ATMIYA UNIVERSITY

For the award of the degree of

Bachelor of Science in Information Technology

Semester – VI (2022-2023)

Project Report on

Apartment Visitor Management System

Submitted by

Radadiya Devalkumar Chandulal

Submitted to

Department of Computer Application & Information Technology

Atmiya University

Apartment Visitor Management System (Secure your space, streamline your guests)

A PROJECT SUBMITTED TO

Atmiya University

Department of Computer Application & Information Technology

RAJKOT

Submitted in partial fulfillment of the requirements for the degree of

"Bachelor of Science in Information Technology"

Sem-6

(Year 2022-2023)

Submitted By :-

Devalkumar C. Radadiya

Ridham N. Kakadiya

Guided By :-

Dr. Ripal Ranpara

Mr. Malay Solanki

Acknowledgement

We are very much thankful to our Project guide **Dr. Ripal Ranpara** and **Mr. Malay Solanki**, our Lecturer for his unconditional and unbiased support. He explained about the System and guided us during the entire Project. His constant encouragement and co-operation have been a source of inspiration for us all the time.

And finally, our sincere thanks to our batch-mates, who had provided us with innumerable discussions on many technicalities and friendly tips. Without their cordial and friendly support, this activity had been much tougher. The project has given us immense knowledge to use in our future ventures and many moments to cherish.

We thank **Mr. Divyesh Gohel**, Head of the computer department for their support through the project. Last but not the least; we would like to thank Lab Teachers, friends and our Parents who had helped us indirectly throughout the project and have been the source of encouragement.

Finally, we would like to express our gratitude for the support and encouragement we have received from them.

Most successful feats involve efforts of many people. This Project is no exception. It is our pleasure to take this opportunity to thank all those who helped us directly or indirectly in development of the project. It has prepared us to become a good professional. It requires lot of people's support to complete the project. We take this opportunity to acknowledge their support to us.

We sincerely thank computer department for the academic advancement it has provided to us during the year and finally provided us an opportunity for project work.

Firstly, we would thank Atmiya University, who gave us chance to develop project. It was a nice experience and it helped get idea of the Real time Systems and their development procedure.

Abstract

Apartment Visitor Management System deals with the security provided at society premises from the unauthorized or unwanted visitors and provide entry pass to the regular visitor.

Now a days, in most society visitor management consists of visitors scribbling their name in a paper book.

Instead, Apartment Visitor Management System will assist you the professionalized way in which you welcome your visitors. This software is a complete Visitor Management service to improve the efficiency, productivity and security.

The system provides a user-friendly interface that enables visitors to check-in quickly and easily. Visitors can pre-register their visit using an online portal, reducing wait times and improving the overall visitor experience. The system also allows residents to pre-authorize visitors, so they can quickly gain access to the building.

The apartment visitor management system improves security by verifying the identity of visitors and ensuring they have a legitimate reason to be on the premises. The system generates visitor badges that can be easily identified, and it tracks the movements of visitors throughout the building.

Chapter 1: Introduction

1.1 Project Summary

The apartment visitor management system is a software solution that aims to streamline the visitor check-in process in residential apartments. The system will be accessible through a web portal, which will allow both residents and management staff to easily manage and monitor visitor activity.

The system will allow residents to pre-register their visitors, enabling a smoother check-in experience for their guests. Upon arrival, visitors will be able to sign in using a tablet or computer at the front desk or lobby. The system will automatically verify their identity, print a visitor badge, and notify the resident of their arrival via text message or email.

The management staff will have access to a dashboard that displays real-time visitor data, including the number of visitors, their duration of stay, and their purpose of visit. The system will also allow staff to blacklist unwanted visitors, restrict access to certain areas of the building, and generate reports on visitor activity.

Overall, the apartment visitor management system aims to improve security and convenience for both residents and staff, while providing valuable data and insights into visitor behavior within the building.

1.2 Purpose

1. Security: By using an apartment visitor management system, apartment management can ensure that only authorized visitors are allowed to enter the building. The system also provides a record of visitors, which can be used to track any security incidents or investigations.
2. Convenience: With a visitor management system in place, residents can pre-register their visitors, which saves time and avoids delays in allowing visitors access to the building.
3. Monitoring: An apartment visitor management system allows management to monitor the number of visitors in the building at any given time, which can help prevent overcrowding and ensure compliance with fire safety regulations.
4. Communication: The system allows management to communicate with residents and visitors, providing important information and updates about the building.

1.3 Scope

Apartment Visitor Management System project is developed as a web application and it will work over web. The project Apartment Visitor Management system includes creation of entry pass and storing the details of visitor into the system. The software has the facility to give a unique entry pass id to regular visitor like maids, milkman, washer man etc.

The Apartment Visitor Management System can be entered using a username and password. It is accessible only by an administrator. Only admin can create the pass and add visitors details. The data can be retrieved easily.

