SkyGate – A Wireless RFID Authentication System

by

Tan Mao Cheng

A dissertation submitted to the

Electrical and Electronics Engineering Programme

Universiti Teknologi PETRONAS

in partial fulfillment of the requirement for the

BACHELOR OF ENGINEERING (Hons)

(ELECTRICAL AND ELECTRONICS ENGINEERING)

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

June 2009

CERTIFICATION OF APPROVAL

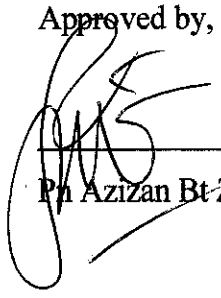
Skygate – A Wireless RFID Authentication

by

Tan Mao Cheng

A project dissertation submitted to the
Electrical and Electronics Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
BACHELOR OF ENGINEERING (Hons)
(ELECTRICAL AND ELECTRONICS ENGINEERING)

Approved by,



Azizan bt. Zainal Abidin
Senior Lecturer
Electrical & Electronic Engineering Department
Academic Block 20-03-11
Universiti Teknologi PETRONAS
Tel: 05-3687672 Fax: 05-3655905

Pn Azizan Bt Zainal Abidin

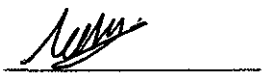
UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

June 2009

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.



TAN MAO CHENG

ABSTRACT

This dissertation is written with the objective of showcasing the feasibility of coupling RFID and Wi-Fi technology and attaching a RDBMS database backend to allow for an effective, simple, robust and efficient authentication control network. This report contains 5 chapters in total. The first chapter will present the problem statement that had led to the attempt to develop the project, as well as defining the project's objective and the limits of its scope as a Final Year Project. Meanwhile, the second chapter will provide a literature review concerning the implementation of RFID/Wi-Fi hybrids on the field. Sources will mainly be based on literature by Syed Ahson, Mohammad Ilyas, Stephen B. Miles, Sanjay E. Sarma & John R. Williams. Chapter 3 highlights the methodology taken during the implementation of the project and provides an updated and more in-depth description of all the technologies involved. This will cover all three aspects of the project which are RFID technology, Wi-Fi networking as well as its underlying software programming construct. In Chapter 4, a report on the results of the project is given detailing completed project components as well as reporting on the problems encountered while implementing the project. Chapter 5 will put forth a conclusion for this report as well as concerning the project itself.

ACKNOWLEDGEMENT

I wish to express my deep appreciations and thanks to those who have contributed to this study and design project. Firstly, I want to express my sincerest appreciations to my Finaly Year Project supervisor, Pn Azizan Bt Zainal Abidin for the immeasurable support she had given as well as the guidance, comments and suggestions through the period of the project. This thesis would not see completion without her kind guidance and support.

I would also like to thank Universiti Teknologi PETRONAS for providing the facilities, equipments and monetary support necessary for the successful completion of this project.

Lastly but not least, a heartfelt thanks goes to my family members and friends for their support and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

UTP ..... Universiti Teknologi Petronas

RFID ..... Radio Frequency IDentification

IP ..... Internet Protocol

IFF ..... Identification of Friend or Foe

RF ..... Radio Frequency

SQL ..... Structured Query Language

WPF ..... Windows Presentation Foundation

PHP ..... Hypertext Pre-Processor

VB.NET ..... Visual Basic .NET

GUI ..... Graphical User Interface

DLL ..... Dynamic Link Library

WPA-PSK ..... Wireless Protected Access - Pre-Shared Key

LTS ..... Long Term Support

HTTP ..... HyperText Transfer Protocol

HTTPS ..... HyperText Transfer Protocol Secure

CHAPTER 1
INTRODUCTION

1.1 - Background of Study

Looking at UTP's current system for authentication and access control within its facilities, it can be seen that automated authentication is used widely throughout all its modern compounds. However, control of access for each of the facilities is decentralized, as is generally the case for many current implementations throughout the country. Therefore, each locking system acts as independent entities separate from each other thereby preventing cohesive control amongst the various units.

In addition to that, the current implementation allows authentication only in a very broad sense whereby access is segregated by groups rather than the finely tuned architecture of individual authentication. This gives the current system little flexibility both in terms of logistics and management as well as negating any possible improvements regarding automated control.

This is further brought down by the fact that each locking system needs to be programmed separately rendering large changes to the system to be an unwieldy effort. This can pose a serious logistics and management problem when massive changes to its access database are required to be done due to its decentralized nature.

1.2 - Problem Statement

From observations made especially on academic facilities such as the university compounds of Universiti Teknologi PETRONAS (UTP), the problems with the current system can be narrowed down to the following:-

- Decentralized locking system.
- No fine-tuned control over each individual user's access.
- Not flexible and prohibits reprogramming to allow new access rules.
- No direct system overview poses serious logistics and management problems.

1.3 - Objective

The objective of this Final Year Project is to design a functional prototype or technology demo for a 'Centralized Wireless Locking, Authentication and Administration System via RFID Technology through Wi-Fi over TCP/IP (Transmission Control Protocol/Internet Protocol)'. This is subsequently rebranded upon request to the ***'SkyGate'*** system for a more user friendly reference phrase. The implementation of such a system will enable a centralized access database to be created containing access credentials as well as automated operations. Both this info can be processed and used to enable not only fine grained control of access to any particular facility, but it can also be configured to enable automatic deployment of any facility in a given span of time without manual intervention.

The database systems is connected through Wi-Fi that allows fine grained control over access rules as well as provide a platform for mass administration of facility access control throughout the university. This will enable individual system units to be unwired and spread out throughout the Wi-Fi cloud without much hassle. Additionally, the Wi-Fi cloud will grant remote management capabilities to the system by allowing direct remove access to the database through a developed web front-end. From there, on-the-field configuration and access requests can become a reality due to the synergistic use of the Wi-Fi cloud for both standard Wi-Fi devices and the actual access control system. Facility access can be given on a personal basis as opposed to group access although such functionality will be available as well for ease of management.

With client-server connectivity being established by the Wi-Fi cloud, RFID technology is used to deliver the authentication mechanism to the system. RFID readers connected to the client computers will be placed at the entrances of the facility for authentication purposes. System users will be provided with a passive RFID tag that will be scanned to process their authentication with the central database server via the connected client systems. This can be done by querying access data based on the RFID's serial number ID by cross-referencing the database for its credentials. The client system will be fed-back with the access credentials and will be programmed to act accordingly based on the information received.

1.4 - Scope of Study

The scope of project will at least cover the components below:

- Design of a centralized database application for data storage and retrieval.
    - Stores user profile for all its users uniquely identified by their own RFID serial number ID which is unique for all RFID tags.
    - Stores client location (IP) and access points under its jurisdiction.
    - Stores data for automated tasks such as door manipulation at a specific programmable interval or time span.
- Design of an administration panel capable of both web-based remote management as well as traditional Microsoft Windows based interface.
    - Administration panel capable of being run from any computer connected to the server's network.
    - Two flavors, full-fledged Windows based administration panel and a web-based access panel allowing raw editing of the database through MySQL queries by authenticated users.
- Design of client application for on-the-field control of facilities' locking and access control as well as RFID reader management interface.
    - Can be considered a simple version of the administration panel.
    - Autonomously controls access points under its jurisdiction based on data queried from the central database.
    - Connected to the server's network via Wi-Fi.
    - Also tasked with receiving RFID data from placed readers and taking care of the generated event.
- Implementation of a Wi-Fi communication medium to enable wireless communication between client computers on-the-field and the central administration server for access authentication and mass operation purposes.

The above list will cover the basic features of the proposed project. Additional features may be added if deemed feasible to increase the value and quality of the project throughout its development period.

CHAPTER 2
LITERATURE REVIEW

2.1 - Introduction to RFID (Radio Frequency Identification)

In areas where control of access to a specific area is of importance, many different techniques and technologies had been developed and applied in the field ranging from traditional measures such as lock and key to advanced devices such as biometric sensors. One particular technology that had been gaining popularity for such implementations is the usage of RFID systems to handle the authentication and access control of a particular facility. This is in addition to the other widespread uses of RFID especially in the areas of logistics and commodity transfer.

By itself, RFID technology had proven itself useful wherever the tagging of items is concerned and is poised to replace the traditional bar-code system wherever it is cost effective to do so. However, the fundamental characteristic of RFID's tracking and tagging capability had also raised concerns regarding privacy amongst privacy advocate groups. Returning to the example of an RFID based access control system, such a system can easily be developed or modified to not only manage access but to also track these access events to give a clear historical view of a person's movement within the facility.

2.2 - RFID History

RFID technology was first developed and applied together with radar technology during the World War II for IFF (Identification of Friend or Foe) systems.[1] This application consists of two elements namely the RF transponder (tag) and the interrogator (reader). During the time, it was designed for the detection of friendly airplanes as opposed to those of the enemy's. Meanwhile, for passive RFID technology, it has its roots in electronic article surveillance or EAS systems that were deployed in retail stores in the 1970s. This was used for anti-theft detection within the stores itself. RFID utilization basically has the methodology as described below:-

1. Generate, process and couple signals to an electromagnetic environment through the use of a transducer or an antenna;
2. Another transducer receives the electromagnetic signals from the environment and converts it to voltage and current.

This is applicable for an active RFID system since the source signals are generated. For a passive system, the tag doesn't generate its own signals but reflects information from the reader back to it. An illustration of the methodology is given in Figure 1.

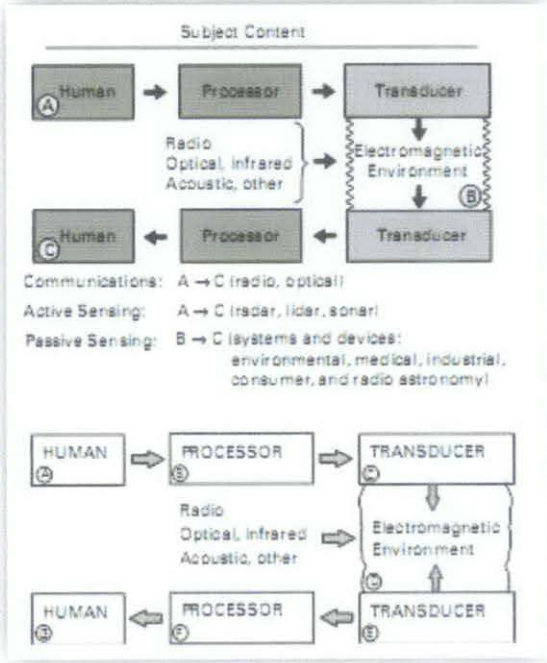


Figure 1: Receivers, antennas, and signals [2]

2.3 - Modern RFID Applications

In current times, RFID technology is divided into 4 different categories which are passive, active, semi-passive and extended capability. However, only the active and passive variety had acquired a large following. The main differentiator between these classes is the presence of an internal power supply used within the RFID tag. Passive RFID tags have no internal power supply. It functions by having current being induced in its antenna by an incoming frequency signal generally from the interrogator/reader device. This induced current provides just enough power for the CMOS integrated circuit within the tag to power up and transmit a response. Due to the limited power being generated tag-side, most passive tags communicate with the reader through backscattering of the carrier wave from the reader. Such tags don't have a large range characteristic with distances ranging from 11 centimeters in the near-field up to 10 meters in far field. However, coupled with phased array transmission, such tags are demonstrated to have a maximum range of 183 meters. Most passive tags being manufactured today are designed for operation within the 13.56 MHz range as defined in the ISO standard namely ISO 18000-3.

While passive tags aren't internally powered, active tags categorize the exact opposite by defining RFID tags with its own internal power source. The presence of an internal power supply greatly increases the tag's capability at the expense of size and portability. Since active tags have an onboard power supply, it is capable of transmitting at much higher power levels thereby increasing its potential range to hundreds of meters. Additionally, active tags can afford to have much larger memories as well as specialized sensors attached such as temperature, humidity, shock or vibration sensors as per needed by its designed application. The drawbacks of active RFID systems are its increased size due to its power pack as well as its higher manufacturing price. However, an interesting fact to note is that that despite its reliance on a battery source, its shelf life is considered comparable to passive tags since the corrosion of aluminated printed circuit competes with the batteries' self discharge rate.

2.4 - RFID Security

An RFID system is normally deployed with 3 different entities: tags, readers, and a back-end server.[3] Due to its separate nature, every layer of communication between the separate entities is open to subversion which could compromise the privacy of its users. These layers can be distinguished into 3 layers which at the lowest would be the physical layer involving the RF coupling element, the data link layer, and the application layer.[4] Therefore, such a system can be insidiously subverted to acquire private 'positional' data of its users.

On the other hand, whenever such hostile subversions such as the exploits above are mitigated, there also exists the fundamental problem with the design itself which 'legally' stores user data and movement within its database. Therefore, in addition to considering the threat of privacy compromises by outsiders, an implementation must also evaluate the potential for abuse by the system's operators and maintainers. Such privacy concerns are referred to as 'Big Brother' privacy concerns.[5]

Nonetheless, for this project, it is worthwhile to exempt the back-end server from suspicion while only considering the threat of insidious subversion of the system itself. This is due to the fact that privacy concerns can be mitigated by various means such as by storing only trivial information in the database as well as preventing it from keeping the access logs of its users. By storing only trivial information, the negative effect of a compromise can be reduced despite no increase in the design of the system's security measures.

2.5 - RFID/Wi-Fi Hybrid Applications

Looking into the scope of this project, one unique side to it is the use of Wi-Fi technology for the client-server interconnects of the system. While RFID and Wi-Fi are both rather mature technologies on its own, the integration of both these technologies together has not been a widespread practice before 2007. This is highlighted by Beth Bacheldor of RFID Journal in her article 'Wi-Fi Based RFID Improves Elderly Care' where active RFID tags coupled together with Wi-Fi gives quick and easy access to help for its users. It is interesting to note that Wi-Fi/RFID hybrids had until now only being applied in medical facilities with little documentation of it being applied in other fields. Despite this, Gerry Blackwell in his article 'Tuning In to RFID and Wi-Fi' points out that there is a growth in applications outside asset and people tracking, especially when extra information is provided by the RFID tags. This can be seen in its usage to track temperature data for prescriptions storage to ensure compliance to medical statutes.

