This Project's goal was to develop a secure Automated Lock System which inherently keeps a log of people entering a hall of residence providing easy and safe entrance.

COMP3901: Group 4 Document

Automated Lock System

Andrew Philp
Jevoi Harvey
Samath Scott
Markland Payne

Semester 2,
26 May 2017

The University of the West Indies, Mona Campus
Professor Daniel N. Coore

TABLE OF CONTENTS

INTRODUCTION

In recent times, there has been several requests and attempts to improve the efficiency of security within the halls of residence of the University of the West Indies. Currently the halls are secured by security personnel's who perform manual verification checks on each entrant to the hall.

The existing system is flawed in a way that a person seeking entry to a hall may need identification which they may or may not have. Residents are allowed within the hall once the security guard on duty recognizes their faces as true residents to the hall. This verification may lead to a resident being denied or delayed access to the hall. In the case of visitors, security guards use a hand written "Visitor's Log Book" for storing visitor's information. In addition to being a tedious process, it has also proven futile in allowing security guards to efficiently keep track of all the visitors who went over the log out time regulations put in place by each hall.

As the system stands, persons may be able to enter the hall without being a resident through means of bypassing the view of the security; there is no true reflection of all the visitors on the hall's premises. Students have raised issues of discomfort and harassment due to the constant manual identification checks if they are not recognized by the security guard on duty, as well as hall regulations concerning visitors are not being adhered to.

PROJECT PROBLEM

Considering the above issues, a computer system was designed to automate varying aspects of the hall's entry process. The designed system facilitates the automatic opening of a gate lock through integration of several computer applications and hardware connections. Figure 1.0 highlights the aspects of the process that the development team took into consideration when creating the solution for the current system.

Upon requesting entrance, an individual must provide their personal identification information. The system must then process this information and make checks against the record of residents to verify if the individual is a resident of the hall. If the entrant is a resident, the lock should open allowing the person to enter. If the individual requesting entrance is not a resident, he/she will be treated as a visitor and the name and room number of the resident being visited should be further provided and all such information must be recorded in a log of visitor's.

In ensuring that all the functionalities were met, several issues were considered to make the solution optimal.

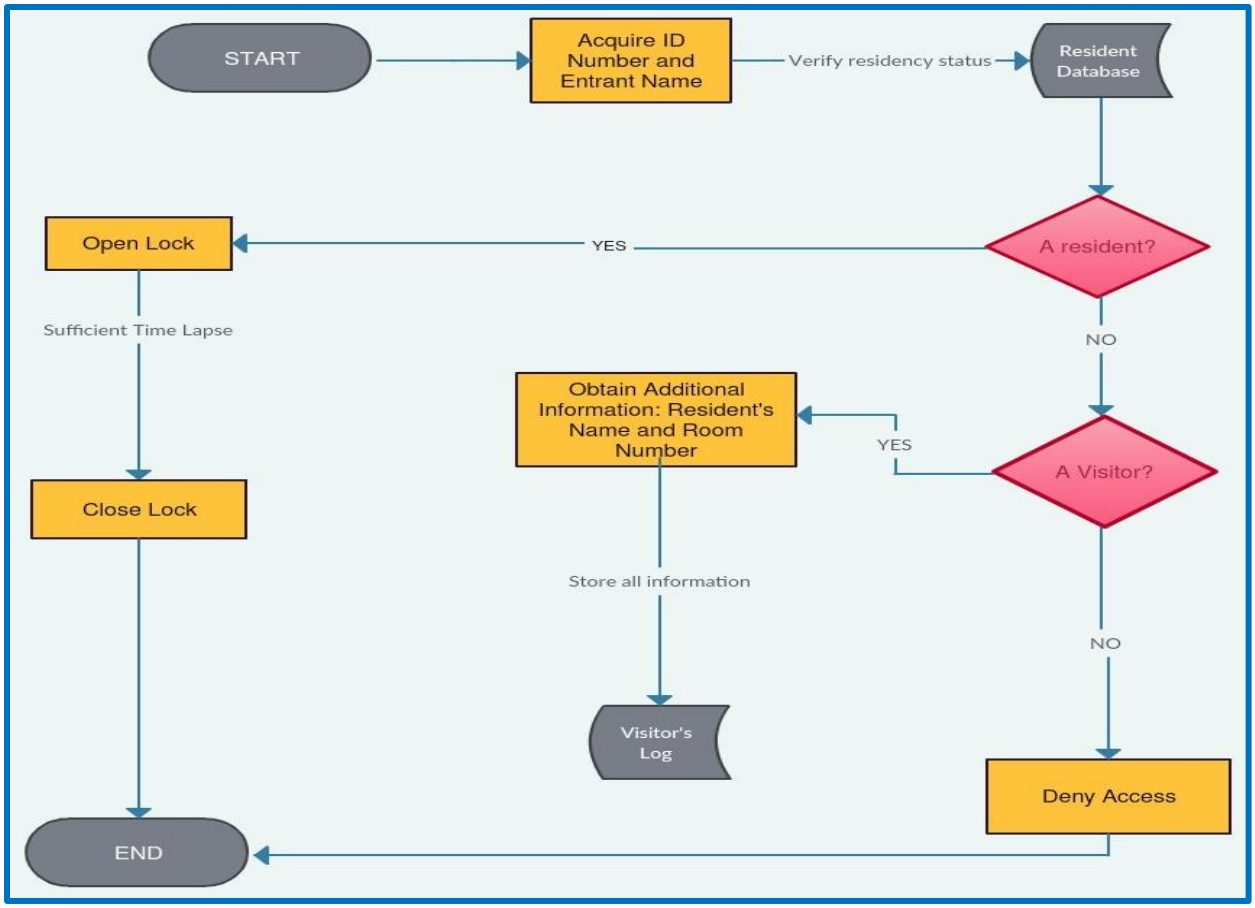

Figure 1.0: Shows the general process flow involved in gaining access to a hall.

ISSUES TO BE ADDRESSED

- The collection and storage of resident's and non-resident's necessary information.
- Accurate verification of a resident, i.e. properly and efficiently identifying residents.
- Proper logging of the information of those who access the hall; visitors and residents alike.
- Ensuring that hall regulations are enforced and the system designed allows for the altering of those regulations by authorized persons. Regulations include time curfews for visitors, etc.
- Mechanism to control opening and closing of the lock.
- Ensuring that only approved residents are always allowed to enter.

SECURITY ISSUES

- Ensuring that only authorized persons have access to the database that stores information of persons deemed as residents. The database for residents should enforce authorization constraints to allow only administration to add or remove a resident from the database. The system should also ensure that such constraints are not easily defeated or breached, for example via spoofing.

- Ensuring safe network communications between all components of the designed system. As well as ensuring that no unauthorized device or application can maliciously send a signal to open the lock.

- Ensuring that the use of overriding the system is not misused. Overriding the system unnecessarily or without proper permission was deemed an issue essential to be addressed.

This system was developed to perform as a secure Automated Lock System which inherently keeps a log of individuals entering a hall of residence whilst providing an easy and safe entrance. This goal was accomplished and the team highlighted the problems which may arise or arose during the design process of the system. More details on this is found further in the document.

BACKGROUND

PROJECT SOLUTION

The solution that was generated to address the functionalities and issues highlighted consists of several components, both hardware and software, which works together to create the Automated Lock System. The system was created to keep a log of all entrants to the hall. Once the gate is opened, a summary of the person's information, and the time and date is stored in a temporary log on the system. This log is available to security guards and administrators for inquiry and statistical purposes.

MAIN SUMMARY

Initially, our team made the decision to read the barcode from the student's ID for opening the lock. However, it was then analysed that a current problem being faced is that students often do not have their physical ID present and therefore entering the hall would be denied. The team decided to build an Android Mobile Application (App) which would ease the pressure off the necessity of ID(s). The App became another method of entering the Hall; the App would become available for download for all residents. Given that a resident may opt out of using the App due to likely constraints such as the absence of some wireless internet connection or other reasons. The third method of entering the Hall is to enter the ID number through a desktop interface which was designed to be used by the security.

Each option essentially catches the short-comings of the other two not necessarily due to design issues but moreover unpredictability of human behaviour. With the use of any options, the necessary inputs are gathered and sent to the database server to query if the individual is a resident of the hall.

The system designed incorporated the use of a Camera used as a Barcode Scanner, a Mobile Application (Android), a Raspberry Pi, a Desktop Application, and a Database Server. Each component was designed and built to handle an individual function of the entire system and integrated to make the entire system functional.

The ways in which the system opens the lock are dependent on the following components:

- ✓ **Mobile Application** – which allows a resident to log in using the same ID and Password used with a regular UWI system. Upon logging in, the mobile application presents a button labelled 'Open', which the user then presses to initiate the request to open the lock.

- ✓ **Barcode Scanner** – this component scans the identification cards of individuals to retrieve the necessary personal information such as ID Number, name, and image.

- ✓ **Desktop Application** – provides a user-friendly user interface for the security guard. It is one of the main component of the system, as most the communication is handled by it before sending to the Pi. The authorization for opening the lock involves sending a secured message from the desktop application to the Raspberry Pi. The Pi is then responsible for sending the electronic signal to the lock, and initiating a timer; the lock then closes after the set time is reached.

The system was designed to allow for the adjusting of this timing constraint on the lock, by a security guard or administrator, in the case of an emergency, i.e. fire etc. on the hall premises. Visitors will provide information which includes the individual's legal name, a student or government recognized ID and the room number of the resident being visited. A time stamp is kept for all visitors on the hall. It automatically notifies the security, via a message on the desktop, stating all visitors who are not yet logged out, i.e. overspending their time after 1:00 A.M. or any time stipulated by the administrative rules of the hall.

