Chapter I

**INTRODUCTION**

**Project Context**

This chapter presents to introduce the principles behind Automated Billing to Centralized Dormitories targeted for dormitories as well as its intentions.

The chapter includes the Background of the Project, Theoretical Framework and Statement of the Problem and Objectives of the study.

**Purpose and Description of the Project**

The Automated Billing to Centralized Dormitories (ABCD)is to mainly assist the management upon billing its tenants.

There are no systems in place to cater the needs of these businesses. Basically, the methods involved in dormitories are similar to hotels. It involves registration of tenants in a unit for a specific length of time, and billing them according to what has been discussed. This process is not difficult but is repetitive. At times, inconsistencies may present conflict towards the renter and the management which could have easily been avoided. It is the main reason why a system must be put in place to handle the processes involved.

ABCD’s sole purpose is to replace the repetitive manual system ongoing in the dormitories. It will present itself similar to a hotel billing system but in lighter means. It will handle profiling of each tenant, room monitoring and invoicing to their accounts. All computations done in the system with the input from an authorized personnel. Each tenant will be able to view their accounts from the system as well.

The system is web based making it accessible from anywhere with internet accessibility. This also enables the system to cater to multiple dormitories as long as they involve the same processes in the company the system is designed for.

In summation, system will make the work of managers or accountants easier in terms of dormitory room management and invoicing to individual tenants easier, more reliable, and consistent.

**Objectives of the Study**

The objectives of the study with the use of Automated Billing to Centralized Dormitories (ABCD) for dormitories are the following:

1. To enumerate the processes involved in the existing manual system of dormitories.
2. To enumerate the problems encountered in the existing inventory system of dormitories.
3. To emerge design features of the Automated Billing to Centralized Dormitories that will ensure maximum efficiency to dormitories.
4. To enumerate the benefits that the Automated Billing to Centralized Dormitories will for any dormitories.
5. To identify the security measures to be imposed once Automated Billing to Centralized Dormitories is implemented.

**Scope and Limitation of the project**

The system can maintain a fluid computing involving full arithmetic of all accounts of the tenants. Doing this with security in terms of keeping away any chance of the data from being tampered and preventing any chance of data loss due to misplacement of records or even stealing. Manager or the user no longer needs to compute any data to fulfill the payments. All possible computations including a room availability handler and a complaint system will be by the system automatically.

The Automated Billing to Centralized Dormitories (ABCD) has the potential to be a sole system for the billing of multiple businesses. Being web based, it can handle multiple users while maintaining crucial calculations for all. It has the potential to compete with global standards because of its simplicity and flexibility. It can handle different rates anytime and show a summary of all the times the rates changed even mid-semester. Increase are effective after each semester.

a dormitory or an apartment complex having multiple tenants. The system is limited to the automation of the computations alone involving the billing of tenants. With this, the system’s shortcomings are discussed hereafter.

1. The proposed system is dependent on the partial involvement of the user, or the manager. This means that for the system to function, inputs and actions by the manager is still needed. Functions such as registration of each tenant and changing of their rooms. Upon registration, the manager must manually take down the basic details of the tenant and input them to the system. There is no social media integration on ABCD for creating accounts which is termed as one click registration. Upon changing of rooms, the manager must be the one to give special discounts since this cannot be integrated to the system as each discount varies depending on the agreements made by him and the tenant.
2. The physical collection of payments also resides in the manager still, and not the system. ABCD does not handle gathering of payments because it is not capable to do so. Because of it being web based and mobile,

integrating such functionality in the Philippine setting is highly unlikely and very difficult for the system. Also, the checking of electric wattage consumption resides with the manager since readings must be taken manually.

Finally, the major limiting factor to ABCD would be the lack of internet access. Without it, there would be no access to the system.

**Significance of the project**

This study intends to provide an accurate and effective Billing system which will benefit the company and its tenants. The proposed system will lessen the time, enhance the level

of work, and help its processes to be more reliable and efficient, and providing accuracy and security of data and information.

To the Company. The development of better data manipulation, retrieval, and data storage to Web based system can strengthen the company’s transactions and operations which may eventually lead to business growth and success. It will enable the management to retrieve and render correct data on their records since computerization of files and documents saves time and space.

To the Manager. This will lessen his work in means of accounting. This alleviates his task on calculating accounts of every tenant more than once. The proposed system will lessen his work. It will also reduce or eliminate the percentage of human error in tallying the Bill. Monitoring and updating of Billing records can also be easily accommodated.

This however does not remove him from his work but will enable him to focus on management of the business for its growth and success.

To the Tenants. Assurance that the records and agreements remains unchanged as well as accurate billing calculation which they can review will be to their advantage. This will eliminate bias thinking towards the company leading to a more comfortable living in the company premise.

To the Researchers. This system will be a statement to the capability of the researchers as well as the students of Baguio College of Technology upon accomplishing a fully functional online billing system that meets global standards.

To the Future Researchers. This study could be used as reference to constructing and implementing a web-based system. Results gathered will be beneficial for their research leading to more advanced and practical projects.

Chapter II

**REVIEW OF RELATED LITERATURE**

**Theoretical Background**

**--**

**Related Literature**

A reliable automated billing system is essential for the benefit of the company and the tenants by ensuring a system that will rid of human errors in computations and record keeping.

For this reason, the researchers developed and designed   
the online billing system for the company which will handle

all computations and store the records as soon as it is inputted in the system. By using technology to advantage, the study focused on the reliance on the internet which is prevalent at present.

The use of input-process-output model is implemented in this study. Input being in control of the administrator or the manager – listing tenant details and agreements in the system. Process, handled by the system without the need for manual assistance after the input. And output being the display for both the tenant and the management to monitor – the basis for rent collection and room availability management.

**Related Studies**

This research will serve as a basis for revolutionizing the fundamental anatomy of the company, and serve as indicators in accumulating the needs of the tenants more than what has been used to. Results of this study will be analyzed and scrutinized cautiously to achieve best output.

As cited in the article entitled “Computerized Billing for Time Improves Client Communication and Firm Profitability” by Murphy (1996), he stated that, accounting firms can improve their profitability and relation with clients by automating client billing with computers. Accounting firms can use personal computer as an economical means of computerizing the billing process. A Computerized Billing System allows accounting firms to more effectively manage their employee’s productivity and give them the means to analyze the hours available and hours billed of each employee. A Computerized Billing System provides more control over a accounts receivable and client work processes.

Raetta Leiker, billing administrator for the Parker Water and Sanitation District in Parker, CO, is thrilled that the utility signed up for the BillMaster utility billing software distributed by Data West in 2002. It has since made life far easier for the employees in the utility’s billing department, the utility’s billing software automatically sends statements, either through e-mail or regular mail, to its customers. This combination saves an incredible amount of time and labor for the utility. It also provides better service to the utility’s paying customers.

The proposed system will focus on related literature available to attain similar positive results.

Chapter III

**TECHNICAL BACKGROUND**

Presented in this chapter is the overview of adopted technologies that will be incorporated in the development of the proposed system and the mechanism of the system.

**Technicality of the Project**

The proposed system is web based. The functionality is categorized into two different layers. The front-end and the back-end.

The front-end will handle the visual presentation of the system. It is the graphical user interface (GUI. It is how the user will interact with the functionality of the system. It will be written in HTML with CSS and can be accessed in any updated browser.

The back-end will handle all data validation, computation, and storage of data processed. Data inputs from the front-end will be passed on to the back-end layer for data handling. This layer will authorize and limit accessibility depending on the user, compute each tenant’s bills, and store all information passed to it. It will be written in PHP and JavaScript for functionality and in MySQL for databasing. All the data will be encrypted through md5 by this layer for security purposes.

The two layers will comprise the web based system for full, secured functionality by combining the mentioned programming languages.

**Details of the Technologies to be Used**

The following technologies will be used in developing the proposed system.

Bootstrap. An open source HTML, CSS, and JavaScript framework for developing responsive, mobile first projects on the web. This enables fast web development with responsive capability for full compatibility to pc and mobile devices.

(CSS) Cascading Style Sheet. A mechanism or technique for adding style such as fonts, colors, spacing, and frames to web documents. It is usually if not always, used with HTML. It is used to describe the appearance of a webpage.

(HTML) HyperText Markup Language. The most basic building block of the web. HyperText refers to links that connect webpages to one another, either within a single website or between websites. It uses markup to annotate text, images, and other content for display in a Web browser. HTML markup includes special elements such as <head>, <title>, <body>, <article>, <section>, <p>, <div>, <img> and many others. This to specify each part of the web page.

JavaScript. A programming language. Originally developed to add dynamic and interactive elements to websites. It is a client-side scripting language, which means the [source code](https://techterms.com/definition/sourcecode) is processed by the client's [web browser](https://techterms.com/definition/web_browser) rather than on the [web server](https://techterms.com/definition/web_server). This means JavaScript [functions](https://techterms.com/definition/function) can run after a webpage has loaded without communicating with the server. The JavaScript code can produce an error message before any information is transmitted to the server.

(Ajax) Asynchronous JavaScript.A [client-side JavaScript](http://www.seguetech.com/blog/2013/02/07/what-are-the-pros-and-cons-of-client-side-scripting) that communicates to and from a server/database without the need for a [post back](http://www.c-sharpcorner.com/uploadfile/2f73dd/what-is-postback-in-Asp-Net/) or a complete page refresh.

JQuery. A concise and fast JavaScript library that can be used to simplify event handling, HTML document traversing, Ajax interactions and animation for speedy website development. jQuery simplifies the HTML's client-side scripting, thus simplifying Web 2.0 applications development.

(MySQL) My Structured Query Language. A powerful open source database server built based on a relational database management system (RDBMS) and can handle a large concurrent database connection. It is used for adding, removing, and modifying information in the database. MySQL can be used for a variety of applications, but is most commonly found on Web servers. A website that uses MySQL may include Web pages that access information from a database. These pages are often referred to as dynamic, meaning the content of each page is generated from a database as the page loads. Websites that use dynamic Web pages are often referred to as database-driven websites.

(PHP) Hypertext Preprocessor. A widely used open source general purpose scripting language that is especially suited for web development and can be embedded into HTML. It is a server side script which means all data and requests are processed in the server. The client cannot see these processes and can only see the results. PHP is best suited for controlling MySQL databases easily without compromising security.

**How the Project Will Work.**

To access the proposed system, it must first be uploaded to a web hosting server. Then, a corresponding domain name must be registered depending on the availability of the name and preference of WynHope. The database structure, populated only with the account of the manager, must then be uploaded to the web server. When the manager will access the website, he can immediately login with his credentials and the system will validate it.

For the manager. When he first logs in, he will see a dashboard summarizing the number of floors in the building, the number of rooms, number of tenant, and total incoming collection of rent fees. The first thing he must do is to register how many floors are to be included in the system, and the units in those floors. Floor names are numeric and unit numbers are string unique only in each floor. This offers flexibility to the naming scheme of the company.

If a customer then wishes to rent a room, the manager must navigate to add rent page and fill up a form to register his full name, email address, contact number, and the password the customer wishes. The email and the password will serve as his credentials for his login. The manager will select a floor, and the system will determine which unit rooms are available in that floor this is handled by JavaScript and Ajax in the back-end layer. The rent is not predefined. This is because instances such as discounts, special agreements, and bargains are to be made between the customer and the manager. In this way, the program is not limited to the agreements. He just needs to input the rent per month. Before completion of registration, the customer must agree an agreement form in the program, if he doesn’t agree, all information will be void. Upon submission, a JavaScript will check for empty fields, if one is found, the data will not be saved and the user will be notified of the empty field. If all fields are filled, then a PHP script will process the data and pass it on to the MySQL database for saving.

Upon billing, the manager needs to navigate to the add fair page. A form will be provided, and he just needs to select a floor and occupied rooms will be shown where he would select. Through similar process of adding a renter, the system will show who is renting that unit and show the rent per month specified, this time, cannot be changed. An option to add an additional bill, electricity bill, water bill, is also present for the manager to edit. Including the date issued, he can post it and it will be updated in the database. The manager will be redirected to a summary page showing the bills issued with options to toggle if they have already paid. The available balance will also be shown.

For the tenant. To view his account and list of issued bills, he must simply login to the same website with his email and password set in the registration. Upon successful login, he will see a dashboard different from what the manager can see. In his own page, he can see a summary of his balance, rent per month and the room or rooms he is occupying. To see the current issued bill, he must navigate to the bills page and his summary of account will be presented.

The payments made must be given personally to the manager upon collection and the manager must tick the paid option to the paid tenant.

All data will be stored even if the tenant will move out. This will be useful for future referencing and avoid constant registration of tenants who always rent after a while.

As long as the web server is online, so is the system. If there is internet connection the user can access the system at will by providing valid credentials.

Chapter IV

**METHODOLOGY**

**Environment**

The proposed system will be used by dormitories and its tenants. The main functionality of the system will be handled by the manager whom has direct control of its features.

**Population of the study**

The population of the study to be benefited and targeted by the proposed system includes the manager of the company and its tenants at present.

**Organizational Chart**

**Requirements Specification**

This part describes the intended purpose and environment for software under development. It includes operational feasibility, technical feasibility, schedule feasibility, economic feasibility, requirement modeling and object modeling.

Operational Feasibility

The researchers determined the problems of the existing system by determining the cause and effect through fishbone diagram. Determining these properly, and comparing it hereafter to the capabilities of the proposed system will mean beneficial.

Fishbone Diagram

Fishbone diagram, Ishikawa diagram or the Cause-and-Effect diagram was used by the researchers to determine systematically the factors that affect and contribute to the current process of the dormitories. Using this method, the researchers can study the root cause and arrived with a solution for the current system.

Fishbone Diagram of the Existing System

MANPOWER

MATERIALS

METHOD

Inaccurate computations

Easily manipulated by tenants

Redundant process

Inconsistent rate of rooms

SLOW, INACCURATE, AND UNSECURED PROCESS OF BILLING SYSTEM

Prone to human error

Loss and damage of records

Incorrect computations

Not utilized properly

MACHINE

Functional Decomposition Diagram

Technical Feasibility

With the data gathered, it was found that the company can easily integrate the proposed system to the business. Internet access, well equipped computers, smartphones capable of accessing the internet, are all present leading to a firm conclusion of proposed system viability.