In most cases, the implementation of a Wi-Fi/RFID system outlines the cooperative nature both technologies have to synergistically solve a particular problem. It not only extends both the range and effectiveness of both wireless technologies but also opens up new fields of implementation for both these systems. For this Final Year Project, applications of hybrid RFID/Wi-Fi for centralized authentication and access control management is a novel project that had not, to the best of the author's knowledge, been done before. Such an application will therefore not only introduce new methods for access control but may also prove to be an expansion of RFID application fields when coupled with Wi-Fi technology.

2.6 - Wi-Fi (Wireless Fidelity)

The main networking medium for this design project, Wi-Fi is a popular wireless networking technology covered under the various IEEE 802.11 technologies. As of now, Wi-Fi uptake had proliferated to various electronic devices and is supported by nearly every modern PC operating systems; most advanced gaming consoles and even includes laptops, printers as well as other peripherals. In many cases, Wi-Fi is used to provide wireless access to digital content be it applications, multimedia, the Internet, or various other generic digital data. This not only allows mobility of both servers and clients within the system but also removes the physical restraints imposed by wiring. With that, such a system is easy to grow despite limited physical connection infrastructure.

Wi-Fi uses single carrier direct-sequence spread spectrum radio technology which is a part of the larger family of spread spectrum systems. Additionally, it also utilizes the multi-carrier Orthogonal Frequency Division Multiplexing radio technology. Spread-spectrum techniques are methods by which electromagnetic energy generated in a particular bandwidth is deliberately spread in the frequency domain. Therefore, the signal will have a wider bandwidth as compared to one which is focused on a particular frequency whereby it contains signal structuring techniques that employs direct sequence, frequency hopping, or a hybrid of these which can allow multiple accesses or functions. Additionally this technique decreases potential interference to other receivers while also achieving privacy. In the case of Wi-Fi, spread spectrum allows Wi-Fi to occupy 13 channels over the 2.4GHz band. However, only 5 of the channels are used which are channels 1, 3, 5, 9 and 13.

The range of a typical router is normally around 32 meters indoor and 95 meters outdoors. Additionally, range also varies with the frequency band. Wi-Fi in the 2.4 GHz block (802.11b/g) has slightly better range than Wi-Fi in the 5 GHz block (802.11a). However, in the case of Long-range Wi-Fi, it is used for low-cost, unregulated point-to-point connections as an alternative to cellular networks, microwave or satellite links. Main applications of long range Wi-Fi include providing coverage to a large office or business complex or campus, establishing point-to-point link between large skyscrapers or office building as well as allowing the design of a neighborhood Wi-Fi network. Amongst the applications given, the first example would be a good fit to this project of implementing Wi-Fi as a propagation network of RFID authentication data throughout the UTP campus.

As is in this project, Wi-Fi is typically used to overcome the shortages of traditional wired LAN (Local Area Networks) by allowing the system to be deployed without cabling for client devices. This allows a reduction in network deployment cost as well as allowing rapid expansion. The typical range of a single router is around 32 meters indoors and 100 meters outdoors. However, this can be easily accounted for by the placement of signal repeaters around the routers to boost the range of the signals. Another ideal alternative is through the use of an improved directional antenna that can be used at a range of several kilometers or more with line of sight. This is especially so if a region had been targeted beforehand for the usage of Wi-Fi.

Looking at the objective of this project, a directional large area antenna would be the most suitable Wi-Fi router that can be implemented. However, due to the limited scope of the project as well as its budget limitations, an implementation using a Wi-Fi system hosted on standard routers is a more preferable option. Such an implementation would only have a practical difference of range and coverage area but the actual Wireless LAN will remain unchanged for prototyping purposes.

2.7 - Relational Database Management Systems (RDBMS)

A database management system is defined as computer software that manages databases. Meanwhile, a database is a structured collection of data or records that are stored on a computer system. In this case, the structure can vary according to which database model it is being implemented. Many database models had developed over time beginning with the development of the Navigational model in the 1960s. In 1969, E. F. Codd first formulated and proposed the relational model for database management.[6] The relational model is heavily based on First-order Logic which is a formal deductive system used in a wide range of applications ranging from mathematics to philosophy, linguistics and computer science.

In the case of databases, the fundamental property of a relational model database is that all data is represented as mathematical n-ary relations, with any n-ary relation being a subset of the Cartesian product of n domains. With that, data are being operated upon through the use of relational calculus or relational algebra, thereby sharing its advantage in expressive power. By having a relational model to data, database designers are therefore able to create a consistent logical representation of information.

The main purpose of the relational database model is to provide a declarative method for specifying data and queries. In this case, the user is able to directly state what information the database will contain and what information the user wants from it. Since the 1980s, the method for such declaration had been standardized into what had been known as SQL (Structured Query Language). Through the use of SQL statements sent to the database management system software, the user is able to manipulate the database as well as retrieve information from it as defined and constraint within the SQL queries being sent.

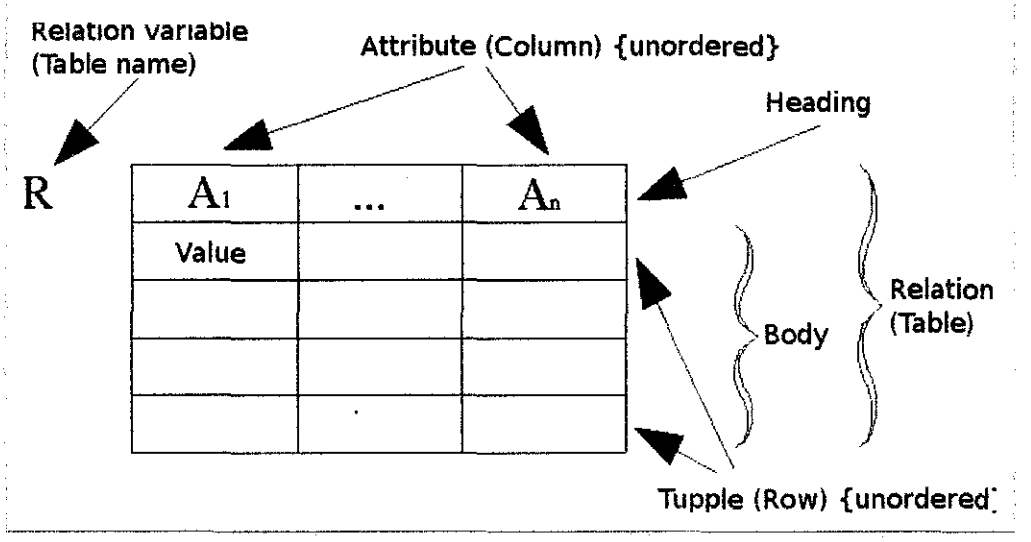


Figure 2: Relational Model Concepts [7]

Figure 2 demonstrates basic building blocks within a relational database is known as a data type, or more modernly known as just 'type'. Meanwhile, a 'tuple' is an unordered set of attribute values while an 'attribute' is an ordered pair of attribute name and type name. A relation consists of a 'heading' and a 'body' whereby a heading is a set of attributes while the body is a set of n-tuples. The heading of a relation is also considered the headings of each of its tuples. With that, a relation is defined as a set of n-tuples.

In order to understand this relation, a table with column headers is a good visual representation of a relation. In this form, attributes are the individual columns of the table, whereby the columns are named according to the headings. The body is the part of the table containing the data rows with each row being a tuple in the relational model. The name of the table would be the relational variable while the data or value of the relational model is stored in the cells within the table. In the relational model, both the columns and the rows within the relation are considered unordered. However, many database management systems impose a specific order in order to simplify calculation and present a more manageable view of the data once queried.

When the relational model is applied to databases, many of the terms are redefined in order to make them more applicable in a database system. In database applications, a 'type' may be any computer object type such as integers, a set of character strings. It can even be a set of dates or Boolean expressions and so on as implemented by the database management system. Each type will have corresponding type names which are stored as strings such as "int" for integers, "char" for character strings, "date" for date data-type in the sortable ISO format and "bool" for Boolean constructs. Additionally, modern database system also supports the ability for user-defined types in addition to the built-in types provided by the system. Meanwhile, with the advent of SQL, concepts such as 'table', 'column' and 'row' are used instead of 'relvar', 'attribute' and 'tuple'.

The operation of a relational database management system basically begins when a user or program requests data from it through the use of a query written in a special language, usually a dialect of SQL. In response to this query, the database well therefore returns a result set, a subset of the database itself, which is a list of the rows containing the answers matching the sent query. The simplest query, which is "SELECT *" is just to return all the rows from a table. Nonetheless, most queries are done with specified constraints in order to filter data one way or another.

For this project, MySQL is the relational database management system that will be used to provide a database backend to the system. MySQL's source code is available under the terms of the GNU General Public License. As of April 2009, MySQL supports a broad subset of the official ANSI SQL 99 specifications as well as extensions. It is capable of multiple storage engines, where a primary high-performance among them would be InnoDB which will be used in this project. Since the MySQL Server is available as free software under the GNU General Public License in contrast to other commercial database systems, it proves to be a prime candidate in which cost is a prime matter of concern by being a free component to the system.

2.8 – Virtualization of the MySQL Database Backend

System virtualization is the act of creating an efficient, isolated duplicate of a real machine whereby a complete system platform is provided which supports the execution of a complete operating system. The main advantages of virtual machines are:-

- multiple operating systems can co-exist on the same computer with strong isolation from each other
- the virtual machine can provide a somewhat different instruction set architecture from the actual machine
- virtual machines can provide application provisioning, good maintenance capabilities, high availability and good disaster recovery options.

Therefore, by applying virtualization to the MySQL database, a dramatic improvement in terms of efficiency and availability can be reached. In tests done on with a traditional system against a virtualized system, the virtualized system is observed to perform almost equally to a physical system despite the added complexity of the virtualization layer. Figure XX shows the performance results in terms of Transactions per Second (TPS) as well as latency between a virtual machine and a physical system. With negligible performance loss, the added advantage in terms of high availability and disaster recovery becomes more evident as an advantage in favor of virtualizing the MySQL subsystem of the project.

Figure 3: MySQL Performance Benchmark, Physical versus Virtualized [8]

CHAPTER 3
METHODOLOGY

3.1 - Planned Project Timeline

3.1.1 - Gantt Chart

Planned project implementation Gantt chart is bundled in the appendices and is available as Appendix 1.

3.1.2 - Project Milestones and Progress

MILESTONE DESCRIPTION	PROGRESS
Preliminary Research Work	COMPLETED
Set-up of Server Virtualized Hardware	COMPLETED
Set-up of Server Software	COMPLETED
Database Design and Implementation	COMPLETED
Database Access Layer Shared Module Code	COMPLETED
RFID Equipment Acquisition	COMPLETED
Wi-Fi Networking Infrastructure	COMPLETED
Main System Administration Interface	COMPLETED
Client System Application	COMPLETED
Web Access Interface	COMPLETED
RFID ⇔ Client Application Interaction	COMPLETED
System Alpha / Beta Testing	COMPLETED

Figure 4: Project Milestone and Completion Rate

3.2 - Project Flowchart

3.2.1 - Server Interface Flowchart

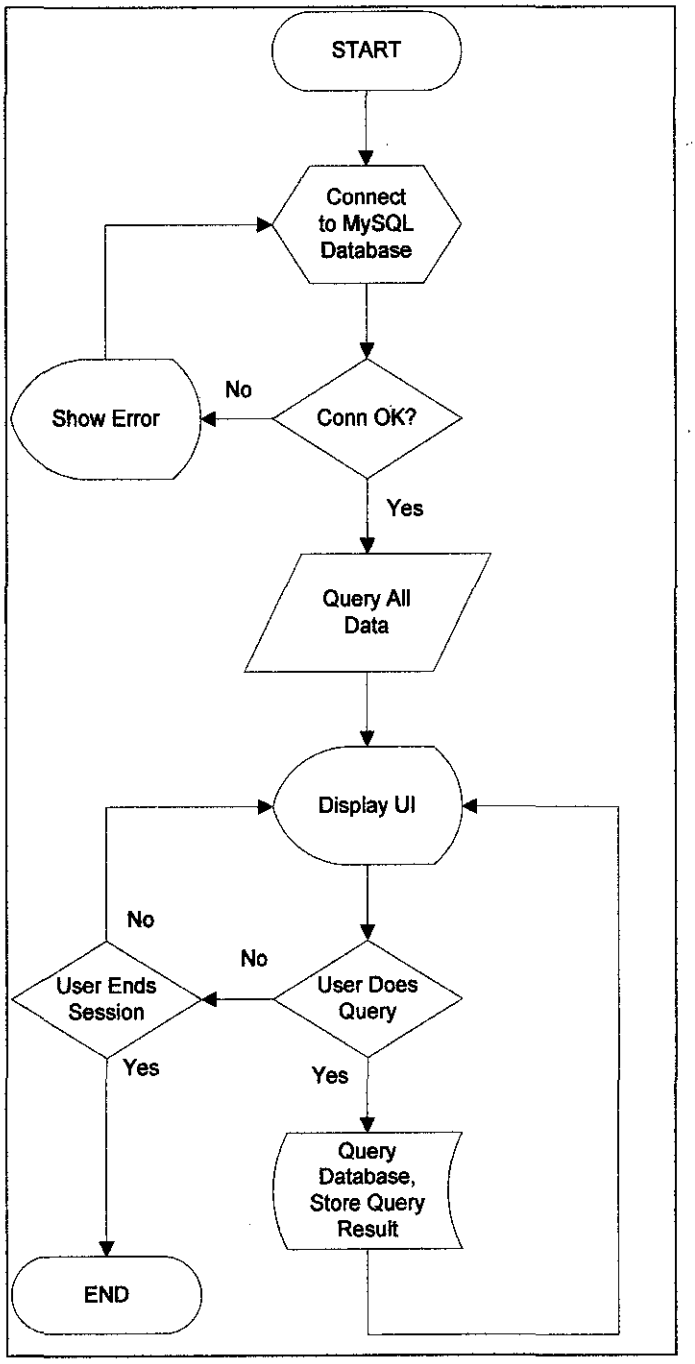


Figure 5: Server User Interface Flowchart

Figure 5 depicts the flowchart for the operation of the server interface, also known as the system's administration control panel. Upon application startup, it first attempts to connect to the system database based on the credentials provided by the user. Administrators will have read-write access while normal users will only have read-only access. If the database connection cannot be completed, an error will be given and the application will retry to connect once credentials are entered again. Errors here can be caused by network instability or plainly due to the MySQL server being down for maintenance.

