

1. INTRODUCTION

1.1 Objective

The motivation behind the project-

1. To automate the cumbersome process of storage and maintenance of various records in a residential venture.
2. Ease the usage of building amenities (indoor/outdoor).

Building Management System is a Java based web application. This tool facilitates in reducing the physical storage of confidential information. In addition to allowing the occupants to check their maintenance and other dues, this application also tracks the usage of building amenities (Indoor/Outdoor).

It stores the details of visitors coming into the building, along with automatic message based alert to the concerned resident.

Members of the building association are given different access rights and privileges based on their position.

2.PROJECT REQUIREMENTS

2.1 Functional requirements

2.1.1 Front end - Bootstrap, HTML, CSS

2.1.1.1 Bootstrap - Front-end framework

Bootstrap is a web framework that focuses on simplifying the development of informative web pages (as opposed to web apps). The primary purpose of adding it to a web project is to apply Bootstrap's choices of color, size, font and layout to that project. As such, the primary factor is whether the developers in charge find those choices to their liking. Once added to a project, Bootstrap provides basic style definitions for all HTML elements. The result is a uniform appearance for prose, tables and form elements across web browsers. In addition, developers can take advantage of CSS classes defined in Bootstrap to further customize the appearance of their contents. For example, Bootstrap has provisioned for light- and dark-colored tables, page headings, more prominent pull quotes, and text with a highlight.

Bootstrap also comes with several JavaScript components in the form of jQuery plugins. They provide additional user interface elements such as dialog boxes, tooltips, and carousels. Each Bootstrap component consists of an HTML structure, CSS declarations, and in some cases accompanying JavaScript code. They also extend the functionality of some existing interface elements, including for example an auto-complete function for input fields.

2.1.1.2 HTML

Html, short for hypertext markup language, is the predominant markup language for the creation of web pages. It provides a means to describe the structure of text-based information in a document by denoting certain text as headings, paragraphs, lists, and so on and to supplement that text with interactive forms, embedded images, and other objects. Html can include embedded scripting language code which can affect the behavior of web browsers and other html processors.

Html was originally developed by Tim Berners-lee while at CERN, and popularized by the mosaic browser developed at ncsa. During the course of the 1990s it has blossomed with the explosive growth of the web. During this time, html has been extended in a number of ways. It is preferred to use html for home page and all primary pages and the site.

`<tagname>content</tagname>`

Basic structure of html tags

2.1.1.3 CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a markup language. While most often used to style web pages and interfaces written in HTML and XHTML, the language can be applied to any kind of XML document, including plain XML, SVG and XUL. CSS is a cornerstone specification of the web and almost all web pages use CSS style sheets to describe their presentation. CSS is designed primarily to enable the separation of document content from document presentation, including elements such as layouts, colors and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content.

CSS can also allow the same markup page to be presented in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader and Braille based tactile devices. It can also be used to allow the web page to display differently depending on the screen size or device on which it is being viewed. While the author of a document typically links that document to a CSS file, readers can use a different style sheet, perhaps one on their own computer, to override the one the author has specified. The document to a specific style sheet the default style of the browser will be applied.

2.1.2 Database - MySQL

A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds.

Other kinds of data stores can also be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those types of systems.

Nowadays, we use relational database management systems (RDBMS) to store and manage huge volume of data. This is called relational database because all the data is stored in different tables and relations are established using primary keys or other keys known as Foreign Keys.

2.1.3 OS - Windows / Linux

2.1.3.1 Windows

Windows is the most recent version of the operating system from Microsoft. Officially it was released in 2015 and was initially offered free of charge to legitimate users of Windows 7 and Windows 8.1. This new version combines features from those two previous installments to suit the users in a better way for both desktop/laptop computers as well as mobile devices.

The most notable change in Windows 10 is that Microsoft replaced the Start screen tiles from Windows 8, and brought back the Start Menu. They also removed the vertical toolbars (or “charms”) that appeared from the sides of the screen. These changes make this Windows version easier to use for users of both desktop/laptops and mobile devices.

2.1.3.2 Linux

Linux is an open-source operating system. It is like Windows, Mac, Android, etc.

Unix is also an operating system like Linux. It is a commercial OS. It consists of three parts: Kernel, Shell and Programs. Most of the Unix and Linux commands are similar in nature.

Our Linux tutorial includes all topics of Linux OS such as Linux commands, Directories, Files, Man Pages, File Contents, File Permissions, shells, VI editor etc. There is also given Linux interview questions to help you better understand the Linux operating system.

2.1.4 IDE - Eclipse

In the context of computing, Eclipse is an integrated development environment (IDE) for developing applications using the Java programming language and other programming languages such as C/C++, Python, PERL, Ruby etc.

The Eclipse platform which provides the foundation for the Eclipse IDE is composed of plug-ins and is designed to be extensible using additional plug-ins. Developed using Java, the Eclipse platform can be used to develop rich client applications, integrated development environments and other tools. Eclipse can be used as an IDE for any programming language for which a plug-in is available.

The Java Development Tools (JDT) project provides a plug-in that allows Eclipse to be used as a Java IDE, PyDev is a plugin that allows Eclipse to be used as a Python IDE, C/C++ Development Tools (CDT) is a plug-in that allows Eclipse to be used for developing application using C/C++, the Eclipse Scala plug-in allows Eclipse to be used an IDE to develop Scala applications and PHP eclipse is a plug-in to eclipse that provides complete development tool for PHP.

2.2 Technical Requirements

Software Requirements:

- Chosen IDE - Eclipse for Java EE
- Preferred OS - Windows, Linux, Mac

Technology Stack -

- Front-end: Bootstrap 4, CSS 4, HTML 5, Javascript
- Back-end: Advanced Java 8(Servlets, JSP, JDBC)
- Database: MySQL

Hardware Requirements:

- Intel i5 processor
- Screen resolution-1600*900
- 38.5MB RAM

3. SYSTEM DESIGN

3.1 ARCHITECTURE

Architecture diagram is a diagram of a system, in which the principal parts or functions are represented by blocks connected by lines that show the relationships of the blocks. The block diagram is typically used for a higher level, less detailed description aimed more at understanding the overall concepts and less at understanding the details of implementation.

3.2 UNIFIED MODELING LANGUAGE (UML)

The unified modeling is a standard language for specifying, visualizing, constructing and documenting the system and its components is a graphical language which provides a vocabulary and set of semantics and rules. The UML focuses on the conceptual and physical representation of the system. It captures the decisions and understandings about systems that must be constructed. It is used to understand, design, configure and control information about the systems.

Depending on the development culture, some of these artifacts are treated more or less formally than others. Such artifacts are not only the deliverables of a project; they are also critical in controlling, measuring, and communicating about a system during its development and after its deployment. The UML addresses the documentation of a system's architecture and all of its details. The UML also provides a language for expressing requirements and for tests. Finally, the UML provides a language for modeling the activities of project planning and release management. It is a very important part of developing object oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

Building blocks of UML:

The vocabulary of the UML encompasses three kinds of building blocks:

- Things

- Relationships
- Diagrams

Things are the abstractions that are first-class citizens in a model; relationships tie these things together; diagrams group interesting collections of things.