Compatibility Checking

The researchers tabulated similarities and differences of hardware present in the company and required by the proposed system and found no conflict or requirement that can hinder the system.

Relevance of the technologies

Schedule Feasibility

The researchers followed a time frame and completion dates for all the activities within the research project to meet deadlines and comply with the requirement of study despite delays in schedule

Gantt Chart

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Task Performed** | **January**  **(2017)** | | | **February (2017)** | | | **March**  **(2017)** | |
| **W1** | **W2** | **W3** | **W1** | **W2** | **W3** | **W1** | **W2** |
| Interview |  |  |  |  |  |  |  |  |
| Consultation(Documentation review) Chapter 1 Creation  -Project |  |  |  |  |  |  |  |  |
| Chapter 2 Creation  -Design Methodology |  |  |  |  |  |  |  |  |
| Chapter 3 Creation  -Technicality of the Project |  |  |  |  |  |  |  |  |
| Chapter 4 Creation  -Summary, Conclusion and Recommendation   -Program Design |  |  |  |  |  |  |  |  |
| Making of Presentation |  |  |  |  |  |  |  |  |
| Presentation |  |  |  |  |  |  |  |  |
| Revision |  |  |  |  |  |  |  |  |
| Final Submission |  |  |  |  |  |  |  |  |

Economic Feasibility

A negative economic benefit is non-existent. The use of the proposed system will not yield negative economic value but will ensure positive cash flow. This, not considering the possible errors being made through manual

computations which will be removed through the proposed system. If this is to be included, a more positive economic benefit will arise.

Cost-Benefit Analysis

To evaluate the actual figures that show worth viability, the researchers constructed a Cost and Benefit Analysis. This includes development Cost, the entire cost of creation of the proposed system, operating costs, cost of running the existing system and the proposed system, and the return on investment, the percentage of the initial investment return after the investment has equaled the costs of operation per year.

1. Development cost

Total Development Cost

|  |  |
| --- | --- |
| **Particular** | **Cost(Php)** |
| Programmer’s Fee | 0.00 |
| Training Fee | 0.00 |
| Total | 0.00 |

1. Existing operating cost

Office Materials

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Particulars** | **Unit** | **Quantity** | **Unit Cost (Php)** | **Monthly Cost (Php)** | **Annual Cost (Php)** |
| Ball pen | pieces | 2 | 7.00 | 60.00 | 720.00 |
| Folder (Long) | pieces | 7 | 15.00 | 105.00 | 1260.00 |
| Logbook | pieces | 1 | 113.75 | 113.75 | 1,365.00 |
| Staple Wire | box | 1 | 20.00 | 20.00 | 240.00 |
|  |  |  |  | Total(Php) | 3,585.00 |

Employee Salary

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Position** | **Working hours per month (h)** | **Hourly salary (Php)** | **Monthly Salary (Php)** | **Annual Salary (Php)** |
| Operation Manager | 28 | 87.5 | 2,450.00 | 29,400.00 |
|  |  |  | Total(Php) | 29,400.00 |

Total Existing Operating Cost

|  |  |
| --- | --- |
|  | **Total Cost(Php)** |
| Office Materials | 3,585.00 |
| Salary | 29,400.00 |
| Total | **32,985.00** |

1. Proposed Operating Cost

Office Materials

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Particulars** | **Unit** | **Quantity** | **Unit Cost (Php)** | **Monthly Cost (Php)** | **Annual Cost (Php)** |
|  |  |  |  |  |  |
|  |  |  |  | Total(Php) |  |

Employee Salary

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Position** | **Working hours per month (h)** | **Per hour salary (Php)** | **Monthly Salary (Php)** | **Annual Salary (Php)** |
| Operation Manager | 0.5 | 87.5 | 175.00 | 2,100.00 |
|  |  |  | Total(Php) | 2,100.00 |

Server

|  |  |  |
| --- | --- | --- |
|  | **Monthly Cost (Php)** | **Annual Cost (Php)** |
| System Rent | 1,000.00 | 12,000.00 |
|  | Total(Php) | 12,000.00 |

Total Proposed Operating Cost

|  |  |
| --- | --- |
|  | **Total Cost(Php)** |
| Office Materials | 0.00 |
| Salary | 2,100.00 |
| Server | 12,000.00 |
| **Total(Php)** | **14,100.00** |

1. Savings

Savings (S) = Existing Operating Cost (EOC) – Proposed Operating Cost(POC)

= Php 32,985.00 – Php 14,100.00

= Php 18,885.00

Interest

= (interest rate +1)n [n = year; interest rate = 1.2%]

1. Comparative Table 30615

Comparative Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Year | Savings (Php) | Interest | Present Value | Cumulative Present Value (CPV) | Mark  (x or /) |
| 1 | 18,885.00 | 1.20 | 24,679.17 | 24,679.17 | X |
| 2 | 18,885.00 | 1.44 | 20,565.97 | 45,245.14 | X |
| 3 | 18,885.00 | 1.73 | 17,118.50 | 62,363.67 | X |
| 4 | 18,885.00 | 2.07 | 14,306.76 | 76,670.43 | √ |
| 5 | 18,885.00 | 2.49 | 11,893.57 | 88,564.00 | √ |

1. Pay Back Period

Payback Period (PBP)= (DC-CPV of last “X” mark /PV of 1st “/”) +number of “X” mark.

PBP = ((75,000 – 62,363.67)/( 14,306.76))+3

PBP = 3 years, months and days(3.88 day and months)

1. Net Present Value

Net Present Value (NPV)= CPV of 1st“/” mark-DC

NPV = 76,670.43– 75,000

NPV = Php 1670.43

1. Return on Investment

Return on Investment (ROI)= (NPV/DC)\*100%

ROI = (4,429.01/75,000)\*100

ROI = 2.23%

Cost Recovery Scheme

Requirements Modeling

This section shows the analysis of the system. It also describes what the system is and what it does.

Input

The following are the inputs used in the proposed system:

* Registration of tenant as renter
* Registration of tenant in the system for access to his account
* Assignment of tenant to available room
* Assignment of rent per month (including additional payment if present)
* Collection periods
* Feedbacks to the company by the tenants

Process

The following are the processes in the proposed system:

* Authenticate user credential to limit function per access level. Depending on the user account registered, the system will determine what account the user belongs may it be administrator or tenant. This is checked through PHP query in the MySQL database.
* Monitor number of available rooms and store it. The system always checks the room availability real-time and always keeps the data available to be presented in the dashboard. Again, this is accomplished by querying the number of rooms with status set as ‘0’ meaning available.
* Compute rent of each tenant and store it. This is determined by the set agreement of rent price by the owner and the tenant. The system adds the current rent and the balance if present.
* Compute additional payments and store it. Additional payments include the excess electricity bill and payments of appliances brought by the tenants as well as if any tenants incurred any damages to the property owned by the dormitory.
* Store status of agreement on tenant if expired or active.
* Store feedbacks of tenant to manager. Feedbacks sent by the tenants will be stored in the database as well and can be will be queried for display on the manager’ account.

Output

The following are the output of the proposed system:

* Show available rooms to the manager
* Show feedbacks of tenants to the manager
* Show bills of dues to tenants.
* Show expired or active agreements.
* Show rent, payment status, balance, and amount paid by tenants to tenants accountable and to the manager.

Control

* The manager and the tenants can access with different functions. The manager handles all inputs. The tenant only handles input of feedbacks and view his account.

Object Modeling

To demonstrate the logical structure of *the* existing system and the proposed system - the researchers constructed a Use Case Diagram for illustration of the existing system and a Sequence Diagram for showing the proposed system structure.

**Use Case Diagram of Existing System**

Use Case Diagram of the Existing System (Registration)

Tenant

Registration

Manager

Use Case Diagram of the Existing System (First Payment)

Manager

Tenant

First Payment

<<Include>>

Use Case Diagram of the Existing System (Monthly Payment)

Manager

Tenant

Monthly Payment

<<Include>>

<<Include>>

<<Extend>>

Use case Diagram of the Existing System (Changing Room Type)

Manager

Tenant

Changing Rooms

<<Include>>

Use Case Diagram of the Existing System (Final Payment)