Once database connection is successful, it will query all the statistical data on the server such as total number of users, total accesses within the last 12 hours etc. Once the data had been acquired it will display the user interface listing the acquired data. That interface will also serve as a portal to other interfaces from which data manipulation can be done. There will be 3 different manipulation interfaces each catering to a particular aspect of the system which are the users, the usergroups and finally the gates itself.

As mentioned above, manipulation of each data is done through the individual interfaces. Manipulation is done through the application queries based on user interaction with the interface. If a user adds data, an INSERT command will be sent to the database. If a user modifies data, an UPDATE command will be sent while data removal will result in the REMOVE query being sent. The SELECT query will be sent in order to refresh the data display on the various user interfaces.

3.2.2 - Client Operation Flowchart

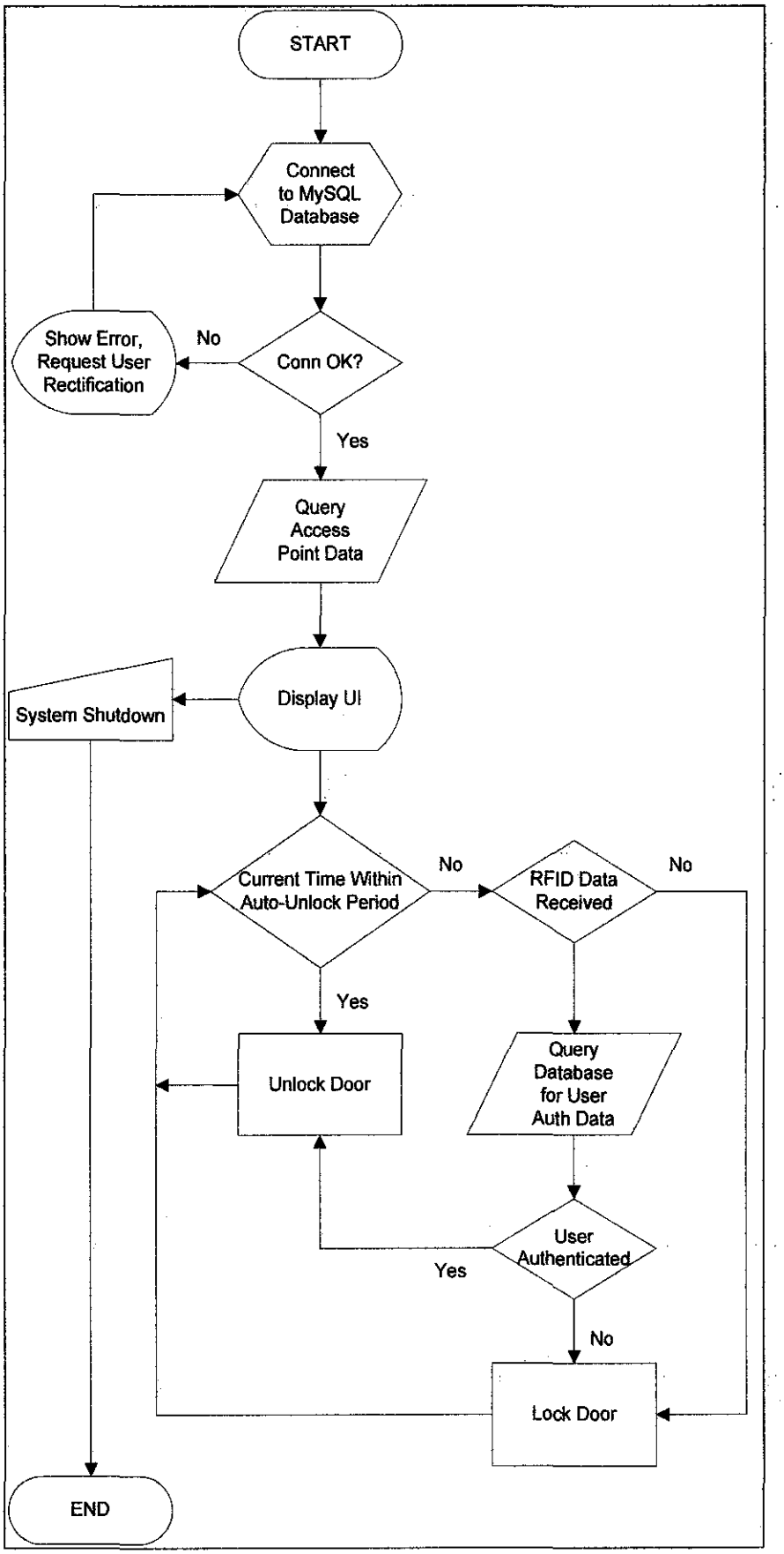


Figure 6: Client Operation Flowchart

As shown in Figure 6, the client operation flowchart is a major part of the system's operation. Upon the startup of the client application, it first attempts to connect to the central database, raising an error if there is a problem with the connection attempt similar to the administration control panel. If the connection is successfully started, it queries the database for the gate properties including the auto-unlock times and durations and stores them within the application for further processing. After that, the process basically displays its UI and waits for either an RFID tap or until the current time is within the auto-unlock period.

The flowchart in Figure 6 after the "Display UI" block begins as a user taps an RFID tag onto the connected RFID reader. Upon tapping, the RFID tag is energized by the electromagnetic field of the reader and begins to transmit its stored information, in this case, the RFID tag's unique ID string. The RFID receives the transmitted string and sends it to the client PC by raising an event within the RFID class module. Upon receipt of the RFID tag's unique ID through the raised event, it attempts to establish the connection to the central database and calls the helper interface for authentication based on the tag's unique ID and the gate ID. The algorithm used is explained in further detail in Chapter 4.1.1. If the user is authenticated, the door will be unlocked by sending an UNLOCK string by opening a Windows Socket connection to local host on Port 11111. Otherwise, nothing will happen and no unlock signal will be sent.

The UNLOCK string sent via Windows Socket Port 11111 will allow 3rd party applications to read for this signal and control the lock based on the lock's design. By decoupling the unlocking interface from the actual lock itself, a variety of locks can be controlled by just listening for the UNLOCK signal on Port 11111. For the case of this prototype, an application will be developed to handle the unlock signal and display it in a virtual manner instead of a hardware implementation.

3.3 - Applied Tools and Technology

3.3.1 - RFID Tags

For the developed system, Radio Frequency Identification or RFID technology is used exclusively for the storage of authentication credential for personnel. Many different RFID tags are available for use ranging in size from those resembling tiny specks of dust to large active tags a few centimeters in width. In Figure 7, we can see an illustration of some of the various form factors passive RFID had been designed.

Figure 7: Passive RFID Tags in Various Form Factors [9]

This is due to the various application demands for RFID technology in various industries where tagging and identification is needed. After taking into account the various architectures of RFID tags and its specific usage scenarios, it is deemed that small tags of the passive variety will suite the role for this project. Being of the passive variety, it doesn't require external power sources rendering it ideal for personal usage. In order for the system to be implemented within the university, everyone using the technology will be issued their own tag containing their own customized authentication credentials for use when activating the lock of a particular facility.

3.3.2 - RFID Readers

For the RFID reader, it consists of both the antenna and reader package. The reader package is interfaced to a PC to process acquired RFID tag information through dedicated client software. The easiest reader packages are those featuring a USB interface to connect it to the host PC as shown in Figure 8.

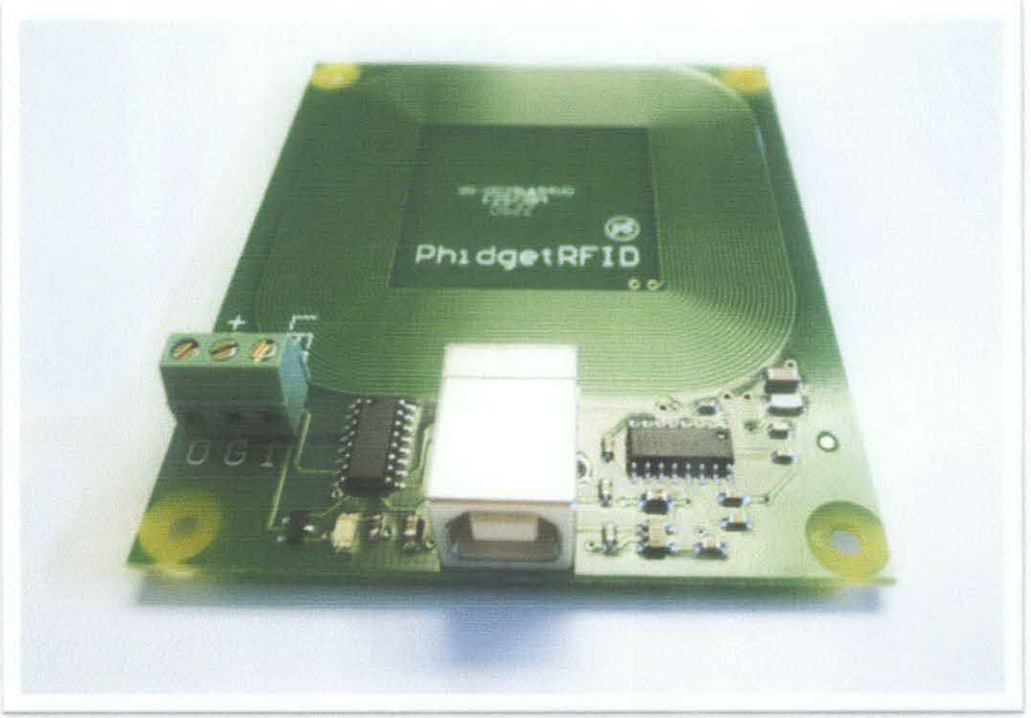


Figure 8: Phidget© 13.56MHz RFID Reader with USB Interface [9]

Due to the passive nature of the implemented RFID tags and taking account its low-range requirements for application, a low-frequency RFID antenna is deemed suitable and sufficient for proper operation. These antennas are attached to the perimeter of doors to be controlled by the system to enable easy access to its users. The operating frequencies for these passive tags are 13.56 MHz at a range of 3 centimeters. This should be optimum for application at door access points as required by the project.

3.3.3 - Authentication and Access Control Database Server

This component of the project is responsible for providing both a centralized database as well as a management user interface for the system. Therefore, it consists of 2 main units which are the database system and both its web-based and Windows GUI management panel. These two units work hand-in-hand to provide not only storage of access data but also to enable mass-control of all connected locking mechanisms throughout its service area. Figure 9 below shows the registered trademark of the MySQL Database System Logo.

Figure 9: MySQL Database System Logo [8]

The database system is powered by the open source MySQL Relational Database Management System. Therefore, all relevant data is stored in tables within the database system. Data storage and retrieval is done both remotely by both the remote client software and the central server through Structured Query Language (SQL) queries over the network via TCP/IP. SQL is a standard interactive and programming language for querying and modifying data as well as managing the database. Due to MySQL's adherence to ANSI/ISO standards under SQL/PSM (ISO/IEC 9075/4:2003), it can be expected to provide a standards compliant database interface and robust industry grade performance.

In addition to that, another factor for the selection of MySQL as the database server is its interoperability with the PHP web scripting language. This enables the construction of a web interface for the whole authentication system through PHP/MySQL web applications. PHP by default had been designed with integration with MySQL and many of the web applications or websites found online are built upon this implementation. As a bonus, an open source web application, 'phpMyAdmin', is available for direct administration of the database server if the need arises.

As for the server interfaces, two alternatives are be provided. The first one is a Windows based user interface for efficient administration of the system. This program is designed with Windows Presentation Foundation (WPF) through Visual Basic .NET 2008 utilizing the .NET 3.5 Framework. Both WPF and VB.NET 2008 are very recent technologies that introduce many new fundamental technologies including Presentation/Logic Layer separation which applies very much to the requirements of the system to be designed. The presence of this interface will significantly ease the administration task of the system by having an intuitive GUI that provides complete control of the system to its operators. Main features of this interface includes the listing of all available access points graphically displayed, editing of automatic door access configurations, editing of all personal profiles in the database as well as various database management capabilities including database backup and restore in the case of catastrophic system failure.

The second interface is the web-based interface. Its main purpose is to enable remote administration of the system through Wi-Fi networks thereby increasing the system's flexibility. It is hosted on the Apache Web Server for enhanced communication between its MySQL database backend. It should be noted that this front-end is not designed to be a full-fledged administration interface but rather a remote control system for on-the-field control of the various access points. This is most useful where ad-hoc requests are being made by its users for access to a particular facility. The system is hosted on the open source Apache web server for access over the system's Wi-Fi cloud. By accessing this system, users knowledgeable of MySQL queries are able to manipulate the database directly by sending queries directly to the database via this web front-end.