The interface is very user-friendly. The data are well protected for admin use and makes the data processing very fast.

Apartment Visitor Management System is powerful, flexible, and easy to use and is designed and developed to deliver real conceivable benefits to societies.

Chapter 2: Literature Review (You used Technology)

- **PHP :**
  - PHP is an "HTML-embedded scripting language" primarily used for dynamic Web applications. The first part of this definition means that PHP code can be interspersed with HTML, making it simple to generate dynamic pieces of Web pages on the fly.

  - As a scripting language, PHP code requires the presence of the PHP processor. PHP code is
    normally run in plain-text scripts that will only run on PHP-enabled computers (conversely
    programming languages can create standalone binary executable files, a.k.a. programs).
  - PHP takes most of its syntax from C, Java, and Perl. It is an open source technology and runs
    on most operating systems and with most Web servers.

- **HTML :**
  - HTML was originated by Tim Berners-Lee.
  - HTML developed a few years ago as a subset of SGM (Standard Generalized Mark-up Language), which is a higher-lever mark-up language that has long been a favourite of the Defence.
  - Any HTML document is also valid for SGML.
  - HTML is a Hyper Text Mark-up Language that is used to develop web pages.
  - HTML is not a programming language like C, C++ and Java etc.
  - It is a cross platform mark-up language that is design to be flexible enough to display text and other elements like graphical on a variety of views.
  - The HTML document Consist of special Tags that are embedded in an ASCII document.

- **Bootstrap :**
  - Originally created by a designer and a developer at Twitter, Bootstrap has become one of the most popular front-end frameworks and open source projects in the world.
  - Bootstrap makes responsive web design a reality.
  - It makes it possible for a web page or app to detect the visitor's screen size and orientation and automatically adapt the display accordingly.

- **JavaScript :**
  - JavaScript is Netscape's cross-platform, object-oriented scripting language. JavaScript is a small, lightweight language.
  - It is not useful as a standalone language, but is designed for easy embedding in other products and applications, such as web browsers.
  - Inside a host environment, JavaScript can be connected to the objects of its environment to provide programmatic control over them.
  - A scripting language for web pages. Scripts written with JavaScript can be embedded into HTML documents. JavaScript is an interpreted language means that scripts executed without preliminary compilation.
  - JavaScript can function as both a procedural and an object oriented language.
  - Core JavaScript contains a core set of objects, such as Array, Date, and Math, and a core set of language elements such as operators, control structures, and statements.

- **AJAX:**
  - Asynchronous JavaScript and XML.
  - AJAX is a technique for creating fast and dynamic web pages.
  - AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes.
  - This means that it is possible to update parts of a web page, without reloading the whole page.

- **CSS :**
  - CSS stands for Cascading Style Sheet.
  - CSS are a series of instruments that specify how text should appear on web pages.
  - You can use CSS to set styles for text, font, border, list and background. When most of us first learn html, we get taught to set the font , face, size, color etc. every time is wasted with CSS, you occurs on a page.
  - That time is wasted with CSS you only occurs have to specify these details once for any elements.

- **MySQLi:**
  - Mysqli provides an object-oriented interface to interact with MySQL databases, which makes it easier to work with and maintain code.
  - Mysqli supports prepared statements, which allows you to execute a SQL statement repeatedly with different parameters, improving performance and security.
  - Multiple statements execution: mysqli allows executing multiple statements in a single query, reducing the number of database calls and improving performance.
  - Mysqli includes several security features such as support for parameterized queries and secure connections, making it less vulnerable to SQL injection attacks.
  - Mysqli supports transactions, which allow you to group a set of SQL statements into a single unit of work that either completes fully or is rolled back if an error occurs.

Chapter 3: Project Management

3.1 Project Planning and scheduling

Define the project scope : Clearly define the scope of the project, including the goals, objectives, and deliverables.

Identify the stakeholders : Identify all the stakeholders involved in the project, such as the property managers, security personnel, and residents.

Create a project plan : Create a detailed project plan that includes all the tasks and activities required to complete the project. Break down the project into smaller tasks and assign deadlines for each task.

Develop a project schedule : Develop a project schedule that outlines the timeline for each task and the overall project completion date. Use tools such as Gantt charts or project management software to create and manage the project schedule.

Allocate resources : Determine the resources required for each task, such as personnel, equipment, and software.

Manage risks: Identify potential risks and develop a risk management plan to mitigate or avoid them.

Monitor and track progress : Monitor the progress of the project and track the actual progress against the planned schedule. Make adjustments as necessary to ensure the project stays on track.

Review and evaluate : Once the project is complete, review and evaluate the project to identify areas for improvement and to ensure that all project goals and objectives were met.