UNDERLYING FEATURES

Necessary for the system to function correctly and efficiently, features such as access control and other security features, as well as managerial and maintenance features were also implemented. To ensure that the system's core components are only accessible to authorized users, each authorised personnel is given a username and password to the system. Features may be unique to administrators and likewise for that of security guards.

Firstly, residents are entered and/or removed from the system's database by authorized administrators via the appropriate form accessible on the desktop interface of the system. Via the desktop application, an administrator is also capable of temporarily disabling residency of a person for reasons that may comply to hall policies. For example, students who have not paid hall fee can be banned from entering. Thus, by disabling their residency status, their identification credentials will no longer be authorized to open the lock.

While designing the system, we made note of the fact that technology malfunctions or failures may lead to situations arising where a duly authorized resident is not able to enter the premises of the hall. As such, overrides of the system were incorporated into the system design. Therefore, a security guard or administrator can manually override the system. Thus, allowing the gate to open in situations when it isn't opening and it should, as well as preventing the gate from opening.

For this manual override, the security guard would press the physical button that currently exists in the manual system to open the gate. Pressing the button initiates an override of the automatic designed system which will cause the system to make a log of this override. Such a log is to allow administration to be able to see how many times an individual initiates an override and serves as a means of discouraging authorized users from override the system unnecessarily or constantly.

Finally, in addressing the issue of safe communication between all applications and components, we incorporated the AES Algorithm. All messages sent between the components of the system are hashed and then sent using AES Algorithm via a TCP connection.

PROJECT SCOPE

IN-SCOPE

Listed are the tasks that were accomplished:

- Implemented the barcode reader.
- Created and designed the desktop interface and database.
- Programmed the Raspberry Pi.
- Created the mobile application.
- Kept logs of entrants.
- Integrated the desktop application, the server, the mobile application, the Raspberry PI, the Breadboard and the LED lights to open a lock.
- Created an interface to allow future implementation of facial recognition.

OUT OF SCOPE

- **Opening the physical lock** – Essentially, it was realized that the physical lock cannot be done solely using the raspberry pi, as the voltage released from the pi is not enough to trigger the movement of a lock. Thus, we would've needed to do some amplification of the voltage which would require the use of a relay, a NPN transistor, resistors, and a diode to safely generate the required voltage, i.e. approximately 12V. In addition to being more focused on the mechanical aspect of the design rather than the software aspects, the process was considered as a bit time consuming and was a risk to the successful completion of the project in the given time.

- **Automatic facial recognition** – this is a method that was considered to help the security recognise the face of the individual and open the lock while security need not do anything but observe. The system we designed will facilitate this method in the future.

- **GPS Monitoring** – this would've been used with App to reduce overloading and sending an Open request from anywhere.

- **Facilitating exit of the hall** – This was not incorporated in the scope of the project as it is more concerned with the physically opening of the lock from the inside without the need for verification of identity and any other service provided by the components of the system.

SIMILAR SYSTEMS THAT EXISTS

FEATURE	AUTOMATED LOCK SYSTEM	GATEHOUSE	ISN SECURITY GATEKEEPER
Desktop Application	✓	✓ "Only temporary visitor information can be added or amended in this interface"	✓
Barcode Scanner Application	✓	✓ Used for expedited and self-re-entry of visitors	✗ Uses RFID on Cars
Mobile Application	✓	✗ Planned Implementation	✗
Proper Logging	✓	✓	✓
Easy Manipulation of System Features to Match Changing Organizational Needs	✓	✓	✗
Secured Communication between all devices	✓	✓	✓
Works without Internet Connection/ Distributed Database	✗	✓	✗
Interface for facial recognition	✓	✗	✓
Visitor Management	✓	✓	✓
Restriction of Residents based on Authorized Conditions	✓	✓	✗
Resident Interfaces (to manage their own visitors beforehand)	✗	✓	✓
Free of Cost	✓	✗	✗

FUNCTIONAL REQUIREMENTS

USER REQUIREMENTS

- The Administrator can:
  - ✓ Register new residents.
  - ✓ Delete a retired resident from the database.

- The security guard can:
  - ✓ Input resident's ID number to provide the resident access to the hall.
  - ✓ See all relevant information of resident's, i.e. name, picture, and room number.
  - ✓ Log a visitor's information in the system.
  - ✓ Log in a recurring visitor without the need to re-enter their information if said visitor visited the premises within a specified time span.
  - ✓ Log out a visitor upon them leaving the hall of residence.
  - ✓ View all visitors who are not yet logged out by 1:00 A.M.
  - ✓ Verify/Approve opening of the lock for anyone seeking entrance to the premises.

- A resident can:
  - ✓ Scan their ID cards to gain entrance.
  - ✓ Login to the mobile application with proper credentials.
  - ✓ Input their ID number and a created password using App.
  - ✓ Request entrance to the hall via mobile application.

- A visitor can gain entrance to the hall after providing security guard with required information.

SYSTEM REQUIREMENTS

- The system allows:
  - ✓ Verification of a user's identity upon seeking entrance.
  - ✓ Authorized registration/deletion of new residents.
  - ✓ Authorized personnel to manually open the lock if the system refuses to open the gate at an instance where it should.

- The system facilitates:
  - ✓ Scanning and decoding of an ID barcode.
  - ✓ Overrides from authorized personnel, i.e. for emergencies, etc.

- The system is adaptable to any hall specific "entrance policy" changes that may occur.

NON-FUNCTIONAL REQUIREMENTS

SECURITY REQUIREMENTS

- The system makes logs of all occurrences of overrides.
- Logs includes the personnel that initiated the override, and the date and time of the override.
- The system ensures a secure communication between server, desktop application, mobile application, and the raspberry pi.
- The system authenticates all security guard and administrative users before allowing them access to the system's database and other functions.
- Being on a network that can be disturbed by hackers, protection systems such as firewalls and virus protection systems shall be in place on each computer system within the organization that uses the automated lock system.

PERFORMANCE REQUIREMENTS

- The system's user interface takes no longer than 1 minute to boot.
- Evolution qualities such as testability, maintainability, scalability, and extensibility are embodied in the structure of the software system.
- The system shall take no more than 1 minute between the collection and verification of all information necessary to open the lock, and the actual opening of the lock.

