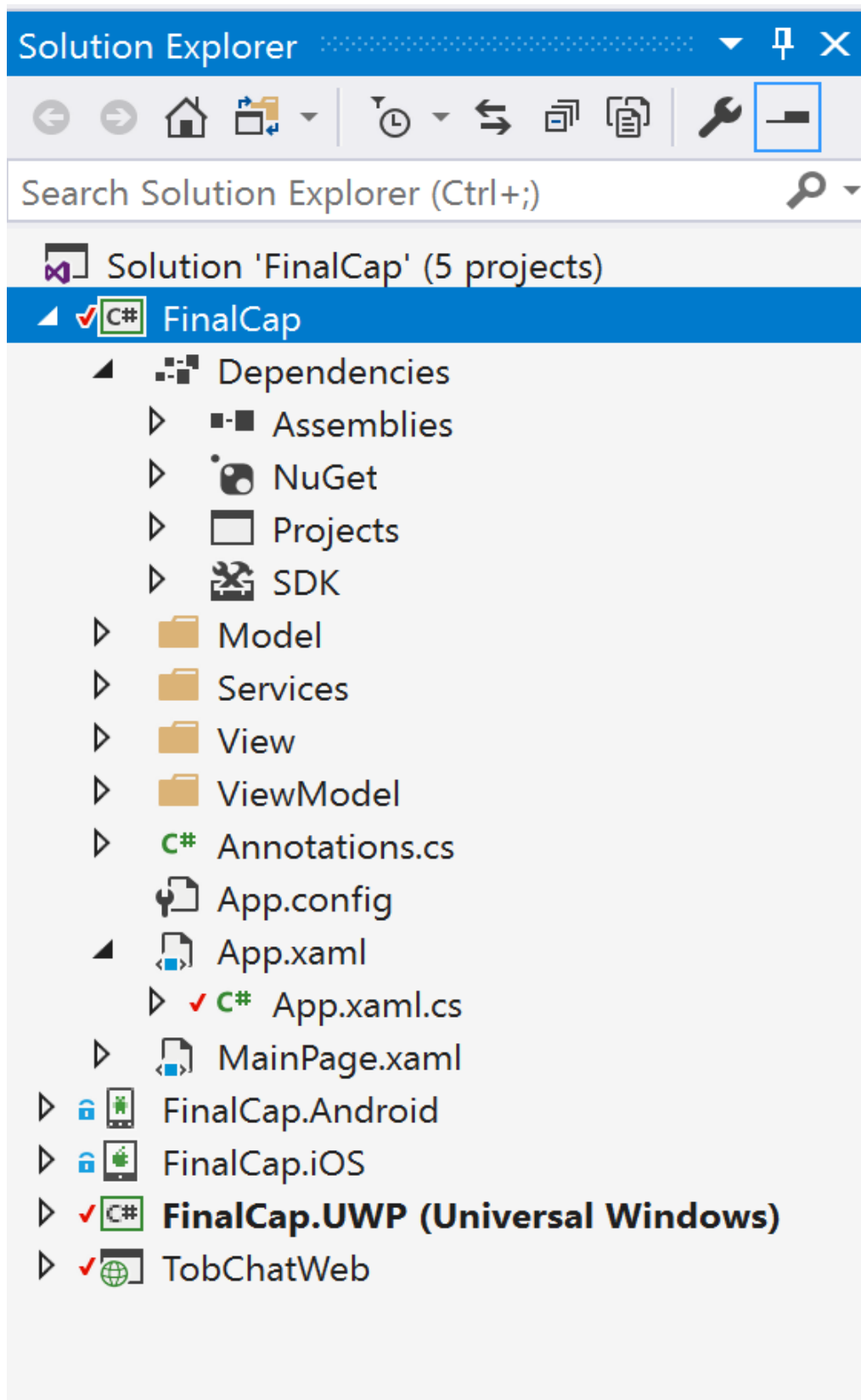


Figure 3.16 FinalCap Xamarin.Forms Project folder structure

Dependencies:

These contains the packages that will be used for the project, it also houses software assemblies, the NuGet packages, the TobChatWeb project reference and the SDK library.

Model:

This folder contains the model objects that connects to the database to store and retrieve information. For this project, the model objects are elements and fields from the MVC project.

Services:

The services folder holds the classes that are necessary to communicate with the ASP.NET MVC model backend. In this project, the services folder also contains interfaces and mock setups where I swap out the real service and use something that has the same interface but deals with local data for initial set ups.

View:

This is the folder that contains files that render the user interface. This folder typically has its files with the *.xaml* extension to indicate the markup language used to build the user interface.

ViewModel:

Otherwise known as the Controller, this folder contains the business logic and behind codes from the files in the View Model. Transfer of communication is done through data binding.

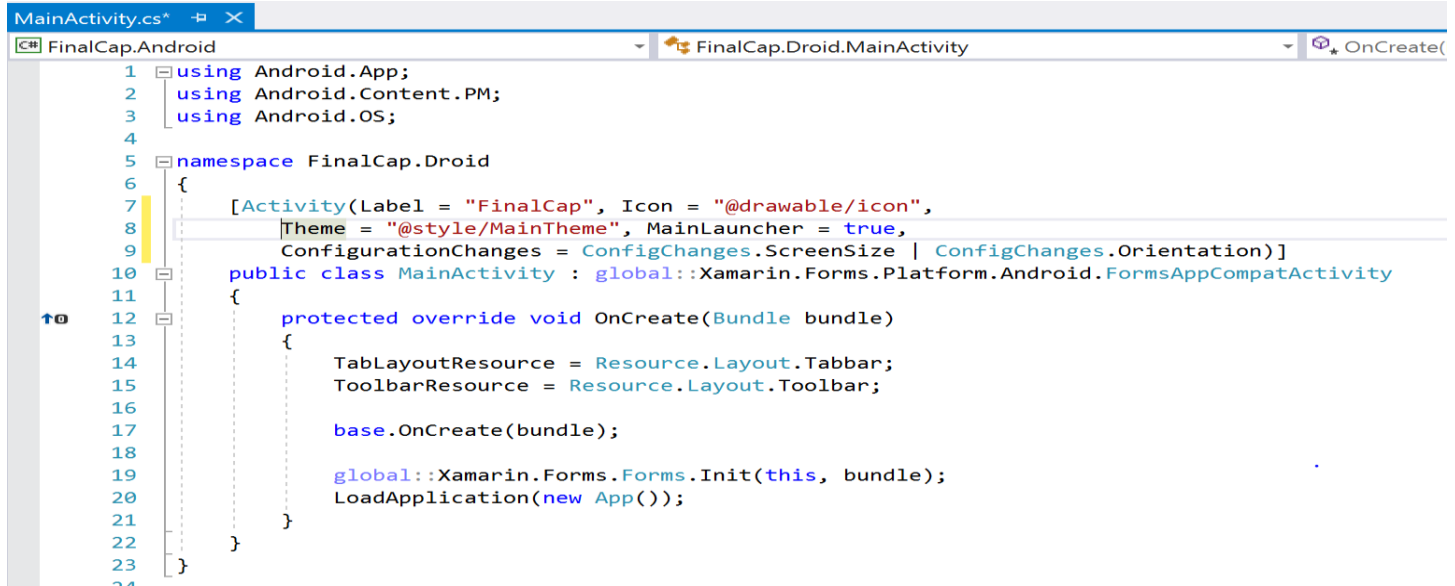
App.Xaml / App.Xaml.cs:

This class is the main Application class whose constructor only sets the main page property of the application class to an object of type Page.

FinalCap.Android:

This folder contains Android specific codes. In the Android application, the typical MainActivity class is derived from the Xamarin.Forms class named FormsApplicationActivity. This class goes ahead to override

OnCreate method which calls a LoadApplication method passing in an argument of the App class in the FinalCap namespace to display the content of the FinalCap application.



```
1 using Android.App;
2 using Android.Content.PM;
3 using Android.OS;
4
5 namespace FinalCap.Droid
6 {
7     [Activity(Label = "FinalCap", Icon = "@drawable/icon",
8         Theme = "@style/MainTheme", MainLauncher = true,
9         ConfigurationChanges = ConfigChanges.ScreenSize | ConfigChanges.Orientation)]
10    public class MainActivity : global::Xamarin.Forms.Platform.Android.FormsAppCompatActivity
11    {
12        protected override void OnCreate(Bundle bundle)
13        {
14            TabLayoutResource = Resource.Layout.Tabbar;
15            ToolbarResource = Resource.Layout.Toolbar;
16
17            base.OnCreate(bundle);
18
19            global::Xamarin.Forms.Forms.Init(this, bundle);
20            LoadApplication(new App());
21        }
22    }
23 }
```

Figure 3.17 FinalCap.Android MainActivity class

FinalCap.iOS:

This folder is for the iOS project. The iOS project here contains a class that derives from an alternative FormsApplicationDelegate instead of the regular UIApplicationDelegate. It overrides a method called, FinishedLaunching and then calls LoadApplication method passing in an argument of the App class from the FinalCap namespace to render the page content.

```

AppDelegate.cs
FinalCap.iOS
FinalCap.iOS.AppDelegate
FinishedLaunchir

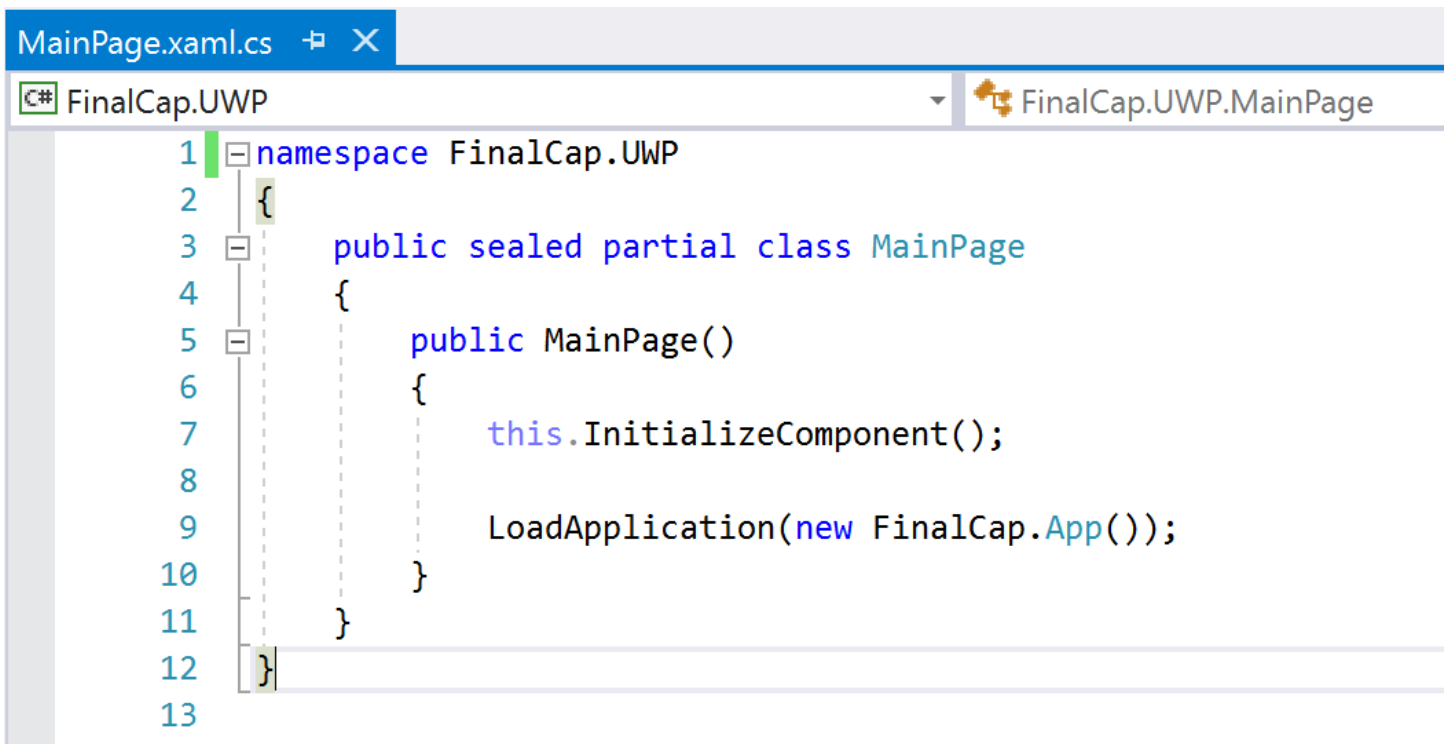
1 using Foundation;
2 using UIKit;
3
4 namespace FinalCap.iOS
5 {
6     // The UIApplicationDelegate for the application. This class is responsible for launching the
7     // User Interface of the application, as well as listening (and optionally responding) to
8     // application events from iOS.
9     [Register("AppDelegate")]
10    public partial class AppDelegate : global::Xamarin.Forms.Platform.iOS.FormsApplicationDelegate
11    {
12        //
13        // This method is invoked when the application has loaded and is ready to run. In this
14        // method you should instantiate the window, load the UI into it and then make the window
15        // visible.
16        //
17        // You have 17 seconds to return from this method, or iOS will terminate your application.
18        //
19        public override bool FinishedLaunching(UIApplication app, NSDictionary options)
20        {
21            global::Xamarin.Forms.Forms.Init();
22            LoadApplication(new App());
23
24            return base.FinishedLaunching(app, options);
25        }
26    }
27 }
28

```

Figure 3.18 FinalCap.iOS AppDelegate class

FinalCap.UWP (Universal Windows):

This is the Windows project folder. The main entry point here is the MainPage.Xaml.cs which calls the LoadApplication method and passes the App class. It doesn't seem like so much happening in here to load a windows project. It is worthy to note that this MainPage.Xaml.cs actually does derive from the Xamarin.Forms class.



```
1 namespace FinalCap.UWP
2 {
3     public sealed partial class MainPage
4     {
5         public MainPage()
6         {
7             this.InitializeComponent();
8
9             LoadApplication(new FinalCap.App());
10        }
11    }
12 }
13
```

Figure 3.19 FinalCap.UWP MainPage class

3.5.2 Testing the Mobile Applications.

The Xamarin.Forms C# framework debugs successfully and it was deployed on the Android Nexus 4 running API 21 Emulator, the iPhone 7 Simulator and the Windows Phone Emulator. To run the program on an iOS simulator, there must be a connection to a MacBook computer running appropriate version of XCode software and Xamarin For Mac. These computers must be remotely connected through a Xamarin Mac Agent using the MacBook's IP Address and information for the connection. The successful test can be seen in the screenshots below for individual devices.