3.3.4 - Facility Access Control Client

The control client software, as with the server software, is coded in WPF/VB.NET. Its main task is to interface with both the RFID reader and the MySQL database to control the lock status of all the locks configured to be under its jurisdiction. Therefore, an interface is provided as well to give users easy access to the system for maintenance purposes. These thin clients are designed to query the central database both at set intervals and whenever an access request is received via the attached RFID reader. It will take care of authentication and unlocking of the particular access point should access be granted. Meanwhile, it can also maintain a local cache of the access point's automated lock/unlock cycles so as to ensure that the point under its jurisdiction will function as programmed through the central database.

3.3.5 - Wi-Fi Network Communication

The system utilizes the 802.11 standards for Wi-Fi communication. The application of this technology enables the decoupling of its client and server functionality wirelessly thereby allowing easy scaling of the project when needed. Wireless interconnects also enables more flexibility in both the addition of new locking nodes as well as allow remote administration of the system. This is especially true for remote administration from Wi-Fi enabled handheld devices which are very much commonplace in recent years. In Figure 10 below, we can see the official logo by the Wi-Fi Alliance which is a registered trademark:-

Figure 10: The Wi-Fi Logo [10]

Implementation of the Wi-Fi network will be through strategically placed Wi-Fi routers and repeaters to extend the Wi-Fi cloud which will be responsible for both the address resolution of its connected system clients as well as any devices logging on to the network. The address of connected clients will be identified by client-server communication over the Wi-Fi network and stored as well within the database itself.

CHAPTER 4
RESULTS & DISCUSSION

4.1 - Project Results

4.1.1 - Database Server Installation & Schema Design

One of the first implementation efforts for the project was the installation of the database server and the subsequent design of the database and table structure in which the needed data are to be stored. Database setup and design was done through the phpMyAdmin interface as shown below:-

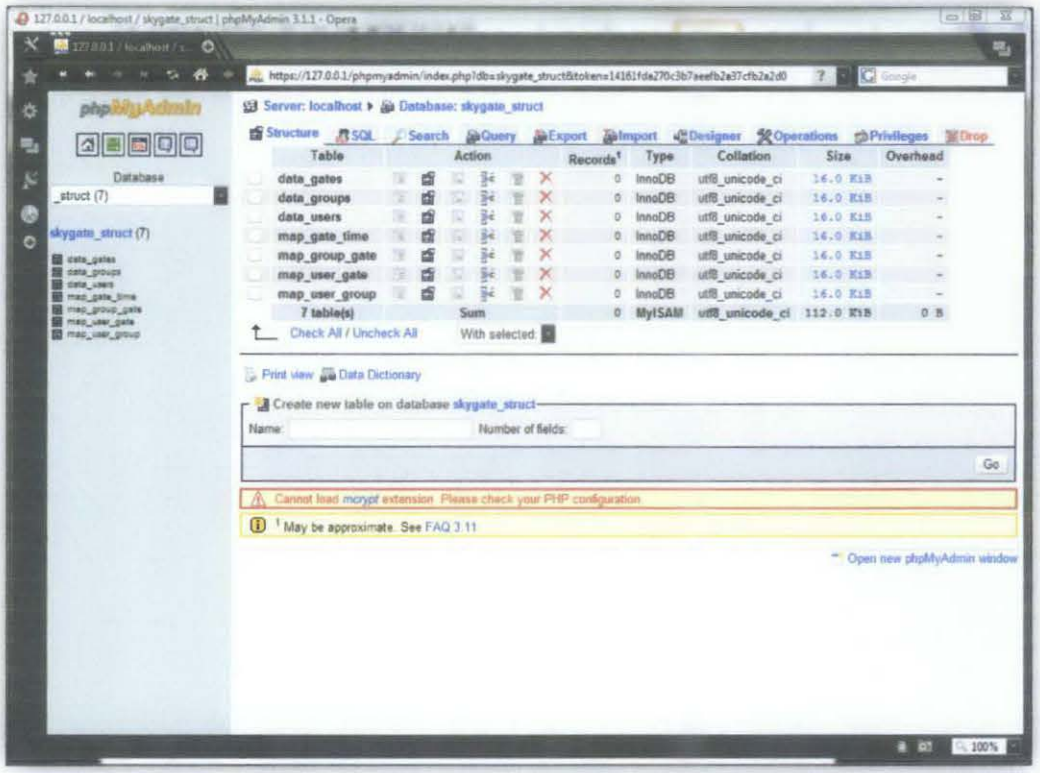


Figure 11: Main Database View from phpMyAdmin

The database, hereby named 'skygate' after the project name, was designed to hold tables. There are two functionally different table styles in this database. The first are plain data storage tables, in this case, 3 tables namely 'data_gates', 'data_groups' and 'data_users'. These 3 tables store pure data regarding gates, user groups and user info respectively.

Meanwhile, the other different table style in this database is the mapping tables. The function of mapping tables is to store authentication relationships between the data tables above. They mainly contain 3 columns which are the first table row index, second table row index and the mapping property which can be either Allow or Deny. For this project, the authentication bindings are:-

- A particular gate authenticated against a span of time (map_gate_time)
- A particular group against a particular gate (map_group_gate)
- A particular user against a particular gate (map_user_gate)
- A particular user being in a particular group (map_user_group)

When a user-gate authentication check is requested, authentication checking will follow the flow below:-

1. Receives user and gate ID
2. Checks for DENY mapping in 'map_user_gate', if exists, return DENY
3. Checks for ALLOW mapping in 'map_user_gate', if exists, return ALLOW
4. If no user-gate mapping found, acquire user's group from 'map_user_group'
5. For every user-group mapping found
    1. Checks for DENY mapping in 'map_user_group' for the current group, if exists, return DENY
    2. If no DENY mappings found, redo iteration and check for ALLOW mapping in 'map_user_group', if exists, return ALLOW
6. If no user-group mapping found, return DENY

Table schema for the database had been designed as follows. Due to the availability of mapping tables, additional data tables can be added with ease since mapping can be done separately without the need to modify current mapping or data tables.

Table 1: Gates Data Table Schema (data_gates)

Field	Type	Description
id	int(11)	Auto incrementing, unique, gate ID
location	text	Info specifying the gate's location

Table 2: Groups Data Table Schema (data_groups)

Field	Type	Description
id	int(11)	Auto incrementing, unique, group ID
name	text	Info specifying the group's name

Table 3: Users Data Table Schema (data_users)

Field	Type	Description
id	int(11)	Auto incrementing, unique, user ID
rfid	text	Unique ID of the user's RFID tag
name	text	User's full name
title	text	User's working title, if any

Table 4: Gate-Time Mapping Table Schema (map_gate_time)

Field	Type	Description
id	int(11)	Auto incrementing, unique, mapping ID
gateid	int(11)	ID of the gate being mapped
start	int(11)	Time to start automatic gate open (in mins from 12AM)
duration	int(11)	Duration gate will remain open (in minutes)

Table 5: Group-Gate Mapping Table Schema (map_group_gate)

Field	Type	Description
id	int(11)	Auto incrementing, unique, mapping ID
groupid	int(11)	ID of the group being mapped
gateid	int(11)	ID of the gate being mapped
isdeny	tinyint(1)	Sets mapping property, 1 for DENY, 0 for ALLOW

Table 6: User-Gate Mapping Table Schema (map_user_gate)

Field	Type	Description
id	int(11)	Auto incrementing, unique, mapping ID
userid	int(11)	ID of the user being mapped
gateid	int(11)	ID of the gate being mapped
isdeny	tinyint(1)	Sets mapping property, 1 for DENY, 0 for ALLOW

Table 7: User-Group Mapping Table Schema (map_user_group)

Field	Type	Description
id	int(11)	Auto incrementing, unique, mapping ID
userid	int(11)	ID of the user being mapped
groupid	int(11)	ID of the group the user is part of

4.1.2 - Database Access Application Layer

The database access application layer is a class designed to provide a unified layer of access to the 'skygate' database from the various clients that will be using it. For this project, both the administration control panel and the field client software will be using this class as a helper interface to access the MySQL central database backend for the system. The class will be compiled as a Dynamic Link Library (DLL) instead of an executable to ensure that it can be integrated into any systems that may be designed in the future as well as to separate the data access codepath and the other application codepaths.

Being a helper interface, the class exposes various functions and methods in order to assist the user in acquiring data from the database. Additionally, it also performs formatting of the raw data retrieved from queries into standard .NET data tables for increased compatibility. Below is the list of exposed methods, constructors and functions as well as a description of its use and functionality.

Constructor Functions

```
Sub New()
Sub New(ByVal Host As String)
Sub New(ByVal Host As String, ByVal User As String, ByVal Pass As String)
```

Above are the constructor function used to initialize the class. When called without any inputs, the database is assumed to be at the local host and accessed in read-only mode. By giving the first string, which is the host string, an alternate or specific address is given. In this case, access is still read-only. Meanwhile, the last calling procedure, which gives the server host address as well as username and password, allows the access to be elevated based on the credentials given.

Raw Execute Query Function

```
Public Function Execute(ByVal SqlQuery As String) As Integer
```

This function executes the raw SQL query passed as a string onto the database server. Execute queries doesn't return results, but the total rows affected is returned as an Integer value.

Raw Standard Query Function

```
Public Function Query(ByVal SqlQuery As String) As DataTable
```

Similar to the Execute function, this function executes the raw SQL query passed as a string to the database server as well. However, this function expects the results returned to be datarows from the database. Therefore this function is normally done to execute SELECT queries which acquire data, rather than the INSERT or UPDATE queries which are executed using the Execute function instead. This function returns a formatted .NET datatable wherewith the data can be accessed as rows and columns in the datatable itself or through the 2-dimensional array format notation.

User Access Checking Function

```
Public Function CheckUserAccess(ByVal UserID As Integer, ByVal GateID As Integer) As Boolean
Public Function CheckUserAccess(ByVal UserRFID As String, ByVal GateID As Integer) As Boolean
```

This function checks a user's ability to access a certain gate based on either his UserID passed as an Integer or UserRFID (RFID Unique ID) passed as a string. It follows the algorithm given in Chapter 4.1.1 of this study and returns a TRUE if the user is allowed access or FALSE otherwise.

Data Addition Functions

```
Public Function AddGroup(ByVal GroupName As String) As Integer
Public Function AddGate(ByVal GateLocation As String) As Integer
Public Function AddUser(ByVal RFID As String, ByVal UserName As String, Optional ByVal
    UserTitle As String = "") As Integer
```

This collection of functions adds datarows to the respective tables. The usage of this function is safer compared to direct query since it checks for duplicates before adding the respective data.

Data Modification Functions

```
Public Function ModifyGroup(ByVal ID As Integer, ByVal GroupName As String) As Integer
Public Function ModifyGate(ByVal ID As Integer, ByVal GateLocation As String) As Integer
Public Function ModifyUser(ByVal ID As Integer, ByVal RFID As String, ByVal UserName As
    String, Optional ByVal UserTitle As String = "") As Integer
```

Similar to the Addition functions above, these functions modify data in the respective datatables with the values given as inputs.

4.1.3 – Administration Panel User Interface

For this project, an administration interfaces had been completed to allow an administrator to control and oversee the operation of the system via this interface. The administration interface is divided into 3 main sections, each of which manages a different aspect of the system. These 3 aspects are:-

1. Management of individual users
2. Management of groups
3. Management of individual access gates

The interface for managing users will allow the administrator to edit user info, add a user to a group as well as manipulate allow/deny permissions for a user at gate level. Likewise, the group's management interface will allow manipulation of a group's data, list of registered users within the group as well as group level allow/deny permissions. Finally, the gate management interface will show gate information as well as both user and group permissions for a particular gate. It is noticeable that the data manipulated by each interface are interlinked with the others. This is because the same data are being manipulated as linked pair where each interface shows the manipulated data from a different perspective.