BEHAVIOURAL REQUIREMENTS

USE CASE DIAGRAM

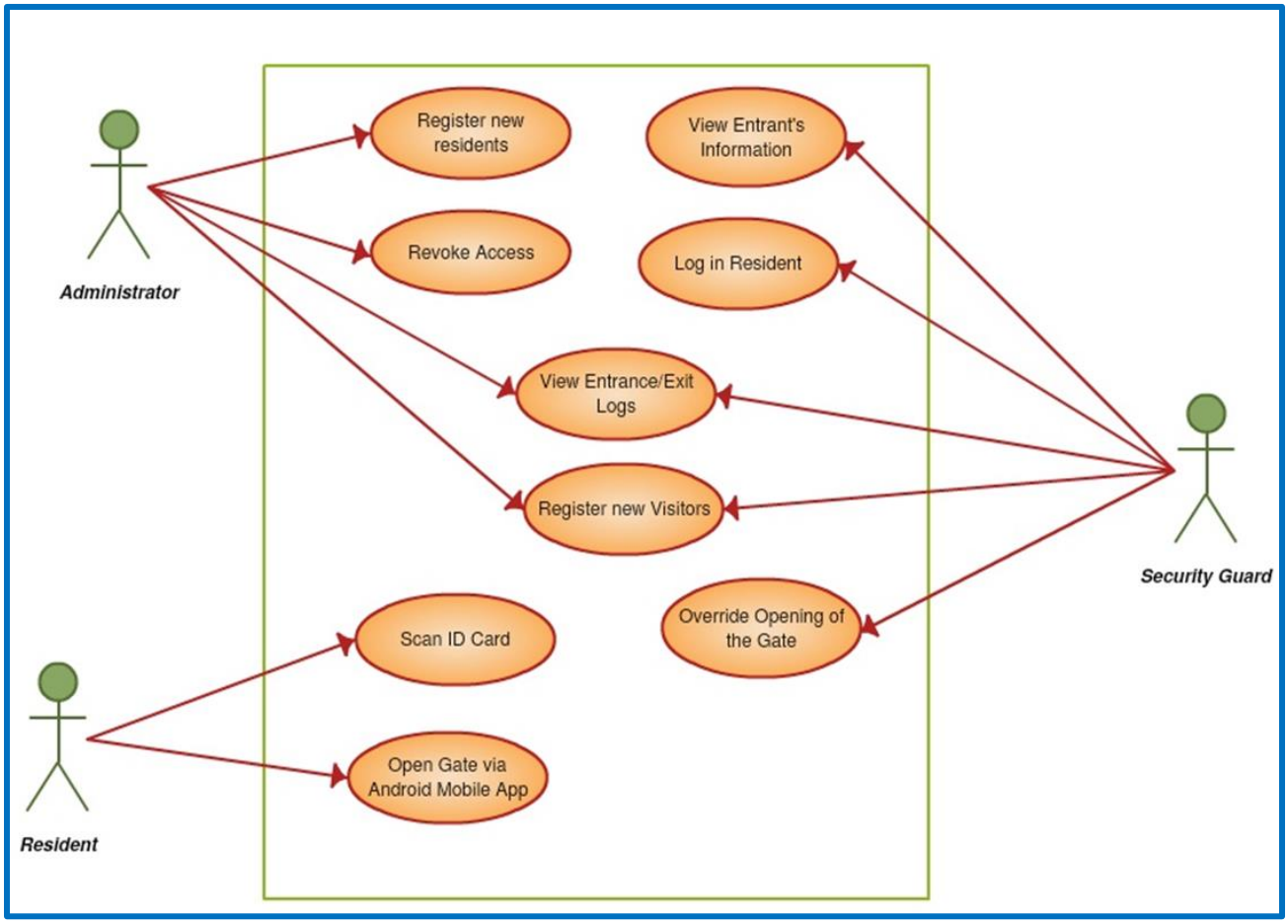

Figure 2.0: Use Case Diagram

DETAILED USE CASE FOR LOGIN AS A RESIDENT

Use Case Name	Resident Login
Description	This use case depicts how the security guard uses the system to log in a resident if a resident does not have present his/her physical identification card present.
Participating Actors	Security
Trigger/Action	Security clicks log in resident icon.
Pre-Condition	1. The desktop application is opened. 2. Security guard correctly entered login information on prompt.
Exit Condition	The LED lights up, representing the opening of the lock; or the LED lights remain off, representing denied access to the hall.
Main Scenario	1. The system will load a new window where the user will be required to input the id number of the resident seeking entry. 2. User will click a button to send the ID number to the database server to query if the person is a resident. 3. The server will return the residency status of the entrant, that is, whether he/she is a valid resident; and information of the resident, which includes their name and stored picture. 4. The security will match the picture stored and the face of the entrant and verify the entrant is the resident. 5. The system provides a 5 second delay of the automatic opening of the lock. If the identities do not match, the security guard will override the system within this time, by clicking a button to cancel the automatic opening of the gate. 6. If the entrant is a fully verified resident the desktop application will send an open signal to the raspberry pi. 7. The pi will then send an electronic signal to the LED causing it to light up which represents the opening of the lock. 8. If the entrant is not a resident the security will either log the person as a visitor or deny access to the hall.

DETAILED USE CASE FOR MOBILE APP OPEN LOCK

Use Case Name	Open Gate via Mobile App
Description	This use case depicts how a resident request for the gate to be opened with the use of the mobile app.
Participating Actors	Resident
Trigger/Action	Resident presses the open button on the interface of the mobile app.
Pre-Condition	1. The mobile phone has internet connection. 2. The mobile application is opened. 3. Resident correctly logged in to system with university stored ID number and password.
Exit Condition	The LED lights up, representing the opening of the lock; or the LED lights remain off, representing denied access to the hall.
Main Scenario	1. The resident will be brought to a screen where they are allowed to press a button to activate the open signal for the lock system. 2. User will click the button and their ID number will be sent via internet connection, to the desktop application which will send the information to the database server to verify their residency. 3. The server will send the residency status of the entrant as well as their identification information (name and image) to the desktop application. 4. The security will match the picture stored and the face of the entrant and verify the entrant is the resident. 5. The system provides a 5 second delay of the automatic opening of the lock. If the identities do not match, the security guard will override the system within this time, by clicking a button to cancel the automatic opening of the gate. 6. If the entrant is a fully verified resident the desktop application will send an open signal to the raspberry pi. 7. The pi will then send an electronic signal to the LED causing it to light up which represents the opening of the lock. 8. If the entrant is not a resident the security will either log the person as a visitor or deny access to the hall.

ACTIVITY DIAGRAM

ACTIVITY DIAGRAM FOR A RESIDENT

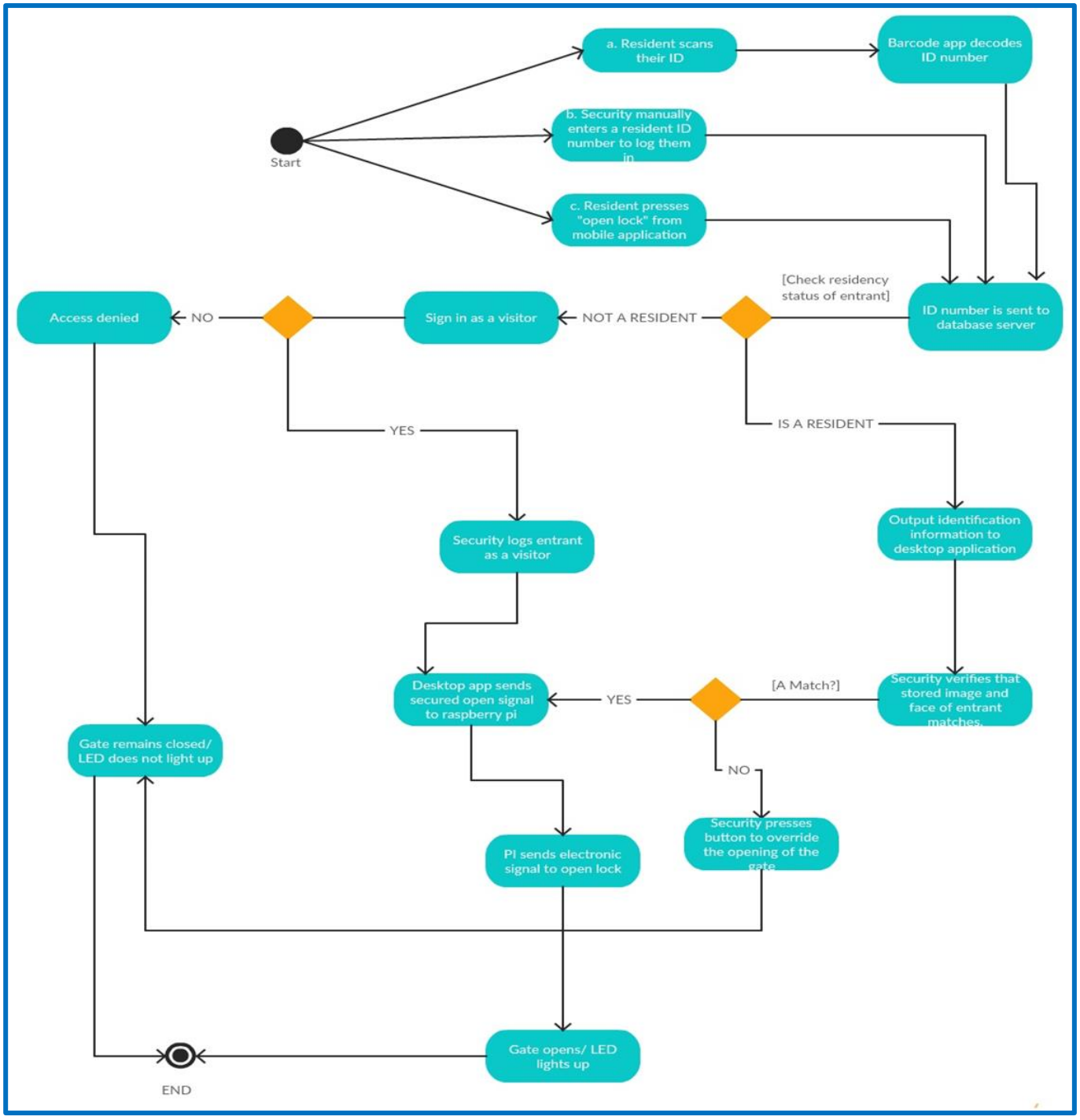

Figure 2.1: Activity Diagram for a Resident.