Things in the UML:

There are four kinds of things in the UML:

- Structural things
- Behavioral things
- Grouping things
- Annotational things

Structural things are the nouns of UML models. The structural things used in the project design are:

First, a **class** is a description of a set of objects that share the same attributes, operations, relationships and semantics.

Window
Origin Size
open() close() move() display()

Fig 1: Classes

Second, a **use case** is a description of a set of sequence of actions that a system performs that yields an observable result of value to a particular actor.

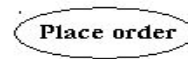


Fig 2: Use Cases

Third, a node is a physical element that exists at runtime and represents a computational resource, generally having at least some memory and often processing capability.

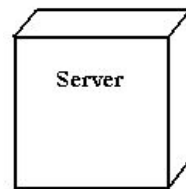


Fig 3: Nodes

Behavioral things are the dynamic parts of UML models. The behavioral thing used is:

Interaction:

An interaction is a behavior that comprises a set of messages exchanged among a set of objects within a particular context to accomplish a specific purpose. An interaction involves a number of other elements, including messages, action sequences (the behavior invoked by a message, and links (the connection between objects).



Fig 4: Messages

Relationships in the UML:

There are four kinds of relationships in the UML:

- Dependency
- Association
- Generalization
- Realization

A **dependency** is a semantic relationship between two things in which a change to one thing may affect the semantics of the other thing (the dependent thing).



Fig 5: Dependency

An **association** is a structural relationship that describes a set of links, a link being a connection among objects. Aggregation is a special kind of association, representing a structural relationship between a whole and its parts.



Fig 6: Association

A **generalization** is a specialization/ generalization relationship in which objects of the specialized element (the child) are substitutable for objects of the generalized element (the parent).



Fig 7: Generalization

A **realization** is a semantic relationship between classifiers, where in one classifier specifies a contract that another classifier guarantees to carry out.

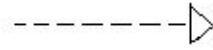
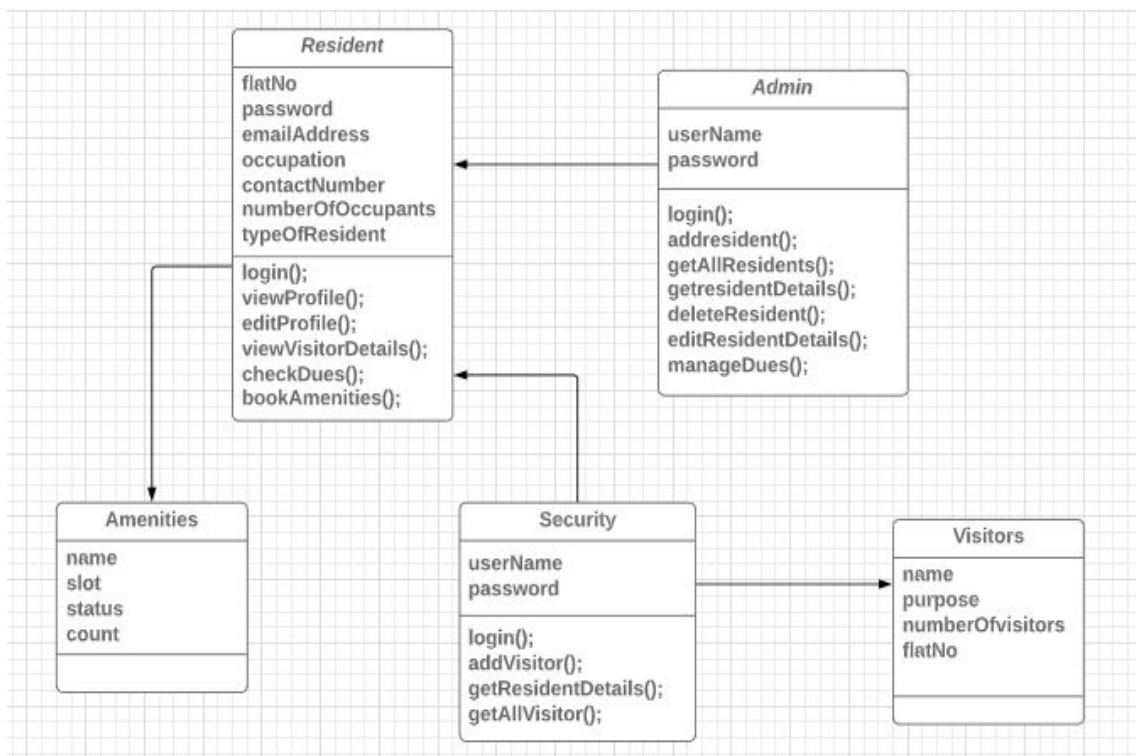