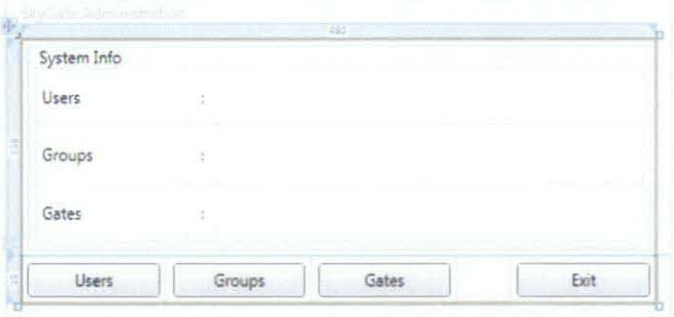


Figure 12: Simplified Main Portal Screen

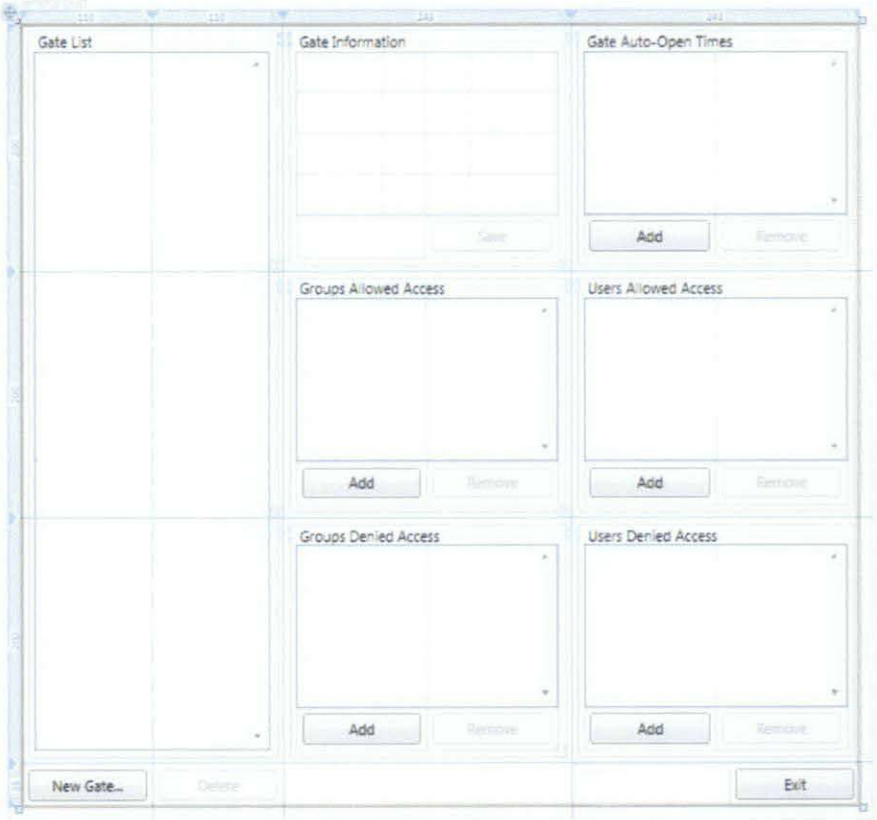


Figure 13: Gates Management Screen

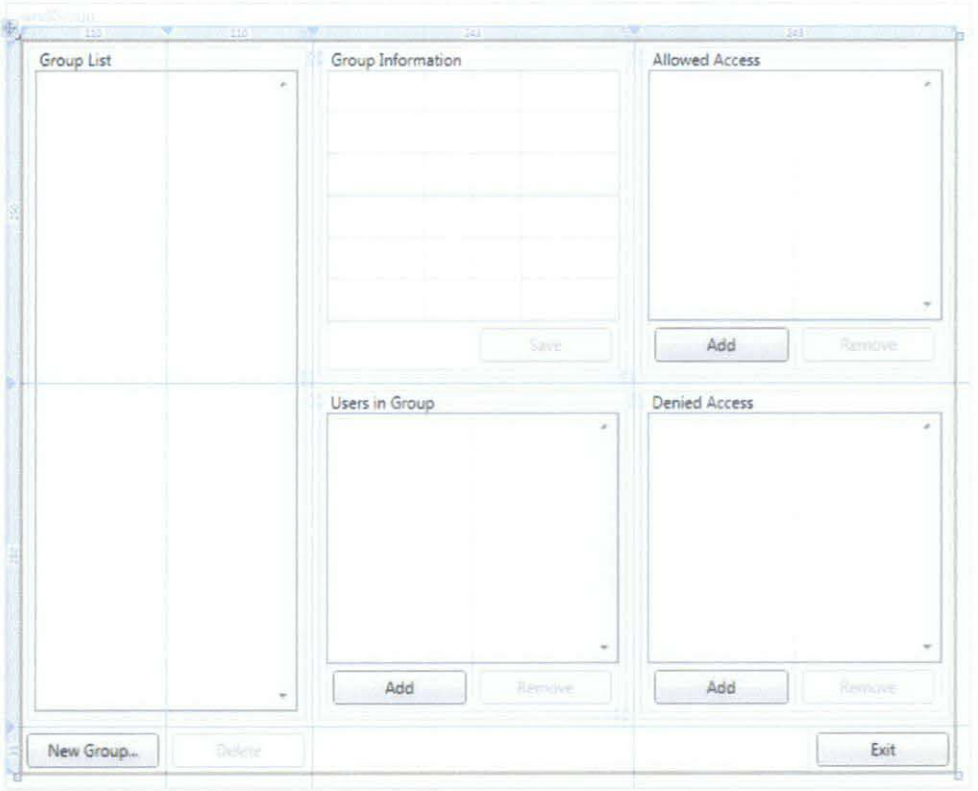


Figure 14: Group Management Screen

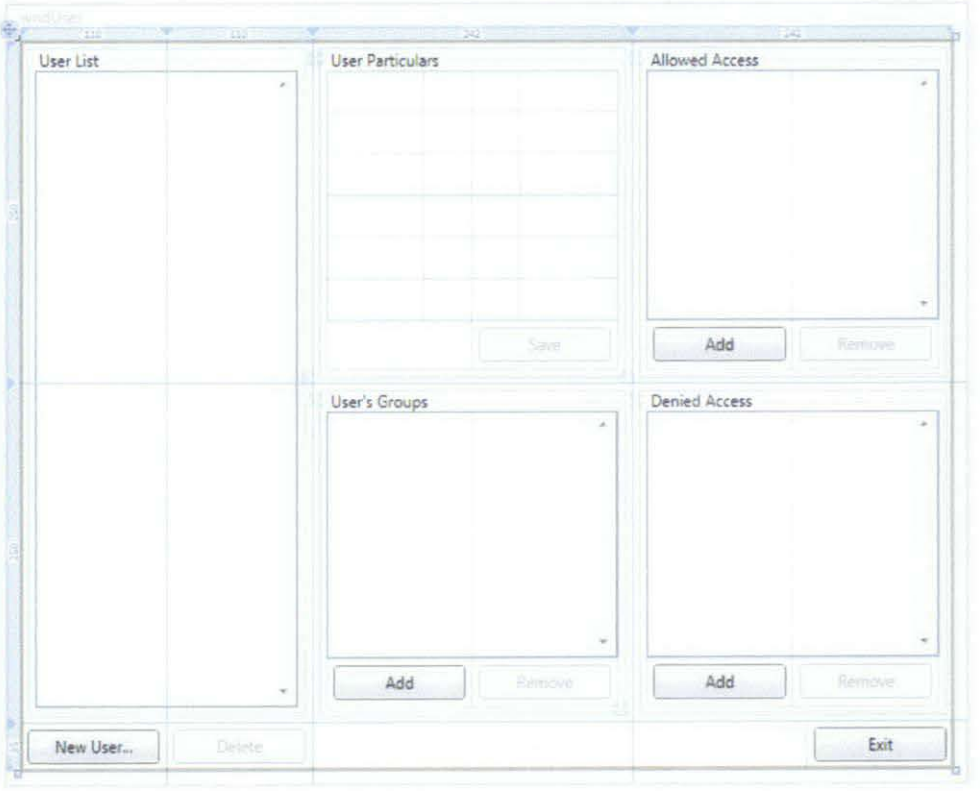


Figure 15: User Management Screen

4.1.4 - Hardware Acquisition

Hardware that is required for the project had been either purchased or sourced. More specifically, the Wi-Fi switch/bridge had been purchased from Belkin® which is of model number F5D7230-4. This network switch/bridge is capable of emitting an 802.11b/g Wi-Fi cloud to an outdoor range of 100 meters. It is also capable of providing bridging capabilities to connect a wireless and a wired network together. As in any other Wi-Fi device, it operates over a frequency of 2.4GHz. An added advantage of this device is its capability of encrypting wireless communication using the highly secure WPA-PSK (Wireless Protected Access, Pre Shared Key) algorithm thereby ensuring the security of the system against hostile intrusion.

Figure 16: Belkin© 802.11g Switch

Figure 17: Phidget© RFID Kit

Meanwhile, for the RFID part, appropriate RFID equipment had been sourced together with other FYP students to be used for all our projects. The equipment will utilize passive RFID technology and will include both the RFID readers as well as the passive tags. The RFID kit by Phidget© is used which contains both a reader and a variety of tags in different form factors.

4.1.5 - Virtual Machine and JEOS (Just Enough OS) Installation

A virtual machine system containing a JEOS powered by Turnkey Linux had been prepared for the purpose of this project as planned. Turnkey Linux is a derivative of the highly acclaimed Ubuntu Linux which had been streamlined to provide only the services that are needed for this project. In this case, Ubuntu Linux LTS 8.04 is used. Overall installation of this virtual machine only occupies near 200MB of space which results in a highly portable environment. This eases development of the project since its high portability ensures that development on the software side can be done without much delay.

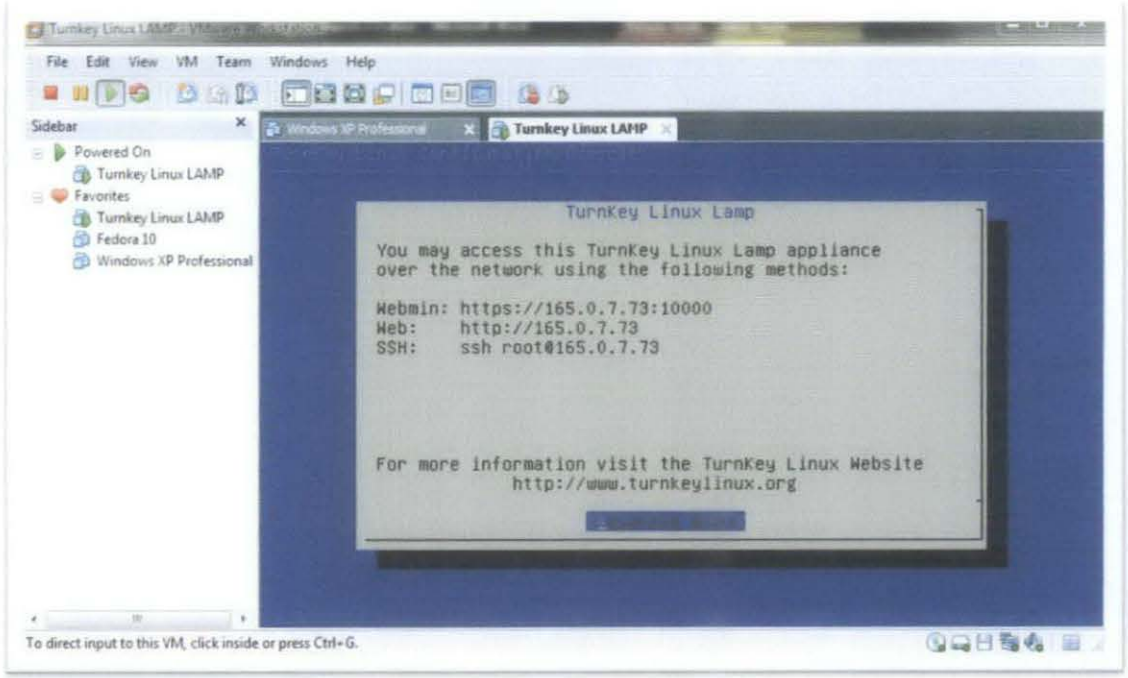


Figure 18: Turnkey Linux Virtual Appliance Screen

4.1.6 - Network Services Setup

Network services needed by the project had been set up which includes the database server (MySQL 5) and web server (Apache). An administration interface to manage these network services had also been made available. These interfaces are capable of providing remote administration capability to the database and web servers as well as the hardware it is running on. These administration interfaces are secured with the HTTPS protocol for added security. On the next page are screenshots of the various parts of the available services as well as its administration interfaces.

Figure 19: Web Administration Main Menu

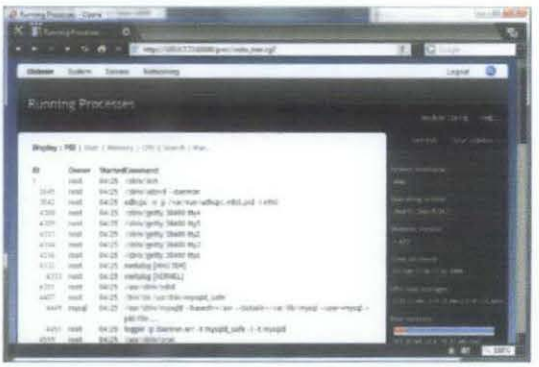


Figure 20: System Stats Screen

Figure 21: Apache Administration Panel

Figure 22: MySQL Administration Panel

4.2 - Discussion of Problems Encountered

4.2.1 - Hardware Acquisition

Throughout the implementation of the project the past few weeks, problem had mostly centered on acquiring the hardware for the project. As mentioned earlier, the scale of the project inhibits the purchase of needed devices at acceptable prices. This is due to the fact that suppliers mainly supply in bulk sums of the RFID devices which is prohibitive for the implementation of the project.

A proposed alternative would be to determine if there are still any unused and available RFID devices of the passive variety still in stock within the university. Up to now, no such known stock exist within the Electrical and Electronic Faculty. In any case, this had been solved by the shared purchase of a Phidget RFID kit which contains a passive USB connected RFID reader. Since it is a shared purchase, the cost of the kit is acceptable.

4.2.2 - GPL Licensing of Used Software Components

Other than the hardware sourcing difficulties, another minor issue regarding the project will be the status of the source code that will be developed for it. Since the project is mainly going to be built off open source tools and systems, it is required that the source code be rendered open source as well according to the GNU General Public License (GPL). The GPL is a widely used free software license that requires derived works to be available under the same copyleft license. This mainly applies to the MySQL/.NET connector for now although it is very likely that future coding efforts will introduce more components that are licensed as such For this licensing issue, it is decided that the code for this project will also be released under the GPL as required. This will allow the GPL-licensed code and components to be used within the project as well as to ensure that the codes for the project will be available for public perusal for improvements and/or adaptations. Indirectly, it is hoped that this will be a good contribution to the software community at large.

CHAPTER 5
CONCLUSION & RECOMMENDATION

5.1 - Conclusion

To conclude this report, it is proven that RFID data propagation over a Wi-Fi wireless network is a feasible, flexible and efficient way to expand on the range limitations of RFID technology. Additionally, by having a database backend powered by a relational database management system, both sweeping modifications and small updates to the system can be applied with ease and simplicity. Therefore, it can be concluded that RFID/Wi-Fi is a step in the right direction in the tagging, transfer and processing of large amounts of data over a large working area.

5.2 - Recommendations

Future efforts that can be undertaken for this project that is outside the current scope will be as follows:-

- Integrate the RFID readers into the network directly. Some readers are capable of RJ-45 LAN connections; therefore, a single multi-port switch can be used to connect many of these readers together into a single sub-network that can be managed by a single client computer.
- Use embedded systems instead of PCs to relay RFID reader data to the central server. Embedded systems will be much smaller and can be implemented in more robust ways superior to that of a PC client.