3.1.1 Project Development Approach

- **Flexibility** : An Agile approach is well-suited for projects that require flexibility and adaptability to changing requirements. In the case of an apartment visitor management system, the requirements may evolve over time based on user feedback and changing needs.
- **Iterative development** : Agile methodologies involve iterative development, which means that the system is developed incrementally in small, manageable pieces. This approach allows for continuous feedback and improvement, which is essential for creating an effective visitor management system.
- **Collaboration** : Agile methodologies emphasize collaboration and teamwork, which is important for ensuring that all stakeholders are involved in the development process. In the case of an apartment visitor management system, this may involve collaboration between property managers, residents, and security personnel.
- **Time-to-market** : An Agile approach is designed to deliver software quickly and frequently. This is important for a visitor management system, which needs to be implemented as soon as possible to improve security and streamline visitor management processes.
- **Testing and quality assurance** : Agile methodologies place a strong emphasis on testing and quality assurance, which is critical for ensuring that the system is reliable, secure, and easy to use.

3.1.2 Project Plan

Define project scope - 1 day

Develop project plan - 5 days

Develop project schedule - 3 days

Allocate resources - 2 days

Manage risks - Ongoing

Monitor and track progress - Ongoing

Review and evaluate - 1 day

Week 1:

- Define project scope, identify stakeholders
- Develop project plan
- Develop project schedule

Week 2:

- Allocate resources
- Begin development of Apartment Visitor Management System

Week 3-6:

- Develop and test Apartment Visitor Management System

Week 7:

- Implementation and training

Week 8:

- Ongoing monitoring and tracking of progress
- Risk management as necessary

Week 9:

- Review and evaluation of project

3.1.3 Schedule Representation

1. Break down the project into tasks
    - Identify all the tasks required to complete the project
    - Break down each task into smaller, more manageable sub-tasks

2. Determine the duration of each task
    - Estimate the amount of time required to complete each task
    - Include any dependencies between tasks

3. Determine the start and end dates of each task
    - Determine the start date of each task based on any dependencies and the project timeline
    - Calculate the end date of each task based on the duration of the task and the start date

4. Create the Gantt chart
    - Use a spreadsheet or Gantt chart software to create the chart
    - List all the tasks along the left-hand side of the chart
    - Create a horizontal bar for each task that spans the duration of the task
    - Color-code the bars to indicate the status of each task (e.g., not started, in progress, completed)
    - Include milestones and critical path tasks to highlight important points in the project timeline

5. Update the Gantt chart regularly
    - Update the chart as the project progresses to reflect any changes in task duration, dependencies, or status
    - Use the chart to track progress, identify delays, and adjust the project schedule as necessary

Chapter 4: System Requirements Specification

4.1 User Characteristics

- Level of Security
- Customizable
- Data Reports

4.2 Hardware and Software Requirements

- **Hardware Requirements :**

RAM	2 GB
Hard Disk	160 GB
Processor	1.0 GHz

- **Software Requirements :**

Browser	Google Chrome or any compatible browser
Database Server	SQL server 2008
Operating System	Windows 7 and above

Chapter 5: System Analysis

5.1 Feasibility Study

All projects are feasible when given unlimited resources and infinite time. It is both necessary and prudent to evaluate the feasibility of a project at the earliest possible time. A feasibility study is not warranted for systems in which economic justification is obvious, technical risk is low, few legal problems are expected and no reasonable alternative exists. An estimate is made of whether the identified user needs may be satisfied using current software and hardware technologies. The study will decide if the proposed system will be cost effective from the business point of view and if it can be developed in the given existing budgetary constraints. The feasibility study should be relatively cheap and quick. The result should inform the decision of whether to go ahead with a more detailed analysis.

Feasibility study may be documented as a separated report to higher officials of the top-level management and can be included as an appendix to the system specification. Feasibility and risk analysis is related in many ways. If there is more project risk then the feasibility of producing the quality software is reduced. The study is done in these phases.

- Technical feasibility
- Economic feasibility
- 0perational feasibility
- Behavioural feasibility

5.1.1 Technical Feasibility

This is related to the technicality of the project feasibility if check the cost to conduct a full system investigation, cost of hardware and software.

The apartment management system supports the economic feasibility to a great extends. Development of the system and the cost of hardware and software are not high. This reduces effort and time of us. This makes software economically feasible.

5.1.2 Economical Feasibility

A system that can be developed technically and that will be used, if installed, must be sill good. Always the financial benefits must be equal or exceed the cost. Economic analysis is the most frequently used method for evaluating the effectiveness of a candidate system or more commonly known as cost or benefits analysis.

5.1.3 Operational Feasibility

Proposed systems are beneficial only if they can be turned into information systems. That is it will meet the organizations operating requirements and also checks that whether the system will work when it is developed and installed. Therefore it is understandable that the introduction of a candidate system requires special efforts to educate, sell and train others.

The Apartment Management system supports the operational feasibility to a great extends. The performance of this software is more accurate, more user friendly, effective, error free.