ACTIVITY DIAGRAM FOR A VISITOR

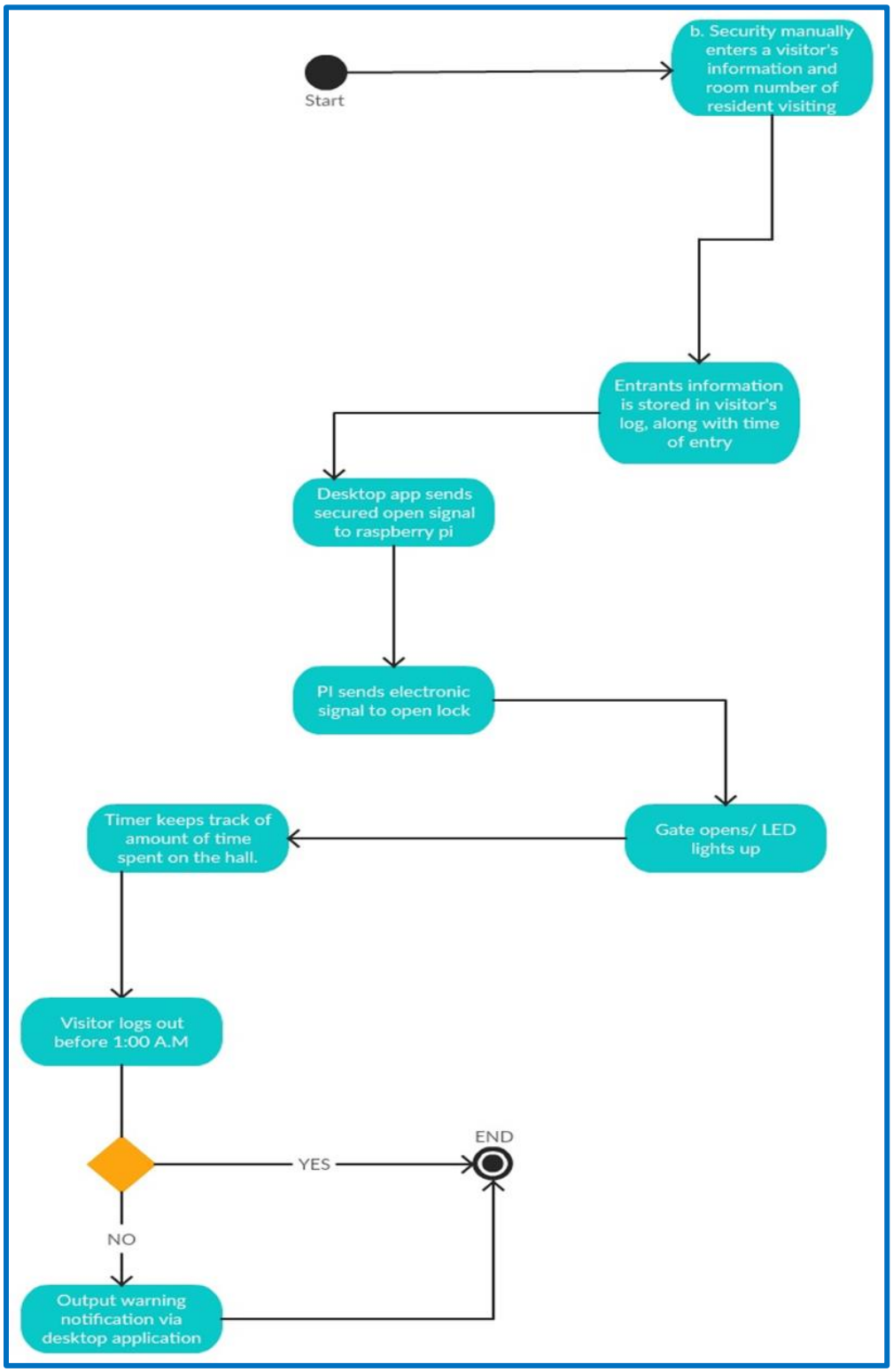

Figure 2.2: Activity Diagram for a Visitor.

ANALYSIS CLASS DIAGRAM

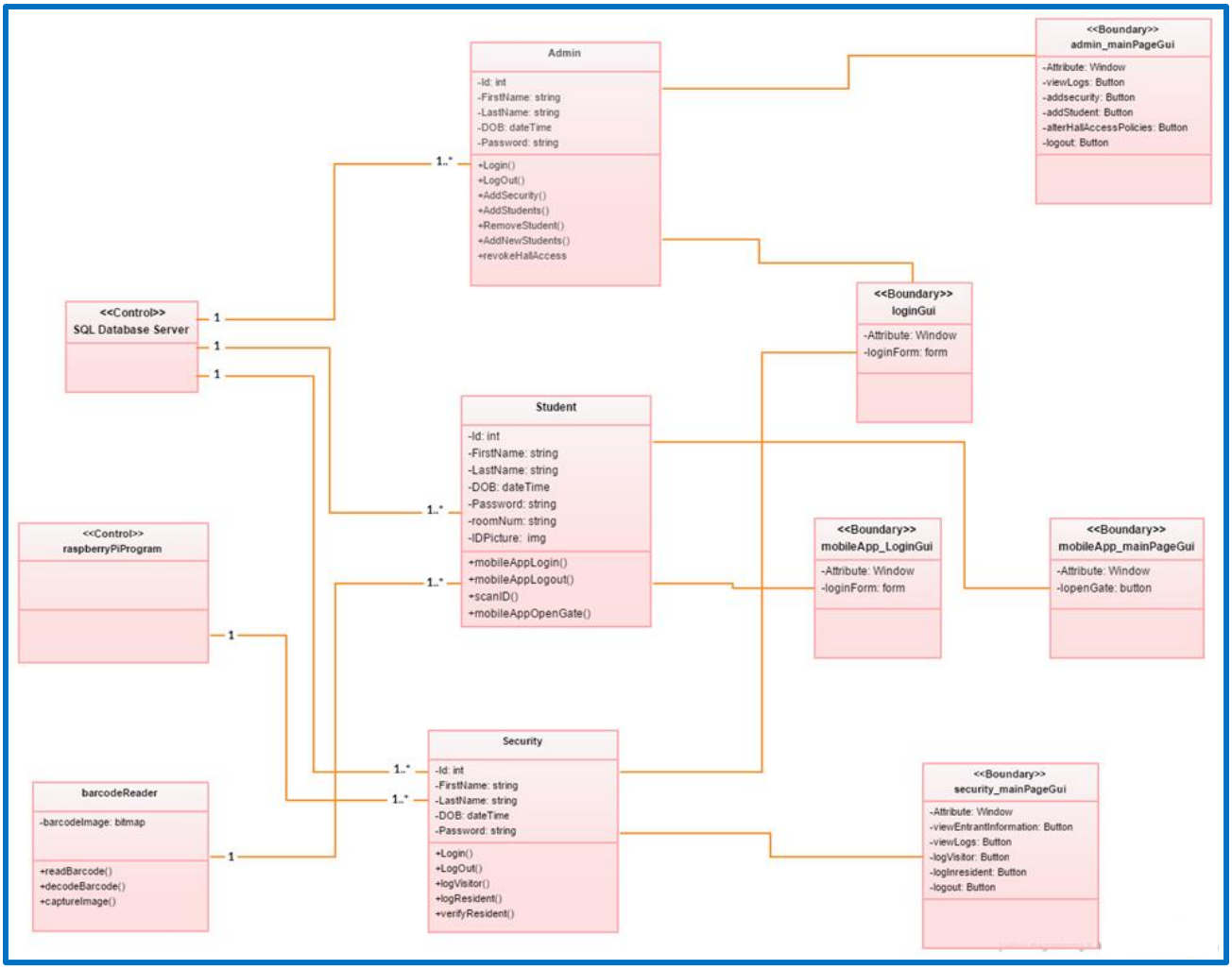

Figure 2.3: Class Diagram.

SEQUENCE DIAGRAM

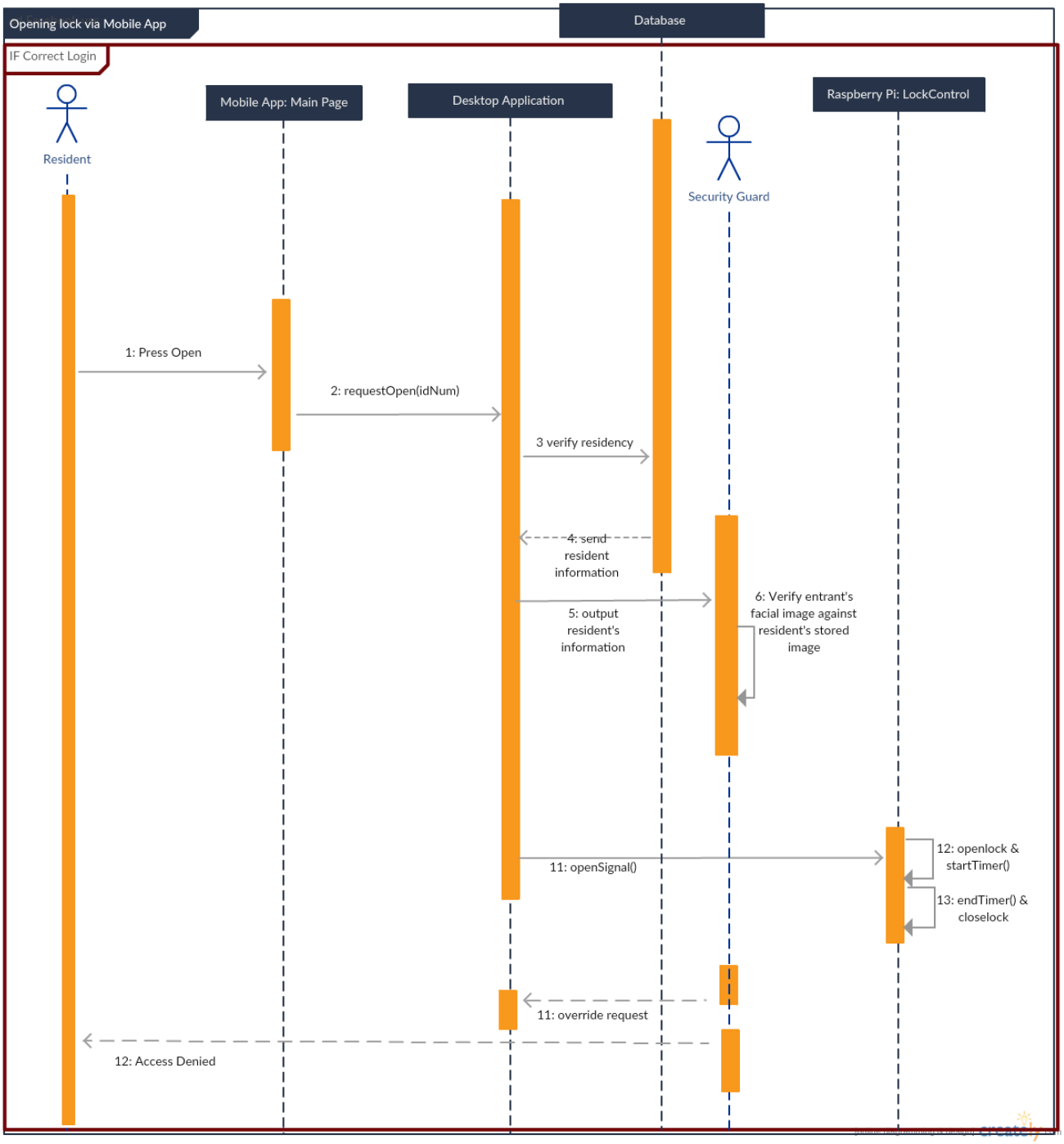

Figure 2.4: Sequence Diagram showing the opening of lock via App. The diagram assumes login authentication has already being verified.