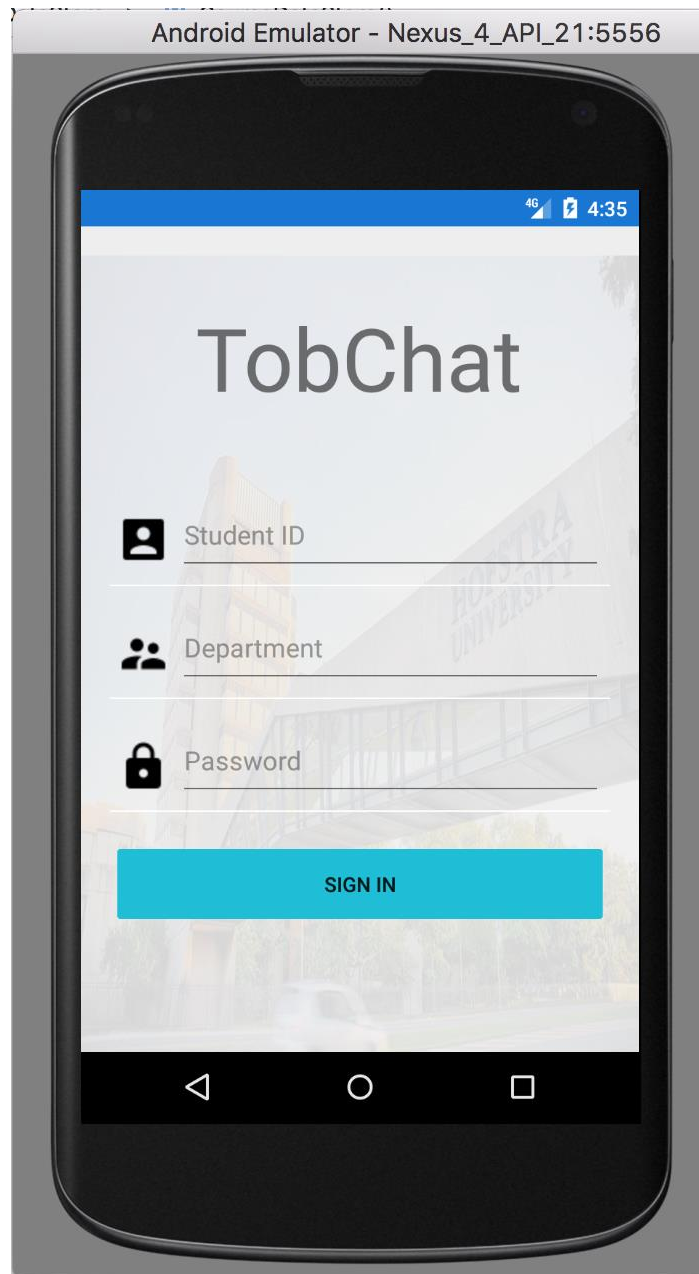


Figure 3.20 FinalCap Android Mobile Application Project

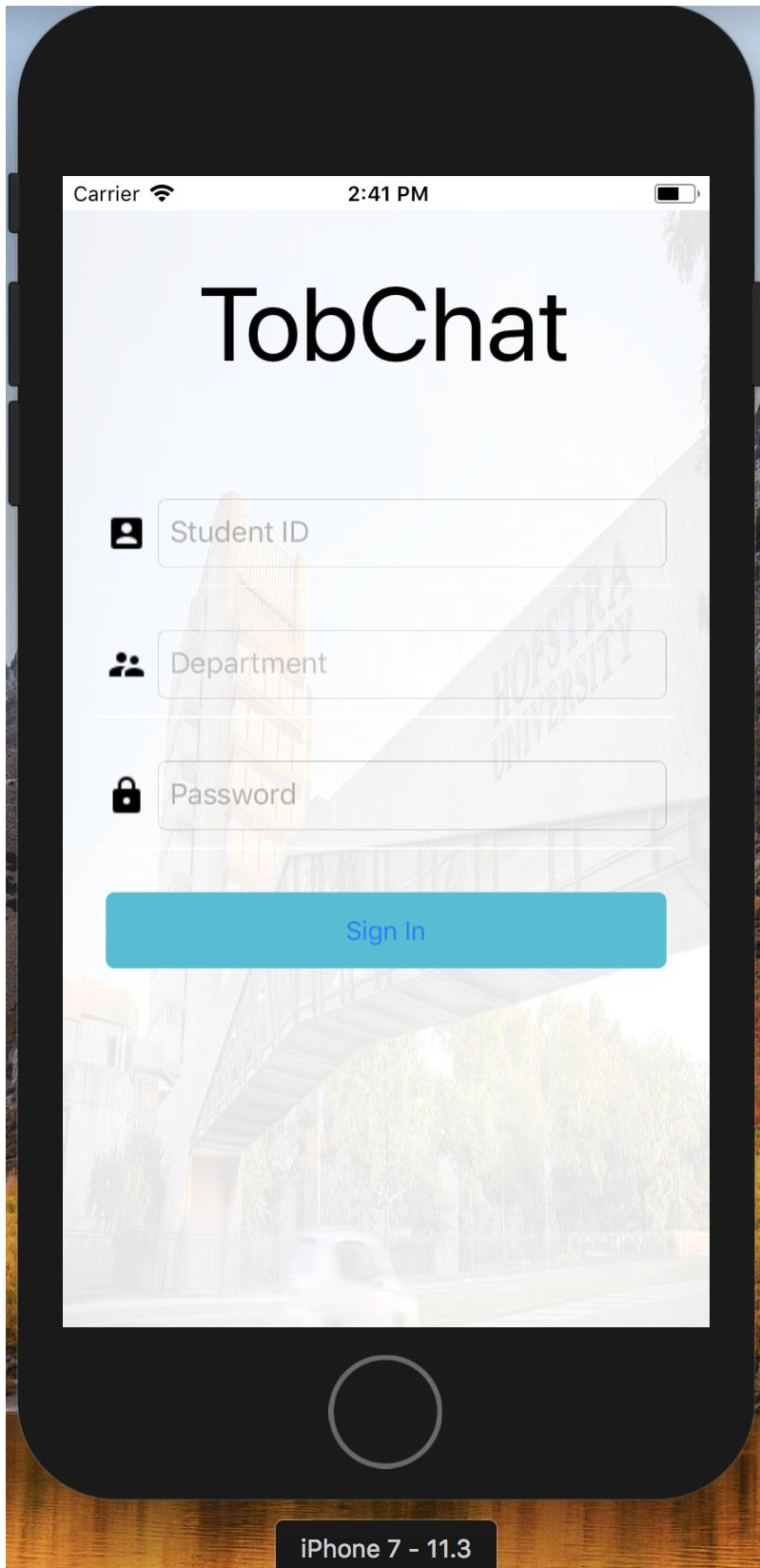


Figure 3.21 FinalCap iOS Mobile Application Project

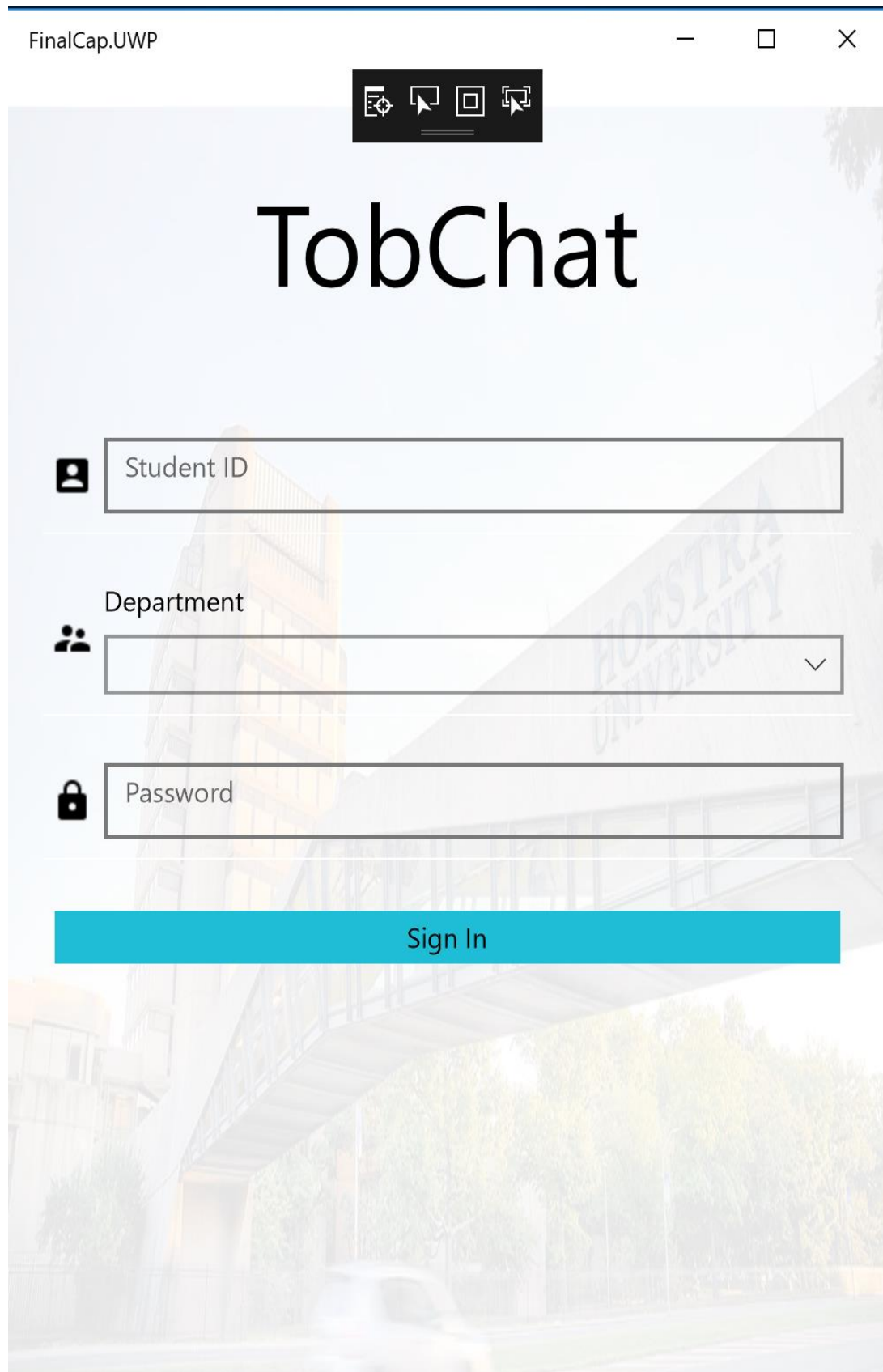


Figure 3.22 FinalCap Windows Phone Mobile Application Project

CHAPTER 4 – RESULTS AND ANALYSIS

This chapter presents the results and its analysis according to the programming codes. A short discussion is made on the methods used to achieve the results. Shortly, I will be discussing the software implementation measures taken to achieve the system design on individual technology. The overall collection of these technologies after been put together results in a solution for Digital Transformation. According to (Lankshear & Knobel, 2008), digital transformation in its ultimate stage means that digital usages inherently enable new types of innovation and creativity in a domain to stimulate a significant change, rather than simply enhancing and support traditional methods. In the context of this project, digital transformation is enabling institutions in the developing and under-developed countries go paperless by transforming how course documents are transferred from teachers to students for educational advancement. To implement this paperless solution, an approach which is paperless is mostly desired which result in software and hardware development. With this development, comes the need for different technology implementations which are discussed below.

4.1 Technologies Chosen to Design and Develop the System.

To design this software and hardware-based connectivity tool system, different sophisticated and modern techniques of computer programming and configurations must be used for this purpose. In the following sub-sections, I will illustrate how individual technology resulted in the desired solution and its analysis in the order of development.

4.1.1 Microsoft SQL Server 2014 Management Studio

According to (Mistry & Misner, 2010, pp. 3-10) as of the year of their publication, they argued that Microsoft SQL is the most advanced, trusted and scalable data platform. Given that I share their view, SQL became the database that serves as a file system in this project. To create a database, a Database Management Studio (DBMS) must be installed. The DBMS is a client software that connects to the database server. For this project, Microsoft SQL Management Studio is the DBMS used. Below figure shows the main interface of the Microsoft SQL Management studio program.