5.2 Functions Of System

5.2.1 Use case Diagram

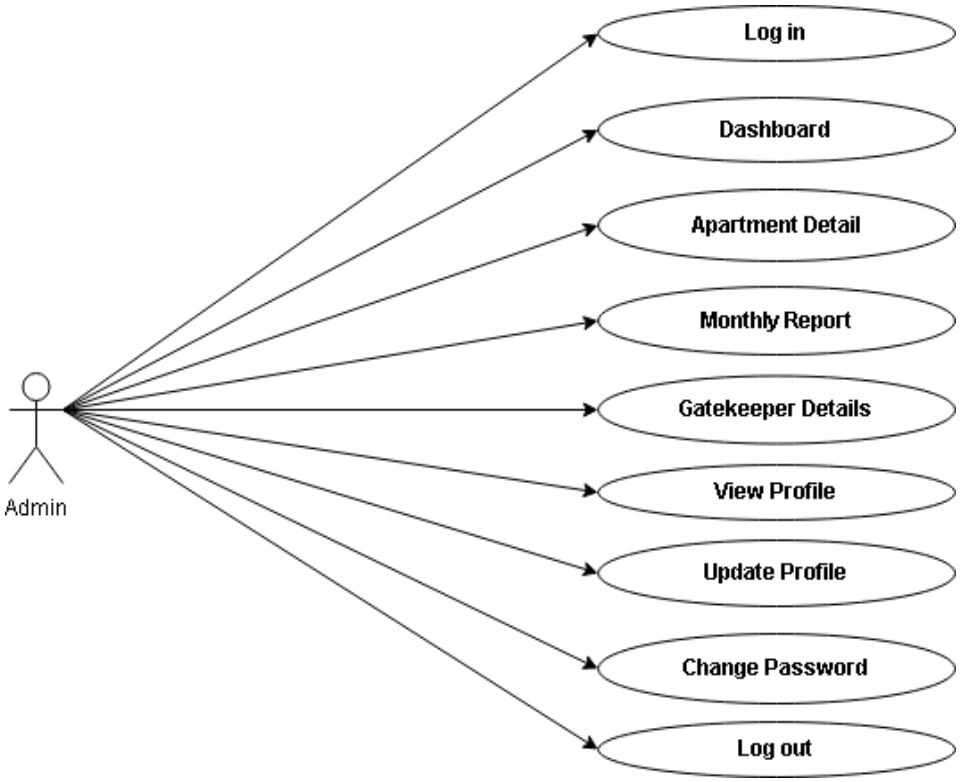
Use case diagrams model behavior within a system and helps the developers understand of what the user require. The stick man represents what's called an actor.

Use case diagram can be useful for getting an overall view of the system and clarifying who can do and more importantly what they can't do.

A Use case is a description of set of sequence of actions. Graphically it is rendered as an ellipse with solid line including only its name. Use case diagram is a behavioral diagram that shows a set of use cases and actors and their relationship. It is an association between the use cases and actors. An actor represents a real-world object. Primary Actor – Sender, Secondary Actor Receiver.

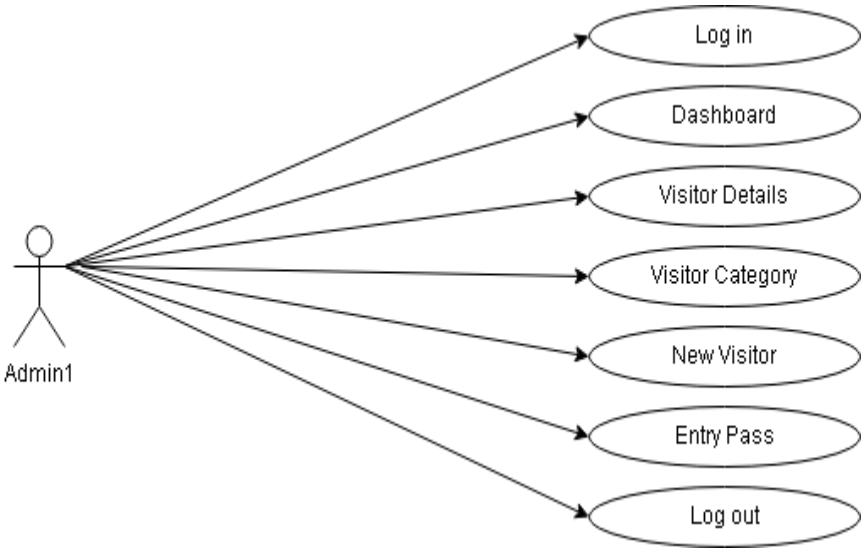
Diagram :

Admin:



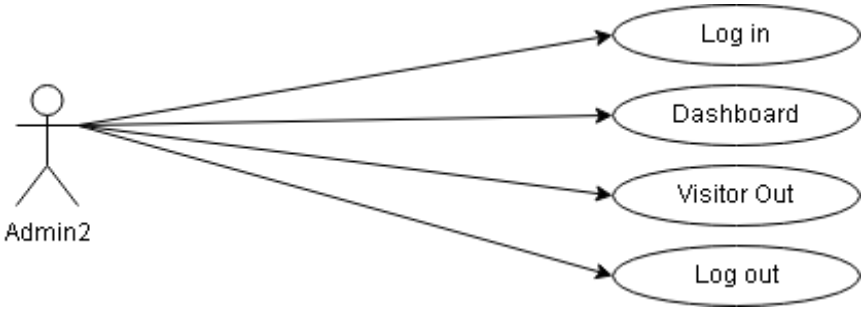
Use Case Diagram for Admin

GatekeeperIn:



Use Case Diagram for Gatekeeper In

GatekeeperOut:



Use Case Diagram for Gatekeeper Out

5.3 Data Modelling

5.3.1 E-R Diagram

E-R diagram stands for **Entity-Relationship** Diagram.

The Entity-Relationship (ER) model was originally proposed by Peter in 1976 [Chen76] as a way to unify the network and relational database views. Simply stated the ER model is a conceptual data model that views the real world as entities and relationships. A basic component of the model is the Entity - Relationship diagram which is used to visually represent data objects.

Since Chen wrote his paper the model has been extended and today it is commonly used for database design for the database designer, the utility of the ER model is:

- It maps well to the relational model. The constructs used in the ER model can easily be transformed into relational tables.
- It is simple and easy to understand with a minimum of training. Therefore, the model can be used by the database designer to communicate the design to the end user.
- In addition, the model can be used as a design plan by the database developer to implement a data model in specific database management software.

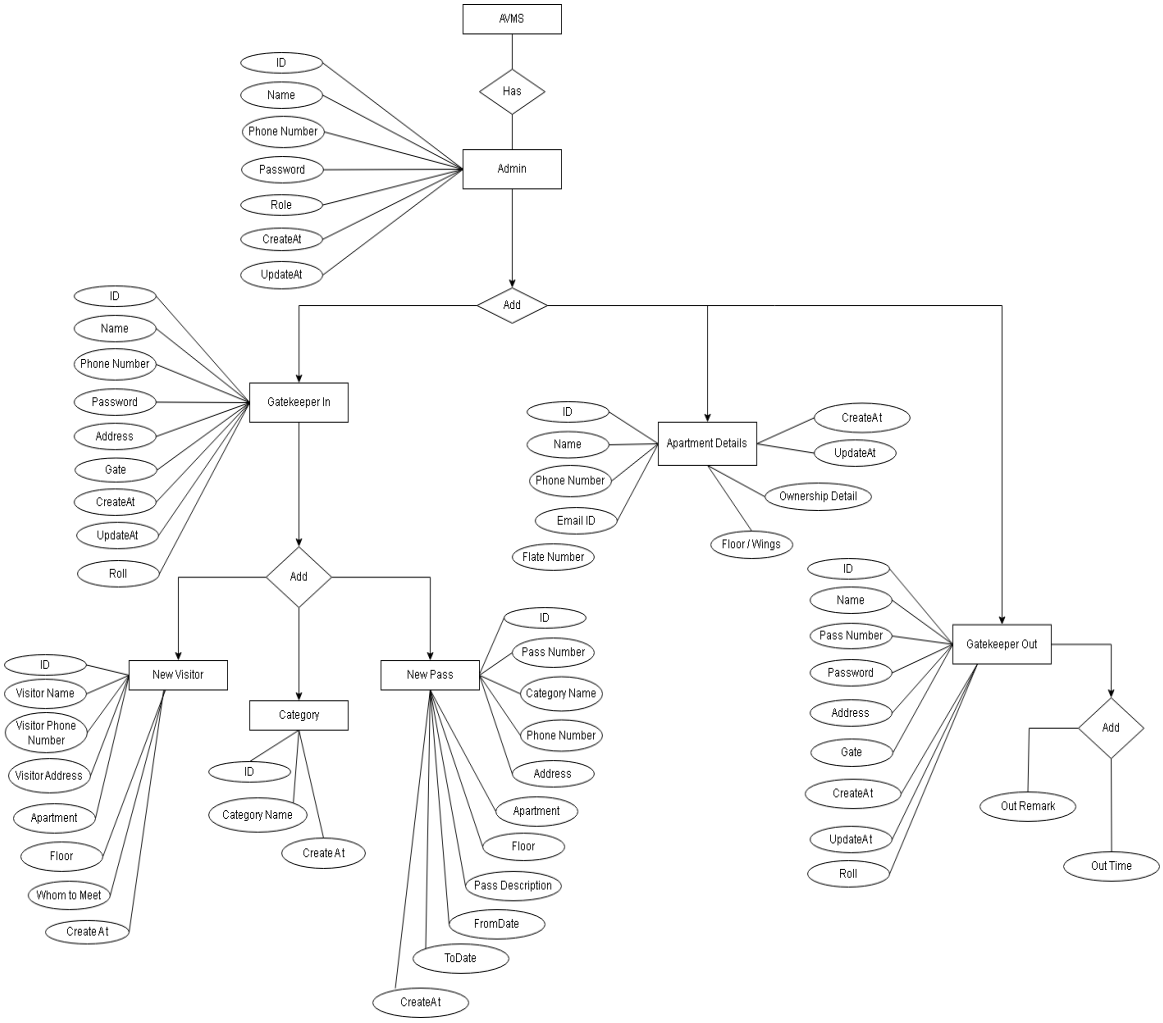
Entities are represented by labeled rectangles. The label is the name of the entity. Entity should be singular nouns.