Manager

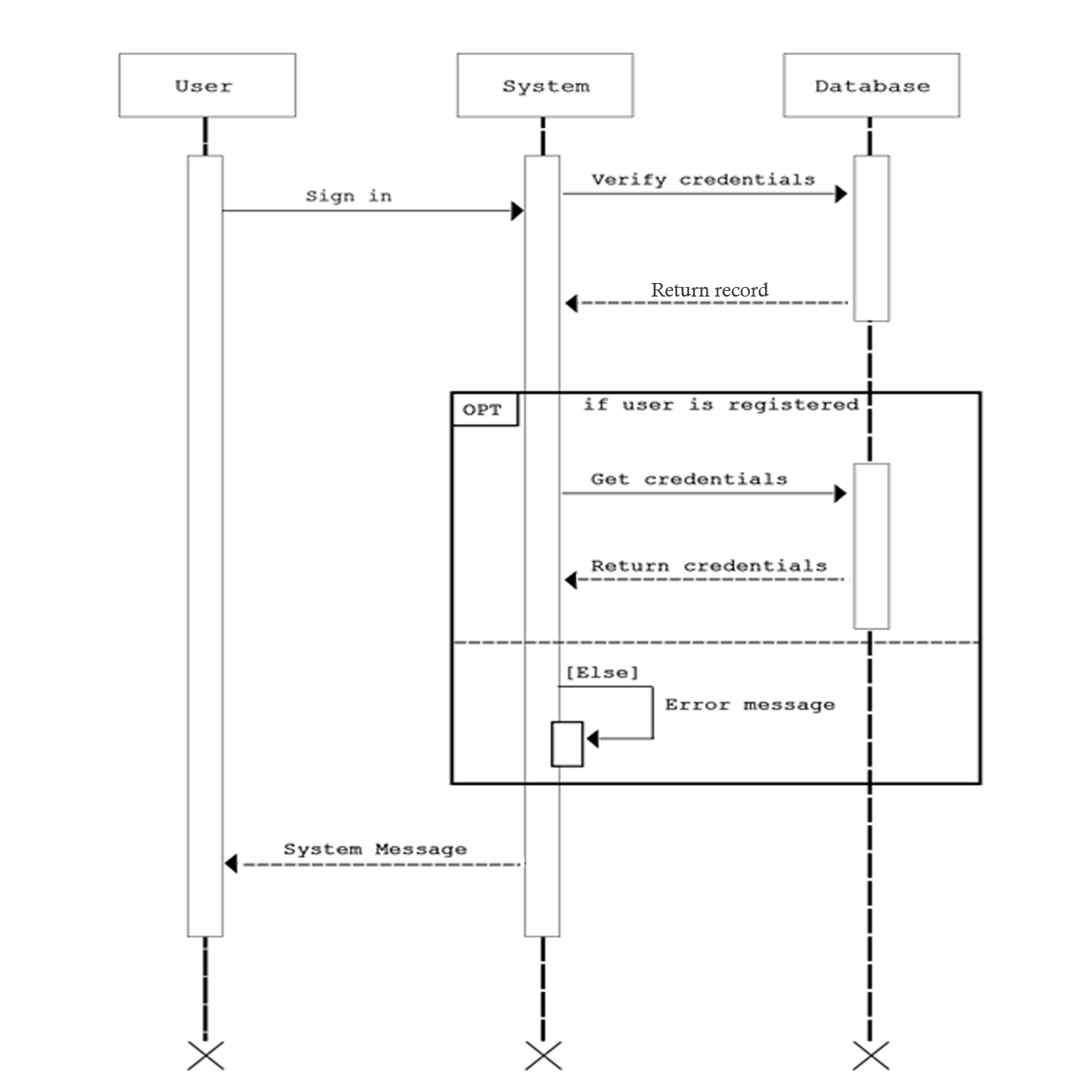
Tenant

Final Payment

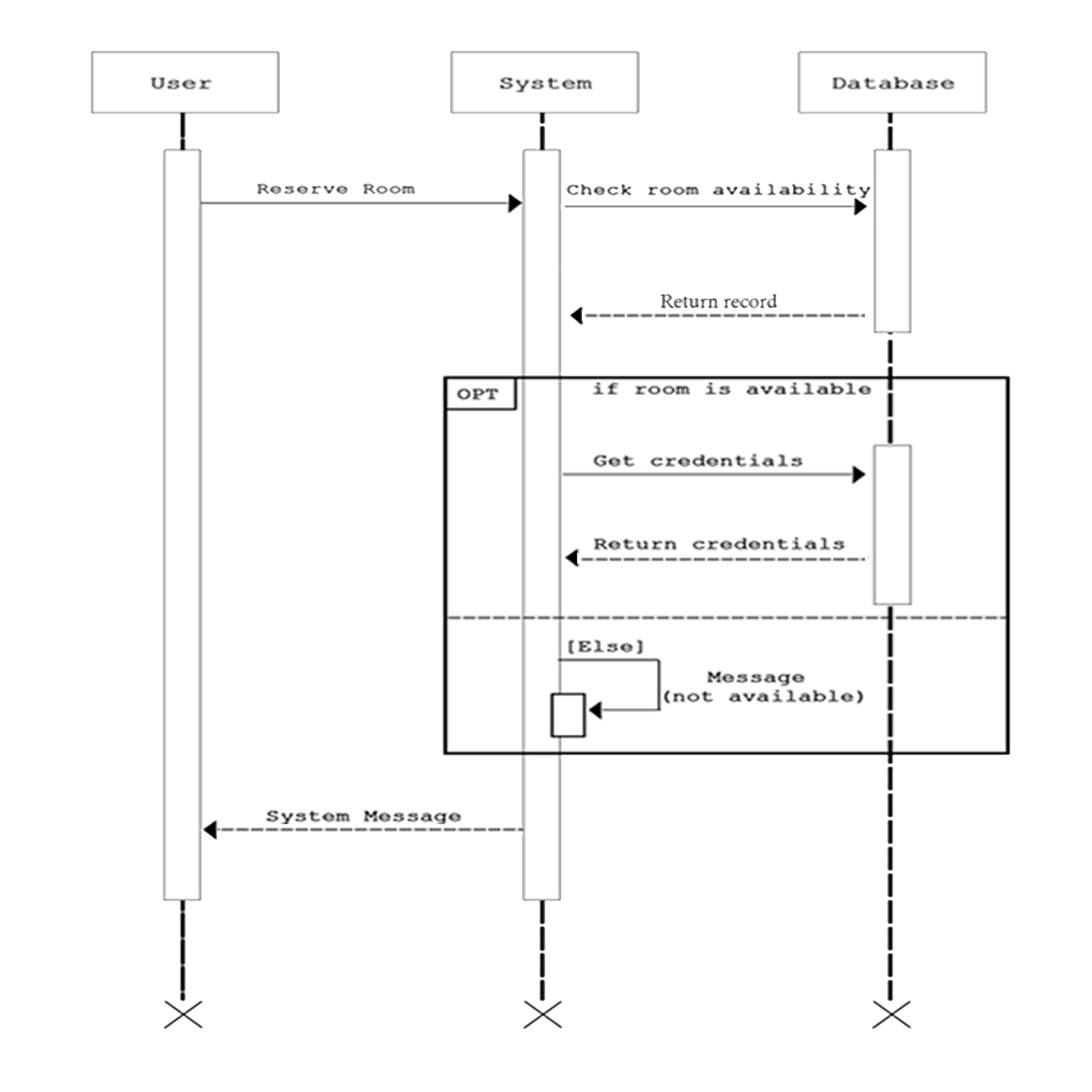
<<Include>>

Class Diagram

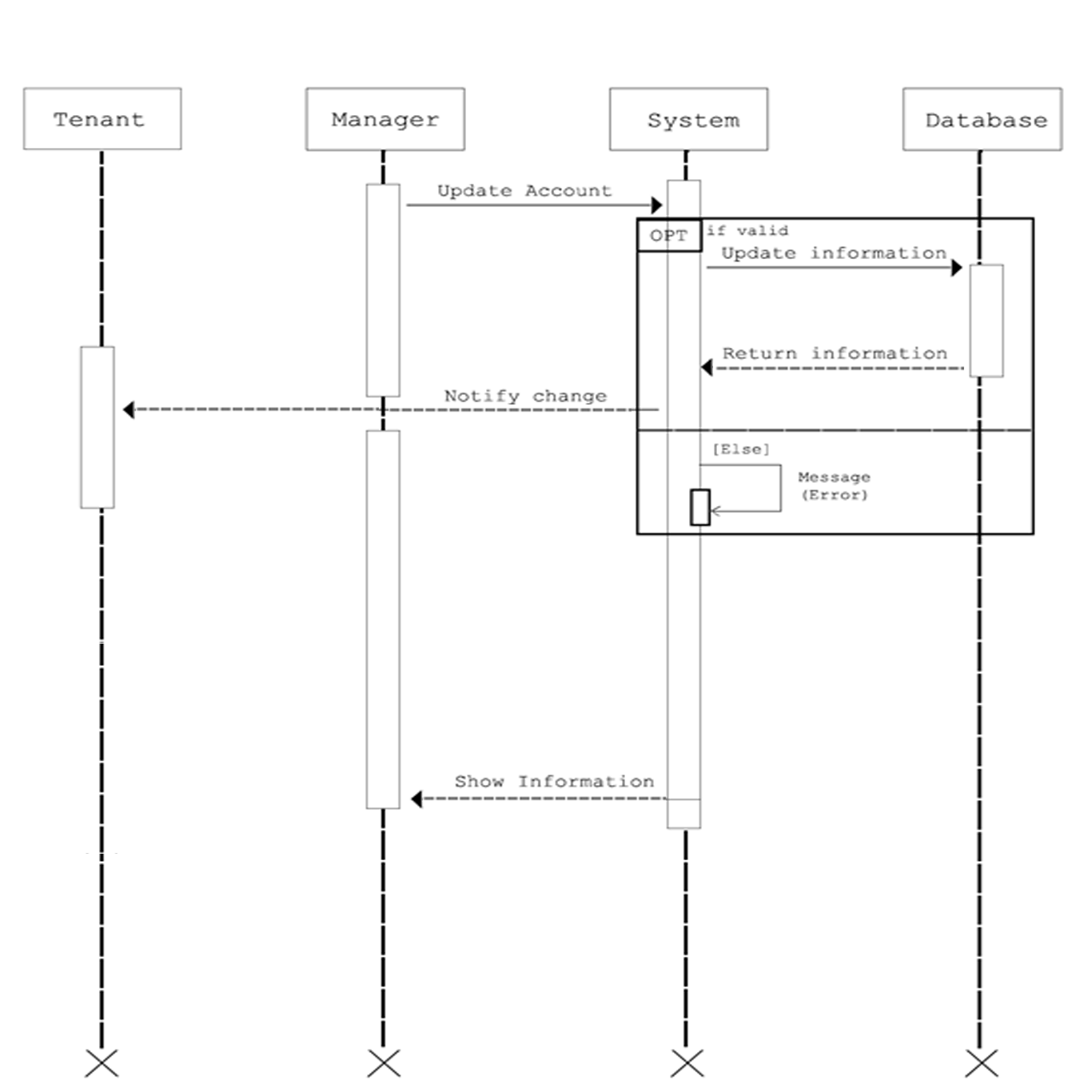
Sequence Diagram of the Proposed System (User Login)



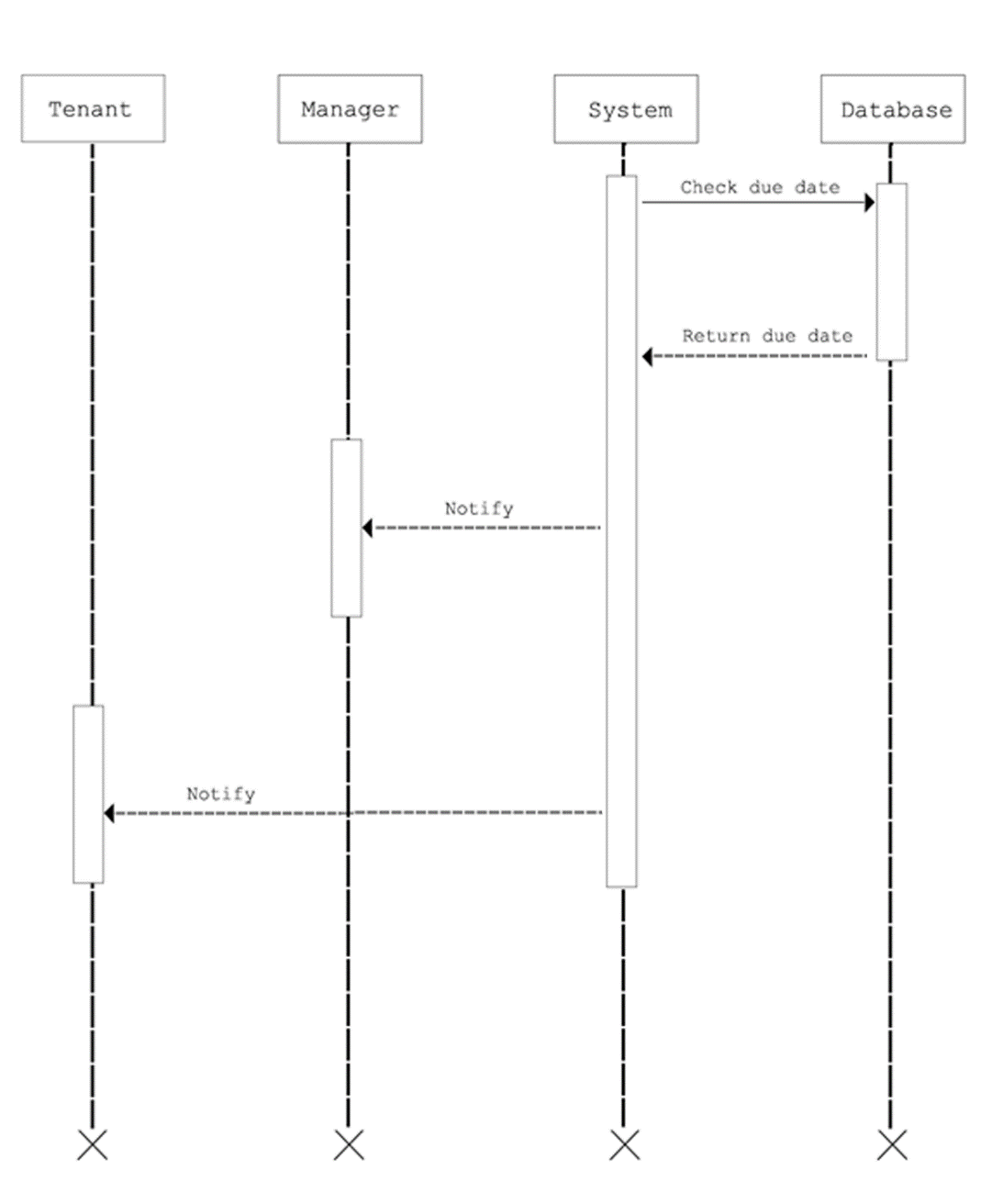
Sequence Diagram of the Proposed System (Room Reservation)



Sequence Diagram of the Proposed System (Cost Adjustment)



Sequence Diagram of the Proposed System (Payment Scheduling)

****

Activity Diagram

Risk Assessment/Analysis

The system involves basic personal information of the tenants and the owner. Their names, addresses, and contact numbers, must never be exposed to the public. These are very sensitive information which can put the users in danger.

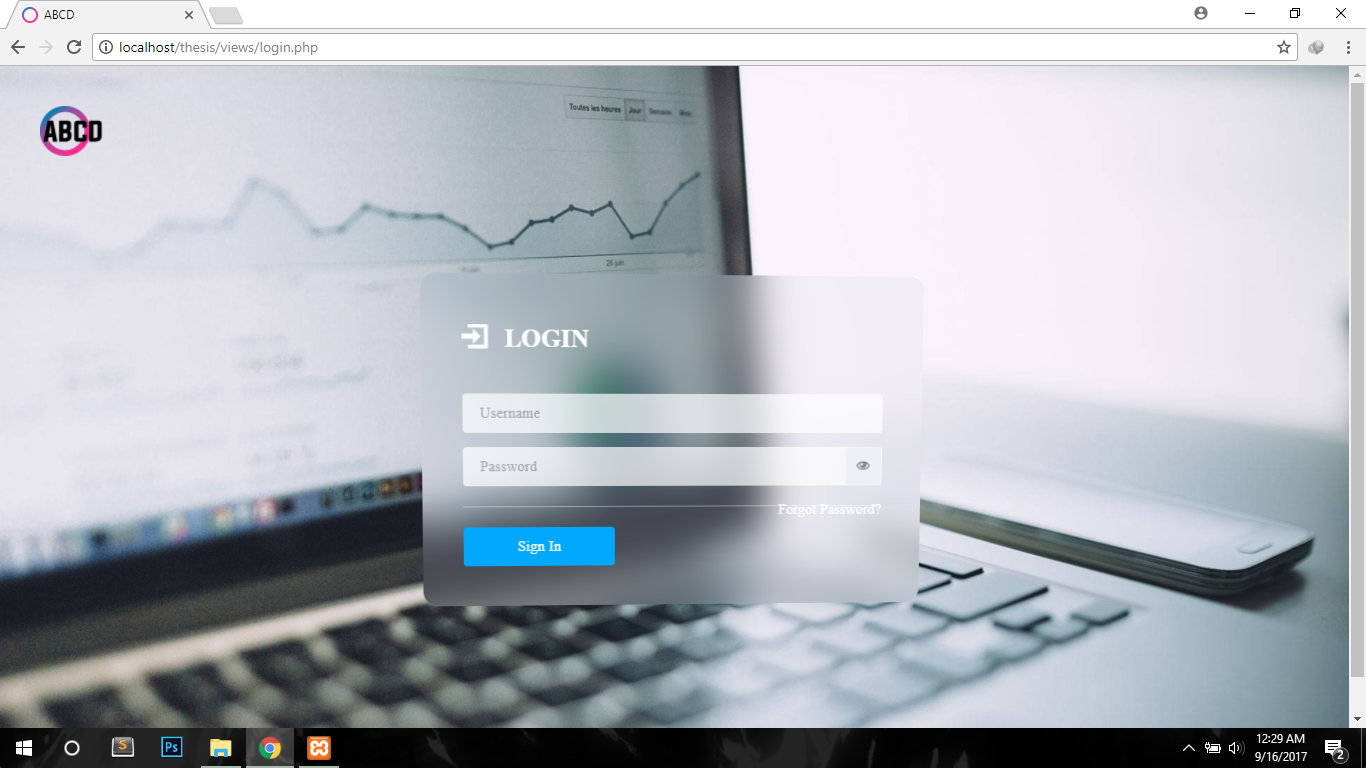
The system is web based and uses an SQL database. This makes the system vulnerable to php and MySQL attacks such as injection attacks, denial of service, blind address traversal, and others alike. The system is available for the public to view but only those registered can access the system.

With these possible vulnerabilities, encryption and careful coding to prevent possible intrusions are essential in the system. Even though only a few people will use the system, security is a must.