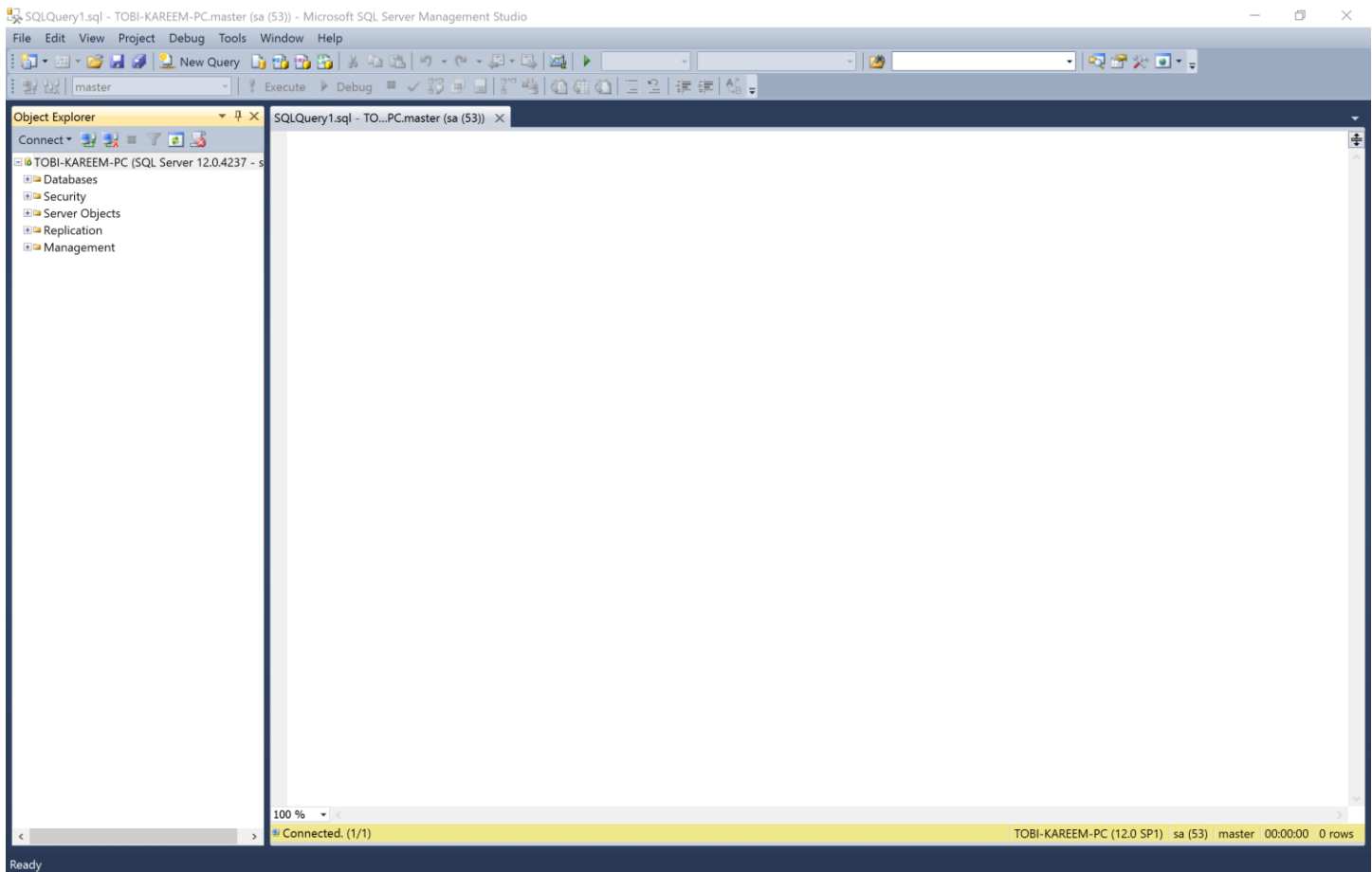


Figure 4.1 The main interface of the SQL Server Management Studio

The SQL server uses database keyword queries to produce results. The keyword below created a database with the name TobChat:

```
CREATE DATABASE TobChat
```

The code snippet below creates tables in the TobChat database.

Use TobChat

GO

-- Creating Student Table --

```
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'[dbo].[Student]'))
```

AND type in (N'U'))

BEGIN

CREATE TABLE Student

(

StudentID int IDENTITY(1000, 1) NOT NULL PRIMARY KEY,

FirstName varchar(55) NOT NULL,

LastName varchar(55) NOT NULL,

MiddleName varchar(55) NULL

)

END

GO

-- Creating Course Table --

*IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'[dbo].[Course]')*

AND type in (N'U'))

BEGIN

CREATE TABLE Course

(

CourseID *varchar*(55) *NOT NULL PRIMARY KEY*,

CourseName *varchar*(55) *NOT NULL*,

DepartmentID *varchar*(55) *NOT NULL*,

TeacherID *int* *NOT NULL*

)

END

GO

-- Creating Teacher Table --

IF NOT EXISTS (*SELECT* * *FROM* *sys.objects* *WHERE* *object_id* = *OBJECT_ID*(*N'[dbo].[Teacher]'*)

AND type *in* (*N'U'*))

BEGIN

CREATE TABLE *Teacher*

(

TeacherID *int* *NOT NULL PRIMARY KEY*,

FirstName *varchar*(55) *NOT NULL*,

LastName *varchar*(55) *NOT NULL*,

MiddleName *varchar*(55) *NULL*,

CourseID *varchar*(55) *NOT NULL*

)

END

GO

-- Creating Department Table --

IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'[dbo].[Department]'))

AND type in (N'U'))

BEGIN

CREATE TABLE Department

(

DepartmentID varchar(55) NOT NULL PRIMARY KEY,

DepartmentName varchar(100) NOT NULL,

Administrator varchar(55) NULL

)

END

GO

Below figure shows the result of the queries above.


```
SELECT CONCAT(TABLE_SCHEMA, '.', TABLE_NAME) AS Table_Name  
FROM INFORMATION_SCHEMA.TABLES  
WHERE TABLE_TYPE='BASE TABLE' AND TABLE_CATALOG='TobChat'
```

Rectangular Snip

100 %

Results Messages

	Table_Name
1	dbo.Student
2	dbo.Course
3	dbo.Teacher
4	dbo.Department
5	dbo.Mapping
6	dbo.sysdiagrams

Figure 4.2 Query showing table creation result

SQLQuery3.sql - TO...C.TobChat (sa (58))*

sp_help Student

100 %

Results Messages

	Name	Owner	Type	Created_datetime
1	Student	dbo	user table	2018-02-28 14:49:39.770

	Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation
1	StudentID	int	no	4	10	0	no	(n/a)	(n/a)	NULL
2	FirstName	varchar	no	55			no	no	no	SQL_Latin1_General_CP1_CI_AS
3	LastName	varchar	no	55			no	no	no	SQL_Latin1_General_CP1_CI_AS
4	MiddleName	varchar	no	55			yes	no	yes	SQL_Latin1_General_CP1_CI_AS
5	DepartmentID	varchar	no	55			yes	no	yes	SQL_Latin1_General_CP1_CI_AS
6	Password	varchar	no	50			yes	no	yes	SQL_Latin1_General_CP1_CI_AS

	Identity	Seed	Increment	Not For Replication
1	StudentID	1000	1	0

	RowGuidCol
1	No rowguidcol column defined.

	Data_located_on_filegroup
1	PRIMARY

	index_name	index_description	index_keys
1	PK_Student_32C52A79AB95CA92	clustered, unique, primary key located on PRIMARY	StudentID

	constraint_type	constraint_name	delete_acti...	update_acti...	status_enabl...	status_for_replicati...	constraint_keys
1	PRIMARY KEY (clustered)	PK_Student_32C52A79AB95CA92	(n/a)	(n/a)	(n/a)	(n/a)	StudentID

Query executed successfully.

TOBI-KAREEM-PC (12.0 SP1) sa (58) TobChat 00:00:00 15 rows

Figure 4.3 Student table information

A different database query is use to insert information into each table. The code snippet below shows how the information is added into the tables. To insert into each table, the concept of stored procedures was used.