Relationships are represented by a solid line connecting two entities.

The name of the relationship is written above the line. Relationship names should be verbs.

Attributes, when included, are listed inside the entity rectangle. Attributes which are identifiers are underlined. Attribute names should be singular nouns.

- E-R Diagram



Entity-Relation Diagram

5.3.2 Activity Diagram

Activity: A specific task or action that occurs within the system. An activity can represent a simple action, such as a calculation or data manipulation, or a more complex process, such as a decision-making process.

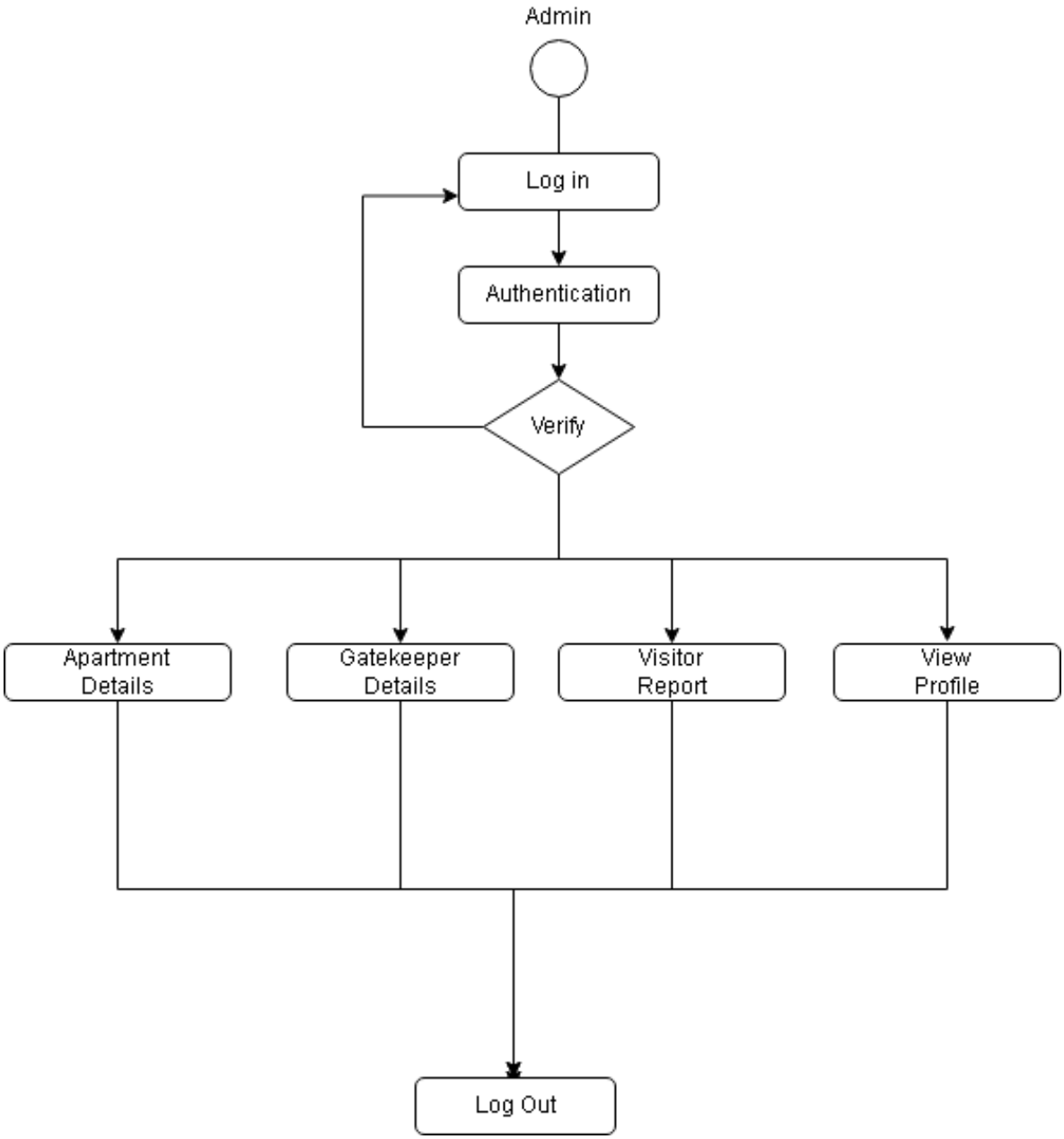
Control flow: The arrows connecting the activities, which show the order in which the activities occur. These arrows can be directed, indicating a specific direction of flow, or undirected, indicating that the order of the activities is not important.