REFERENCES

[1] Landt, J. 2001, *Shrouds of Time. The History of RFID* <www.aimglobal.org/technologies/RFID/resources/shrouds_of_time.pdf>

[2] Staelin, D. H. 2003, *Receivers, Antennas, and Signals. An Electrical Engineering and Computer Science Subject,* MIT OpenCourseware 6.661, Cambridge, MA <http://ocw.mit.edu/OcwWeb/Electrical-Engineering-and-Computer-Science/6-661Spring2003/CourseHome/index.htm>

[3] Sarma, S., Brock, D., and Ashton, K. 2001, *The Networked Physical World, Proposal for Engineering the Next Generation of Computing, Commerce and Automatic Identification*, Auto-ID Center, Cambridge, MA <autoid.mit.edu/whitepapers/MIT-AUTO-ID-WH-001.pdf>

[4] Stephen B. Miles, Sanjay E. Sarma & John R. Williams 2008, *RFID Technology and Applications*, Massachusetts Institute of Technology, Cambridge Press

[5] Sarma, S. 2005, *A History of the EPC in RFID: Applications, Security and Privacy,* Addison-Wesley Professional, New York

[6] E. F. Codd 1990, *The Relational Model for Database Management: Version 2*, Addison-Wesley Longman Publishing

[7] English Wikipedia 2006, *An illustration of Relational model concepts, based on Fragment's Begriffe relationaler Datenbanken.png*, Wikimedia

[8] Mike Frank & Jennifer Glore 2009, *Virtualization for MySQL on VMware® Best Practices and Performance Guide*, Sun Microsystems

[9] Phidget RFID Kit

URL: http://www.phidgets.com/products.php?category=8&product_id=2002

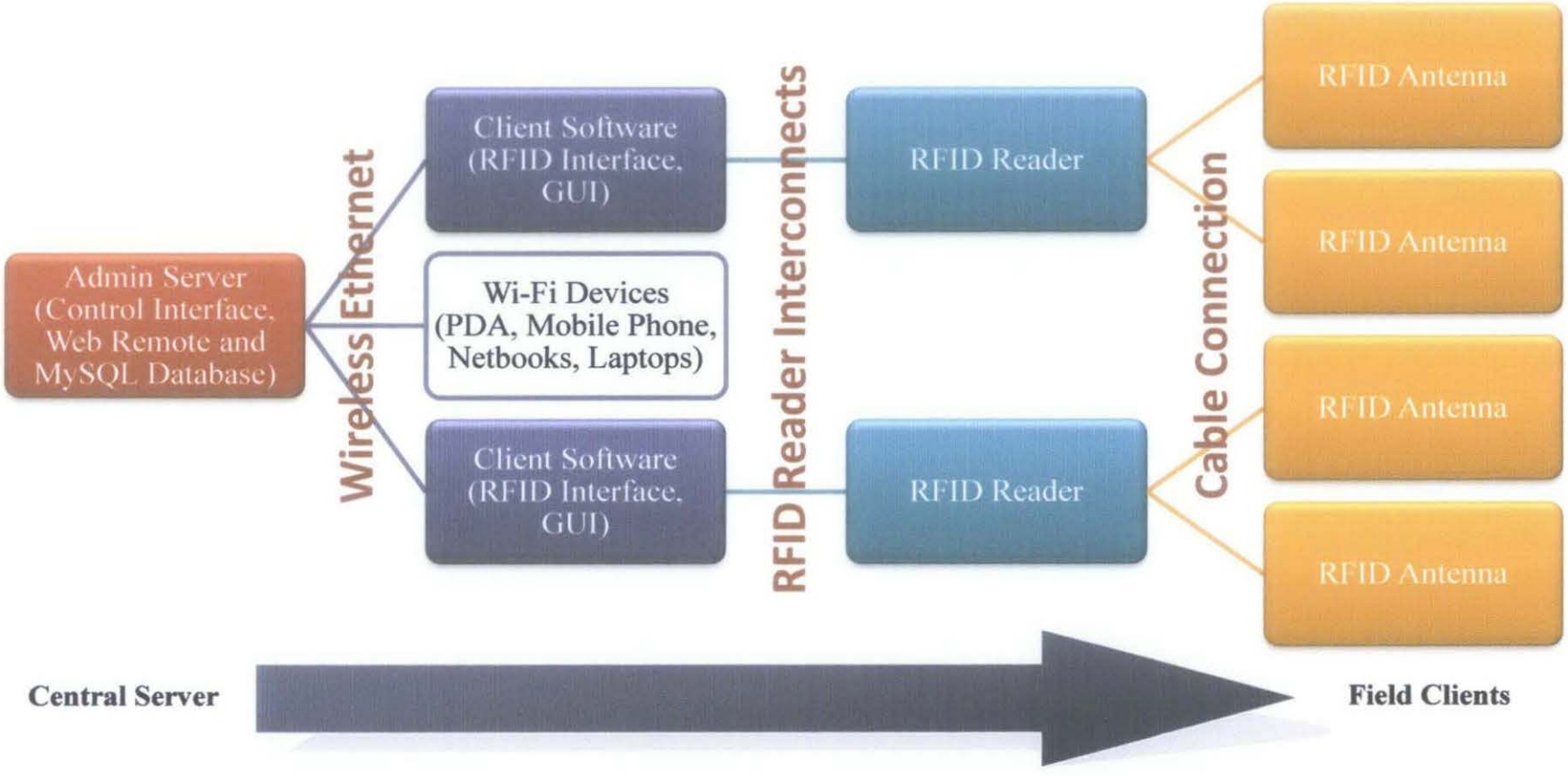
[10] Syed Ahson & Mohammad Ilyas 2008, *Wi-Fi Alliance Brand Styleguide*, Wi-Fi Alliance

APPENDICES

APPENDIX 1: PROJECT GANTT CHART

No.	Detail/ Week	1	2	3	4	5	6	7	8	9	10	11	*	12	13	14
1	Selection of Project Topic												Mid-Semester Break			
2	Preliminary Research Work															
3	Submission of Preliminary Report															
4	Seminar 1 (optional)															
5	Project Work															
6	Submission of Progress Report															
7	Seminar 2 (compulsory)															
8	Project work continues															
9	Submission of Interim Report Final Draft															
10	Oral Presentation															

APPENDIX 2: OVERVIEW OF SYSTEM AND INTERCONNECTS



APPENDIX 3: DATA ACCESS SOURCE CODES

```
Imports MySql.Data.MySqlClient

Namespace SkyGate
    Public Class SkyGateDB

#Region "Structure Definitions"
        Structure _DBConf
            Dim Host As String
            Dim User As String
            Dim Pass As String
        End Structure
#End Region

#Region "Constructors"
        Sub New()
            DBConf.Host = DefHost
            DBConf.User = DefUser
            DBConf.Pass = DefPass
        End Sub
        Sub New(ByVal Host As String)
            DBConf.Host = Host
            DBConf.User = DefUser
            DBConf.Pass = DefPass
        End Sub
        Sub New(ByVal Host As String, ByVal User As String, ByVal Pass As String)
            DBConf.Host = Host
            DBConf.User = User
            DBConf.Pass = Pass
        End Sub
#End Region

#Region "Global Variables"
        Dim DBConf As _DBConf
#End Region
```

```
#Region "Low Level MySQL Queries"
        Private Function GetDbConn() As MySqlConnection
            Dim SqlConnectionString As String = "Database=skygate;Data Source=" & DBConf.Host & ";User Id=" & DBConf.User & ";Password=" & DBConf.Pass
            Dim SqlConnection As New MySqlConnection(SqlConnectionString)
            Return SqlConnection
        End Function

        Public Function Execute(ByVal SqlQuery As String) As Integer
            'Command function for UPDATE, INSERT and DELETE operations
            Dim SqlCommand As New MySqlCommand(SqlQuery, GetDbConn())
            SqlCommand.Connection.Open()
            Dim NumRows As Integer = SqlCommand.ExecuteNonQuery()
            SqlCommand.Connection.Close()
            Return NumRows
        End Function

        Public Function Query(ByVal SqlQuery As String) As DataTable
            'Command function for data-returning queries.
            Dim SqlAdapter As New MySqlDataAdapter()
            Dim SqlDataset As New DataSet
            SqlAdapter.SelectCommand = New MySqlCommand(SqlQuery, GetDbConn())
            SqlAdapter.SelectCommand.Connection.Open()
            SqlAdapter.Fill(SqlDataset)
            SqlAdapter.SelectCommand.Connection.Close()
            Return SqlDataset.Tables(0)
        End Function
#End Region
```

```
#Region "SkyGate Database Query Adapters"
        Public Function CheckUserAccess(ByVal UserID As Integer, ByVal GateID As Integer) As Boolean
            'Queries - Min : 1 , Max : 3 + (2 * User-Group) ~~ 7

            'Query User-Gate mapping WHERE IsDeny = TRUE, IF exist, return FALSE
            If Query("SELECT COUNT(*) FROM `map_user_gate` WHERE `userid`=" & UserID & " AND `gateid`=" & GateID & " AND `isdeny`=TRUE ;").Rows(0)(0) > 0 Then
Return False
            'Query User-Gate mapping WHERE IsDeny = FALSE, IF exist, return TRUE
            If Query("SELECT COUNT(*) FROM `map_user_gate` WHERE `userid`=" & UserID & " AND `gateid`=" & GateID & " AND `isdeny`=FALSE ;").Rows(0)(0) > 0
Then Return True

            'Query User-Group mapping
            Dim UserGroups As DataTable = Query("SELECT `groupid` FROM `map_user_group` WHERE `userid`=" & UserID & " ;")
            '  - For Each User-Group's GROUP :
            For Each GroupID As DataRow In UserGroups.Rows
                '  - Query Group-Gate mapping WHERE IsDeny = TRUE, IF exists, return FALSE
                If Query("SELECT COUNT(*) FROM `map_group_gate` WHERE `groupid`=" & Val(GroupID(0)) & " AND `gateid`=" & GateID & " AND `isdeny`=TRUE
;").Rows(0)(0) > 0 Then Return False
            Next
            '  - For Each User-Group's GROUP :
            For Each GroupID As DataRow In UserGroups.Rows
                '  - Query Group-Gate mapping WHERE IsDeny = FALSE, IF exists, return TRUE
                If Query("SELECT COUNT(*) FROM `map_group_gate` WHERE `groupid`=" & Val(GroupID(0)) & " AND `gateid`=" & GateID & " AND `isdeny`=FALSE
;").Rows(0)(0) > 0 Then Return True
            Next

            'If no mappings found, return FALSE
            Return False
        End Function
        Public Function CheckUserAccess(ByVal UserRFID As String, ByVal GateID As Integer) As Boolean
            'Queries - Min : 1 , Max : 3 + (2 * User-Group) ~~ 7
            Dim UserInfo As DataTable = Query("SELECT `id` FROM `data_users` WHERE `rfid`=""" & UserRFID & """;")
            Dim UserID As Integer
            If UserInfo.Rows.Count > 0 Then
                UserID = Query("SELECT `id` FROM `data_users` WHERE `rfid`=""" & UserRFID & """;")(0)(0)
            Else
                Return False
            End If

            'Query User-Gate mapping WHERE IsDeny = TRUE, IF exist, return FALSE
```

```
            If Query("SELECT COUNT(*) FROM `map_user_gate` WHERE `userid`=" & UserID & " AND `gateid`=" & GateID & " AND `isdeny`=TRUE ;").Rows(0)(0) > 0 Then
Return False
            'Query User-Gate mapping WHERE IsDeny = FALSE, IF exist, return TRUE
            If Query("SELECT COUNT(*) FROM `map_user_gate` WHERE `userid`=" & UserID & " AND `gateid`=" & GateID & " AND `isdeny`=FALSE ;").Rows(0)(0) > 0
Then Return True

            'Query User-Group mapping
            Dim UserGroups As DataTable = Query("SELECT `groupid` FROM `map_user_group` WHERE `userid`=" & UserID & " ;")
            '  - For Each User-Group's GROUP :
            For Each GroupID As DataRow In UserGroups.Rows
                '  - Query Group-Gate mapping WHERE IsDeny = TRUE, IF exists, return FALSE
                If Query("SELECT COUNT(*) FROM `map_group_gate` WHERE `groupid`=" & Val(GroupID(0)) & " AND `gateid`=" & GateID & " AND `isdeny`=TRUE
;").Rows(0)(0) > 0 Then Return False
            Next
            '  - For Each User-Group's GROUP :
            For Each GroupID As DataRow In UserGroups.Rows
                '  - Query Group-Gate mapping WHERE IsDeny = FALSE, IF exists, return TRUE
                If Query("SELECT COUNT(*) FROM `map_group_gate` WHERE `groupid`=" & Val(GroupID(0)) & " AND `gateid`=" & GateID & " AND `isdeny`=FALSE
;").Rows(0)(0) > 0 Then Return True
            Next

            'If no mappings found, return FALSE
            Return False
        End Function

        Public Function AddGroup(ByVal GroupName As String) As Integer
            If Query("SELECT COUNT(*) FROM `data_groups` WHERE `name`=""" & GroupName & """;").Rows(0)(0) > 0 Then Return -1
            Return Execute("INSERT INTO `data_groups` (`name`) VALUES (""" & GroupName & """);")
        End Function

        Public Function AddUser(ByVal RFID As String, ByVal UserName As String, Optional ByVal UserTitle As String = "") As Integer
            If Query("SELECT COUNT(*) FROM `data_users` WHERE `rfid`=""" & RFID & """;").Rows(0)(0) > 0 Then Return -1
            Return Execute("INSERT INTO `data_users` (`rfid`,`name`,`title`) VALUES (""" & RFID & """,""" & UserName & """,""" & UserTitle & """);")
        End Function

        Public Function AddGate(ByVal GateLocation As String) As Integer
            If Query("SELECT COUNT(*) FROM `data_gates` WHERE `location`=""" & GateLocation & """;").Rows(0)(0) > 0 Then Return -1
            Return Execute("INSERT INTO `data_gates` (`location`) VALUES (""" & GateLocation & """);")
        End Function
```

```
        Public Function ModifyGroup(ByVal ID As Integer, ByVal GroupName As String) As Integer
            If Query("SELECT COUNT(*) FROM `data_groups` WHERE `name`=""" & GroupName & """;").Rows(0)(0) > 0 Then
                If Query("SELECT `name` FROM `data_groups` WHERE `id`=" & ID & ";").Rows(0)(0) <> GroupName Then
                    Return -1
                End If
            End If
            Return Execute("UPDATE `data_groups` SET `name`=""" & GroupName & """ WHERE `id`=" & ID & ";")
        End Function

        Public Function ModifyUser(ByVal ID As Integer, ByVal RFID As String, ByVal UserName As String, Optional ByVal UserTitle As String = "") As Integer
            If Query("SELECT COUNT(*) FROM `data_users` WHERE `rfid`=""" & RFID & """;").Rows(0)(0) > 0 Then
                If Query("SELECT `rfid` FROM `data_users` WHERE `id`=" & ID & ";").Rows(0)(0) <> RFID Then
                    Return -1
                End If
            End If
            Return Execute("UPDATE `data_users` SET `rfid`=""" & RFID & """ , `name`=""" & UserName & """ , `title`=""" & UserTitle & """ WHERE `id`=" & ID &
";")
        End Function

        Public Function ModifyGate(ByVal ID As Integer, ByVal GateLocation As String) As Integer
            If Query("SELECT COUNT(*) FROM `data_gates` WHERE `location`=""" & GateLocation & """;").Rows(0)(0) > 0 Then
                If Query("SELECT `location` FROM `data_gates` WHERE `id`=" & ID & ";").Rows(0)(0) <> GateLocation Then
                    Return -1
                End If
            End If
            Return Execute("UPDATE `data_gates` SET `location`=""" & GateLocation & """ WHERE `id`=" & ID & ";")
        End Function

#End Region
    End Class
End Namespace
```