DESIGN MODEL

ARCHITECTURE DIAGRAM

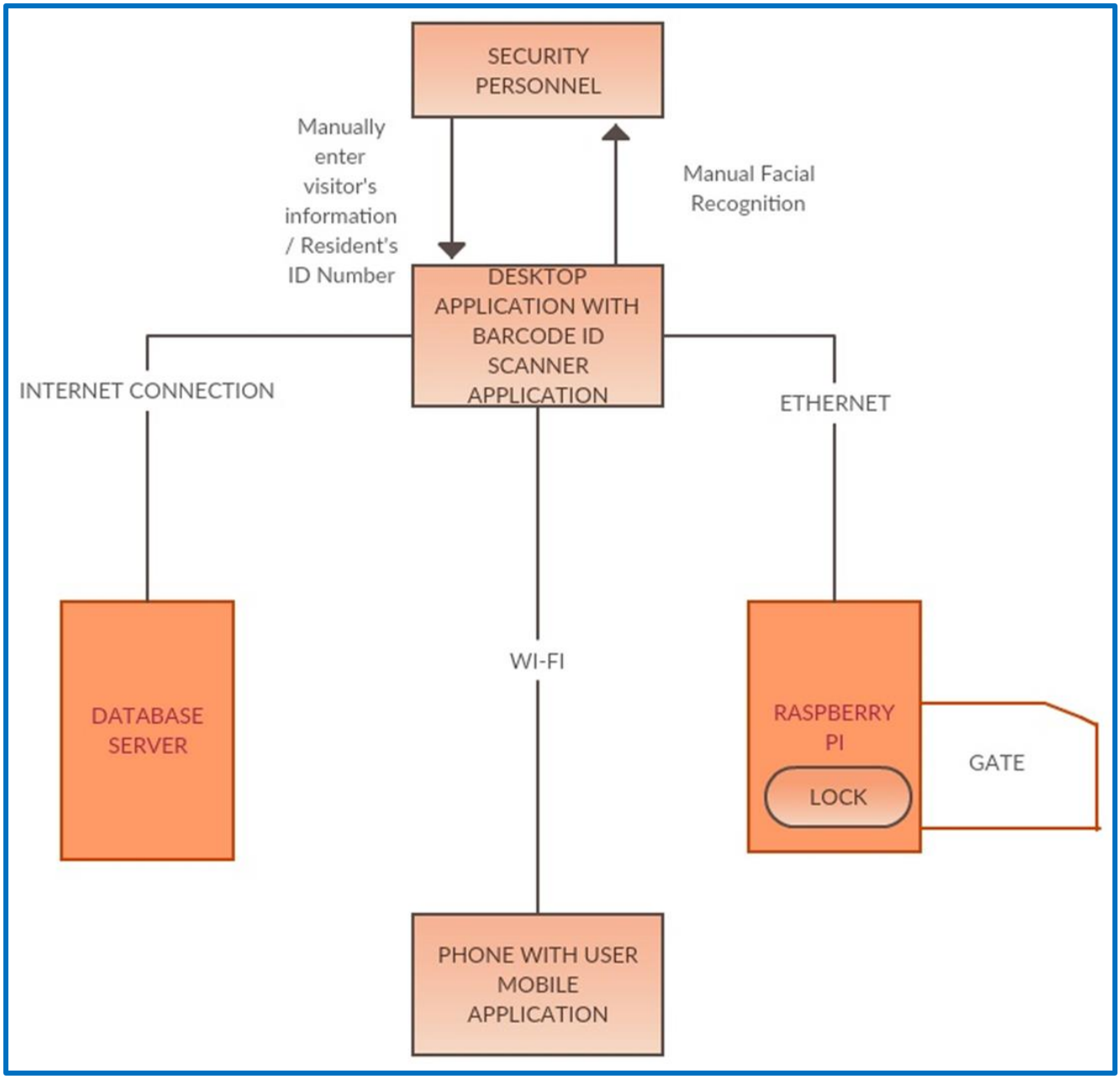

Figure 3.0: System Setup.

PROJECT SETUP

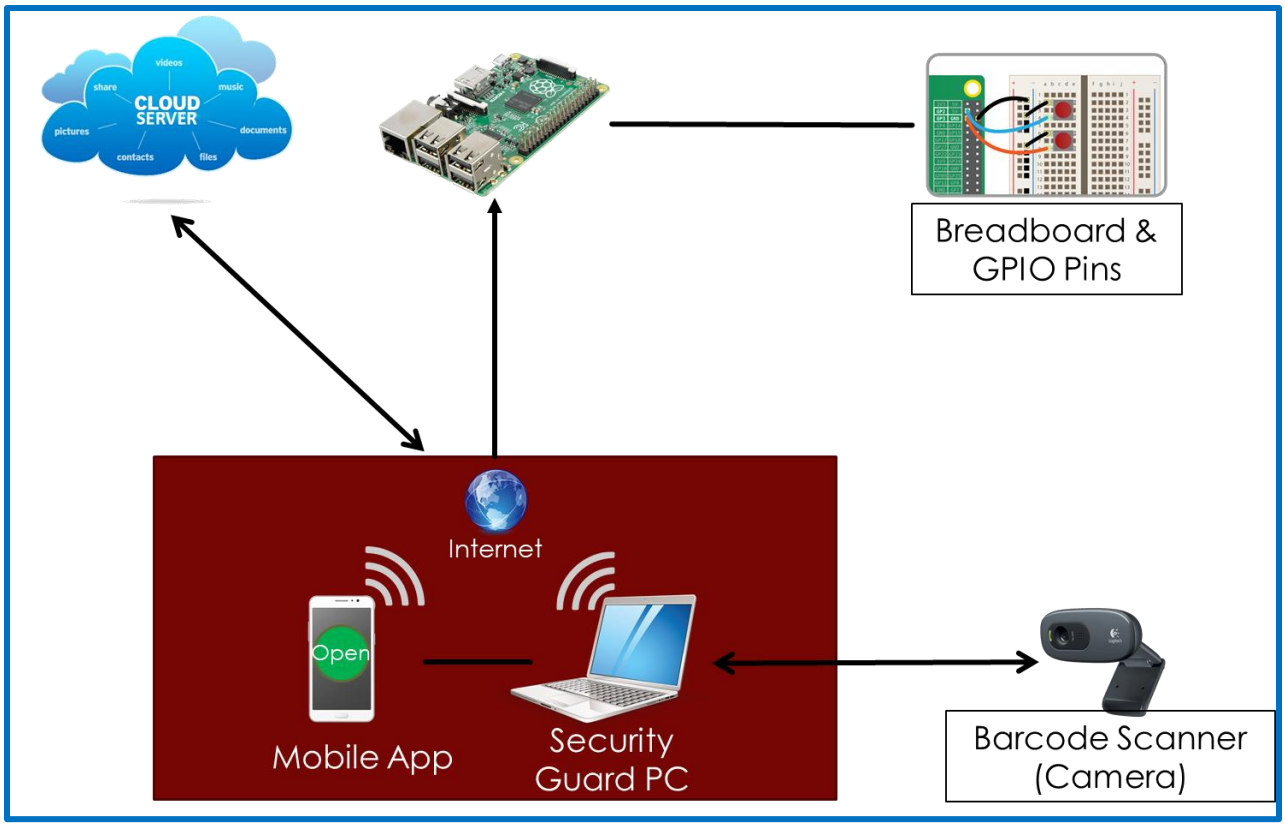

Figure 3.1: Hardware Setup of Project.

METHOD/APPROACH

DESIGN DECISIONS AND TRADE-OFFS

Given the multiple components to the overall system, there were multiple hardware and network connection choices that arose in designing the automated lock system. These design decisions are summarized as follows:

Choice: Micro-Controller

Options: Raspberry Pi
Arduino
Banana Pi

Selection: Raspberry Pi

Reasons: Initially, the development team was only knowledgeable about the Raspberry Pi and as such made an initial decision to use this device. However, after research done, it was discovered that the Arduino and the Banana Pi wouldn't have been the best choice. The Raspberry Pi was chosen as it is a more powerful device with its processing speed being greater than that of the Arduino or Banana Pi. In addition, the Raspberry Pi facilitates the use of multiple programming languages including C# and Python which were the chosen languages for the computer system; whilst the Arduino only supports Arduino, C/C++.

Choice: Storage location of the database

Options: Store database on the same PC that hosts desktop application
Store database in the cloud
Store on Raspberry Pi

Selection: Store database in the cloud

Reasons: Storing the database on the raspberry pi was factored as a risk considering that the pi is a small and fragile device. If the pi was destroyed that would also mean destruction of the database. Such an approach would make the database too dependent on the Pi itself, thus affecting the modularity of the system.

Storing the database in the cloud makes it more easily accessible to all PC's that utilize the automated lock system, without requiring the database to be stored locally on every desktop within the system. In choosing this option it was noted that ensuring that database backup recovery is done locally if internet access is unavailable the system is still functional.

Choice: How to configure communication of the mobile application with the other components of the system.