Decision node: A diamond-shaped symbol that indicates a branching point in the flow of the activities. Depending on the condition or criteria specified, the flow can take one of several paths.

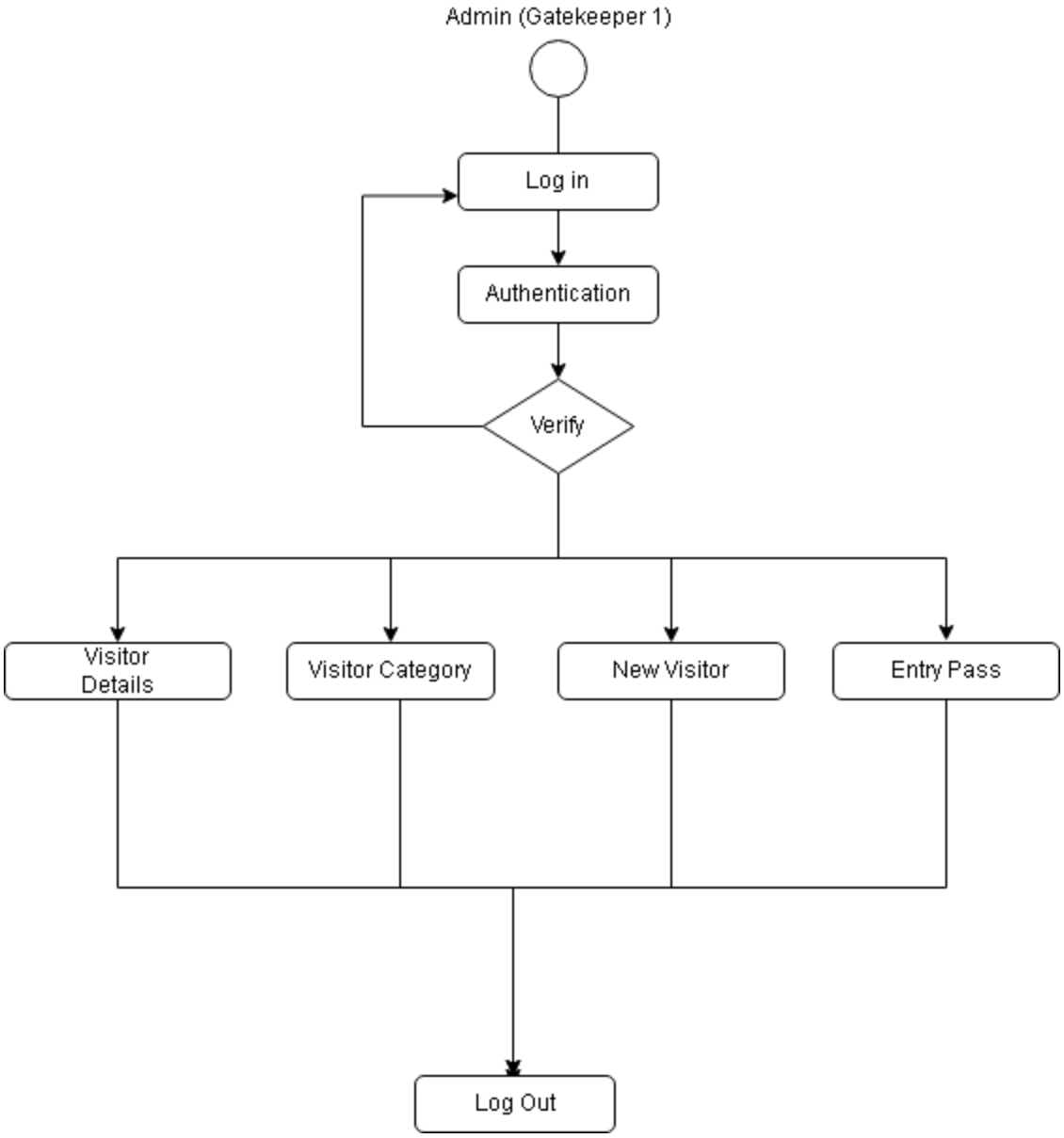
Merge node: A symbol that indicates the point at which the paths that were previously split by a decision node are merged back together.