USE TobChat

GO

-- Insert into Students --

CREATE PROC spInsertStudents

@Firstname varchar(55),

@Lastname *varchar*(55),

@Middlename *varchar*(55),

@Departmentid *varchar*(55)

AS

BEGIN

INSERT INTO Student (FirstName, LastName, MiddleName, DepartmentID) *VALUES*

(@Firstname, @Lastname, @Middlename, @Departmentid);

END

GO

-- Insert into Course --

CREATE PROC spInsertCourses

@CourseID *varchar*(55),

@CourseName *varchar*(55),

@TeacherID *varchar*(55),

@DepartmentID *varchar*(55)

AS

BEGIN

INSERT INTO Course (CourseID, CourseName, DepartmentID, TeacherID) *VALUES*

```
(@CourseID, @CourseName, @TeacherID, @DepartmentID);
```

```
END
```

```
GO
```

```
-- Insert into Department --
```

```
CREATE PROC spInsertDepartment
```

```
    @DepartmentID varchar(55),
```

```
    @DepartmentName varchar(100),
```

```
    @Administrator varchar(55)
```

```
AS
```

```
BEGIN
```

```
INSERT INTO Department (DepartmentID, DepartmentName, Administrator) VALUES
```

```
(@DepartmentID, @DepartmentName, @Administrator);
```

```
END
```

```
GO
```

```
-- Insert into Teachers --
```

```
CREATE PROC spInsertTeachers
```

@TeacherID int,

@FirstName varchar(55),

@LastName varchar(55),

@MiddleName varchar(55),

@DepartmentID varchar(55)

AS

BEGIN

INSERT INTO Teacher(TeacherID, FirstName, LastName, MiddleName, DepartmentID) VALUES

(@TeacherID, @FirstName, @LastName, @MiddleName, @DepartmentID)

END

GO

Below shows the result of the queries above for the Student table insertion by selecting all records in a student table after inserting the records, other tables have information as it was coded:

SQLQuery3.sql - TO...C.TobChat (sa (58))* X

SELECT * FROM Student

100 %

Results Messages

	StudentID	FirstName	LastName	MiddleName	DepartmentID	Password
1	1001	Obed	Ceesay	Jeremy	CSC	Obed
2	1002	Alex	Baddo	Hannah	DRA	Alex
3	1003	Paul	Adeolu	John	PSY	Paul
4	1004	Pordea	Sawyer	Anike	DRA	Pordea
5	1005	Manny	Pacqiou	Boxer	ENGR	MannPaq
6	1006	Tope	Kareem	Emmanuel	CHEM	Tope
7	1007	Mercy	David	NULL	CHEM	Chem

Figure 4.4 Result displaying Student record after insertion

After creating the table, there is a need to add constraints on the tables. Sql constraints are used to specify rules for the data in a table. The code snippet below shows how constraint was added into each table relationships.

USE TobChat

GO

-- Relationship between Student and Department --

```
IF NOT EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[Student_Department_FK]')
AND parent_object_id = OBJECT_ID(N'[dbo].[Course]'))
```

```
ALTER TABLE Student ADD CONSTRAINT Student_Department_FK
```

```
FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)
```

```
GO
```

```
ALTER TABLE Student CHECK CONSTRAINT Student_Department_FK
```

```
GO
```

```
-- Relationship between Course and Teacher --
```

```
IF NOT EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =  
OBJECT_ID(N'[dbo].[Course_Teacher_FK]'))
```

```
AND parent_object_id = OBJECT_ID(N'[dbo].[Course]'))
```

```
ALTER TABLE Course ADD CONSTRAINT Course_Teacher_FK
```

```
FOREIGN KEY (TeacherID) REFERENCES Teacher(TeacherID)
```

```
GO
```

```
ALTER TABLE Course CHECK CONSTRAINT Course_Teacher_FK
```

```
GO
```

```
-- Relationship between Course and Department --
```

```
IF NOT EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =  
OBJECT_ID(N'[dbo].[Course_Department_FK]'))
```

```
AND parent_object_id = OBJECT_ID(N'[dbo].[Course]'))
```

```
ALTER TABLE Course ADD CONSTRAINT Course_Department_FK
```

```
FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)
```

```
GO
```

```
ALTER TABLE Course CHECK CONSTRAINT Course_Department_FK
```

```
GO
```

```
-- Relationship between Teacher and Course --
```

```
IF NOT EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =  
OBJECT_ID(N'[dbo].[Teacher_Course_FK]'))
```

```
AND parent_object_id = OBJECT_ID(N'[dbo].[Teacher]'))
```

```
ALTER TABLE Teacher ADD CONSTRAINT Teacher_Course_FK
```

```
FOREIGN KEY (CourseID) REFERENCES Course(CourseID)
```

```
GO
```

```
ALTER TABLE Teacher CHECK CONSTRAINT Teacher_Course_FK
```

```
GO
```

```
-- Relationship between Teacher and Department --
```

```
IF NOT EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =  
OBJECT_ID(N'[dbo].[Teacher_Department_FK]'))
```

```
AND parent_object_id = OBJECT_ID(N'[dbo].[Teacher]'))
```



```
ALTER TABLE Teacher ADD CONSTRAINT Teacher_Department_FK
```

```
FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)
```

```
GO
```

```
ALTER TABLE Teacher CHECK CONSTRAINT Teacher_Department_FK
```

```
GO
```

4.1.2 Raspberry Pi for LAN Network

As discussed in the methodology section, the raspberry pi offers sophisticated and advanced features that helped created a host access point with a DHCP server. While the creation of the network does not require any coding or compilation, it does require a very basic knowledge of Linux and networking. After implementing the configuration earlier mentioned in its methodology, the raspberry pi is configured as a Wifi hotspot that allows multiple devices to connect to it and for every device that are connected, an automatic IP address is assigned to each device in the range of its specifications. This IP address assignment is done by the DHCP server. The Raspberry Pi enable Wifi access point and broadcast on the channel of your choice but for this project, the Wifi access is configured on channel six. The implementation of the configuration resulted in the below figure.

The Test!

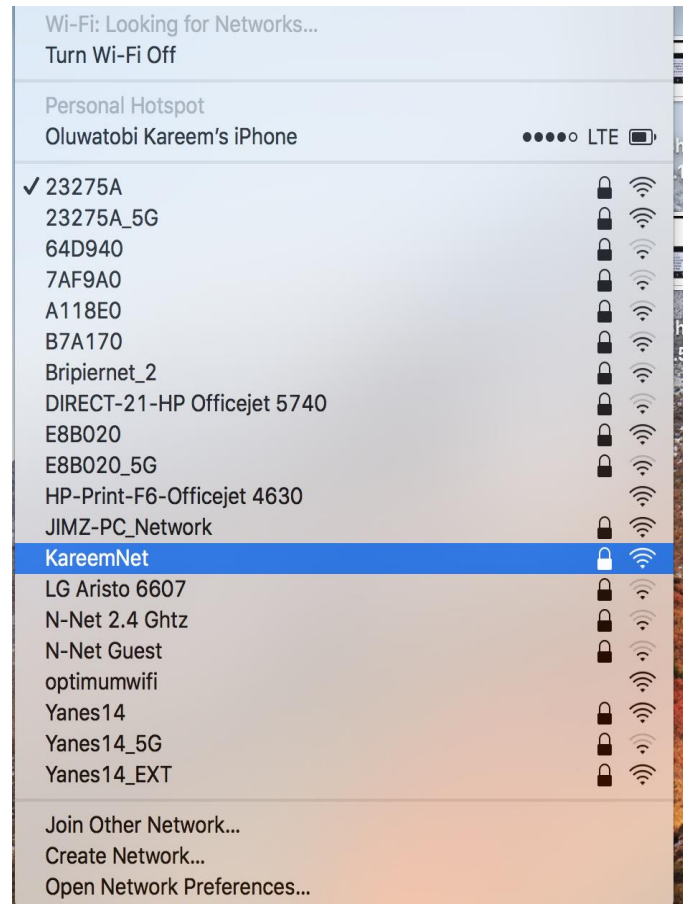
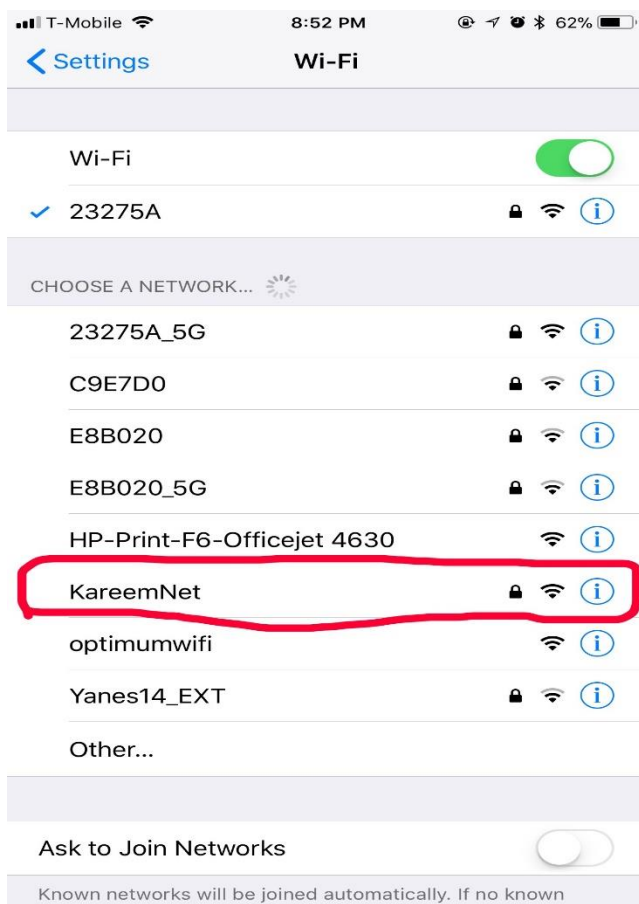
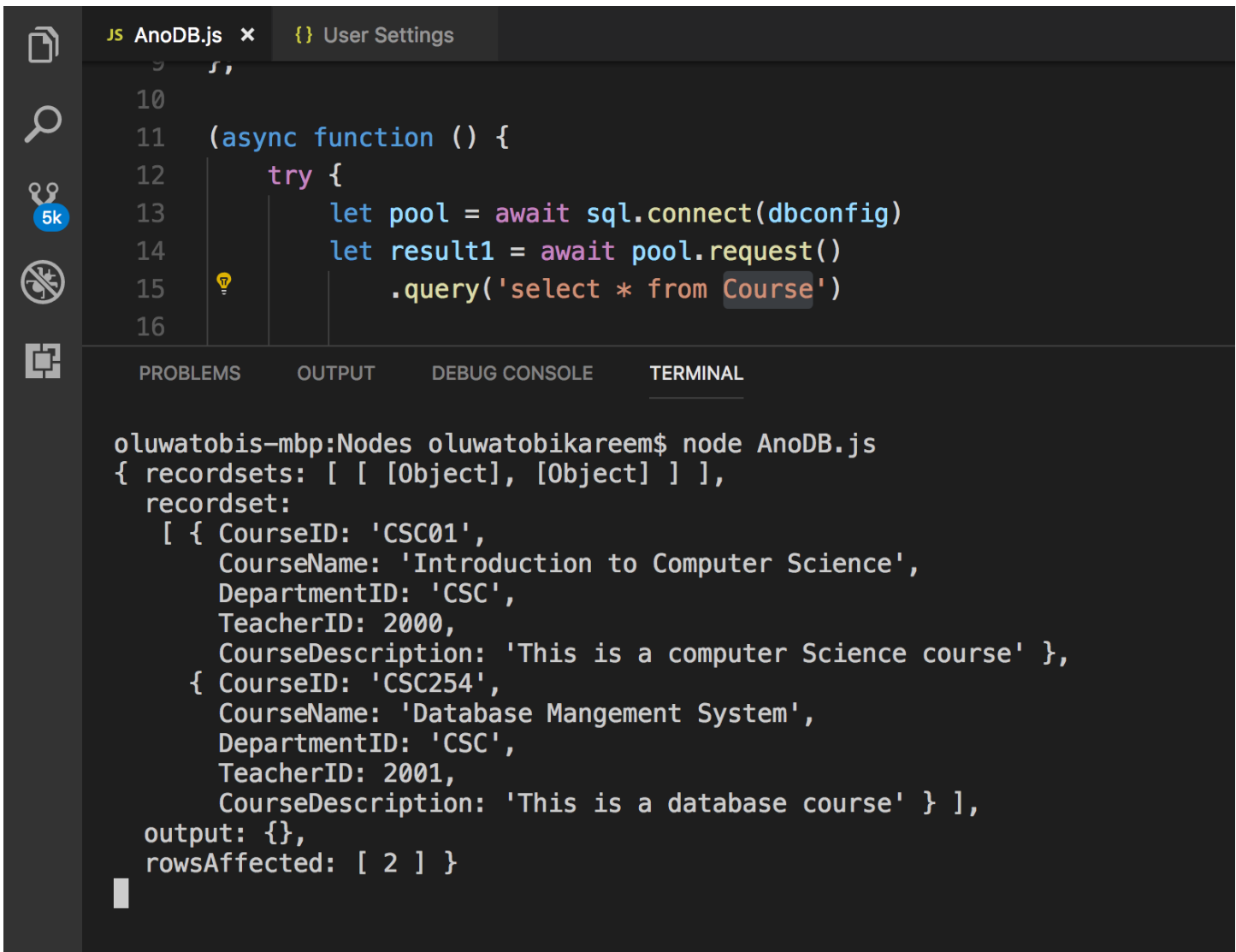