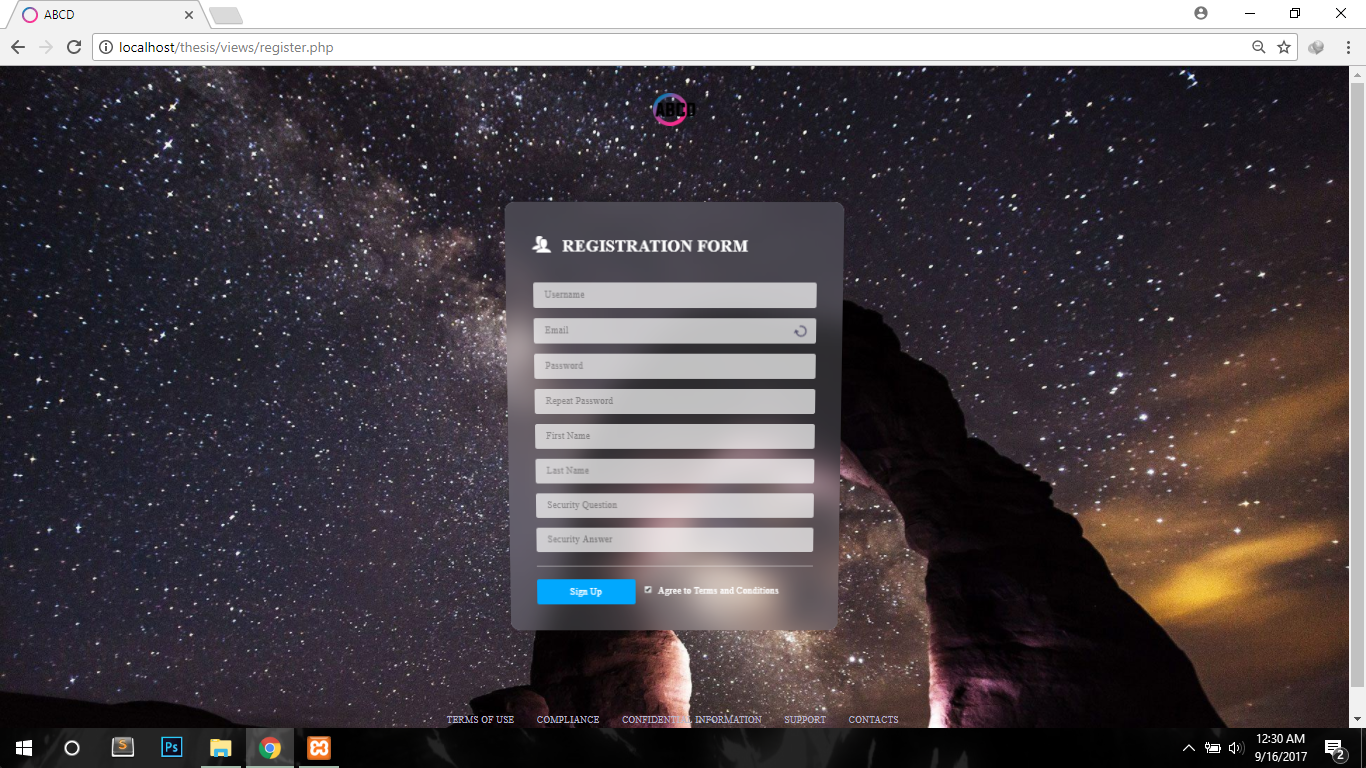
With the system coded with security in mind, the system will be invulnerable sql or php attacks. This ensures up to 99% safety to its users and their personal information.