APPENDIX 4: INTERFACE SOURCE CODES

```
<Window x:Class="wndUserInfo"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="User Info" Height="193" Width="400">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="120"></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition Height="40"></RowDefinition>
            <RowDefinition Height="40"></RowDefinition>
            <RowDefinition Height="40"></RowDefinition>
            <RowDefinition></RowDefinition>
        </Grid.RowDefinitions>
        <Label Grid.Row="0" Grid.Column="0" Margin="5" HorizontalAlignment="Right">RFID Unique ID : </Label>
        <Label Grid.Row="1" Grid.Column="0" Margin="5" HorizontalAlignment="Right">Name :</Label>
        <Label Grid.Row="2" Grid.Column="0" Margin="5" HorizontalAlignment="Right">Title :</Label>
        <TextBox Name="txtRFID" Grid.Row="0" Grid.Column="1" Margin="5"></TextBox>
        <TextBox Name="txtName" Grid.Row="1" Grid.Column="1" Margin="5"></TextBox>
        <TextBox Name="txtTitle" Grid.Row="2" Grid.Column="1" Margin="5"></TextBox>

        <Grid Grid.Row="3" Grid.Column="1">
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="18"></ColumnDefinition>
                <ColumnDefinition></ColumnDefinition>
                <ColumnDefinition></ColumnDefinition>
            </Grid.ColumnDefinitions>
            <Button Grid.Column="1" Name="cmdOK" Margin="5">OK</Button>
            <Button Grid.Column="2" Name="cmdCancel" Margin="5">Cancel</Button>
        </Grid>
    </Grid>
</Window>
```

```
<Window x:Class="wndGate"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="wndGroup" Height="674" Width="728" Name="wndGate">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="110"></ColumnDefinition>
            <ColumnDefinition Width="110"></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition></RowDefinition>
            <RowDefinition></RowDefinition>
            <RowDefinition></RowDefinition>
            <RowDefinition Height="35"></RowDefinition>
        </Grid.RowDefinitions>
        <GroupBox Grid.RowSpan="3" Grid.ColumnSpan="2" Header="Gate List" Margin="5">
            <ListBox Name="lstGate" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
        </GroupBox>
        <Button Grid.Column="0" Grid.Row="3" Margin="5">New Gate...</Button>
        <Button Grid.Column="1" Grid.Row="3" Margin="5" IsEnabled="False">Delete</Button>
        <GroupBox Grid.Column="2" Grid.Row="0" Header="Gate Information" Margin="5">
            <Grid>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition></ColumnDefinition>
                    <ColumnDefinition></ColumnDefinition>
                </Grid.ColumnDefinitions>
                <Grid.RowDefinitions>
                    <RowDefinition></RowDefinition>
                    <RowDefinition Height="35"></RowDefinition>
                </Grid.RowDefinitions>
```

```
        <Grid Grid.Row="0" Grid.ColumnSpan="2">
            <Grid.RowDefinitions>
                <RowDefinition></RowDefinition>
                <RowDefinition></RowDefinition>
                <RowDefinition></RowDefinition>
                <RowDefinition></RowDefinition>
            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
                <ColumnDefinition></ColumnDefinition>
                <ColumnDefinition></ColumnDefinition>
                <ColumnDefinition></ColumnDefinition>
            </Grid.ColumnDefinitions>
        </Grid>
        <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Save</Button>
    </Grid>
</GroupBox>
<GroupBox Grid.Column="2" Grid.Row="1" Header="Groups Allowed Access" Margin="5">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition></RowDefinition>
            <RowDefinition Height="35"></RowDefinition>
        </Grid.RowDefinitions>
        <ListBox Name="lstGroupAllow" Grid.ColumnSpan="2" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
        <Button Grid.Row="1" Grid.Column="0" Margin="5">Add</Button>
        <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Remove</Button>
    </Grid>
</GroupBox>
```

```
<GroupBox Grid.Column="3" Grid.Row="0" Header="Gate Auto-Open Times" Margin="5">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition></RowDefinition>
            <RowDefinition Height="35"></RowDefinition>
        </Grid.RowDefinitions>
        <ListBox Grid.ColumnSpan="2" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
        <Button Grid.Row="1" Grid.Column="0" Margin="5">Add</Button>
        <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Remove</Button>
    </Grid>
</GroupBox>
<GroupBox Grid.Column="3" Grid.Row="1" Header="Users Allowed Access" Margin="5">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition></RowDefinition>
            <RowDefinition Height="35"></RowDefinition>
        </Grid.RowDefinitions>
        <ListBox Name="lstUserAllow" Grid.ColumnSpan="2" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
        <Button Grid.Row="1" Grid.Column="0" Margin="5">Add</Button>
        <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Remove</Button>
    </Grid>
</GroupBox>
```

```
        <GroupBox Grid.Column="2" Grid.Row="2" Header="Groups Denied Access" Margin="5">
            <Grid>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition></ColumnDefinition>
                    <ColumnDefinition></ColumnDefinition>
                </Grid.ColumnDefinitions>
                <Grid.RowDefinitions>
                    <RowDefinition></RowDefinition>
                    <RowDefinition Height="35"></RowDefinition>
                </Grid.RowDefinitions>
                <ListBox Name="lstGroupDeny" Grid.ColumnSpan="2" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
                <Button Grid.Row="1" Grid.Column="0" Margin="5">Add</Button>
                <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Remove</Button>
            </Grid>
        </GroupBox>
        <GroupBox Grid.Column="3" Grid.Row="2" Header="Users Denied Access" Margin="5">
            <Grid>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition></ColumnDefinition>
                    <ColumnDefinition></ColumnDefinition>
                </Grid.ColumnDefinitions>
                <Grid.RowDefinitions>
                    <RowDefinition></RowDefinition>
                    <RowDefinition Height="35"></RowDefinition>
                </Grid.RowDefinitions>
                <ListBox Name="lstUserDeny" Grid.ColumnSpan="2" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
                <Button Grid.Row="1" Grid.Column="0" Margin="5">Add</Button>
                <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Remove</Button>
            </Grid>
        </GroupBox>

        <Grid Grid.Column="3" Grid.Row="3">
            <Grid.ColumnDefinitions>
                <ColumnDefinition></ColumnDefinition>
                <ColumnDefinition Width="110"></ColumnDefinition>
            </Grid.ColumnDefinitions>
            <Button Grid.Column="1" Margin="5">Exit</Button>
        </Grid>
    </Grid>
</Window>
```

```
<Window x:Class="wndGroup"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="wndGroup" Height="572" Width="728" Name="wndGroup">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="110"></ColumnDefinition>
            <ColumnDefinition Width="110"></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition></RowDefinition>
            <RowDefinition></RowDefinition>
            <RowDefinition Height="35"></RowDefinition>
        </Grid.RowDefinitions>
        <GroupBox Grid.RowSpan="2" Grid.ColumnSpan="2" Header="Group List" Margin="5">
            <ListBox Name="lstGroup" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
        </GroupBox>
        <Button Grid.Column="0" Grid.Row="2" Margin="5">New Group...</Button>
        <Button Grid.Column="1" Grid.Row="2" Margin="5" IsEnabled="False">Delete</Button>
        <GroupBox Grid.Column="2" Grid.Row="0" Header="Group Information" Margin="5">
            <Grid>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition></ColumnDefinition>
                    <ColumnDefinition></ColumnDefinition>
                </Grid.ColumnDefinitions>
                <Grid.RowDefinitions>
                    <RowDefinition></RowDefinition>
                    <RowDefinition Height="35"></RowDefinition>
                </Grid.RowDefinitions>
```

```
        <Grid Grid.Row="0" Grid.ColumnSpan="2">
            <Grid.RowDefinitions>
                <RowDefinition></RowDefinition>
                <RowDefinition></RowDefinition>
                <RowDefinition></RowDefinition>
                <RowDefinition></RowDefinition>
                <RowDefinition></RowDefinition>
                <RowDefinition></RowDefinition>
            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
                <ColumnDefinition></ColumnDefinition>
                <ColumnDefinition></ColumnDefinition>
                <ColumnDefinition></ColumnDefinition>
            </Grid.ColumnDefinitions>
        </Grid>
        <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Save</Button>
    </Grid>
</GroupBox>
<GroupBox Grid.Column="2" Grid.Row="1" Header="Users in Group" Margin="5">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition></RowDefinition>
            <RowDefinition Height="35"></RowDefinition>
        </Grid.RowDefinitions>
        <ListBox Name="lstUsers" Grid.ColumnSpan="2" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
        <Button Grid.Row="1" Grid.Column="0" Margin="5">Add</Button>
        <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Remove</Button>
    </Grid>
</GroupBox>
```

```
        <GroupBox Grid.Column="3" Grid.Row="0" Header="Allowed Access" Margin="5">
            <Grid>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition></ColumnDefinition>
                    <ColumnDefinition></ColumnDefinition>
                </Grid.ColumnDefinitions>
                <Grid.RowDefinitions>
                    <RowDefinition></RowDefinition>
                    <RowDefinition Height="35"></RowDefinition>
                </Grid.RowDefinitions>
                <ListBox Name="lstAllow" Grid.ColumnSpan="2" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
                <Button Grid.Row="1" Grid.Column="0" Margin="5">Add</Button>
                <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Remove</Button>
            </Grid>
        </GroupBox>
        <GroupBox Grid.Column="3" Grid.Row="1" Header="Denied Access" Margin="5">
            <Grid>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition></ColumnDefinition>
                    <ColumnDefinition></ColumnDefinition>
                </Grid.ColumnDefinitions>
                <Grid.RowDefinitions>
                    <RowDefinition></RowDefinition>
                    <RowDefinition Height="35"></RowDefinition>
                </Grid.RowDefinitions>
                <ListBox Name="lstDeny" Grid.ColumnSpan="2" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
                <Button Grid.Row="1" Grid.Column="0" Margin="5">Add</Button>
                <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Remove</Button>
            </Grid>
        </GroupBox>
        <Grid Grid.Column="3" Grid.Row="2">
            <Grid.ColumnDefinitions>
                <ColumnDefinition></ColumnDefinition>
                <ColumnDefinition Width="110"></ColumnDefinition>
            </Grid.ColumnDefinitions>
            <Button Grid.Column="1" Margin="5">Exit</Button>
        </Grid>
    </Grid>
</Window>
```

```
<Window x:Class="wndUser"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="wndUser" Height="572" Width="726" Name="wndUser">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="110"></ColumnDefinition>
            <ColumnDefinition Width="110"></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition></RowDefinition>
            <RowDefinition></RowDefinition>
            <RowDefinition Height="35"></RowDefinition>
        </Grid.RowDefinitions>
        <GroupBox Grid.RowSpan="2" Grid.ColumnSpan="2" Header="User List" Margin="5">
            <ListBox Name="lstUsers" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
        </GroupBox>
        <Button Name="cmdAddUser" Grid.Column="0" Grid.Row="2" Margin="5">New User...</Button>
        <Button Name="cmdRemUser" Grid.Column="1" Grid.Row="2" Margin="5" IsEnabled="False">Delete</Button>
        <GroupBox Grid.Column="2" Grid.Row="0" Header="User Particulars" Margin="5">
            <Grid>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition></ColumnDefinition>
                    <ColumnDefinition></ColumnDefinition>
                </Grid.ColumnDefinitions>
                <Grid.RowDefinitions>
                    <RowDefinition></RowDefinition>
                    <RowDefinition Height="35"></RowDefinition>
                </Grid.RowDefinitions>
                <Grid Grid.Row="0" Grid.ColumnSpan="2">
                    <Grid.RowDefinitions>
                        <RowDefinition></RowDefinition>
                        <RowDefinition></RowDefinition>
                        <RowDefinition></RowDefinition>
                        <RowDefinition></RowDefinition>
                        <RowDefinition></RowDefinition>
                        <RowDefinition></RowDefinition>
                    </Grid.RowDefinitions>
```

```
            <Grid.ColumnDefinitions>
                <ColumnDefinition></ColumnDefinition>
                <ColumnDefinition></ColumnDefinition>
                <ColumnDefinition></ColumnDefinition>
            </Grid.ColumnDefinitions>
        </Grid>
        <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Save</Button>
    </Grid>
</GroupBox>
<GroupBox Grid.Column="2" Grid.Row="1" Header="User's Groups" Margin="5">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition></RowDefinition>
            <RowDefinition Height="35"></RowDefinition>
        </Grid.RowDefinitions>
        <ListBox Name="lstGroup" Grid.ColumnSpan="2" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
        <Button Grid.Row="1" Grid.Column="0" Margin="5">Add</Button>
        <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Remove</Button>
    </Grid>
</GroupBox>
<GroupBox Grid.Column="3" Grid.Row="0" Header="Explicitly Allowed Access" Margin="5">
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition></RowDefinition>
            <RowDefinition Height="35"></RowDefinition>
        </Grid.RowDefinitions>
        <ListBox Name="lstAllow" Grid.ColumnSpan="2" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
        <Button Grid.Row="1" Grid.Column="0" Margin="5">Add</Button>
        <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Remove</Button>
    </Grid>
</GroupBox>
```

```
        <GroupBox Grid.Column="3" Grid.Row="1" Header="Explicitly Denied Access" Margin="5">
            <Grid>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition></ColumnDefinition>
                    <ColumnDefinition></ColumnDefinition>
                </Grid.ColumnDefinitions>
                <Grid.RowDefinitions>
                    <RowDefinition></RowDefinition>
                    <RowDefinition Height="35"></RowDefinition>
                </Grid.RowDefinitions>
                <ListBox Name="lstDeny" Grid.ColumnSpan="2" ScrollViewer.VerticalScrollBarVisibility="Visible"></ListBox>
                <Button Grid.Row="1" Grid.Column="0" Margin="5">Add</Button>
                <Button Grid.Row="1" Grid.Column="1" Margin="5" IsEnabled="False">Remove</Button>
            </Grid>
        </GroupBox>
        <Grid Grid.Column="3" Grid.Row="2">
            <Grid.ColumnDefinitions>
                <ColumnDefinition></ColumnDefinition>
                <ColumnDefinition Width="110"></ColumnDefinition>
            </Grid.ColumnDefinitions>
            <Button Grid.Column="1" Margin="5">Exit</Button>
        </Grid>
    </Grid>
</Window>
```

```
<Window x:Class="wndPortal"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="SkyGate Administration" Height="231" Width="502" ResizeMode="NoResize">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition></RowDefinition>
            <RowDefinition Height="35"></RowDefinition>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <GroupBox Header="System Info" Margin="5">
            <Grid>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="0.25*"></ColumnDefinition>
                    <ColumnDefinition Width="0.05*"></ColumnDefinition>
                    <ColumnDefinition Width="0.7*"></ColumnDefinition>
                </Grid.ColumnDefinitions>
                <Grid.RowDefinitions>
                    <RowDefinition></RowDefinition>
                    <RowDefinition></RowDefinition>
                    <RowDefinition></RowDefinition>
                </Grid.RowDefinitions>
                <TextBlock Grid.Row="0" Grid.Column="1" TextAlignment="Center" VerticalAlignment="Center">:</TextBlock>
                <TextBlock Grid.Row="1" Grid.Column="1" TextAlignment="Center" VerticalAlignment="Center">:</TextBlock>
                <TextBlock Grid.Row="2" Grid.Column="1" TextAlignment="Center" VerticalAlignment="Center">:</TextBlock>
                <TextBlock Grid.Row="0" TextAlignment="Left" VerticalAlignment="Center" Margin="5">Users</TextBlock>
                <TextBlock Grid.Row="1" TextAlignment="Left" VerticalAlignment="Center" Margin="5">Groups</TextBlock>
                <TextBlock Grid.Row="2" TextAlignment="Left" VerticalAlignment="Center" Margin="5">Gates</TextBlock>
            </Grid>
        </GroupBox>
```

```
        <Grid Grid.Row="1">
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="110"></ColumnDefinition>
                <ColumnDefinition Width="110"></ColumnDefinition>
                <ColumnDefinition Width="110"></ColumnDefinition>
                <ColumnDefinition></ColumnDefinition>
                <ColumnDefinition Width="110"></ColumnDefinition>
            </Grid.ColumnDefinitions>
            <Button Grid.Column="0" Margin="5" Name="cmdUser">Users</Button>
            <Button Grid.Column="1" Margin="5" Name="cmdGroup">Groups</Button>
            <Button Grid.Column="2" Margin="5" Name="cmdGate">Gates</Button>
            <Button Name="cmdExit" Click="cmdExit_Click" Grid.Column="4" Margin="5">Exit</Button>
        </Grid>
    </Grid>
</Window>
```