Options: Mobile app → Pi → Desktop application → Database server → Desktop application → Pi → Lock.
Mobile app → Desktop application →Database server → Desktop application → Pi → Lock

Selection: Mobile app → Desktop application →Database server → Desktop application → Pi → Lock

Reasons: Initially, the developers wanted to have the mobile application operate independently of the desktop application. However, after thorough brainstorming, we recognized that the communication between the two was necessary for the verification of the identity of the entrant as the true resident, by the security guard. The first option was too repetitious and an unnecessarily cumbersome communication process, as well as a security risk.

RELEVANT TOOLS AND TECHNOLOGIES

In the creation of the Automated Lock System, several existing tools and technologies were utilized to ensure optimum performance and efficiency of the created system. These include, AES algorithm, Ozeki SDK, and the Dynamsoft Barcode Reader Library.

Within the system, the desktop application communicates with the database server to verify the residency of an entrant, and communicates with the raspberry pi to initiate the opening of the lock. The desktop communicates an "on" message if the gate should be opened and an "off" if it should remain closed. These messages are encrypted and decrypted by the desktop application and the raspberry pi respectively using the AES ALGORITHM. This was done to ensure the safe transmission of the messages from one component to the other.

The DYNAMSOFT BARCODE READER LIBRARY was used to make the reading and decoding of the barcode more accurate and efficient. Integrated with the .Net framework, the library enabled us

to use the webcam as a barcode scanner, and made it possible for the system to decode the barcode from a bitmap file.

Initially, our system required the security guard and/or administrator to press a button to allow the web camera to capture the barcode image for it to be decoded. However, after consultation with our advisor we decided that an automatic capture of the barcode would be a more efficient process. Upon research, we found the OZEKI SDK which is a software development kit that enables motion detection. The SDK facilitated the system to automatically capture the barcode by detecting when a card has been placed in front the web camera.

RESULTS

During the development of the system, we conducted a vast amount of test on the system. Please see the Appendix for screenshots as it regards to the system design. Below are some of the test cases carried out.

Test Case(s)	Steps	Expected Results	Status
Test Case:	1: Login for Both Desktop and Mobile App		
1	Verify ID number and password.		
1.1	Blank ID, Blank Password	Return Error	Pass
1.2	Blank ID, Password	Return Error	Pass
1.3	Invalid ID, Blank Password	Return Error	Pass
1.4	Correct ID, Blank Password	Return Error	Pass
1.5	Correct ID, Invalid Password	Return Error	Pass
1.6	Correct ID, Correct Password	Return Home	Pass
1.7	Correct ID, Correct Password, Wrong Interface	Return Error	Pass

Table 1.0: Test Case 1 – Login test cases for the desktop and mobile application.

In this test case, we tested the various ways a user would log into the system. All our cases passed as this was merely enabling a login option for security purposes.

Test Case(s)	Steps	Expected Results	Status
Test Case:	2: Scan Barcode to open lock		
2	Verify barcode		
2.1	Government ID, Correct User	Open Lock	Not Implemented
2.2	Valid ID, Wrong User	Keep Lock Closed (Manually)	Pass
2.3	Valid ID, Wrong User	Keep Lock Closed (Auto)	Not Implemented
2.4	Valid ID, Correct User	Open Lock	Pass
2.5	Invalid ID, Wrong User	Keep Lock Closed	Pass
2.6	Invalid ID, Correct User	Open Lock (Manually)	Pass
2.7	Invalid ID, Correct User	Open Lock (Auto)	Not Implemented
2.8	Random Barcode (Non-ID)	Keep Lock Closed	Blocked
2.9	Staff ID	Open Lock	Pass

Table 1.1: Test Case 2 – Barcode test cases for opening the lock.

Not all tests could be passed as it was out of our scope to implement additional features. For the test cases that were not implemented, these are the cases that would consider the facial recognition feature. As mentioned previously, this is a feature for the future implementation of the system.

As for test case 2.1, this was not a major feature to be added but will be in the future implementation. Test case 2.8 on the other hand, this entered a block state because there was never an exception handler in place to handle all the barcode that are not ID numbers, i.e. out of range. This was tested with an "Iron Supplement" bottle which crashed the program after trying to read its barcode. Test case 2.9 demonstrated that staff IDs have a slight difference to the ID assigned to the user and the ID translated from the barcode.

Test Case(s)	Steps	Expected Results	Status
Test Case:	3: Open lock with Mobile App		
3	Verify user and device		
3.1	Invalid Device, Correct User	Open Lock (Manually)	Pass
3.2	Invalid Device, Correct User	Open Lock (Auto)	Not Implemented
3.3	Invalid Device, Wrong User	Keep Lock Closed	Pass
3.4	Valid Device, Correct User	Open Lock	Pass
3.5	Valid Device, Wrong User	Keep Lock Closed (Manually)	Pass
3.6	Valid Device, Wrong User	Keep Lock Closed (Auto)	Not Implemented
3.7	In Range	Open Lock	Not Implemented
3.8	Out of Range	Keep Lock Closed	Not Implemented

Table 1.2: Test Case 3 – Mobile App test cases for opening the lock.

A user opening the lock via the mobile App is expected to have correct credentials to send the Open signal. As, the app stays now, an individual using the App can be at the Faculty of Science and Technology but still able to open the lock at Chancellor Hall. This should not be the case. The team was concerned with getting the various methods in place before securing the feature. All other methods can be deployed, including the mobile App. We would only need to implement GPS tracking to identify the proximity of the user from the gate.

The Mobile App method for opening a lock is one which is not very common and as such will need to be researched and discussed more to determine the best measures of implementing it properly. The App and the Desktop do have a secure communication with the Raspberry Pi. They use hashing and the AES Algorithm to ensure there is secure communication between the devices. So, where not all areas of the system are perfect, it still has some level of security.

Test Case(s)	Steps	Expected Results	Status
Test Case:	4: Open lock with Desktop Manually		
4	Verify user ID Number		
4.1	Invalid ID number, Correct User	Open Lock	Pass
4.2	Invalid ID number, Wrong User	Keep Lock Closed	Pass
4.3	Valid ID number, Correct User	Open Lock	Pass
4.4	Valid ID number, Wrong User	Keep Lock Closed	Pass

Table 1.3: Test Case 4 – Manual Desktop Operations test cases for opening the lock.

As mentioned, the desktop interface is used by the security guard. This interface is one in which the security manually enters the information and manually opens the lock on the system by sending the digital signal.

The test cases shown are all the core functions of the system. In addition to the features such as adding a student, ban a student, add visitor, etc. Those features are all great and they helped us in meeting our objectives. The main objective of the system was to open the lock. This was accomplished by substitution of LED lights. The second objective was to automate the system in such a way that it will keep records of entrants, which was never done previously. Hence, the development of the system we have now.

It was deducted from the design of the system that the features implemented and the features we would've liked to have implemented, could be used on a larger scale and not just a simple opening of a lock. This system has the potential to grow and can be a security system which companies can use and not just the university. This system tested our knowledge of Computer Science and highlighted key features like communicating between hardware devices using knowledge may have only known in theory. Other features like keeping a log of residents and visitors alike can be used in emergency protocols.

The system comprised of several hardware which had to no relation/communication to each other and at the end of the project we had all hardware working together seamlessly. In addition to that, the communication was secure and the user had little to no issues. As well as all our functional and non-functional requirements were met.

CONCLUSIONS

PROJECT COMPLETION

The final project reflected approximately 99 per cent of the initial functions set forth and the development team was also able to make unplanned additions to the system to enhance its efficiency. Initial functions that were completed include:

- The creation of desktop application to facilitate user interface for administration and security guards.
- Creation of the barcode scanner application.
- Fully programmed Pi to control the opening of the lock.
- Configured LED lights lock representation.
- Implementation of log keeping and security features for the entire system.

In addition, the team was also able to implement motion detection to the system, to detect when an identification card is placed in front of the camera and then automatically captures the image of the barcode. Initially, we had just intended to allow the security guard to click a button via the desktop application to capture the image, however we saw the motion detection feature as more efficient and more facilitating to the automatic process.

LIMITATIONS

In creating such a hardware encompassed application, being software experts we were bound to face a few limitations. One such limitation was configuring the system to open a physical lock. The heavy hardware structure of such a feature was deemed as too time consuming and given the time limitations of the project we decided to make that aspect out of the scope of this project.

Upon implementing the mobile application feature, the developers were aware that flaws existed with respect to the distances from which a resident could request the opening of the lock. As it stands, once a resident is properly logged in to the mobile application and has internet access on the device, irrespective of how far they are from the gates of the hall, they can successfully cause it open. The developers recognized that implementing GPS technology to identify where the

entrant is in respect to where the gate is would allow for distance constraints to be placed on the lock. However, time limitations of the project were once more against the implementation of such a feature.

LESSONS LEARNT

Venturing into this project taught the developers some key lessons necessary for application to most other team development projects that we will encounter in our career journeys. At the forefront, we learnt that clarifying the how all components of the system will interact to ensure efficiency is best to be done from the brainstorming stages of the project development instead of during implementation. We understand that as implementation occurs, some changes in initial plans may also occur, however it is best to ensure that thorough research is done before implementation begins to save time during the development process and to limit mass changes.

We most importantly learnt how to think critically and act assiduously in solving a complete project from both hardware and software perspective, while conceptualizing security requirements and user needs.