**Design**

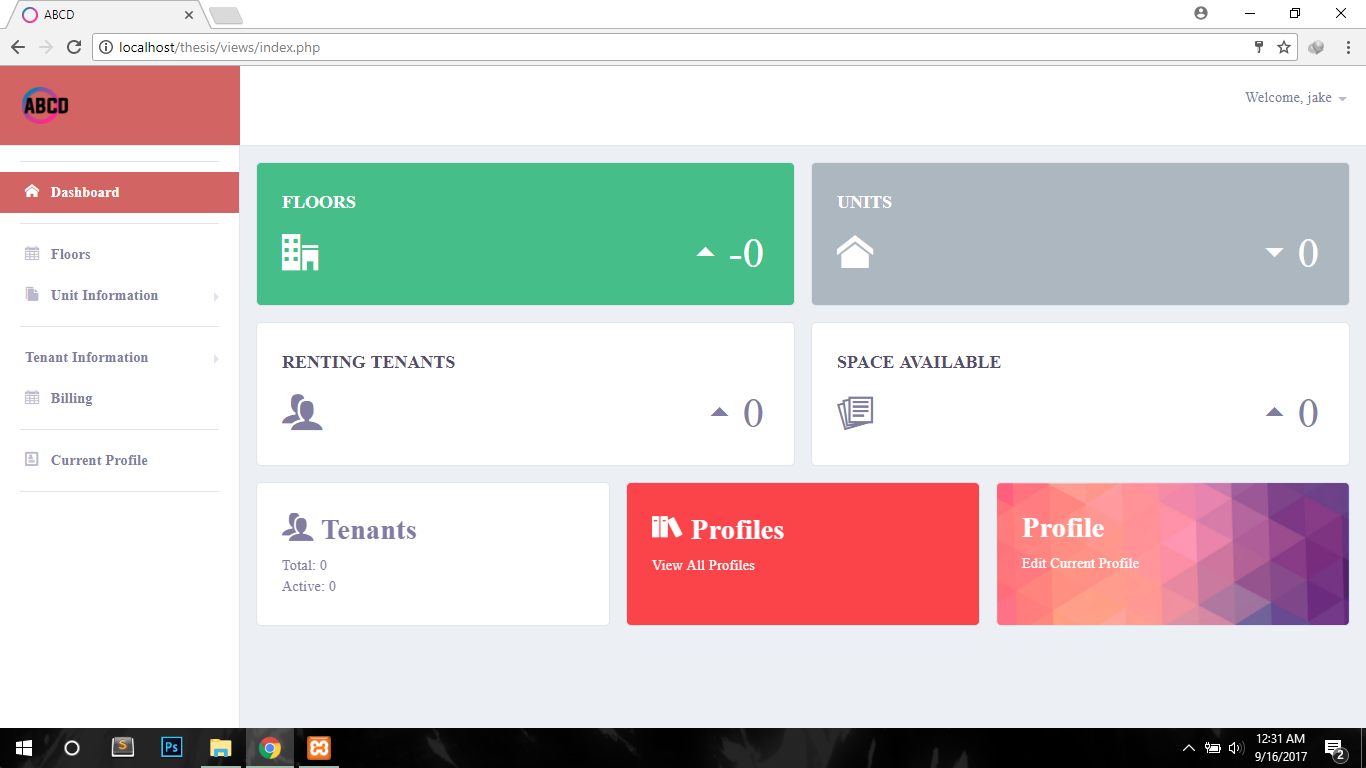
Output and User-Interface Design



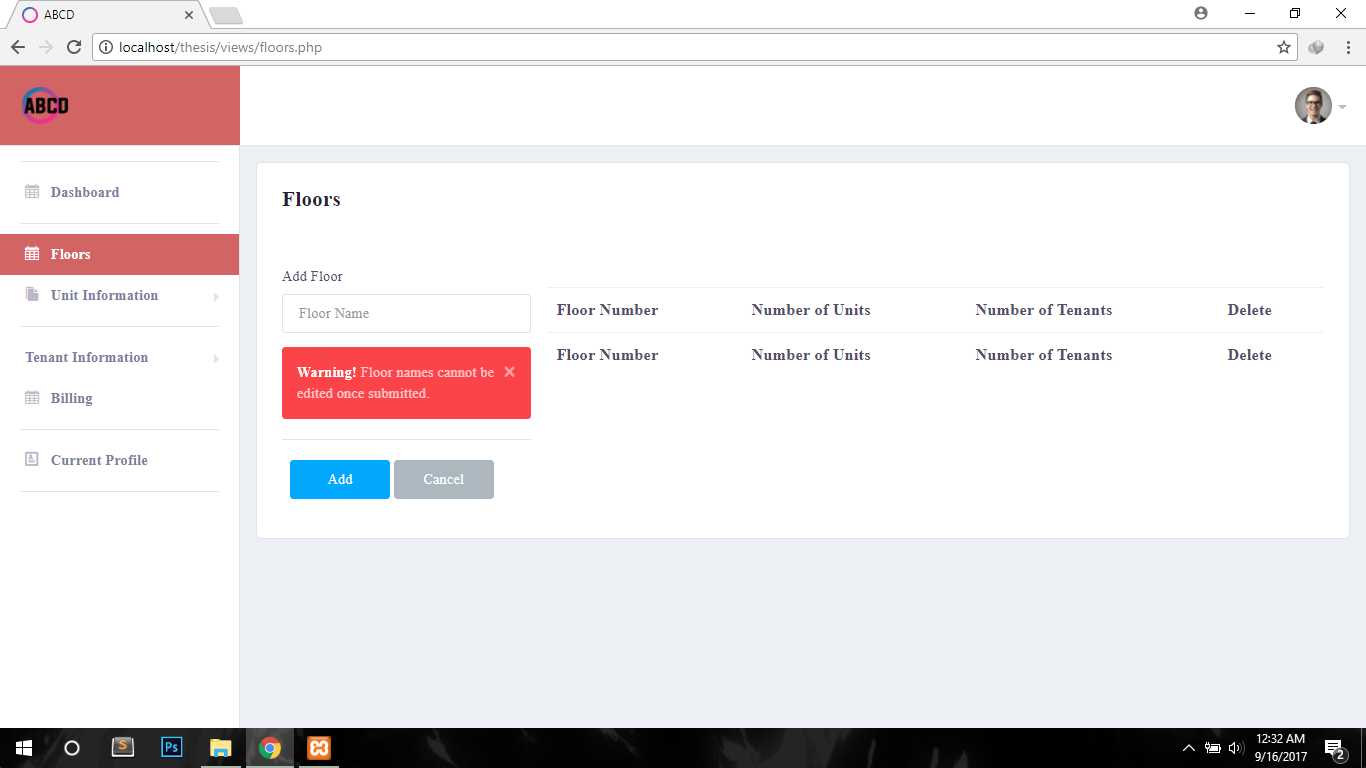
Registration



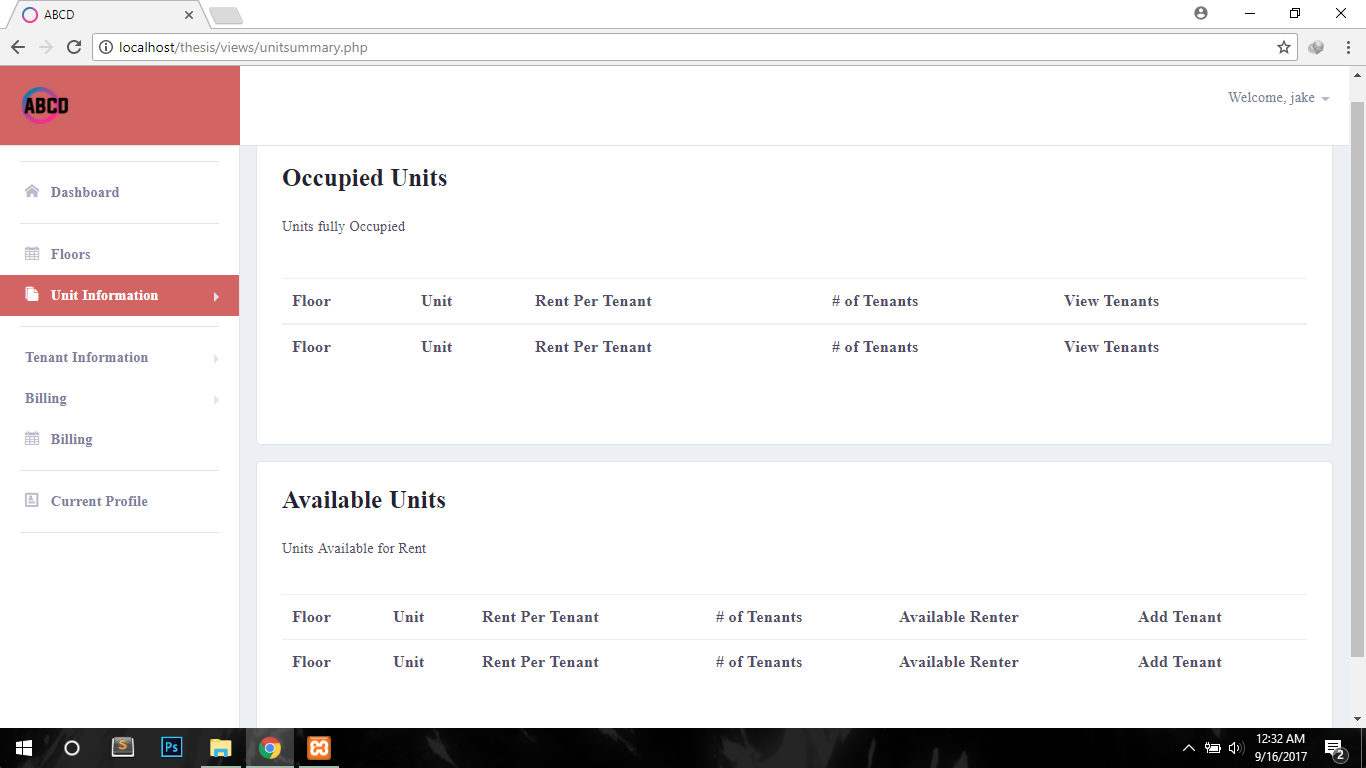
Admin Dashboard



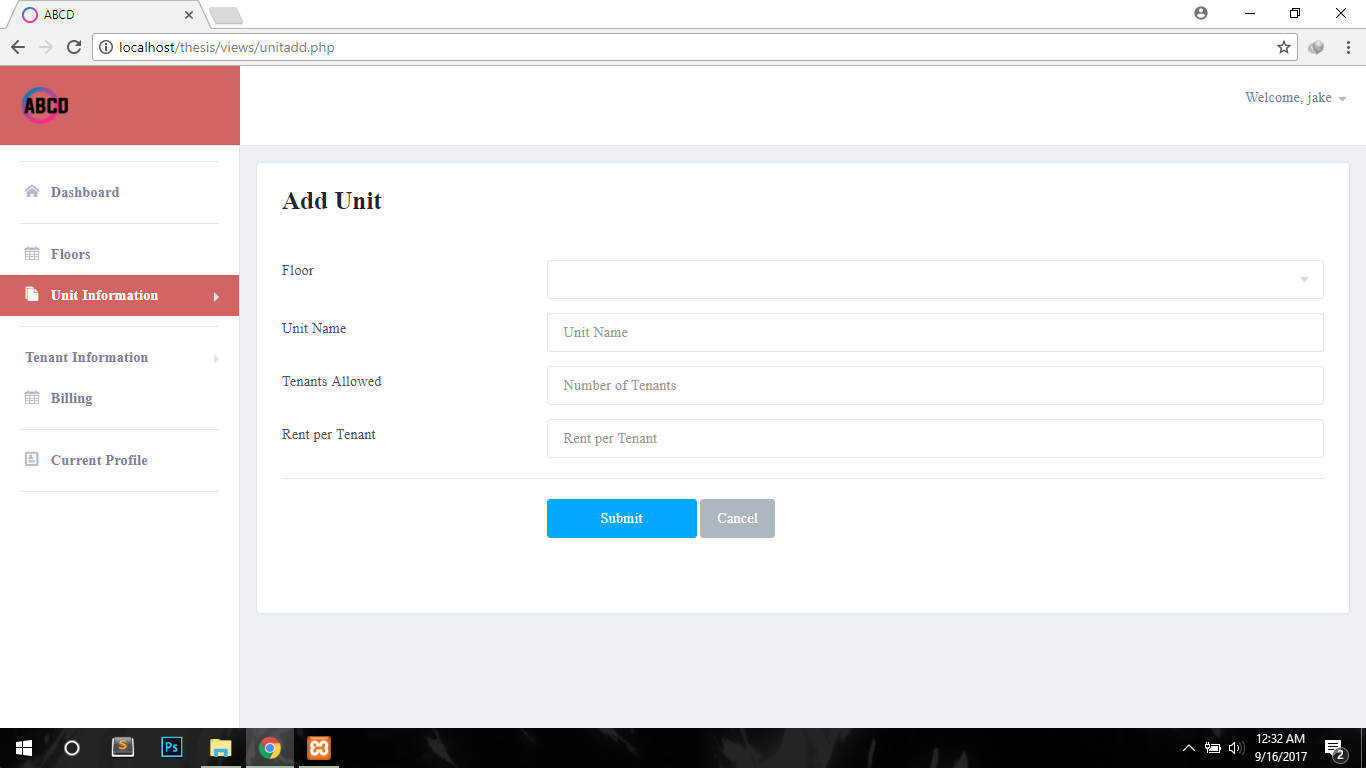
Floor Summary



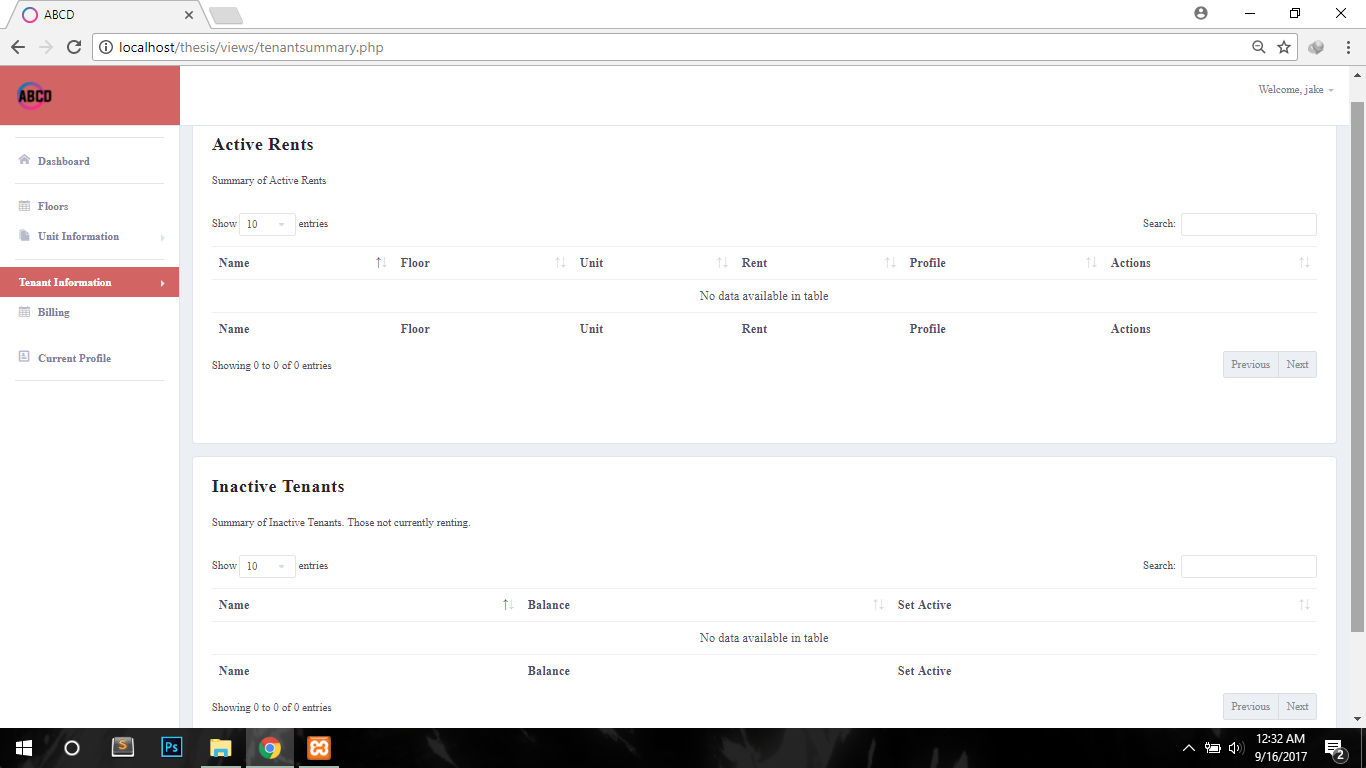
Unit Summary



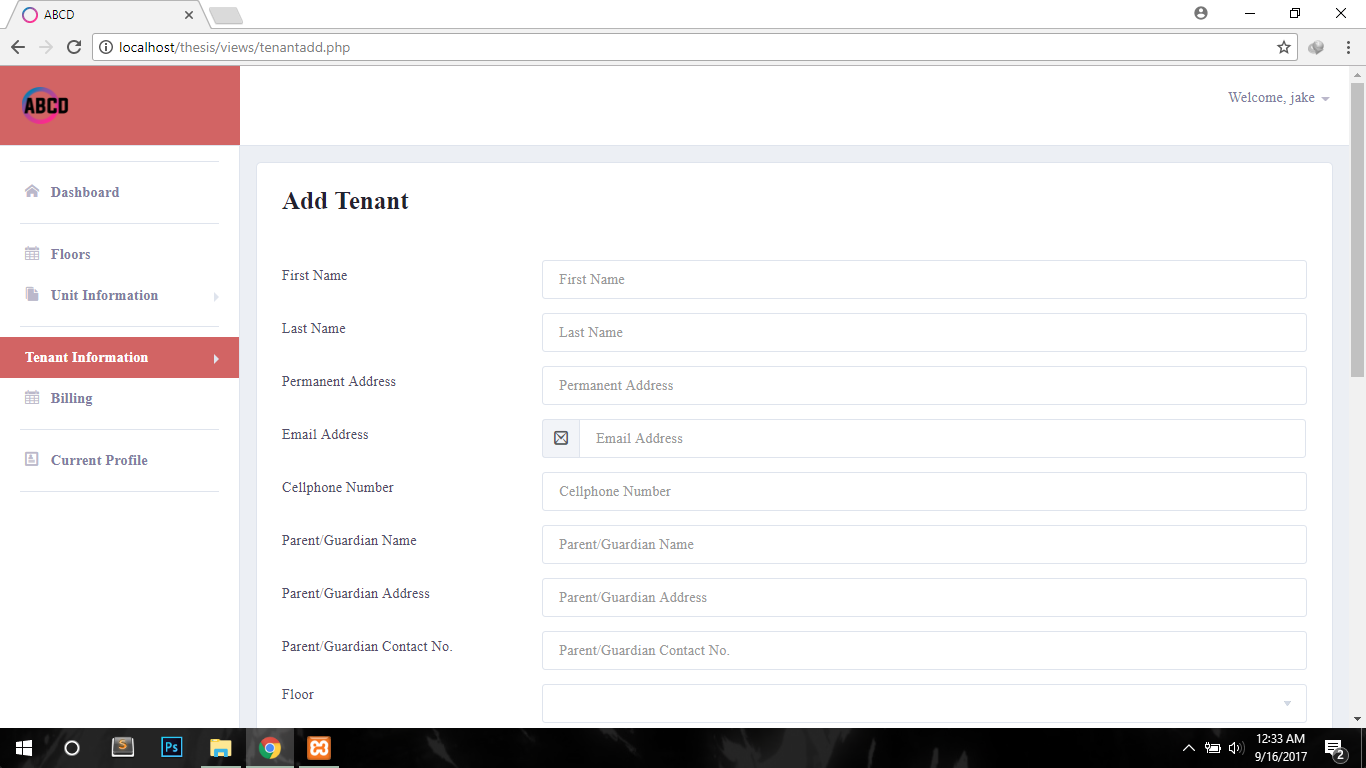
Add Unit

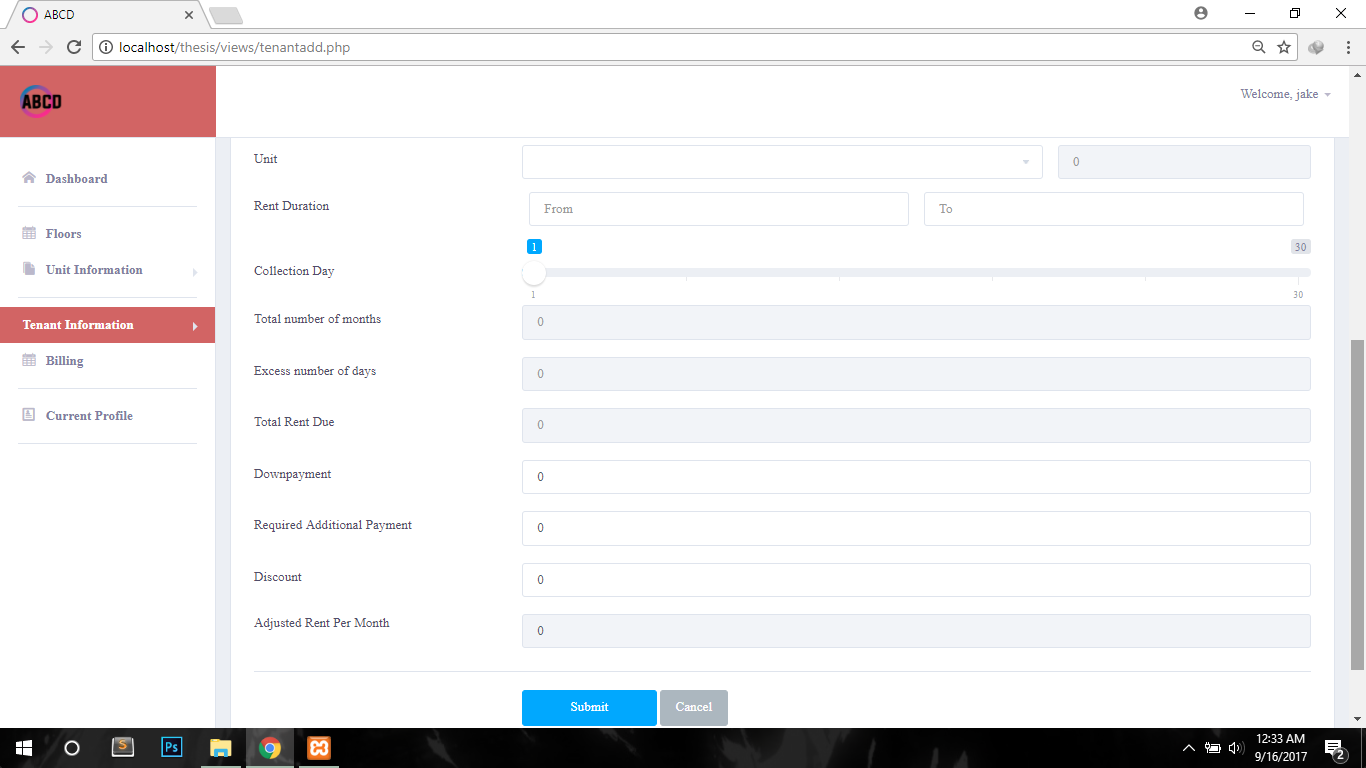


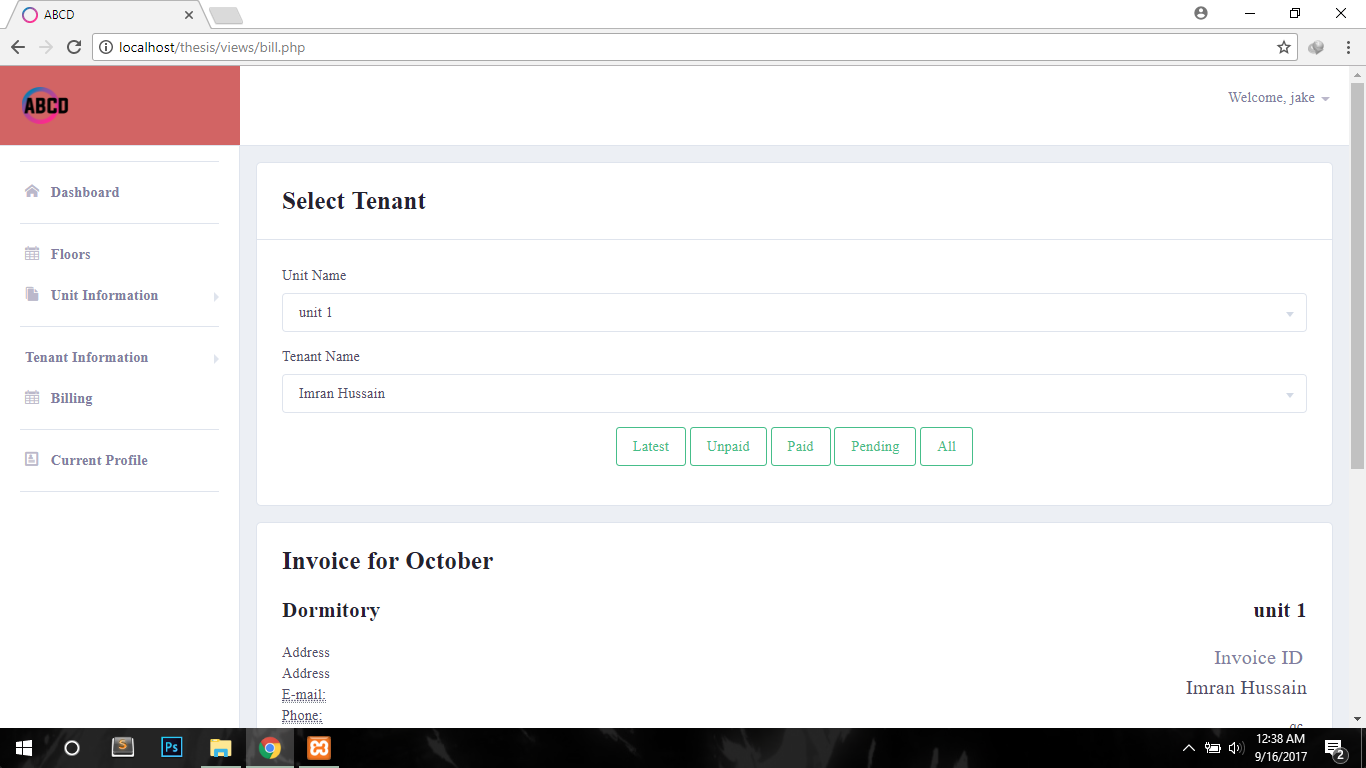
Tenant Summary

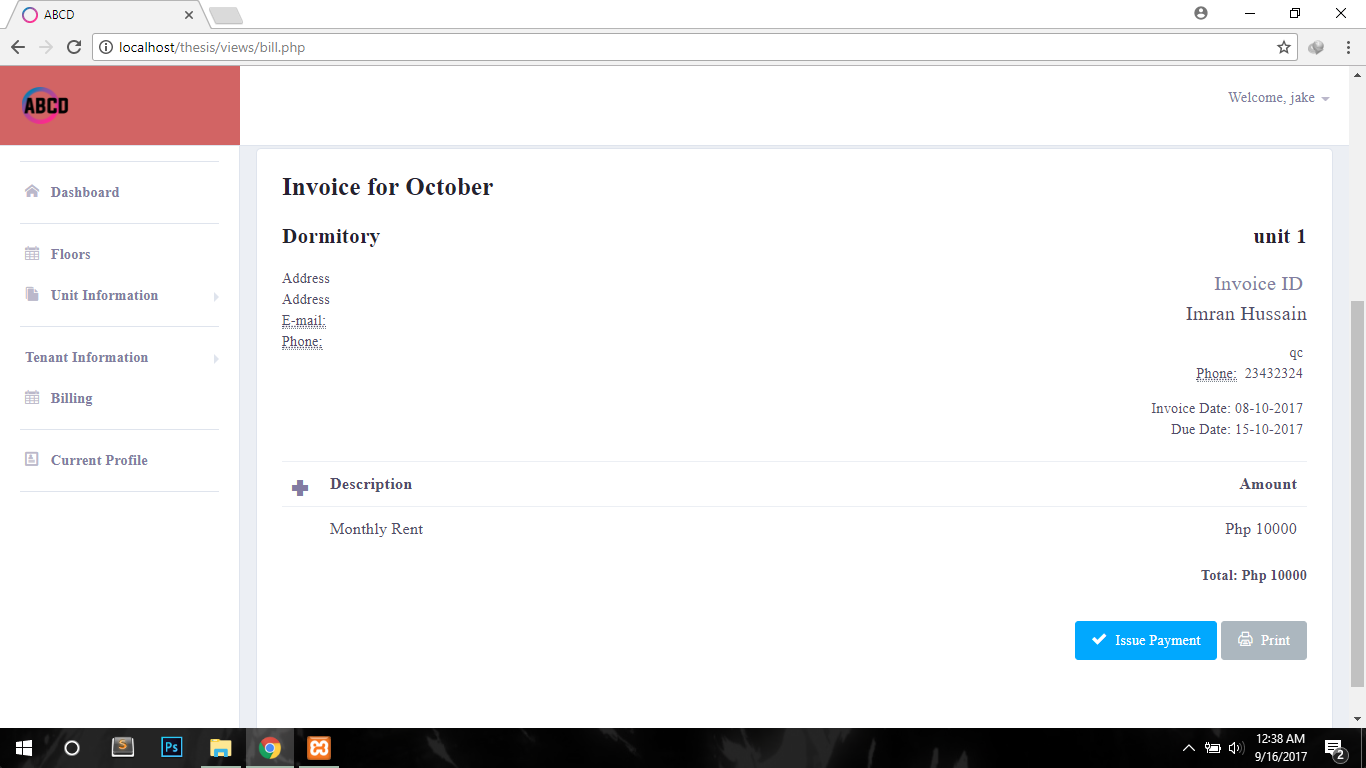


Add Tenant

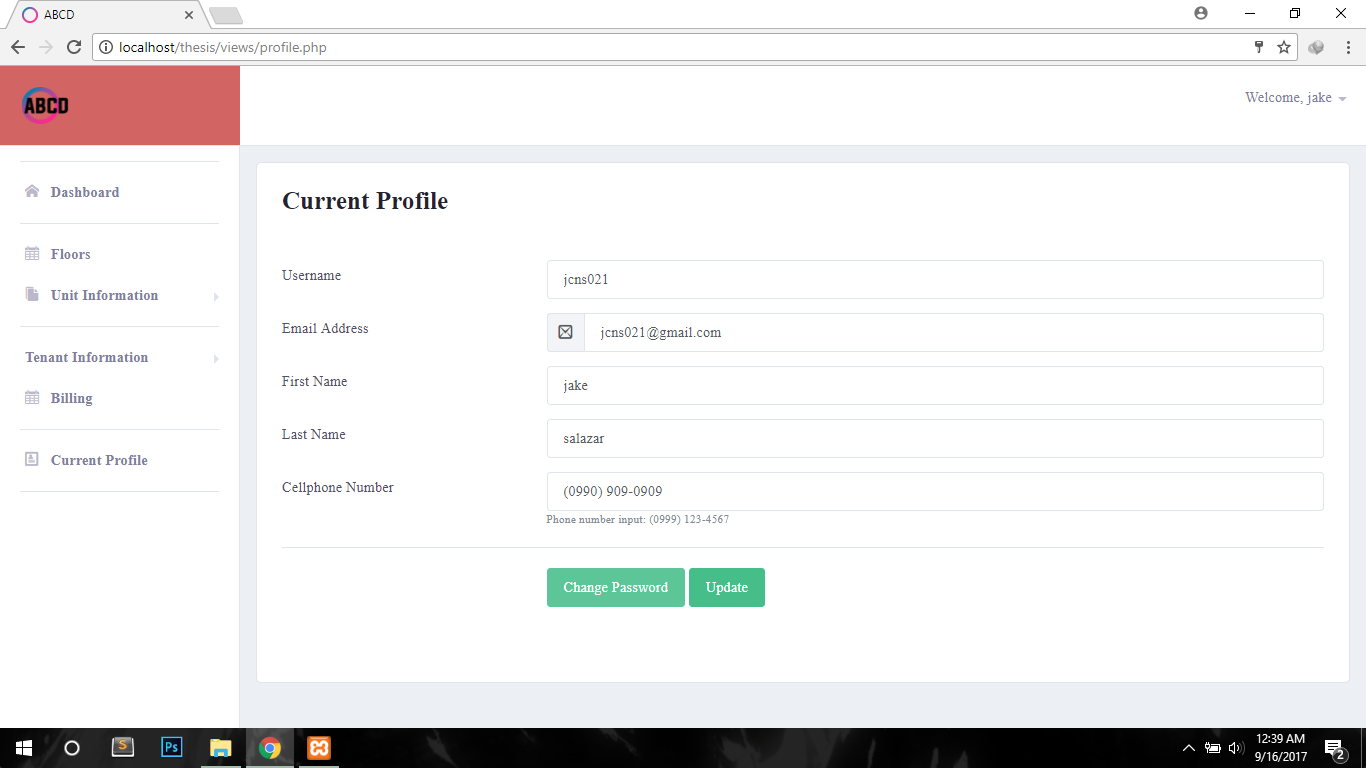




Bills Summary  




Current Profile



**Data Design**

Entity Relationship Diagram

|  |  |
| --- | --- |
| Floor | |
| id | PK |
| oid | FK |
| floorName |  |

|  |  |
| --- | --- |
| Owner | |
| id | PK |
| Username |  |
| Email |  |
| Password |  |
| first\_name |  |
| last\_name |  |
| security\_question |  |
| security\_answer |  |

(1..M)

(M..1)

|  |  |
| --- | --- |
| Floor | |
| Id | PK |
| Oid | FK |
| floorName |  |

(M..1)

(1..M)

(1..M)

(1..M)

(1..M)

|  |  |
| --- | --- |
| Unit | |
| id | PK |
| floor\_id | FK |
| unitName |  |
| tenantAllowed |  |
| rentPerTenant |  |
| totalRent |  |
| currentTenant |  |

(1..M)

(M..M)

(M..1)

..M..

(M..1)

..M..

|  |  |
| --- | --- |
| tenantprofile | |
| id | PK |
| oid | FK |
| firstName |  |
| lastName |  |
| userName |  |
| password |  |
| address |  |
| email |  |

(M..1)

(M..1)

(M..M)

(M..M)

(1..1)

(1..1)

|  |  |
| --- | --- |
| bill  (1..M)  (1..M) | |
| id | PK |
| tid | FK |
| trid | FK |
| description |  |
| amount |  |
| date |  |
| status |  |

(1..1)

(1..1)

|  |  |
| --- | --- |
| bill items  (1..1)  (1..1) | |
| id | PK |
| bid | FK  (1..1)  (1..1) |
| description |  |
| amount |  |

|  |  |
| --- | --- |
| Payments | |
| id | PK |
| tid | FK  (M..1)  (M..1) |
| bid | FK |
| description |  |
| amount |  |
| date |  |

|  |  |
| --- | --- |
| tenantrentinginformation | |
| id | PK |
| tid | FK |
| uid | FK |
| Downpayment |  |
| startDate |  |
| endDate |  |
| totalMonths |  |
| collectionDay |  |
| Balance |  |
| adjustedRentPerMonth |  |

(1..1)

(M..M)

(M..1)

(M..1)

(M..M)

(M..M)

(1..M)

(1..1)

Data Dictionary

|  |  |  |
| --- | --- | --- |
| \_bill | | |
| Field Name | Data Type | Definition |
| id | int | Primary key of the table |
| tid | int | Id of the tenant |
| trid | int | Id of the tenant renting information |
| description | varchar | Description of the bill of the user |
| Amount | double | Amount |
| date | varchar | Due date of the bill |
| status | varchar | Pending, paid, not yet paid |

|  |  |  |
| --- | --- | --- |
| \_bill\_items | | |
| Field Name | Data Type | Definition |
| id | Int | Primary key of the table |
| bid | Int | Id of the bill of the tenant |
| description | varchar | Description of the bill of the tenant |
| Amount | double | Amount of the bill items |

|  |  |  |
| --- | --- | --- |
| \_floor | | |
| Field Name | Data Type | Definition |
| id | Int | Primary key of the table |
| Oid | Int | Id of the owner |
| floorName | varchar | Name of the floor |

|  |  |  |
| --- | --- | --- |
| \_owner | | |
| Field Name | Data Type | Definition |