Figure 4.5 Using the raspberry pi to create a host access point.

4.1.3 Node.js Server

Node.js is a JavaScript framework for creating asynchronous event-driven, non-blocking input and output model. Node.js can create, open, read, write, delete and close file on a server. Added to these features are its ability to add, delete and modify data in a database. These advantageous features are evident during this project for remote connection to a database file system. The Node server is running on a remote laptop and it can fetch database resources from another remote laptop after connecting to the raspberry pi host access point. Certain configuration is a put in place to allow connection and authentication to the remote file system server. The remote file system server is an SQL server which is installed on a windows PC. The figure below shows the result acquired from finding information about the Course table in the TobChat database.



The screenshot shows a code editor with a file named `AnoDB.js` and a `User Settings` tab. The code in `AnoDB.js` is as follows:

```
10
11 (async function () {
12     try {
13         let pool = await sql.connect(dbconfig)
14         let result1 = await pool.request()
15         .query('select * from Course')
16     }
17 }
```

The terminal output shows the execution of `node AnoDB.js` and the resulting JSON data:

```
oluwatobis-mbp:Nodes oluwatobikareem$ node AnoDB.js
{ recordsets: [ [ [Object], [Object] ] ],
  recordset:
    [ { CourseID: 'CSC01',
        CourseName: 'Introduction to Computer Science',
        DepartmentID: 'CSC',
        TeacherID: 2000,
        CourseDescription: 'This is a computer Science course' },
      { CourseID: 'CSC254',
        CourseName: 'Database Mangement System',
        DepartmentID: 'CSC',
        TeacherID: 2001,
        CourseDescription: 'This is a database course' } ],
  output: {},
  rowsAffected: [ 2 ] }
```

Figure 4.6 Node.js remote connection to file system (Course data table)

4.1.4 ASP.NET MVC Web Development

The ASP.NET MVC in this project fulfills the purpose of building a web application for the project and a workspace for the administrator. The web page is hosted on a local server that is running in the LAN and can be retrieved by its appropriate IP address /URL. Here, the administrator of the system can exercise his or her privileges by adding, updating and removing records from and into the filesystem. The administrator can also view details of a record. The successful implementation of this code resulted in the figures below.

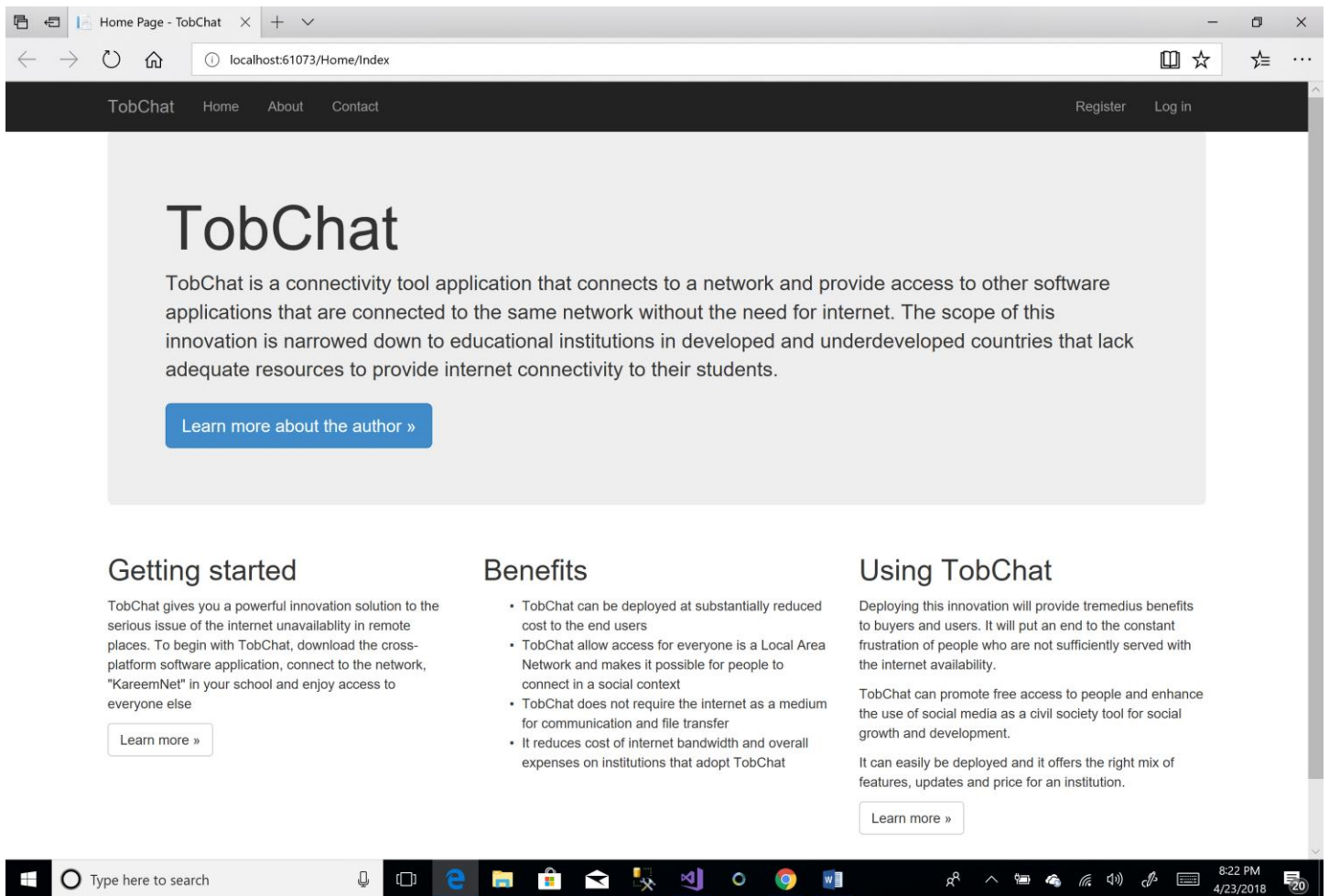


Figure 4.7 TobChatWeb Application Home Page

Figure 4.8 TobChatWeb Create New Student Web Page for Admin

4.1.5 Xamarin. Forms Mobile Applications

The mobile application is successful following its implementation from Chapter 3. It allows the user to sign in with their Id, department and password. The authentication is done based on the three requirements and the system uses inner join statements to list the courses of the student and/or the teachers. Clicking a course in the list displays the details and description of the course, the history of files and a download link. The result of this program can be seen in below figures.