Fig 8: Realization

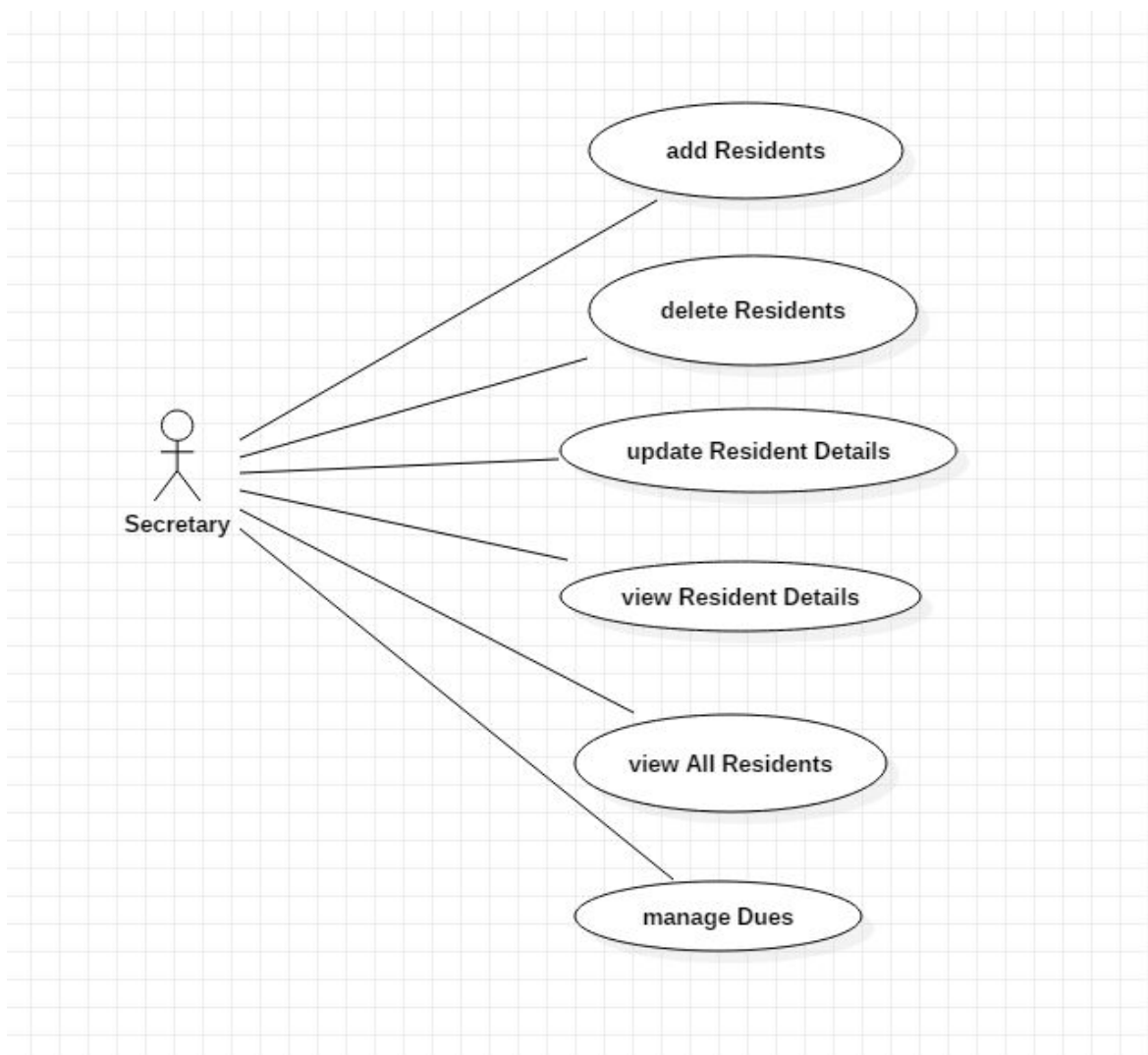
3.3 UML DIAGRAMS FOR BUILDING MANAGEMENT SYSTEM

Class Diagram: It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

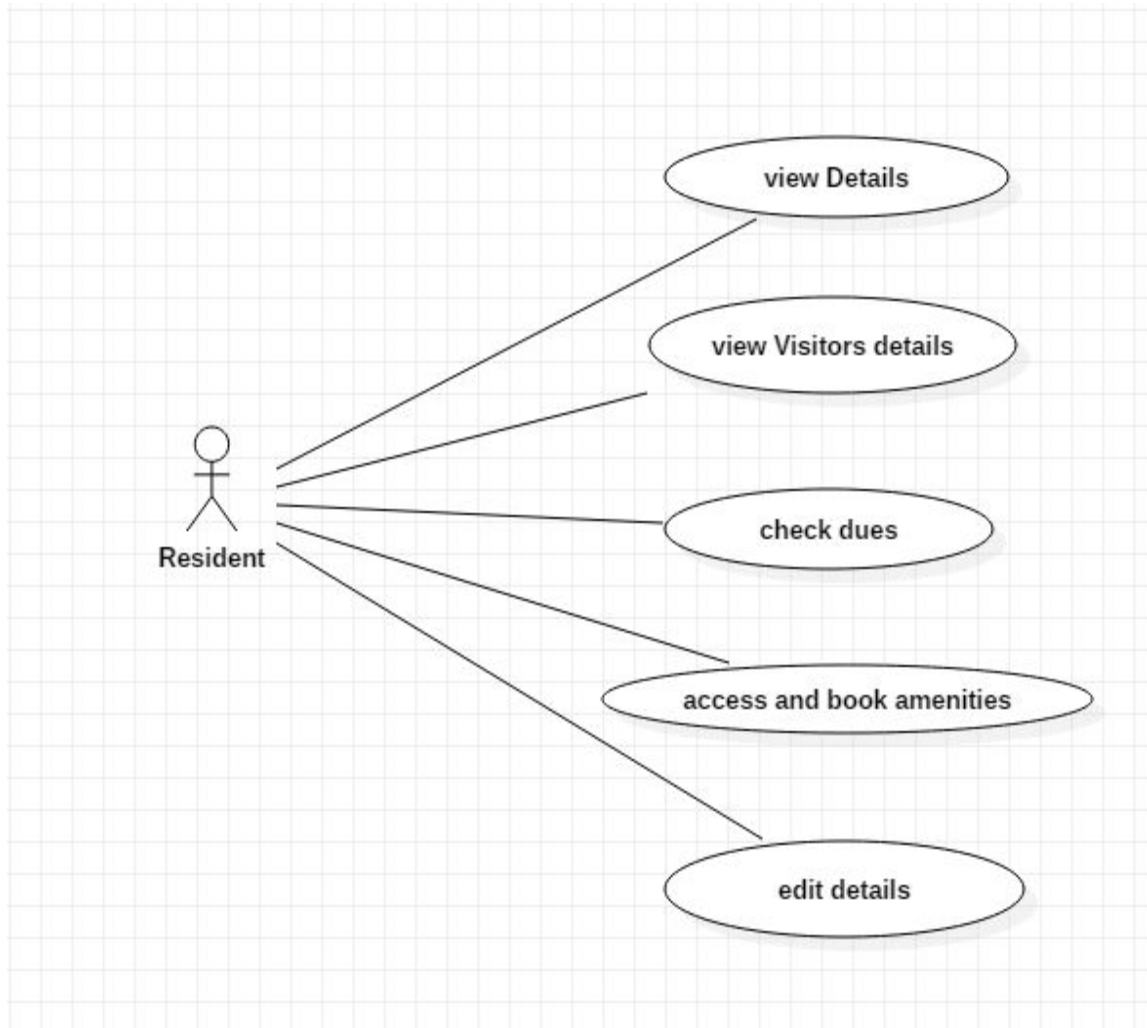


3.1 Class Diagram for Building Management System

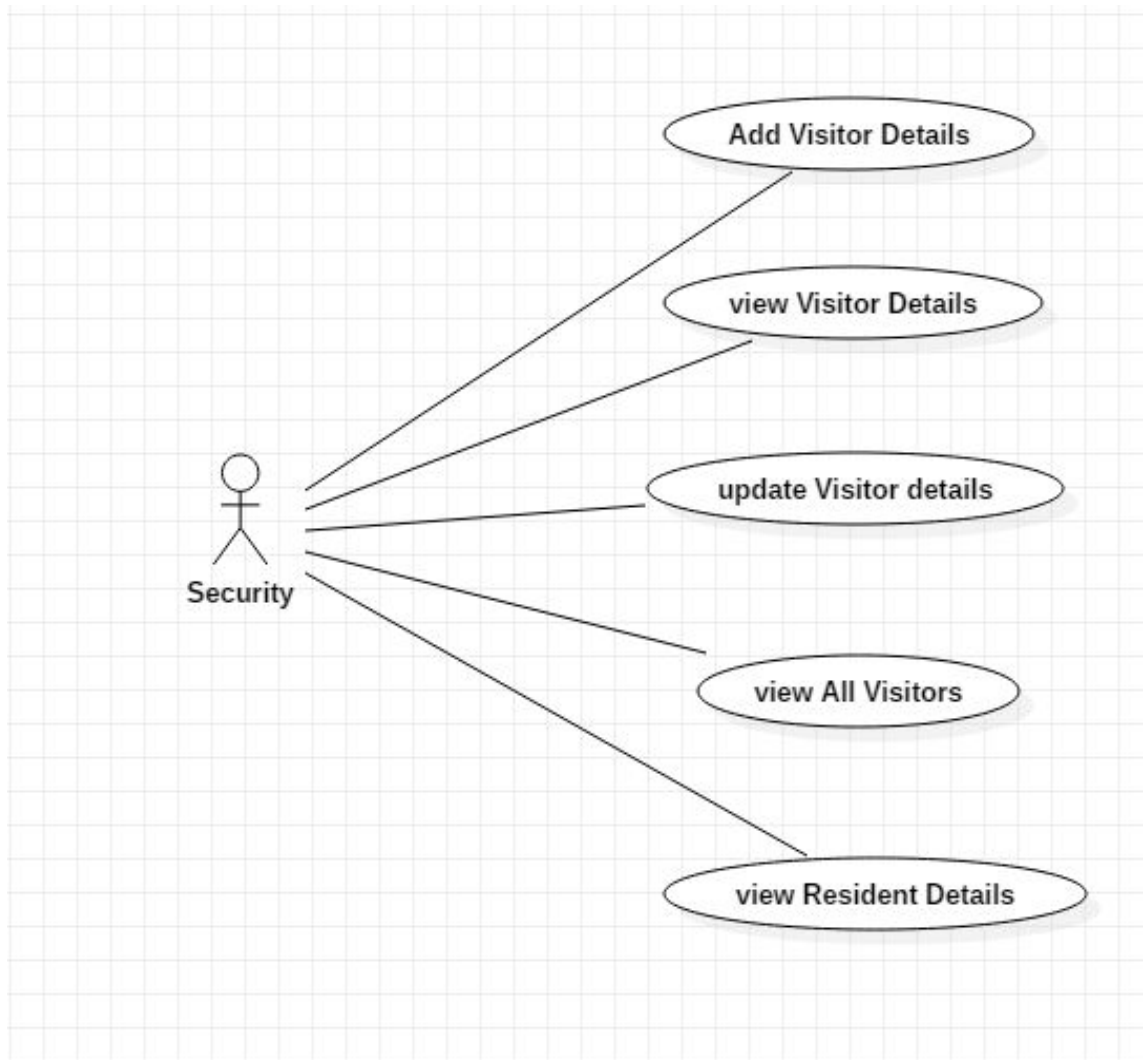
Use Case Diagram: Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified.