| Id | int | Primary key of the table |
| username | varchar | Username of the tenant |
| Email | varchar | Email of the tenant |
| password | varchar | Password of the tenant |
| First\_name | varchar | First name of the tenant |
| Last\_name | varchar | Last name of the tenant |
| security\_question | varchar | Security question of the tenant |
| security\_answer | varchar | Security answer of the tenant |
| contactNumber | varchar | Contact number of the tenant |

|  |  |  |
| --- | --- | --- |
| \_payments | | |
| Field Name | Data Type | Definition |
| id | Int | Primary key of the table |
| tid | Int | Id of the tenant |
| bid | Int | Id of bill of the tenant |
| description | varchar | Description of the payment of the tenant |
| amount | double | Amount paid |
| date | varchar | Date paid |

|  |  |  |
| --- | --- | --- |
| \_tenantprofile | | |
| Field Name | Data Type | Definition |
| id | int | Primary key of the table |
| oid | int | Id of the owner |
| firstName | varchar | First name of the tenant |
| lastName | varchar | Last name of the tenant |
| userName | varchar | Username of the tenant |
| password | varchar | Password of the tenant |
| address | varchar | Address of the tenant |
| email | varchar | Email of the tenant |
| contactName | varchar | Contact number of the tenant |
| guardianName | varchar | Guardian name of the tenant |
| guardianAddress | varchar | Guardian address of the tenant |
| guardianContact | varchar | Guardian contact of the tenant |
| balance | double | Balance of the tenant |

|  |  |  |
| --- | --- | --- |
| \_tenantrentinginformation | | |
| Field Name | Data Type | Definition |
| id | int | Primary key of the table |
| tid | int | Id of the tenant |
| uid | int | Id of the unit where the tenant is currently residing |
| downpayment | int | Down payment of the tenant |
| startDate | varchar | Where the tenant will start renting |
| endDate | varchar | Where the tenant will end the renting |
| totalMonths | int | Total months of the rent |
| collectionDay | int | Collection day of the payments issued by the owner |
| balance | double | Balance of the tenant |
| adjustedRentPerMonth | double | Rent per month |

|  |  |  |
| --- | --- | --- |
| \_unit | | |
| Field Name | Data Type | Description |
| id | int | Primary key of the table |
| Floor\_id | int | Id of the floor |
| unitName | varchar | Name of the unit |
| tenantAllowed | int | Number of tenant allowed in the unit |
| rentParTenant | double | Amount of rent per tenant |
| totalRent | double | Total amount of rent possible in the unit |
| currentTenant | int | Current number of tenants renting in the unit |

**System Architecture**Network Model

A network model is a database model that is designed as a flexible approach to representing objects and their relationships. A unique feature of the network model is its schema, which is viewed as a graph where relationship types are arcs and object types are nodes. Unlike other database models, the network model's schema is not confined to be a lattice or hierarchy; the hierarchical tree is replaced by a graph, which allows for more basic connections with the nodes.