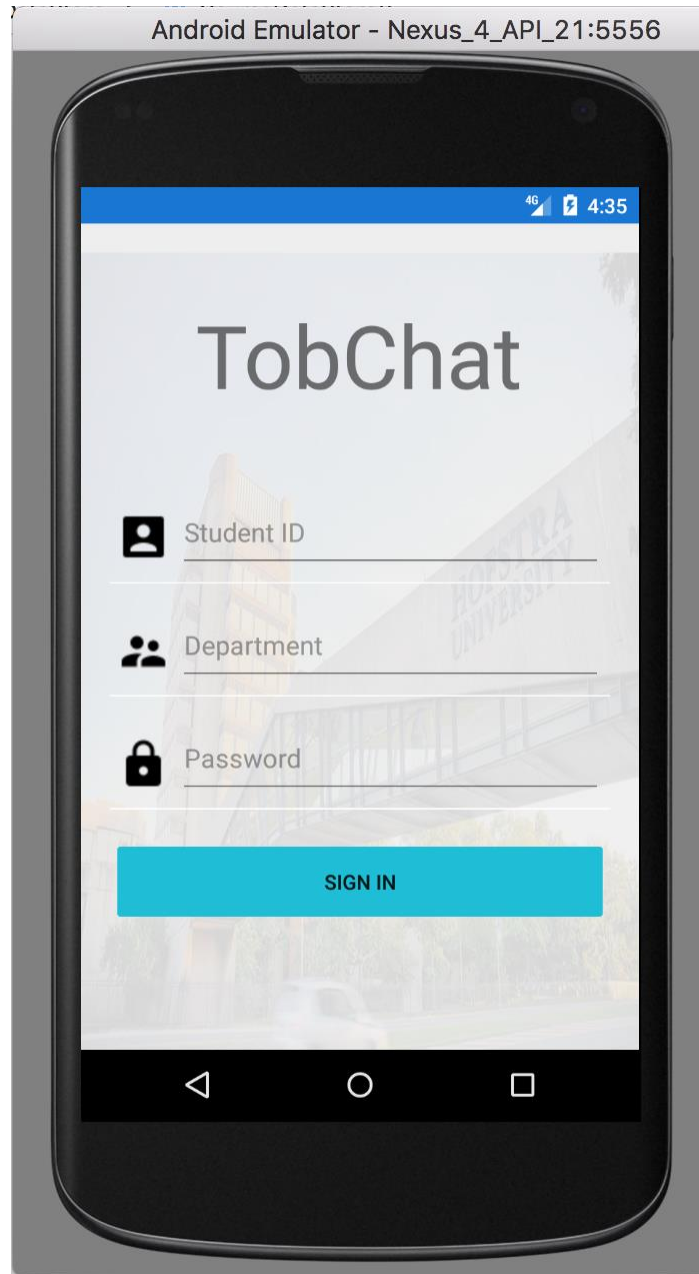


Figure 4.9 Sign in screen on Android device emulator

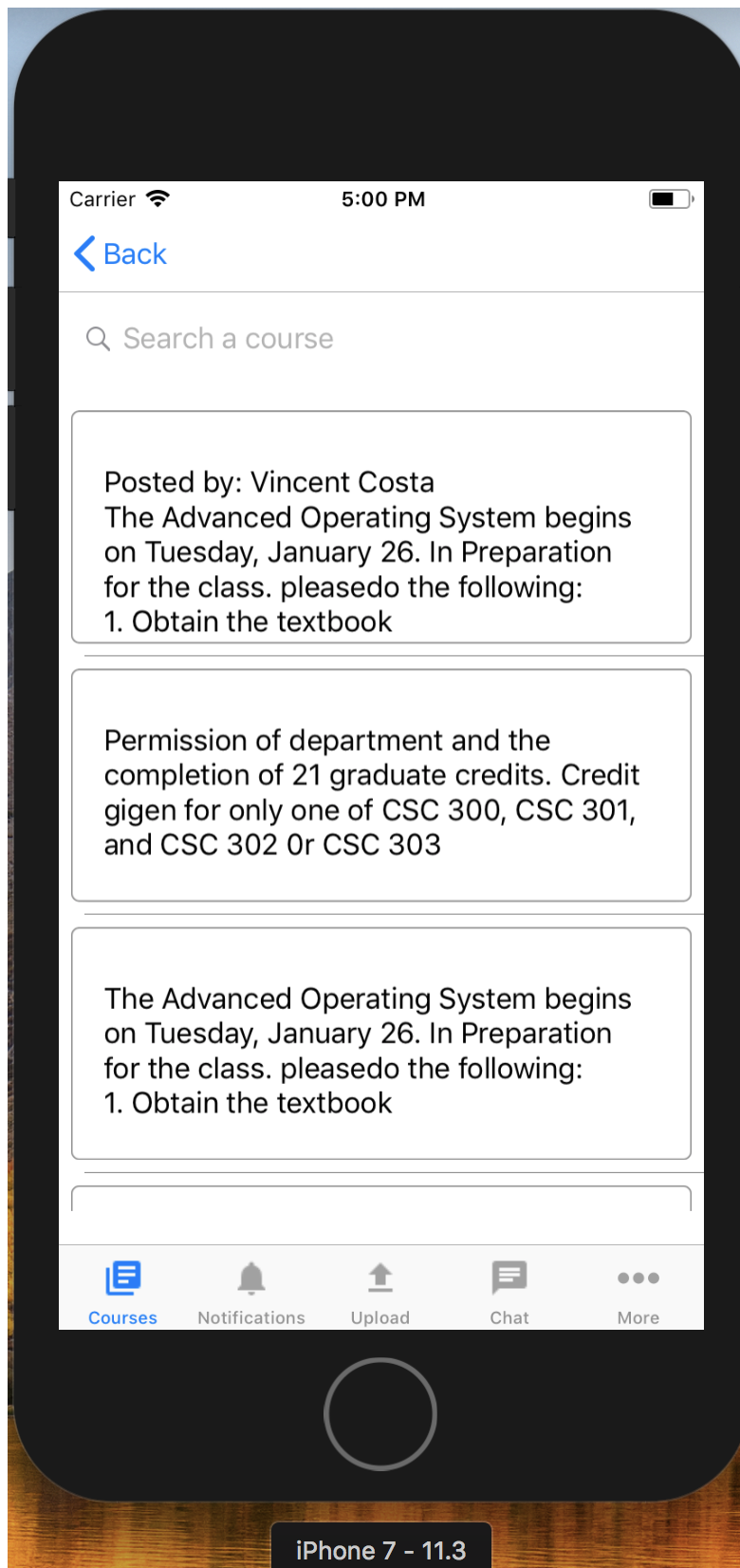


Figure 4.10 List of Courses screen on iOS device simulator

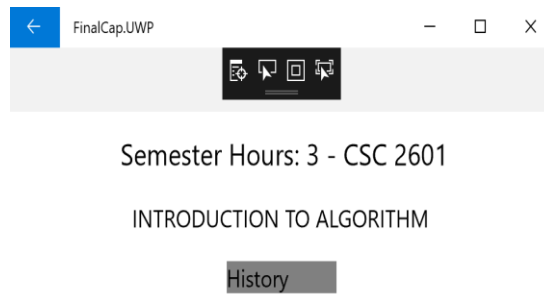


Figure 4.11 Course History screen on Windows Phone device emulator

4.2 Result Analysis

Different technologies such as networking technology and software engineering technology team up to make this project a solution to the problem discussed. The networking part in this project was done using a raspberry pi as a model for having a routing server that will be installed in the final production and delivery sense.

Following suite is the mobile application on the cross-platform devices that simulates the concept of the Blackboard as a content management system. The mobile application is tasked with delivery contents of educational materials electronically instead of the traditional way of distributing papers and hand-outs in these institutions. Students will sign into the mobile application using their student id, departments and password. After the sign in, the content appearing on the next screen is the list of the courses that are attributed to the student Id.