3.2 Use Case Diagram for Secretary

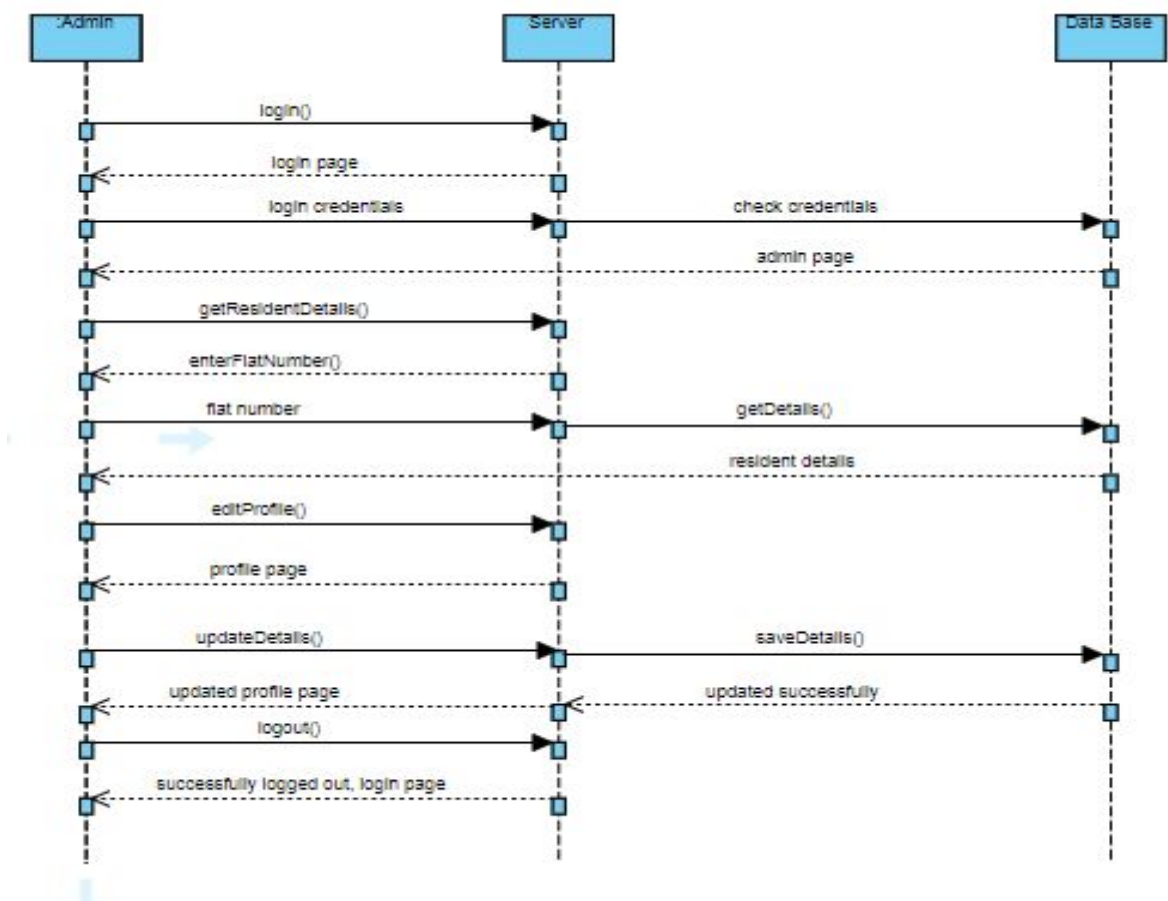


3.3 Use Case Diagram for Resident



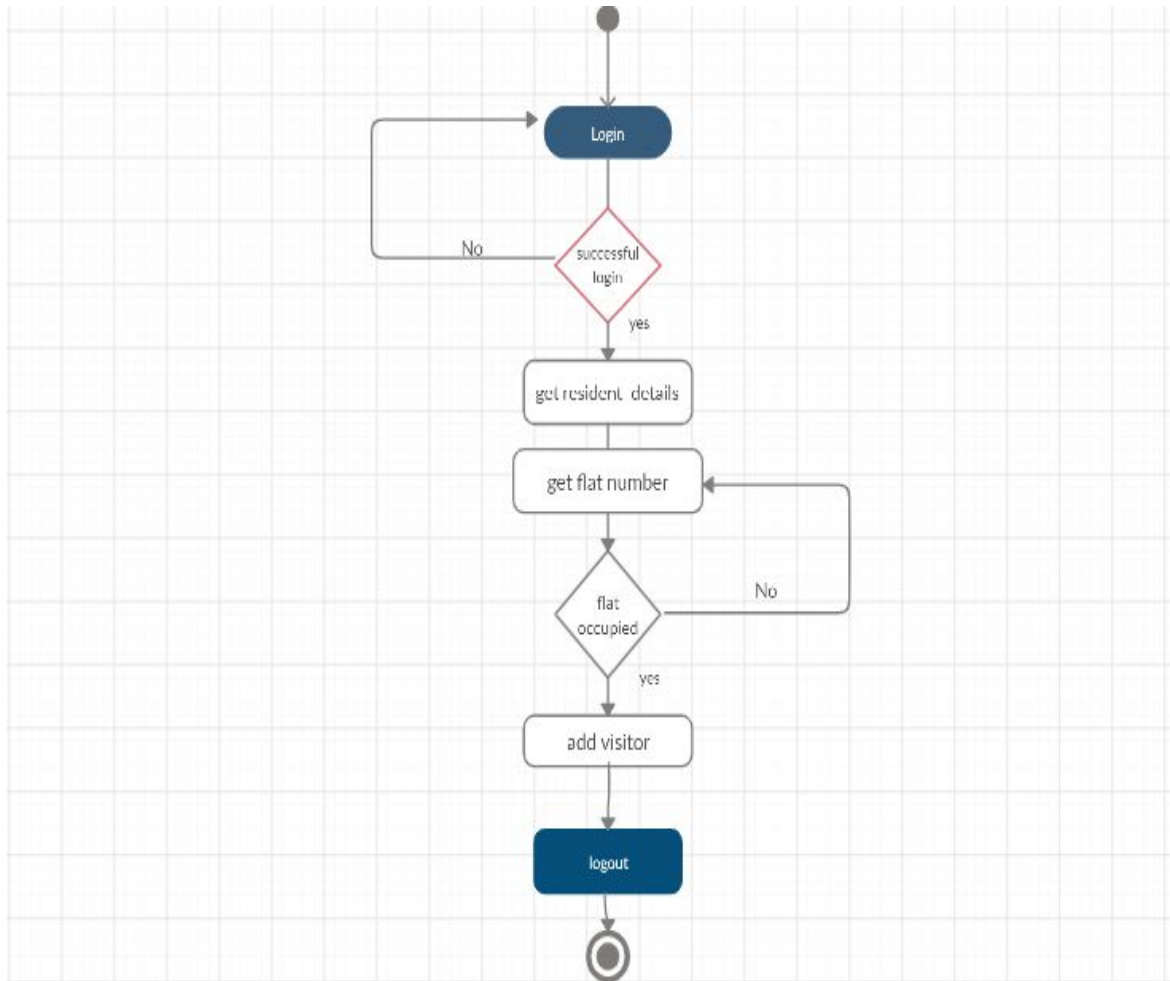
3.4 Use Case Diagram for Security

Sequence Diagram: A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.



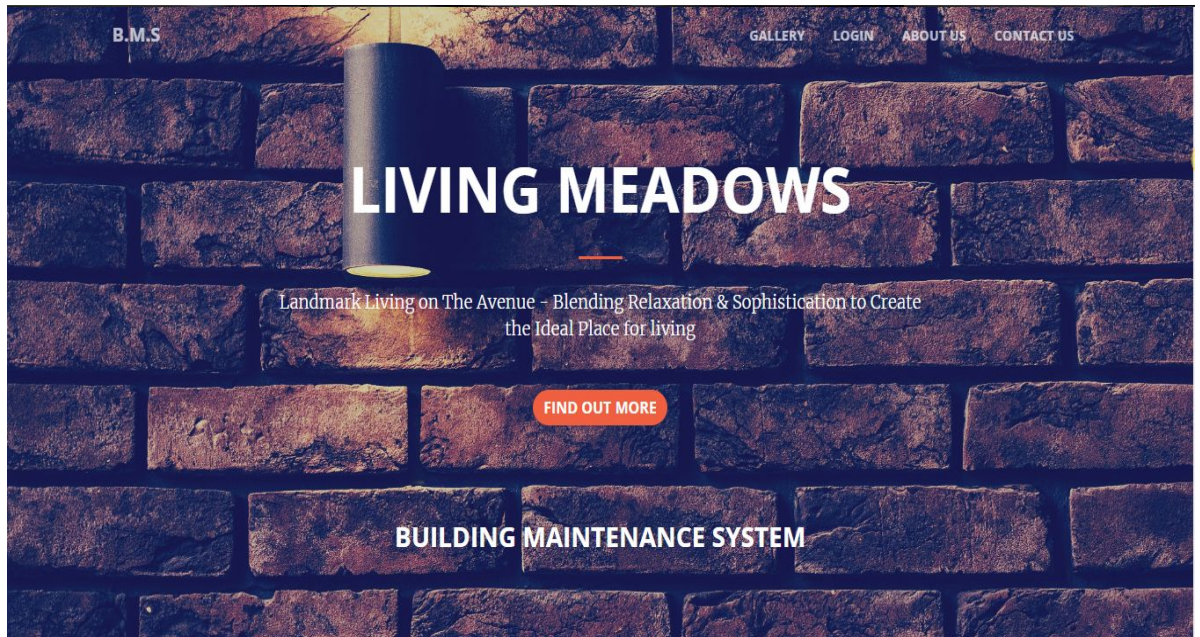
3.5 Sequence Diagram for Secretary editing resident profile

Activity Diagram: The purpose of an activity diagram is to draw the activity flow of a system, to describe the sequence from one activity to another, and to describe the parallel, branched and concurrent flow of the system.

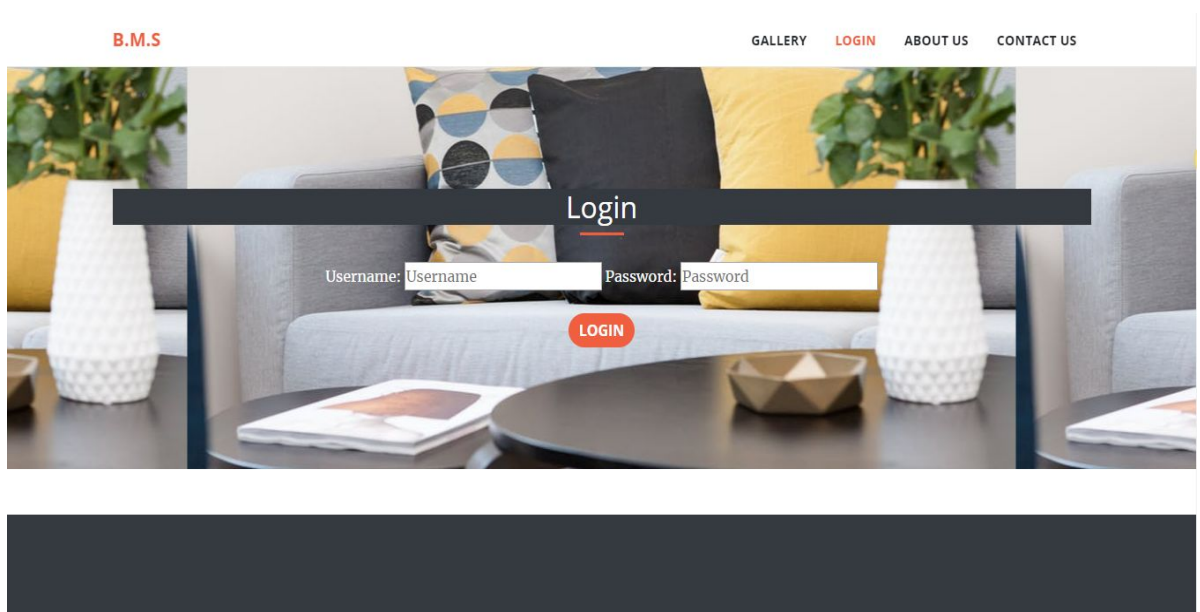


3.6 Activity Diagram for adding visitor for a flat

4. IMPLEMENTATION



4.1 Home page



4.2 Login page

The screenshot displays the Admin home page of the Building Management System. The top navigation bar includes links for '+ Add resident details', 'Display all residents', 'Get resident details', and a 'Log out' button. The main content area is divided into three sections:

- Enter flat number:** A form with a text input labeled 'flat No' and a 'Get Resident details' button.
- Resident details:** A table displaying the details for Flat 100.
- Manage details:** A sidebar with buttons for 'Edit Profile', 'Delete Resident', 'Fill Bill details', and 'Check Dues'.