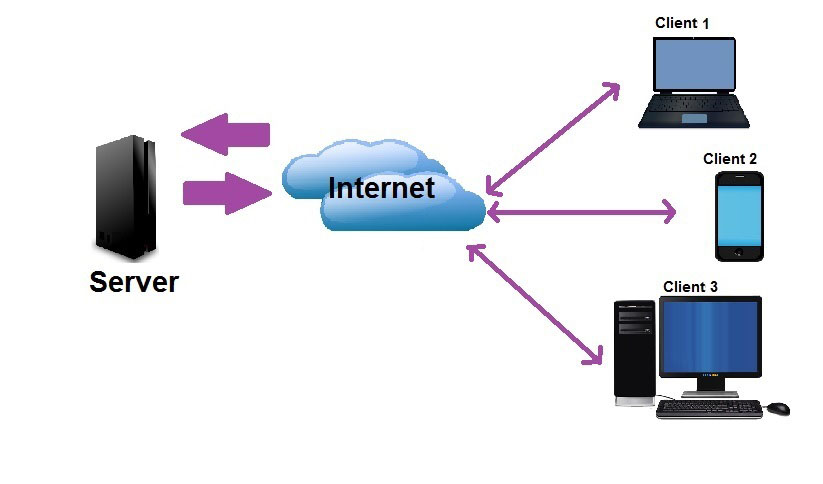
The proposed system will be making use of the client-server model. This model is a distributed communication framework of network processes among service requestors, clients and service providers. The client-server connection is established through a network or the Internet. The client-server model is a core network computing concept also building functionality for email exchange and Web or database access. Web technologies and protocols built around the client-server model are what is used by the proposed system.

Network Topology

The proposed system, with a client-server network model will have a bus network topology. Bus networks are simply the use of a single common [connection](https://www.lifewire.com/definition-of-backbone-817777) to connect all devices or in

this case, to connect a device to the internet. Because the proposed system access the internet, which involves complex topologies, the means of connecting to the internet having only a single device would make use of bus topology as the most practical and efficient way. The access of client-server model in a bus topology is illustrated at the following figure.

Network Topology Illustration



Security

The MySQL database in the server is the biggest security target and is the most essential to protect because of sensitive information stored in it. The researchers would make use of the following to ensure security to the system.

Passwords and usernames are encrypted through AES (Advanced Encryption Standard) 256 bits. It is a symmetric encryption algorithm developed by two Belgian cryptographer Joan Daemen and Vincent Rijmen. AES was designed to be efficient in both hardware and software, and supports a block length of 128 bits and key lengths of 128, 192, and 256 bits. AES encryption is used by most governments for securing sensitive but unclassified material proving its security.Access will only be granted to authorized accounts.

An anti-DDOS (Distributed Denial of Service) will also be enabled in the server to prevent denial of access in the server. DDOS or DOS (Denial of Service) is type of attack where other users or computers would send large quantities of requests to a server leading to overloading and eventually rendering the server or website unreachable. Anti-DDOS functions make sure this would not happen by limiting requests per IP (Internet Protocol) address to ensure no DOS attack is taking place.

Directory traversal attacks on websites will also be impossible since the system resides on PHP framework. The website address does not equal to the folder names in the server.

Finally, by using the latest version of MySQL and using HTML characters like “%20” for a “space” would inhibit SQL injection attacks may it be blind or with keywords. Access to the database would be granted with SQL injection. But with this protection, it would be impossible.

The only security flaw present in the system would be the users, their accounts, and how they secure them in their own.

**Development**

This section discusses the development of the system. This includes software, hardware and program specifications used for development.

Software Specification

The researchers have decided to use the following software as tools in developing the proposed system to its full functionality.

Adobe Photoshop CS6. This application is for editing as well as creating images of custom sizes. This will be useful for designing the graphical user interface to be presented to the user. The image created will be the basis for the website to be designed.

Navicat. This software is capable of connecting to a database in a server and relaying the available information such as, the tables, rows, and data stored in the database. It can also be used to create tables from databases with visual representations. It can also push queries and receive results. This software is useful in easily creating a normalized database and is useful for testing queries to be made by the system.

Sublime Text 3. This application is not an IDE but a text editor. It has no compiler or interpreter. It is different from other text editors because it was made for programming languages which includes plugins for easier coding. Depending on the language selected, it will color code syntaxes used for easier coding. This software will be used for coding the website in HyperText Markup Language (HTML), Cascading Stylesheet (CSS), and HyperText Preprocessor (PHP).

Xampp. This software is capable of starting a localhost server as well as connection to MySQL database. This program is essential to simulate connections to the server being localhost. This will eliminate the need to find other servers online simply for testing. All connections made by the system upon testing will be through the server created with this application.

Hardware Specification

The researchers will be developing the system having the following components.

Hardware Specification

|  |  |
| --- | --- |
| Processor | Intel Core i5 3740k |
| RAM | 8GB |
| Video Card | Sapphire R9 270x 2gb OC Edition |
| Output Device | 32” 1920 x 1080 Resolution |
| Network Card | 1gbps Realtek Network Card |

Program Specification

Presented in this section is the system requirements needed to run the system without experiencing lags. Listed here are the requirements needed for a computer and a smartphone.

Recommended Hardware and Software Requirements for Computer

|  |  |
| --- | --- |
| Hardware Requirements | |
| Processor | Any dual core processor (Intel or AMD) |
| RAM | 2GB or greater |
| Input Device | Mouse, Keyboard |
| Internet Connection Speed | 512 kbps or faster |
| Network Card | Any capable of 512 kbps or faster |
| Graphics Card | 128 MB Integrated or better (Intel or AMD) |
| Display | 1280x720 resolution or higher |
| Software Requirements | |
| Operating System | Windows 7 (x32/x64) and above  Mac OSX or above |
| Internet Browser | Google Chrome, Safari, Opera, Mozilla Firefox (Latest Updated Version) |

Recommended Hardware and Software Requirements for Smartphone

|  |  |
| --- | --- |
| Hardware | |
| Processor | Any dual core processor with frequency greater or equal to 1ghz |
| RAM | 512 MB or greater |
| Graphics Processor | Any (Andreno, Snapdragon, A7, etc) |
| Software Requirements | |
| Operating System | Android 4.4 or IOS 7.0 |
| Web Browser | Native android browser, Google Chrome, Mozilla Firefox, Opera, Safari |

Programming Environment

Here is defined what software and technology is to be used to develop the front end and back end of the proposed system.

1. Front End

The researchers have deemed PHP, HTML, and JavaScript programming languages to develop and design the user-friendly interface. The combination of these will provide a modern design without compromising usability. With scripting, validation of data before processing of the backend will be

possible producing less processing waste. The use of these languages can enable integration to the back end easier and more secure.

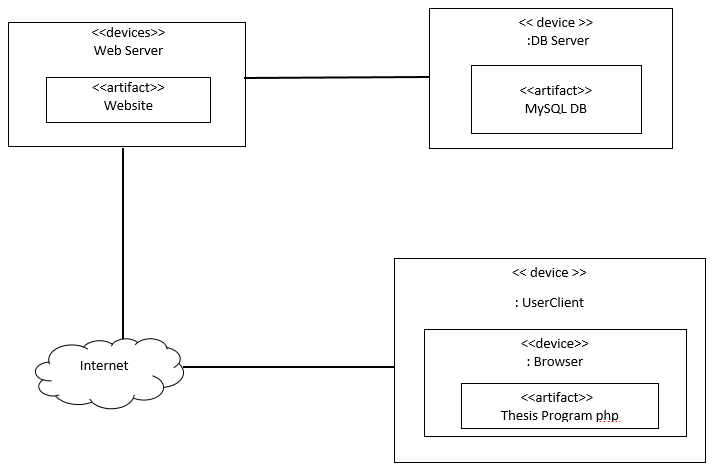
1. Back End

The researchers have used MySQL for the database implementation and PHP, with its server-side functionality to query to the database. Through PHP, requests sent by the front end will be stored in secure variables for posting and requesting results to MySQL. The script in the front end will handle the results stored in PHP variables for display to the user.

**Deployment Diagram**

The researchers used deployment diagram to illustrate the physical aspect of a web-based system on the proposed system; it involves modeling hardware configuration with the software components that lived on.

Deployment Diagram



Verification, Validation and Testing

In this phase, the researchers will test the developed system to verify if the modules will perform correctly without bugs and validate depending on the objectives.

Test Plan

The researchers have used a test plan which is designed to make it easy to collect and organize test cases as well as capture test results. All test results are achieved accordingly.

Test Case

To determine whether the system meets its required functions without errors or faults, a test case template to help find problems in the requirements or design of the proposed system.

Test Data Sample

|  |  |
| --- | --- |
| Elements | Description |
| ID | Id Number of the Case |
| Date Created | Date of Creation |
| Tested by | Name of Tester |
| Date of Tested | Date of Testing |
| Test Specification | Hardware/ Software/ Networks in which the test is executed |
| Objective | Objective of the test cases |
| Test procedure | Step by step procedure to execute the test |
| Result | The Result of the tested system |
| Status | Pass or Failed |
| Remark | comments |

Unit Testing

Unit testing is done in order to show that each part of the program as well as the methods or functions is correct in terms of the requirement and computer specifications. In direct accordance to what it is described.

Integration Testing

The researchers performed the integration testing in order to detect any inconsistencies between the software units or parts of the system that are integrated together and to determine if these parts function correctly.

Compatibility Testing

This is used to ensure compatibility of the system with other subjects such as computer hardware, internet speed, or type of browser. Safari, Mozilla Firefox, Google Chrome, Opera Browser, and Microsoft Edge in their latest versions were tested in each system.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| LAPTOP | | | | |
| Test Environment | Test  Procedure | Expected  Results | Actual  Result | Status |
| CPU: Intel Core i3  RAM: 4GB  OS: Windows 10 64 bit | 1. Install System 2. Run System 3. Login 4. Run the modules | The system will run without encountering any problems | The result is as expected | Passed |
| CPU: Intel Pentium G40  RAM: 4GB  OS: Windows 7 64 bit | 1. Install System 2. Run System 3. Login 4. Run the modules | The system will run without encountering any problems | The result is as expected | Passed |
| CPU: Intel i5 2400  RAM: 6GB  OS: Windows 10 64 bit | 1. Install System 2. Run System 3. Login   Run the modules | The system will run without encountering any problems | The result is as expected | Passed |
| CPU: AMD A6 - 7310  RAM: 4GB  OS: Windows 10 64 bit | 1. Install System 2. Run System 3. Login   Run the modules | The system will run without encountering any problems | The result is as expected | Passed |
| Desktop | | | | |
| CPU: Intel Pentium G4560  RAM: 4GB  OS: Windows 10 64 bit | 1. Install System 2. Run System 3. Login   Run the modules | The system will run without encountering any problems | The result is as expected | Passed |
| CPU: Intel i5 2400  RAM: 16GB  OS: Windows 10 64 bit | 1. Install System 2. Run System 3. Login   Run the modules | The system will run without encountering any problems | The result is as expected | Passed |
| CPU: i5 7100  RAM: 8GB  OS: Windows 10 64 bit | 1. Install System 2. Run System 3. Login   Run the modules | The system will run without encountering any problems | The result is as expected | Passed |
| CPU: AMD A8-6600k  RAM: 4GB  OS: Windows 10 64bits | 1. Install System 2. Run System 3. Login   Run the modules | The system will run without encountering any problems | The result is as expected | Passed |
| CPU: AMD A6-5400k  RAM: 2GB  OS: Windows 7 32bits | 1. Install System 2. Run System 3. Login 4. Run the modules | The system will run without encountering any problems | The result is as expected | Passed |

Performance Testing

Performance testing is used to identify the performance issues of the proposed system. The response time is averaged from 6 tests and shown as less than 1 second for better interpretation by the users.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Environment | | CPU: Intel Core i3 2400  RAM: 4GB  OS: Windows 10 64bit  Internet Speed: 1mbps | | |
| Prerequisites | Test Procedure | Expected Result | Actual Result | Status |
| Owner | Load registration page | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Save registration details to database | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Load login page | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Input login credentials and proceed to index | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Load floor page | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Save floor to database | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Load unit summary | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | View tenants currently renting in the unit | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Delete unit in unit summary | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Load add unit page | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Add unit to database | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Load tenant summary page | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | View tenant profile in tenant summary | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Delete tenant profile in tenant summary | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Load add tenant page | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Save tenant profile to database | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Load bill page | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Generate bill interface of selected tenant | Expected response time less than 4 seconds | Expected response time less than 2 seconds | Passed |
| Owner | Load current profile page | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Save updates in current profile | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Owner | Logout current session | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Tenant | Load login page | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Tenant | View bills page | Expected response time less than 4 seconds | Expected response time less than 2 seconds | Passed |
| Tenant | Load current profile page | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |
| Tenant | Update current profile changes and save to database | Expected response time less than 2 seconds | Actual is less than 1 second | Passed |

Stress Testing

Stress testing is used to determine the stability of the system and its efficiency in utilizing resources available for use. The researcher will perform different scenarios to simulate light multitasking tasks of the users.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID Number: | TCN001 | Date Created: | | 15-Sept-17 | |
| Created By: | Imran | Date Tested: | | 16-Sept-17 | |
| Tested By: | Jake | | | | |
| Objectives: | * To test the response of the proposed system behavior when the computer is under load. * To test system feedback under slow internet connection. | | | | |
| Scenario | Test Procedure | | Results | | Status |
| Running usual office applications prior to using the system under the following specifications:  CPU: Intel i3-2365m  RAM: 4gb  HDD: 500gb 7200 rpm  GPU: Intel HD Graphics 2000  Internet Connection Speed: 1mbps | * Open Microsoft Word 2017 * Open Adobe Photoshop CC 2016 * Open Google Chrome with 5 tabs open (YouTube, Google, Facebook, 9gag, Reddit) * Open the system | | CPU: level is 90% and below,  RAM: level is 91% and below,  Disk: level is 2%  Network: 100% Utilization | | Passed |

Load Testing

In this process the researchers tested the response of the proposed system by applying the normal to extensive actions towards the system.