APPENDIX 5: BUSINESS LOGIC SOURCE CODES

```
Imports System.Data

Partial Public Class wndUser

    Private Sub wndUser_Loaded(ByVal sender As System.Object, ByVal e As System.Windows.RoutedEventArgs) Handles MyBase.Loaded
        For Each user As DataRow In MyDB.Query("SELECT * FROM `data_users` ORDER BY `name`;").Rows
            Dim UserItem As New ListBoxItem
            UserItem.Tag = user("id")
            UserItem.Content = user("name")
            lstUsers.Items.Add(UserItem)
        Next

    End Sub

    Private Sub lstUsers_SelectionChanged(ByVal sender As Object, ByVal e As System.Windows.Controls.SelectionChangedEventArgs) Handles
lstUsers.SelectionChanged
        Dim UserItem As ListBoxItem = lstUsers.SelectedItem
        Dim UserID As Integer = UserItem.Tag
        Dim UserName As String = UserItem.Content

        If lstUsers.SelectedItems.Count > 0 Then
            cmdRemUser.IsEnabled = True
        Else
            cmdRemUser.IsEnabled = False
        End If

        ' Get User's Groups
        lstGroup.Items.Clear()
        For Each group As DataRow In MyDB.Query("SELECT `groupid` FROM `map_user_group` WHERE `userid`=" & UserID & " ;").Rows
            Dim GroupItem As New ListBoxItem
            GroupItem.Tag = group("groupid")
            GroupItem.Content = MyDB.Query("SELECT `name` FROM `data_groups` WHERE `id`=" & Val(GroupItem.Tag) & " ;")(0)(0)
            lstGroup.Items.Add(GroupItem)
        Next
```

```
        ' Get Allowed Gates
        lstAllow.Items.Clear()
        For Each gate As DataRow In MyDB.Query("SELECT `gateid` FROM `map_user_gate` WHERE `userid`=" & UserID & " AND `isdeny`=FALSE ;").Rows
            Dim GateItem As New ListBoxItem
            GateItem.Tag = gate("gateid")
            GateItem.Content = MyDB.Query("SELECT `location` FROM `data_gates` WHERE `id`=" & Val(GateItem.Tag) & " ;")(0)(0)
            lstAllow.Items.Add(GateItem)
        Next

        ' Get Denied Gates
        lstDeny.Items.Clear()
        For Each gate As DataRow In MyDB.Query("SELECT `gateid` FROM `map_user_gate` WHERE `userid`=" & UserID & " AND `isdeny`=TRUE ;").Rows
            Dim GateItem As New ListBoxItem
            GateItem.Tag = gate("gateid")
            GateItem.Content = MyDB.Query("SELECT `location` FROM `data_gates` WHERE `id`=" & Val(GateItem.Tag) & " ;")(0)(0)
            lstDeny.Items.Add(GateItem)
        Next
    End Sub
End Class
```

```
Imports System.Data
Partial Public Class wndGroup

    Private Sub wndGroup_Loaded(ByVal sender As Object, ByVal e As System.Windows.RoutedEventArgs) Handles Me.Loaded
        For Each group As DataRow In MyDB.Query("SELECT * FROM `data_groups` ;").Rows
            Dim GroupItem As New ListBoxItem
            GroupItem.Tag = group("id")
            GroupItem.Content = group("name")
            lstGroup.Items.Add(GroupItem)
        Next

    End Sub

    Private Sub lstGroup_SelectionChanged(ByVal sender As Object, ByVal e As System.Windows.Controls.SelectionChangedEventArgs) Handles
lstGroup.SelectionChanged
        Dim GroupItem As ListBoxItem = lstGroup.SelectedItem
        Dim GroupID As Integer = GroupItem.Tag
        Dim GroupName As String = GroupItem.Content

        ' Get Group's Users
        lstUsers.Items.Clear()
        For Each user As DataRow In MyDB.Query("SELECT `userid` FROM `map_user_group` WHERE `groupid`=" & GroupID & " ;").Rows
            Dim UserItem As New ListBoxItem
            UserItem.Tag = user("userid")
            UserItem.Content = MyDB.Query("SELECT `name` FROM `data_users` WHERE `id`=" & Val(UserItem.Tag) & " ;").Rows(0)(0)
            lstUsers.Items.Add(UserItem)
        Next
```

```
        ' Get Denied Gates
        lstDeny.Items.Clear()
        For Each gate As DataRow In MyDB.Query("SELECT `gateid` FROM `map_group_gate` WHERE `groupid`=" & GroupID & " AND `isdeny`=TRUE ;").Rows
            Dim GateItem As New ListBoxItem
            GateItem.Tag = gate("gateid")
            GateItem.Content = MyDB.Query("SELECT `location` FROM `data_gates` WHERE `id`=" & Val(GateItem.Tag) & " ;").Rows(0)(0)
            lstDeny.Items.Add(GateItem)
        Next

        ' Get Allowed Gates
        lstAllow.Items.Clear()
        For Each gate As DataRow In MyDB.Query("SELECT `gateid` FROM `map_group_gate` WHERE `groupid`=" & GroupID & " AND `isdeny`=FALSE ;").Rows
            Dim GateItem As New ListBoxItem
            GateItem.Tag = gate("gateid")
            GateItem.Content = MyDB.Query("SELECT `location` FROM `data_gates` WHERE `id`=" & Val(GateItem.Tag) & " ;").Rows(0)(0)
            lstAllow.Items.Add(GateItem)
        Next
    End Sub
End Class
```

```
Imports System.Data
Partial Public Class wndGate

    Private Sub wndGate_Loaded(ByVal sender As Object, ByVal e As System.Windows.RoutedEventArgs) Handles Me.Loaded
        For Each gate As DataRow In MyDB.Query("SELECT * FROM `data_gates` ;").Rows
            Dim GateItem As New ListBoxItem
            GateItem.Tag = gate("id")
            GateItem.Content = gate("location")
            lstGate.Items.Add(GateItem)
        Next
    End Sub

    Private Sub lstGate_SelectionChanged(ByVal sender As Object, ByVal e As System.Windows.Controls.SelectionChangedEventArgs) Handles
lstGate.SelectionChanged
        Dim GateItem As ListBoxItem = lstGate.SelectedItem
        Dim GateID As Integer = GateItem.Tag
        Dim GateLocation As String = GateItem.Content

        'Get Denied Groups
        lstGroupDeny.Items.Clear()
        For Each group As DataRow In MyDB.Query("SELECT `groupid` FROM `map_group_gate` WHERE `gateid`=" & GateID & " AND isdeny=TRUE ;").Rows
            Dim DeniedGroupItem As New ListBoxItem
            DeniedGroupItem.Tag = group("groupid")
            DeniedGroupItem.Content = MyDB.Query("SELECT `name` FROM `data_groups` WHERE `id`=" & Val(DeniedGroupItem.Tag) & " ;").Rows(0)("name")
            lstGroupDeny.Items.Add(DeniedGroupItem)
        Next

        'Get Allowed Groups
        lstGroupAllow.Items.Clear()
        For Each group As DataRow In MyDB.Query("SELECT `groupid` FROM `map_group_gate` WHERE `gateid`=" & GateID & " AND isdeny=FALSE ;").Rows
            Dim AllowedGroupItem As New ListBoxItem
            AllowedGroupItem.Tag = group("groupid")
            AllowedGroupItem.Content = MyDB.Query("SELECT `name` FROM `data_groups` WHERE `id`=" & Val(AllowedGroupItem.Tag) & " ;").Rows(0)("name")
            lstGroupAllow.Items.Add(AllowedGroupItem)
        Next
```

```
        'Get Denied Users
        lstUserDeny.Items.Clear()
        For Each group As DataRow In MyDB.Query("SELECT `userid` FROM `map_user_gate` WHERE `gateid`=" & GateID & " AND isdeny=TRUE ;").Rows
            Dim DeniedUserItem As New ListBoxItem
            DeniedUserItem.Tag = group("userid")
            DeniedUserItem.Content = MyDB.Query("SELECT `name` FROM `data_users` WHERE `id`=" & Val(DeniedUserItem.Tag) & " ;").Rows(0)("name")
            lstUserDeny.Items.Add(DeniedUserItem)
        Next

        'Get Allowed Users
        lstUserAllow.Items.Clear()
        For Each group As DataRow In MyDB.Query("SELECT `userid` FROM `map_user_gate` WHERE `gateid`=" & GateID & " AND isdeny=FALSE ;").Rows
            Dim AllowedUserItem As New ListBoxItem
            AllowedUserItem.Tag = group("userid")
            AllowedUserItem.Content = MyDB.Query("SELECT `name` FROM `data_users` WHERE `id`=" & Val(AllowedUserItem.Tag) & " ;").Rows(0)("name")
            lstUserAllow.Items.Add(AllowedUserItem)
        Next
    End Sub
End Class
```