Resident details	
Flat Number	100
UserName	Kim Tae hyung
No Of Occupants	2
Occupant Contact No	9515292384
Email	kimchTae@gmail.com
Type Of Resident	owner
Occupation	singer
Native	southkorea
Owner Contact No	7036000963

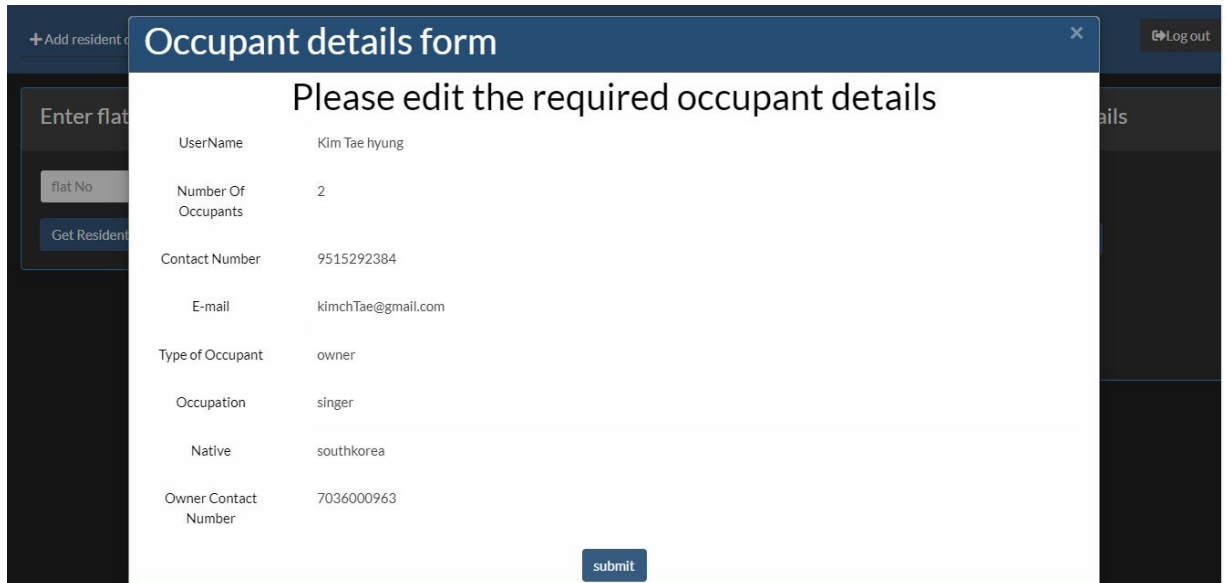
4.3 Admin home page

The screenshot shows the 'Fill resident details' form in the Admin home page. The form is titled 'Fill resident details' and contains the following fields:

- FlatNo
- UserName
- Password
- Number Of Occupants
- Contact Number
- E-mail
- Type of Occupant
- Occupation
- Native

4.4 Resident registration

Admin can register new residents into the application.



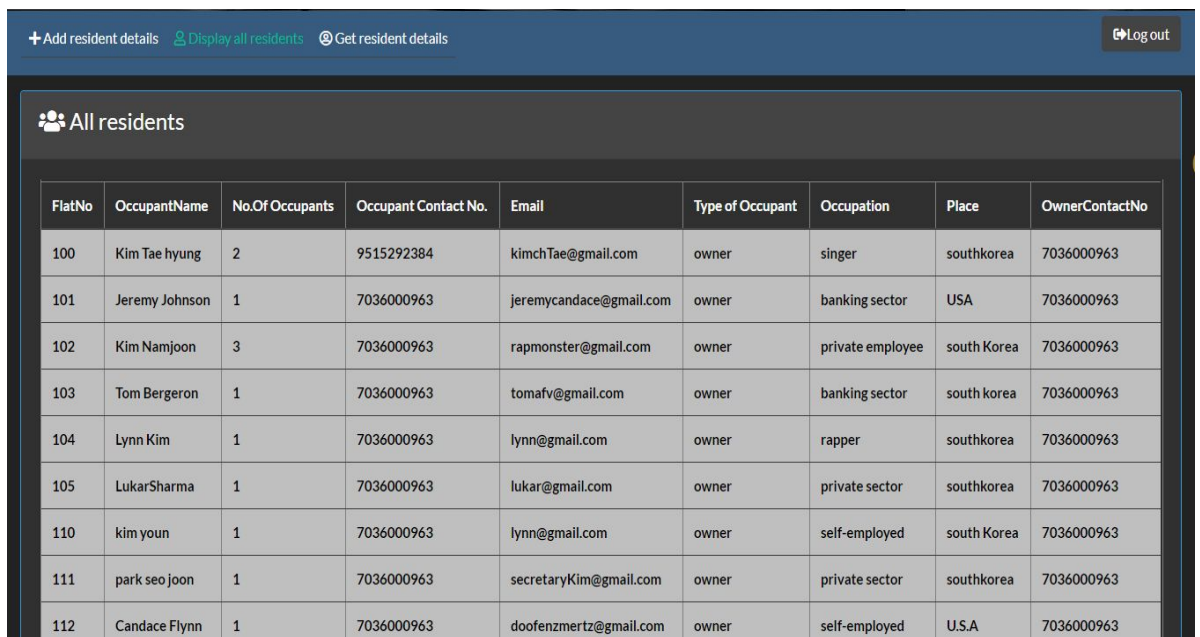
The screenshot shows a web application interface with a dark blue sidebar on the left containing navigation links: '+ Add resident details', 'Enter flat No', and 'Get Resident details'. The main content area features a modal window titled 'Occupant details form' with a close button (X) in the top right corner. Inside the modal, the text 'Please edit the required occupant details' is displayed. Below this text is a form with the following fields and values:

UserName	Kim Tae hyung
Number Of Occupants	2
Contact Number	9515292384
E-mail	kimchTae@gmail.com
Type of Occupant	owner
Occupation	singer
Native	southkorea
Owner Contact Number	7036000963

A 'submit' button is located at the bottom right of the form.

4.5 Edit resident details

Admin can edit the details of resident.



The screenshot shows a web application interface with a dark blue sidebar on the left containing navigation links: '+ Add resident details', 'Display all residents', and 'Get resident details'. The main content area features a table titled 'All residents' with a table icon in the top left corner. The table has the following columns and data:

FlatNo	OccupantName	No.Of Occupants	Occupant Contact No.	Email	Type of Occupant	Occupation	Place	OwnerContactNo
100	Kim Tae hyung	2	9515292384	kimchTae@gmail.com	owner	singer	southkorea	7036000963
101	Jeremy Johnson	1	7036000963	jeremycandace@gmail.com	owner	banking sector	USA	7036000963
102	Kim Namjoon	3	7036000963	rapmonster@gmail.com	owner	private employee	south Korea	7036000963
103	Tom Bergeron	1	7036000963	tomafv@gmail.com	owner	banking sector	south korea	7036000963
104	Lynn Kim	1	7036000963	lynn@gmail.com	owner	rapper	southkorea	7036000963
105	LukarSharma	1	7036000963	lukar@gmail.com	owner	private sector	southkorea	7036000963
110	kim youn	1	7036000963	lynn@gmail.com	owner	self-employed	south Korea	7036000963
111	park seo Joon	1	7036000963	secretaryKim@gmail.com	owner	private sector	southkorea	7036000963
112	Candace Flynn	1	7036000963	doofenzmertz@gmail.com	owner	self-employed	U.S.A	7036000963

4.6 Residents list

Admin can view all the residents registered into the application.