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case ID Number: | TCN002 | Date Created: | 15-Sept-17 |
| Created by: | Imran | Date Tested: | 16-Sept-17 |
| Tested by: | Daniel | | |
| Objective: | To test the proposed system performance by identifying the percentage of each metric. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Percentage | Test Procedures | Results | Status |
| CPU Usage  (Processor) | Open application.  Log-in.  Click some buttons. | The result is 21% and below | Acceptable |
| RAM | Open application.  Log-in.  Click some buttons. | The result is 20% and below excluding native system memory usage | Acceptable |
| Network Utilization | Open application.  Log-in.  Click some buttons. | The result is 20% and below | Acceptable |

System Testing

This test involves the system as a whole used as how the owner or the tenants would utilize it. It was tested to comply with the specified requirements requested by the client. This phase also revealed flaws unnoticed to prior testing. Its functionality’s weaknesses are out flawed in this testing.

Acceptance Testing

This test is all about what the user will think of the system and how they will embrace it to their company. This is to evaluate the compliance of the system’s functionality to the needs of the user. This determines whether the system meets the criteria set by the end user.

It is in this phase that will determine whether the system is likable or not. Since it is made for the client, he must be in full favor of all its functions, processes, and security.

The following types of testing are used in this phase:

1. Internal Acceptance Testing (Alpha Testing)
   * The researchers will test the system among themselves and determine which is favorable or not. Slight changes are possible in this phase and a keen sight for misspells or the user interface itself to further improve the system.
2. External Acceptance Testing
   * This test is performed outside the researchers. It is used to determine if the system is fit for the would-be users of the system. There are two phases:
     1. Customer Acceptance Testing
        + This is performed by the tenants who are currently renting in the dormitory and who will mainly benefit from the system. Whether they feel that the system improves their stay.
     2. User Acceptance Testing
        + This is performed by the owner of the dormitory for each branch. He will check for the overall functions of the program with regards to his standards. This is the Beta phase.

**Conclusions**

This study will introduce a revolutionary way of automation even in micro enterprises when put to use. After crucial investigation and research, it is essential to finalize that the implementation of the Automated Billing to Centralized Dormitories (ABCD) to any dormitories will benefit the business and the people involved in it with a potential to guarantee increase in profit. This, by ensuring a non-biased computational billing.

Very minimal training is required to use the system and therefore it is very easy to use its functionality. If done correctly, the system will be able to minimize the time for calculating the bills of each tenant involved with unbeatable accuracy and lessen the strain being posed on the manager upon making the calculations manually. This eliminates the manually-dependent operations of the company which will make the process faster without sacrificing correctness.

The researchers have successfully designed a secure, reliable, efficient, and mobile system for the billing operations of any dormitories. This system is technologically and economically feasible in the Philippine setting of business management based on data and observations gathered mainly in the company studied. The study has shown that infusing the system online and integrating internet functionality to it will make it mobile which removes the constraint of controlling the system only in a specific place. By introducing mobility, the system functionality has been improved greatly for the benefit of the user of the system which is the manager or the owner.

**Recommendations**

The proper use of the Automated Billing to Centralized Dormitories inclines to the user, the manager. It is essential to keep his account credentials to himself for security purposes. No breach in the system will occur unless his credentials are compromised. Besides securing his credentials to himself it is also important to be creative in his use for his password and username combination. This involves the use of numbers, special characters and a mix of uppercase and lowercase letters. Security above all.

When the user encounters any problems, it is advised to contact the researchers for proper action handling.

Because of the simplicity of the system, there is no need for a training of the user. A brief instruction on how to use it will suffice.

**Implementation Plan**

Project Implementation Checklist

Project implementation is used to carry out the following activities that follow.

Employee training. An authorized personnel will be trained on how to use the system properly before full implementation of the system. This is to ensure full system functionality and integration to the dormitory.

Data conversion. Despite the automation of the system. Manual inputs of the data must be in check since inconsistencies such as misspells of the tenant’s name to incorrect payment amount placements. Although these can be monitored and corrected in the future, getting it right the first time is the most productive.

Implementation Contingency

The system will be monitored by the researchers long after its deployment. But still, there are future risks of uncertain events that may occur which will threaten the system. In this case, a contingency plan was constructed.

The following are the possible problems that may arise and procedures towards fixing them.

* The system may be inaccessible or unreachable
  1. Contact the developers of the system.
* Inaccessible user account
  1. Enter “Forgot Password?” link and fill the required data.
  2. If step 1 does not work, contact the developers of the system.
* In extreme cases, multiple accounts can relate to one user
  1. Contact the developers of the system.

Infrastructure/Deployment

The system is web based. This would need a web hosting site with MySQL database support and PHP scripting. The hosting company must be able to cater multiple user visits from the domain registered with no visitor count limit. The hosting company must also have systems in place for security and database backups in case of any errors in their system. Also, the hosting company’s pricing must be competitive enough but still provide the aforementioned features reliably.

Taking all of these into account, the system will be hosted by HostGator.com under the domain name abcdormitory.com.

**BIBLIOGRAPHY**

* Turban, E. King, D. McKay, J. Marshall, P. Lee, J & Vielhand, D. (2008). Electronic Commerce 2008: A Managerial Perspective. London: Pearson Education Ltd. p.550
* Turban, E. King, D. McKay, J. Marshall, P. Lee, J & Vielhand, D. (2008). Electronic Commerce 2008: A Managerial Perspective. London: Pearson Education Ltd. p.554
* [Mastercard: Security Rules and Procedures-Merchant Edition](http://www.mastercard.com/us/merchant/pdf/SPME-Entire_Manual_public.pdf) ([PDF](https://en.wikipedia.org/wiki/Portable_Document_Format)). 2009. Retrieved: May 12, 2009
* [*Rise of the Network Society*](http://www.wiley.com/WileyCDA/WileyTitle/productCd-1405196866.html), Manual Castells, Wiley-Blackwell, 1996 (1st ed) and 2009 (2nd ed), [ISBN 978-1-4051-9686-4](https://en.wikipedia.org/wiki/Special:BookSources/9781405196864)
* <https://books.google.co.za/books?id=7OjvOmol3CcC&dq=php&printsec=frontcover&source=bl&ots=1oRkbZ44z&hl=en>**.** Retrieved January 20 2017.
* Automated Documentation Error Detection and Notification Improves Anesthesia Billing Performance. Spring, Stephen F, et al., et al. 1, s.l. : Anensthesiology, 2007, Vol. 106.
* Electronic Reminders Improve Procedure Documentation Compliance and Professional Fee Reimbursement. Kheterpal, Sachin, et al., et al. 3, s.l. : Anesthesia & Analgesia, 2007, Vol. 104.
* Development of a Module for Point-of-care Charge Capture and Submission Using an Anesthesia Information Manaagement System. Reich, David L, et al., et al. 1, s.l. : Anesthesiology, 2006, Vol. 105.
* A Systematic Literature Review of Automated Clinical Coding and Classification Systems. Stanfill, Mary H, et al., et al. s.l. : JAMIA, 2010, Vols. 17:646-651.
* <http://getbootstrap.com/components/>. Retrievede February 1 2017
* <http://php.net/manual/en/intro-whatis.php>. Retrieved February 16 2017
* <https://dev.mysql.com/doc/>. Retrieved February 18 2017
* [*"MySQL 5.7 Release Notes"*](https://dev.mysql.com/doc/relnotes/mysql/5.7/en/). mysql.com*. Retrieved 17 July 2017*.
* [*"Changes in Release 8.0.2 (Development Milestone)"*](http://dev.mysql.com/doc/relnotes/mysql/8.0/en/news-8-0-2.html). MySQL 8.0 Reference Manual. Oracle Corporation. 17 July 2017*. Retrieved 17 July 2017*.
* [*"MySQL: MySQL 8.0 Release Notes"*](https://dev.mysql.com/doc/relnotes/mysql/8.0/en/). dev.mysql.com*. Retrieved 2 August 2017*.

**APPENDICES**

Relevant Source Code

Evaluation Tool

This is a confidence log. The reviewers and client would rate the system on its specific aspects based on the following scale.

5 – Very Confident

4 – Confident

3 – Some Confidence

2 – Little Confidence

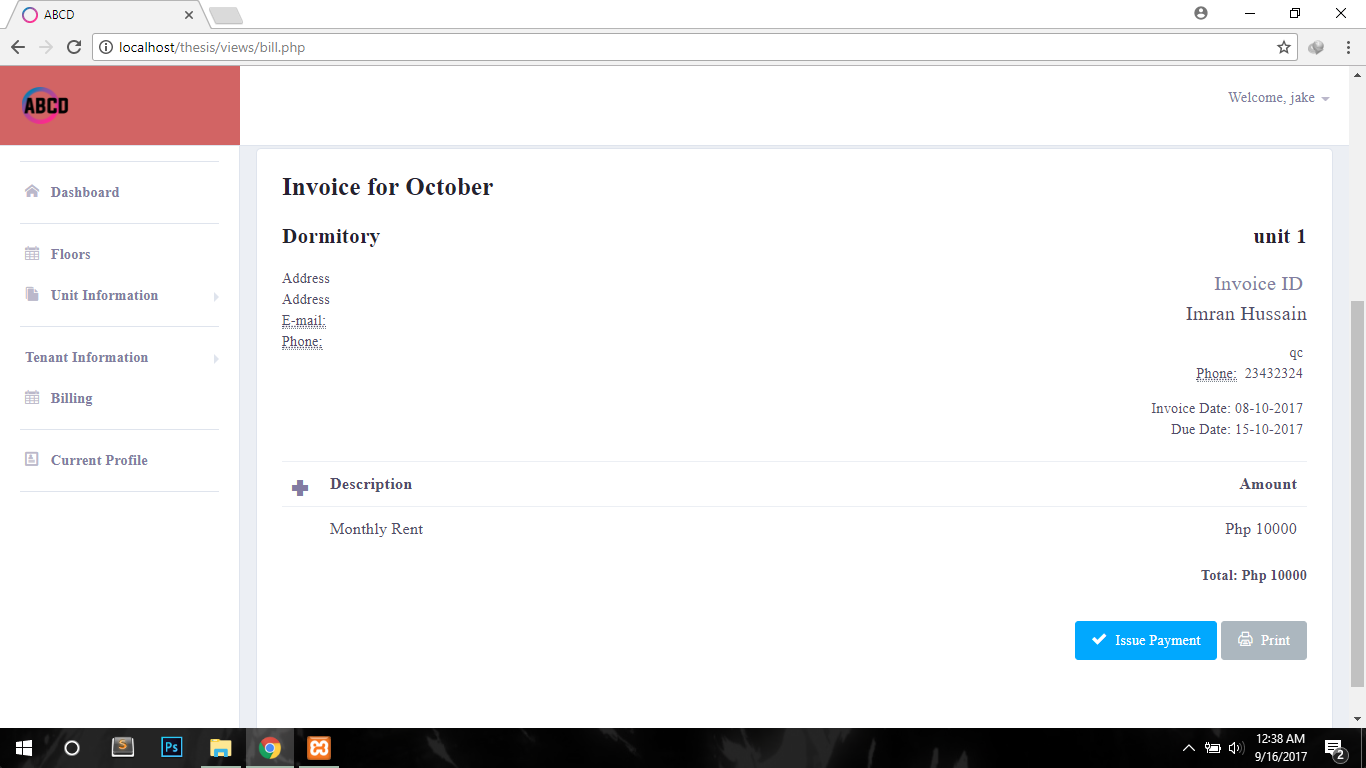
1 – No Confidence

On the following aspects provided, rate the system in the scale provided.

1. Floor management:
2. Unit management:
3. Tenant registration:
4. Tenant billing:
5. Tenant invoicing:
6. Tenant room assignment:
7. Tenant user accounts:
8. User Interface:
9. System security:
10. Overall system functionality:
11. User satisfaction:

Sample Output Reports

Invoice output



**Users Guide**

Before using the system, the following must be noted. These terms have been agreed upon by the developers and the client involved.

1. Upon owner registration, the system will ask for a security question and the answer to that question. This is to be remembered in case of a forgotten password.
2. The length of time of renting is in months and the excess in days. 1 month is measured as 30 days. In order to measure the total duration of renting, the total days are divided by 30 and rounded down and the excess are count as days past the month.
3. On tenant registration, the following are deducted on the sum of the total rent as a whole to lessen the monthly cost of rent:
   1. Discount
   2. Additional Payment
   3. Down payment
4. Tenant information can be viewed and edited as long as they are renting a unit.
5. After expiration of a tenant’s rent duration, his account will remain in the system and can be reactivated by the owner under any available rooms.
6. Floors cannot be deleted once a unit has been added under it.
7. Units cannot be edited or deleted once a tenant has been registered under it.
8. Units can have more than 1 tenants.
9. Units with multiple tenants are charged the same monthly rent excluding their down payments, discounts, and additional payments to ensure fairness.
10. Tenants cannot directly transfer rooms. In order for them to rent another room, one of the following must be fulfilled:
    1. The tenant has finished his rent duration.
    2. The tenant has been deleted and his dues for his current unit has been cleared.
11. Tenants cannot rent more than 1 space at a time. This is for fair use policy in order to cater other students who might want to rent. This is applicable in any circumstance.
12. Tenants are created their own account once registered. Their account credentials will be automatically sent to their emails. The initial username saved is their first name concatenated to the current owner id in the database. Their password goes the same with their last name preceding. The tenants can change their own passwords.
13. Tenant accounts can never be deleted by the owner. Only the developers have the right to do so under request.
14. Floor names and unit names must be unique. No two is allowed to be similar this not case sensitive.
15. If any unwanted circumstance arises, contact the developers.

**Curriculum Vitae**

**GLOSSARY**