APPENDIX

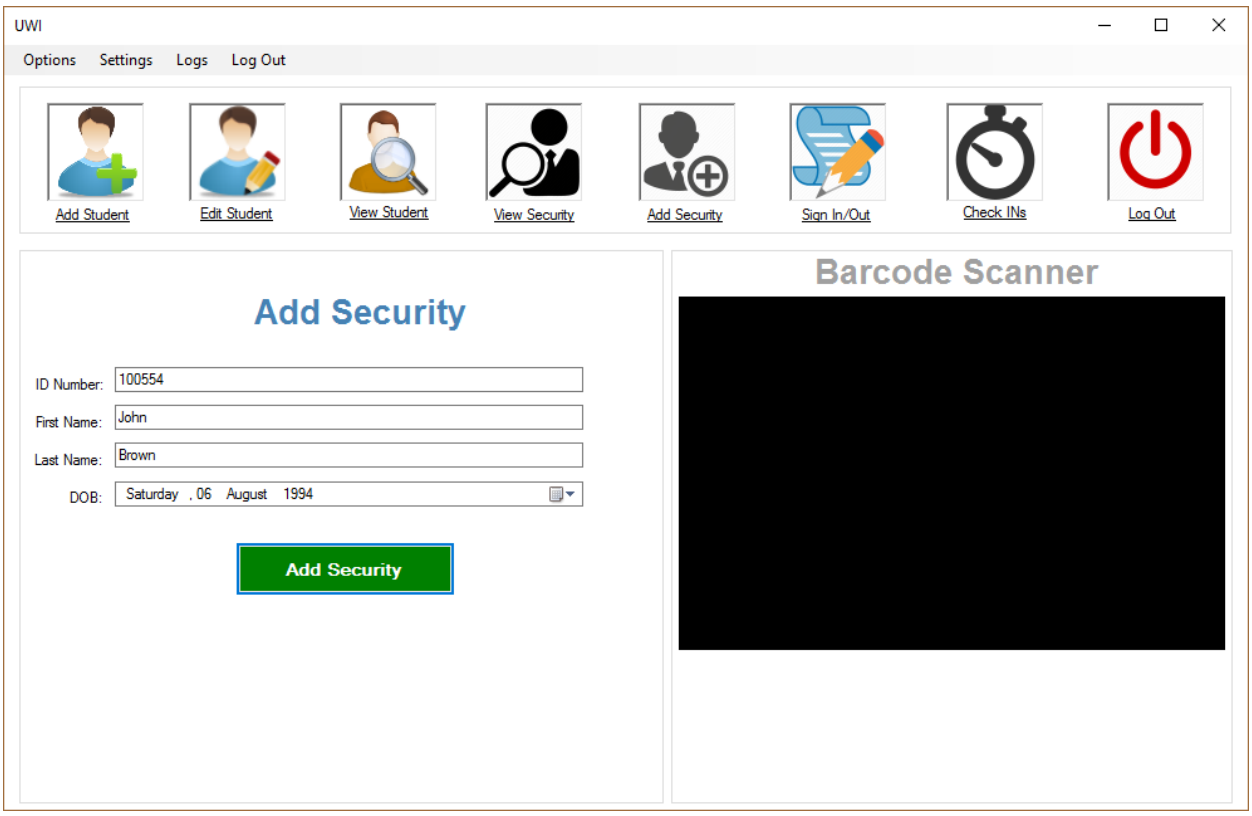

Fig. A.1: Add Security

Fig. A.2: Sign Out Guests

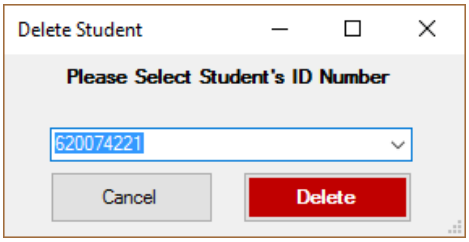

Fig. A.3: Delete a User

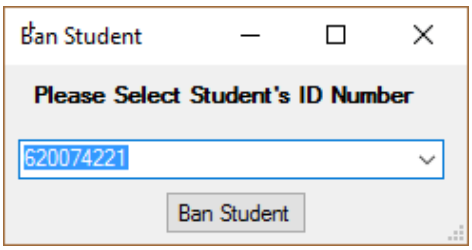

Fig. A.4: Ban Student

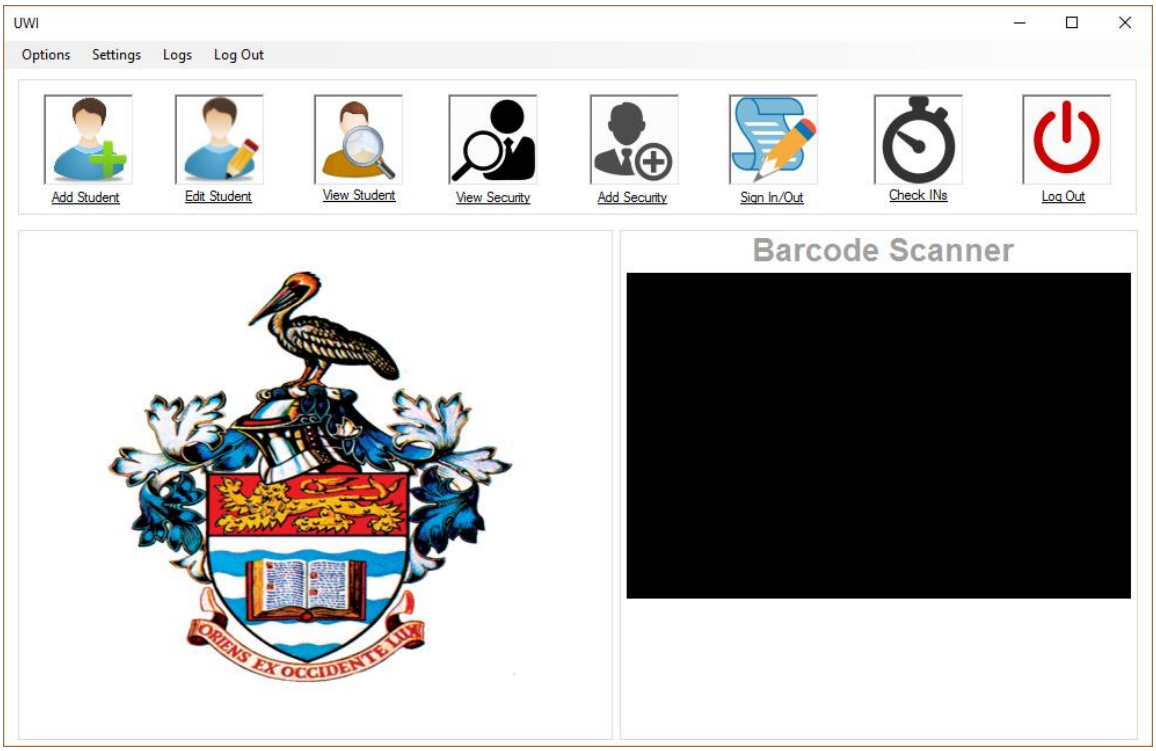

Fig. A.5: Admin Dashboard

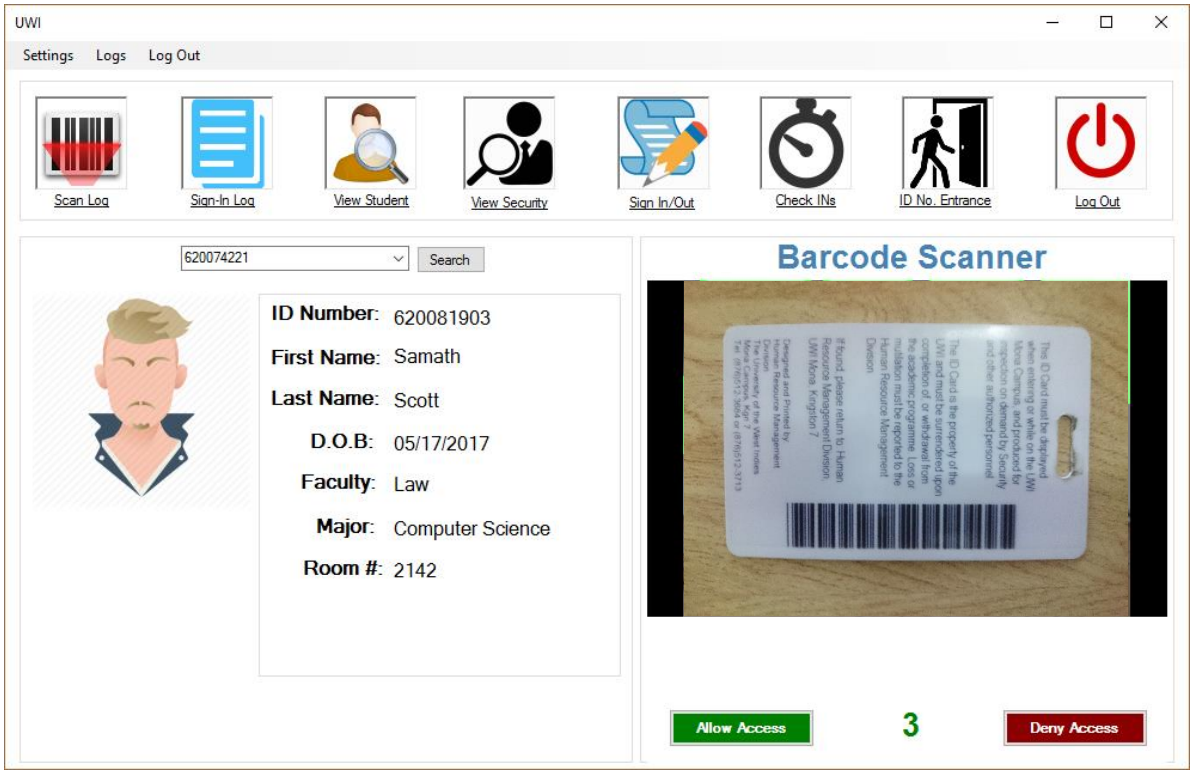

Fig. A.6: ID Scanned