No two students can have the same identity and therefore, makes it easy to distinguish each student with the courses they take. If a student clicks on a particular course in the list of courses, such a student can see the course details, history of course documents that had been sent to the classroom by the teachers and such a student is able to download the course material. The software offers other features like forums, chat, and upload.

The Web application work in a fashion that is different from the mobile application. The web application is designed primarily for administrative functions. The administrator that has control over the system add, update and delete each student information. The role of the administrator is important in this system. Successful information entered by the administrator is what is used as a validation for the student and teacher access to the mobile application.

4.3 System Requirement for Mobile Application Use

This section lists the minimum hardware and software requirements to install and run the mobile application as a client.

4.3.1 Android Mobile Devices

- Install TobChat Mobile App (Installation source path would be provided)
- Minimum Android Version: Android 4.0.3 (API level 15)
- Any processor platform

4.3.2 iOS Mobile Devices

- Install TobChat Mobile App (Installation source path would be provided)
- ARMv7 with A6 processor
- iPhone devices running iOS 6.0 and later

4.3.3 Windows Phone Devices

- Install TobChat Mobile App (Installation source path would be provided)
- Any Windows Phone device

CHAPTER 5 - CONCLUSION

The present scope of the project was set out with the aim of providing a solution to the problem of inefficient internet connectivity in institutions in the developing and under-developed countries. As we observed in chapter two – Literature Review, while many software programs have emerged to connect different individuals from different locations, it is apparent that they all have one thing in common – the use of the internet as their medium of communication. The main objective and scope of this project was to design a connectivity tool by programming and configuring different computing technologies to achieve digital transformation without the need for the internet access. This objective as well as the functions for which the project is designed has been achieved. The system will enable lecturers to upload course materials into a centralized file system and the students will in turn, download these materials for their educational use. The implication of this is that, the traditional way of distributing course materials and handouts which is by printing papers and distributing them has been cut short. The system enables a clear and easy user interface and in addition, enhances positive educational experiences.

Digital Transformation can reduce time and effort required in the process of information dissemination. It takes too much time for paper materials to be distributed evenly among all members in a classroom and this also can be subject to misplacement of paper handout, rough handling and many other things. With the TobChat mobile application, the issues that arises with distributing papers are eradicated. The characteristics and other features of the systems are the essay provided in the purpose of this study.

5.1 Recommendations

The system implementation and use can further offer a great business model for institutions that adopt this technology because most of the internal communication need not go through the internet again and this will have significant decrease in the amount of internet bandwidth cost that is purchased from ISPs and hereby incurring more financial gains for the institutions.

Overall, this project contributes to the existing system designs for connectivity however, takes the scope in a different route by channeling the solution to a specific audience of individuals. Furthermore, the concept of this

design can also provide a foundation for future research into how information can be well disseminated among populations in areas where it is mostly required.

5.2 Limitations

It is important to note that this project narrows its scope for the final capstone. The project is aimed to be expanded in the future and it will cover many other concerns of students offering double or more majors. Due to the time restraint of this work, the project remains a work in progress and the system has not been tested. The budget for the full implementation is not discussed.

References

- Abbate, J. E. (1996--1988). *From ARPANET to Internet: A history of ARPA - sponsored computer networks*. Retrieved from Scholarlycommons: <https://repository.upenn.edu/dissertations/AAI9503730>
- Abhishek, P. (2017, December 28). *Using apt-get Commands In Linux [Complete Beginners Guide]*. Retrieved from It's FOSS: <http://www.itsfoss.com/apt-get-linux-guide/>
- Branzburg, J. (2007). Talk is Cheap: Skype Can Make VoIP a Very Real Communication Option for Your School. *Technology & Learning*, V27, 36.
- Carnevale, D. (2003). *Study of Wisconsin professors finds drawbacks to course management systems*. Wisconsin: Chronicle of Higher Education.
- Caukin, J. (2012, March 5). *35 Million People Concurrently Online on Skype*. Retrieved from The Big Blog: http://web.archive.org/web/20120308033805/http://blogs.skype.com/en/2012/03/35_million_people_concurrently.html
- Date, C. (2000). *An Introduction To Database System*. Reading, Massachusetts: Addison-Wesley Longman, Inc.
- edx, M. (2018, February 18). *Wireless standards*. Retrieved from Microsoft edx.org: <https://courses.edx.org/courses/course-v1>
- Fredrik, J. (2013, February 19). *Wifi access point with Raspberry Pi*. Retrieved from it's a clean machine: <http://www.itsacleanmachine.blogspot.com/2013/02/wifi-access-point-with-raspberry-pi.html>
- GET-IT, g. (2015, November 18). *systematic review*. Retrieved from GET-IT Glossary: getitglossary.org/term/systematic+review
- Gmail. (2016, November 25). *Security and privacy*. Retrieved from Google: <https://support.google.com/mail/answer/7039474>

- Goodin, D. (2012, May 1). *Skype replaces P2P supernodes with Linux boxes hosted by Microsoft (updated)*. Retrieved from ars Technica: <https://arstechnica.com/information-technology/2012/05/skype-replaces-p2p-supernodes-with-linux-boxes-hostwd-by-microsoft/>
- Jeff, H., Ramesh, V., & Heikki, T. (2016). *Modern Database Management*. Essex, England: Pearson Education Limited.
- Johnson, M. (2018, January 12). *Software Engineering*. Retrieved from Node Today: <http://www.morriahjohnson.com>
- Kelley, S. (n.d.). *Dnsmasq*. Retrieved from thekelleys: www.thekelleys.org.uk/dnsmasq/doc.html
- Lammle, T. (2007). *Cisco Certified Network Associate*. Indiana: Wiley Publishing, Inc.,
- Lankshear, C., & Knobel, M. (2008). *Digital literacies: Concepts, Policies and Practices*. New York: Peter Lang Publishing, Inc.
- Microsoft. (n.d.). *ASP.NET MVC Overview*. Retrieved from Microsoft Developer Network: [https://msdn.microsoft.com/en-us/library/dd381412\(v=vs.108\).aspx](https://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx)
- Miller, R. (2016, February 1). *Gmail now has 1 billion monthly active users*. Retrieved from The Verge: <http://www.theverge.com/2016/2/1/10889492/gmail-1-billion-google-alphabet>
- Mistry, R., & Misner, S. (2010). *Introducing Microsoft SQL Server*. Redmond, Washington: Microsoft Corporation.
- Peter Bradford, M. P. (2007). The Blackboard Learning System. *The Journal of Education Technology Sysstems*, 35:301-314.
- Pfleeger, S. L. (1991). *Software Engineering - The Production of Quality Software*. New York: Macmillan Publishing.
- Picolsigns (2009). History of Internet [Recorded by S. Taylor]. United States.

- Rory, C.-J. (2011, May 5). *BBC News dor.Rory* . Retrieved from A 15 Pound computer to inspire young programmers:
www.bbc.co.uk/blogs/thereporters/rorycellanjones/2011/05/a_15_computer_to_inspire_young.html
- Shontell, A. (2011, August 17). *A Year In The Life Of An \$85 Million Startup, GroupMe*. Retrieved from Business Insider: <http://businessinsider.com/groupme-2011-8#present-groupme-turns-one-with-millions-of-users-in-90-countries-16>
- Skype. (2003). *About Skype*. Retrieved from Microsoft Skype: <http://www.skype.com/en/about/>
- Steve, S., & Luke, L. (2017, December 12). *Views in ASP.NET Core MVC*. Retrieved from Microsoft ASP.NET Core 2.1 Preview 2: <https://docs.microsoft.com/en-us/aspnet/core/mvc/views/overview?view=aspnetcore-2.1>
- Suprotim, A. (2008, March 15). *Exploring the Global.asax file in ASP.NET*. Retrieved from et curry.com: <http://www.dotnetcurry.com/aspnet/126/aspnet-global-asax-events-methods>
- Tallinn, J. (2014, December 1). *Affairs Today*. (C. G. Ferdinand, Interviewer)
- Telegraph, T. D. (2017, December 13). *Facebook: a timeline of the social network*. Retrieved from The Telegraph: telegraph.co.uk/technology/facebook/9052743/facebook-a-timeline-of-the-social-network.html
- Timeline of Computer History*. (2018). Retrieved from Computer Histoty: <http://www.computerhistory.org/timeline/>
- w3schools.com. (n.d.). *Node.js and Raspberry Pi*. Retrieved from w3schools.com: https://www.w3schools.com/nodejs/nodejs_raspberrypi.asp
- Wikipedia. (2017, December 26). *Time-sharing*. Retrieved from Wikipedia.org: <https://en.wikipedia.org/wiki/Time-sharing>

Xamarin.Forms. (n.d.). *Xamarin.Forms Namespace*. Retrieved from Xamarin:

<https://developer.xamarin.com/api/namespace/Xamarin.Forms/>

Zivkovic, M. (2016, February 18). *How to connect to a remote SQL Server*. Retrieved from SQLShack:

<http://www.sqlshack.com/how-to-connect-to-a-remote-sql-server/>