The screenshot shows the 'PROFILE' page of a Building Management System. The top navigation bar includes 'Amenities', 'Profile' (active), 'Edit Details', 'Dues', and a 'Log out' button. The main content area is titled 'PROFILE' and displays a table of user information.

Fiat Number	100
UserName	Kim Tae hyung
No Of Occupants	2
Occupant Contact No	9515292384
Email	kimchTae@gmail.com
Type Of Resident	owner
Occupation	singer
Native	southkorea
Owner Contact No	7036000963

4.7 Resident homepage

The screenshot shows two side-by-side forms in the Building Management System. The left form is titled 'Edit Profile' and the right form is titled 'Change Password'.

Edit Profile Form:

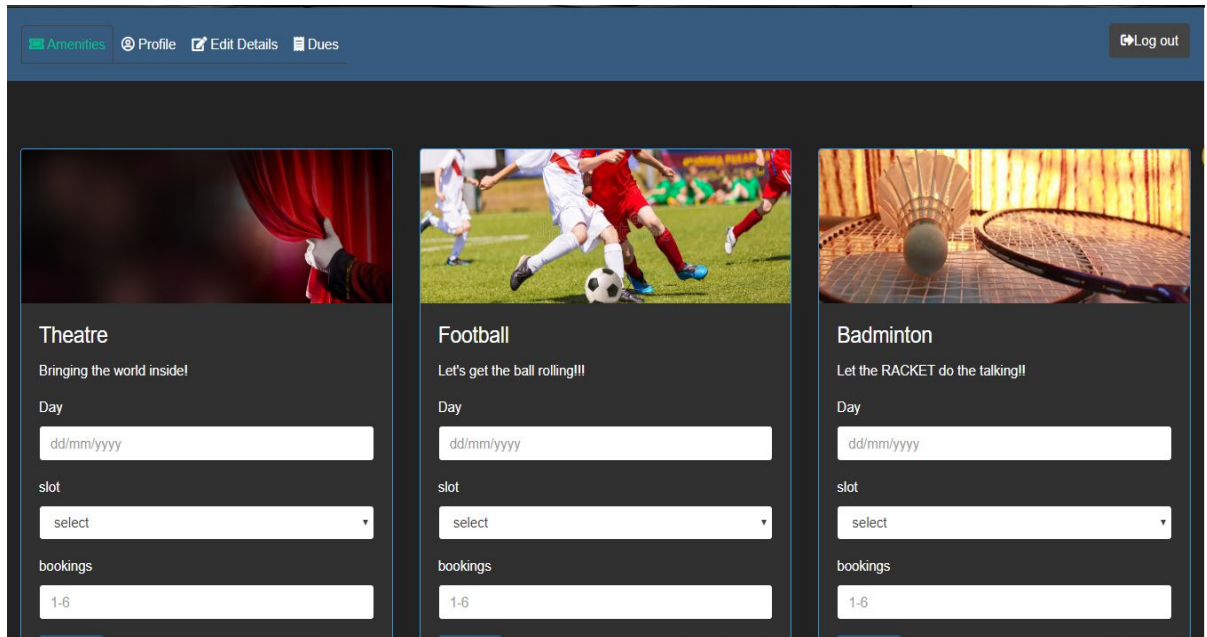
- Email: email@example.com
- Email address: kimchTae@gmail.com
- We'll never share your email with anyone else.
- userName: Kim Tae hyung
- Occupant Contact No: 9515292384
- No Of Occupants: 2
- Occupation: singer

Change Password Form:

- New Password: new password
- Confirm Password: confirm password
- Change Password button

4.8 Edit details

Building Management System



Amenities Profile Edit Details Dues Log out

Theatre

Bringing the world inside!

Day:

slot:

bookings:

Football

Let's get the ball rolling!!!

Day:

slot:

bookings:

Badminton

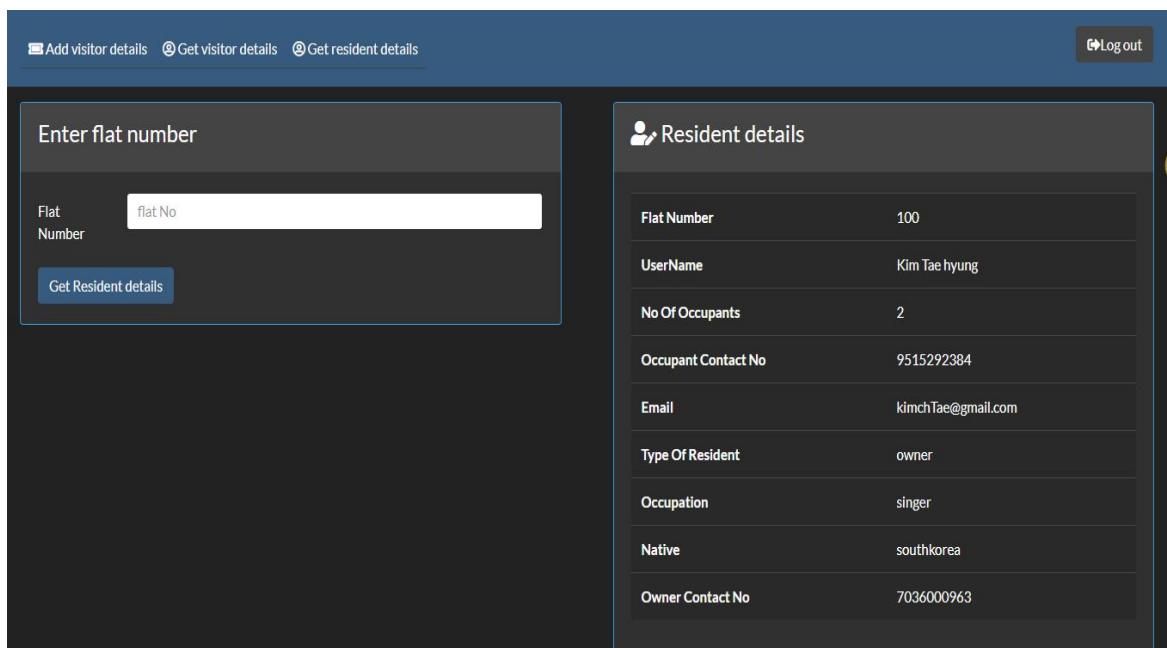
Let the RACKET do the talking!!