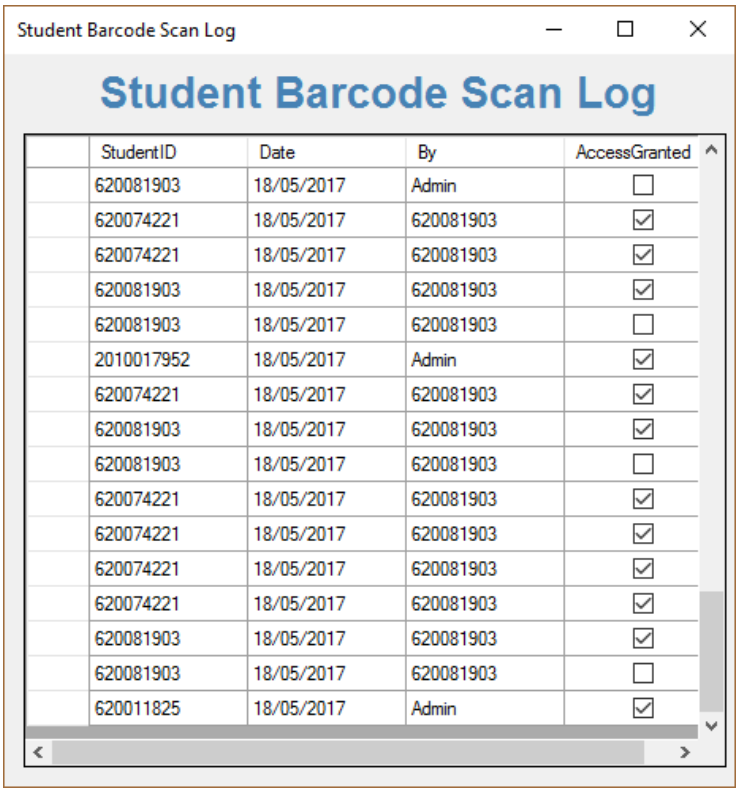

Fig. A.8: Student Logs

Fig. A.7: Sign in Guest

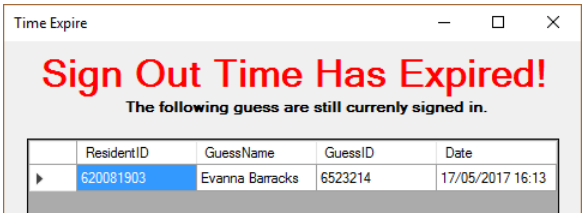

Fig. A.9: Guest time expired

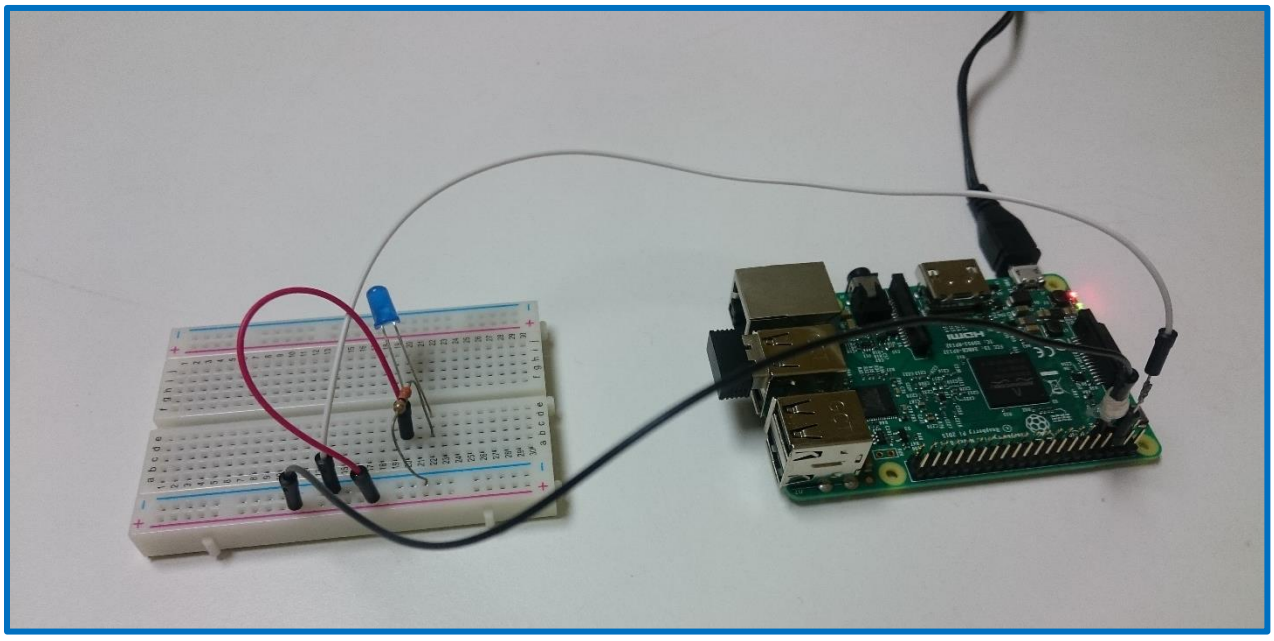

Fig. B.1: Raspberry Pi and Breadboard Connection.

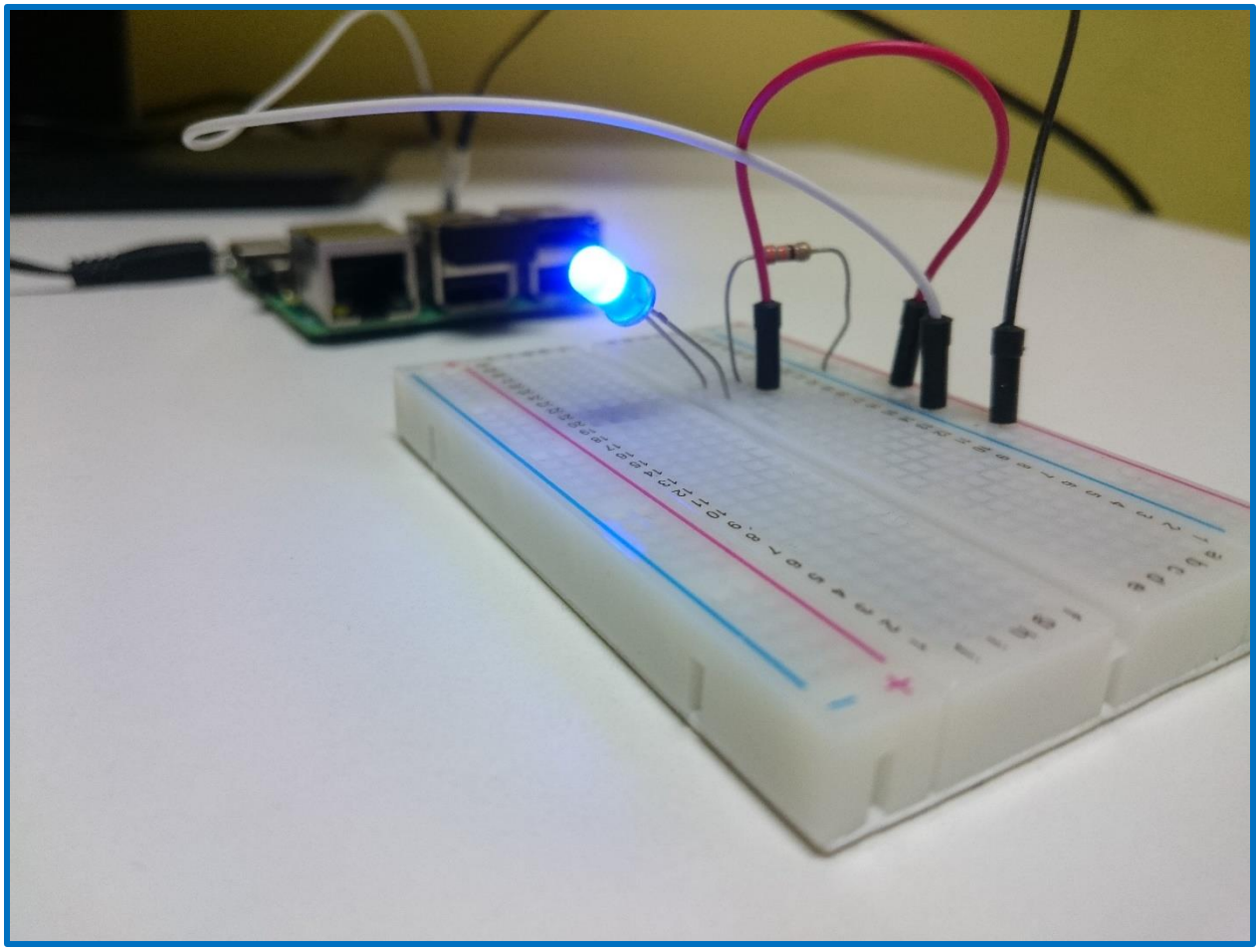

Fig. B.2: Raspberry Pi and Breadboard Connection with LED light 'On' showing that the Lock is Open.