Day:

slot:

bookings:

4.9 Amenities page



Add visitor details Get visitor details Get resident details Log out

Enter flat number

Flat Number:

Get Resident details

Resident details

Flat Number	100
UserName	Kim Tae hyung
No Of Occupants	2
Occupant Contact No	9515292384
Email	kimchTae@gmail.com
Type Of Resident	owner
Occupation	singer
Native	southkorea
Owner Contact No	7036000963

4.10 Security homepage

The screenshot shows a web application interface with a dark blue header. The header contains three navigation links: 'Add visitor details' (highlighted with a green icon), 'Get visitor details', and 'Get resident details'. A 'Log out' button is located in the top right corner. The main content area features a form titled 'Fill visitor details' with a person icon. The form includes four input fields: 'Flat Number' (with placeholder 'Flat No'), 'Visitor Name' (with placeholder 'Visitor Name'), 'Number Of Visitors' (with placeholder 'Number of visitors'), and 'Purpose' (with placeholder 'Family/Package delivery/other'). A 'Submit' button is positioned at the bottom of the form.

4.11 Add visitors

The screenshot displays the 'Visitor details' page. The header is identical to the previous page. The left sidebar contains a section titled 'Enter flat number' with a 'Flat Number' label and a text input field containing 'flat No'. Below this is a 'Get Visitor details' button. The main content area is titled 'Visitor details' and contains a table with the following data:

Visitor Name	No.Of Visitors	Visit Time	Purpose
john wright	1	2019-10-18 21:06:22.0	family
Eric Nam	1	2019-10-26 20:37:29.0	just like that
lee jong suk	1	2019-11-01 13:49:51.0	Friend
Lee Taemin	1	2019-11-01 14:00:51.0	census record
lee jong suk	1	2019-11-01 14:37:50.0	census record
Swapna	1	2019-11-02 11:58:41.0	test
Swapna mam	1	2019-11-02 12:02:36.0	test
Swapna mam	1	2019-11-02 12:04:33.0	test
Swapna mam	1	2019-11-02 12:08:37.0	test

4.12 Visitor details page

Code snippets -

```
import java.sql.Connection;

public class DBConnection {

    public static Connection getConnection() {
        Connection con = null;
        try {
            Class.forName("com.mysql.jdbc.Driver");
            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/bms20","root","mysql123");
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return con;
    }
}
```

4.13 DB Connection code snippet

```
try {
    // Construct data
    String apiKey = "apikey=" + URLEncoder.encode("UgEVRITI+3s-yQoru3YpdRiuFge3GYFJ7h6KLs19tb", "UTF-8");
    String message = "&message=" + URLEncoder.encode(sms_message, "UTF-8");
    String sender = "&sender=" + URLEncoder.encode("TXTLCL", "UTF-8");
    String numbers = "&numbers=" + URLEncoder.encode(ph_number, "UTF-8");

    // Send data
    String data = "https://api.textlocal.in/send/?" + apiKey + numbers + message + sender;
    URL url = new URL(data);
    URLConnection conn = url.openConnection();
    conn.setDoOutput(true);

    // Get the response
    BufferedReader rd = new BufferedReader(new InputStreamReader(conn.getInputStream()));
    String line;
    String sResult="";
    while ((line = rd.readLine()) != null) {
        // Process line...
        sResult=sResult+line+" ";
    }
    rd.close();

    System.out.println(sResult);
} catch (Exception e) {
    System.out.println("Error SMS "+e);
    System.out.println("Error "+e);
}
```

4.14 SMS Gateway code snippet

5. TESTING

5.1 SOFTWARE TESTING

Software testing is a critical element of software quality and assurance and represents an ultimate review of specifications, design and coding. Testing is an exposure of the system to trial input to see whether it produces correct output.

Testing phases

Software testing includes the following:

1. Test activities are determined and test data selected
2. The test is conducted and test results are compared with the expected results.

Testing activities

- **UNIT TESTING:** This finds faults by isolating an individual component using test stubs and drivers and by exercising the components using a test case.
- **INTEGRATION TESTING:** This finds faults by integrating several components together. System testing, which focuses on the complete system, its functional and non-functional requirements and its target environment.

Unit Testing

Unit testing focuses on the building blocks of the software system, that is, objects and subsystems. There are three motivations behind focusing on components. First, unit testing reduces the complexity of the overall test activities, allowing us to focus on smaller units of the system. Unit testing makes it easier to pinpoint and correct faults given that few components are involved in this test. Unit testing allows

parallelism in the testing activities; that is each component can be tested independently of one another.

Integration testing

It detects faults that have not been detected during unit testing, by focusing on small group of components.

5.2 Test Case Design

The design of tests for software and other engineering products can be as challenging as the initial design of the product. Test case methods provide the developer with a systematic approach to testing. Moreover, these methods provide a mechanism that can help to ensure the completeness of tests and provide the highest like hood for uncovering errors in software.

Any Engineered product can be tested in either of the two ways:

- Knowing the specified function that a product has been designed to perform, tests can be conducted. These tests demonstrate whether each function is full operational and at the same time searches for errors in each function.
- Knowing the internal workings of a product, tests can be conducted to ensure that internal operations are performed according to specifications and all internal components hence been adequately exercised.

Test case design methods are divided into two types:

- White-box testing
- Black-box testing

White-box testing

White –box testing, sometimes called glass-box testing is a method that uses the control structure of the procedural design to derive test cases. Using white-box testing methods, the s/w engineer can derive test cases that guarantee that all independent paths within a module have been exercised at least once. Exercise all logical decisions on their true and false sides. Execute all loops at their boundaries

and within their operational bounds. Exercise internal data structures to ensure their validity.

Black-box testing

Black-box testing, also called behavioral testing, focuses on the functional requirements of the s/w. Black-box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements of a program. It is a complementary approach that is likely to uncover a different class of errors that white-box methods could not.

Black-box testing attempted to find errors in the following categories.

- Incorrect or missing functions.
- Interface errors.
- Errors in data structures or external database access.
- Behavior or performance errors.
- Initialization and termination errors.

Black-box testing purposely disregards control structure; attention is focused on information domain. By applying black-box techniques, we derive a set of cases that satisfies the criteria test cases that reduce, by a count that is greater than one, the number of additional test cases that must be designed to achieve reasonable testing. Test cases that tell us something about the presence or absence of classes of errors, rather than an error associated only with the specified test.

5.3 TEST CASES

Test Case 1:

When a user tries to login without submitting the required details it displays “Enter the password” or “Enter the user name”.

Test Case 2:

When the secretary tries to register a resident, Password and Email ID are validated.

Test Case 3:

When the resident tries to book an amenity beyond the specified limit, they are notified of the same.

Test Case 4:

While changing password, the new password and confirm password fields must be the same. Otherwise it is notified that they do not match.

6. CONCLUSION

Ever increasing data and its maintenance nowadays is one of the biggest concerns, and managing it manually is definitely not a good idea. The Building Management System is an attempt to reduce the physical storage of confidential information recorded for the maintenance of a residential venture. Members of the building association are given different access rights based on their position. This tool can definitely facilitate the cumbersome process of building management and give the users a comparatively easy and comfortable experience.

Future enhancements

The possible future enhancements for this application would be -

- To add a payment gateway to pay the dues (maintenance, rent etc.)
- Convert this to an android application for easy access.

7. REFERENCES

S.NO	NAME
1.	https://restfulpi.net
2.	https://www.mulesoft.com/resources/api/restful-api
3.	https://en.m.wikipedia.org/wiki/Web_service
4.	https://getbootstrap.com
5.	https://www.w3schools.com