Initial node: A solid circle that represents the starting point of the activity diagram.

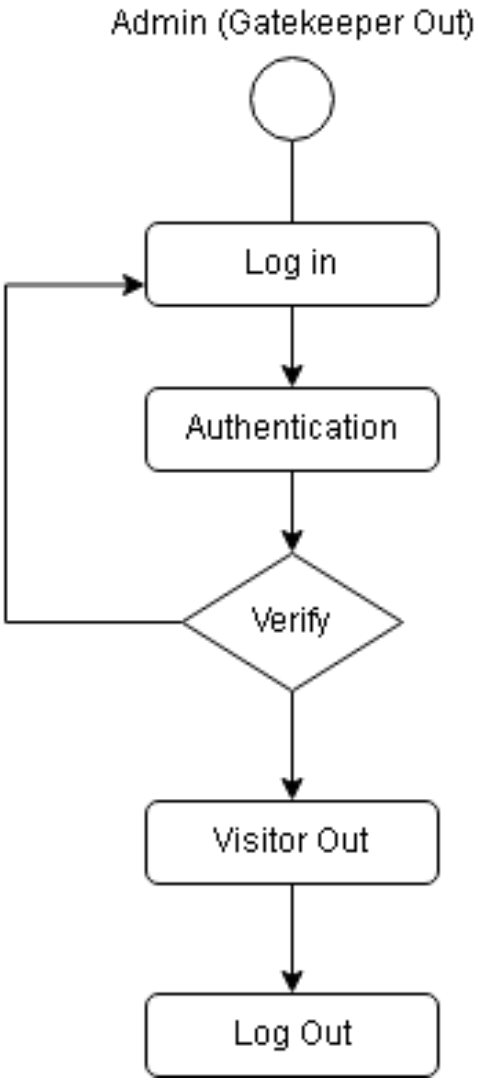
Final node: A solid circle with a border that represents the end point of the activity diagram.



Activity Diagram (Admin)



Activity Diagram (Gatekeeper In)



Activity Diagram (Gatekeeper Out)

5.4 Functional and Behavioural Modelling

5.4.1 Data Flow Diagram

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both.

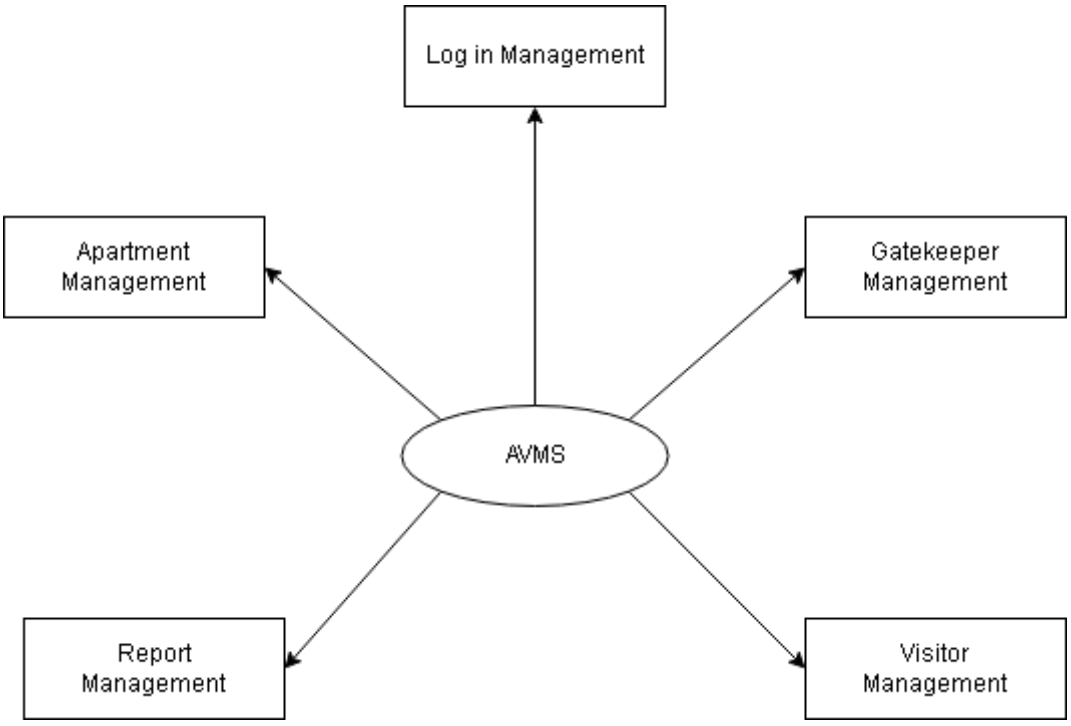
It shows how data enters and leaves the system, what changes the information, and where data is stored.

The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

The following observations about DFDs are essential:

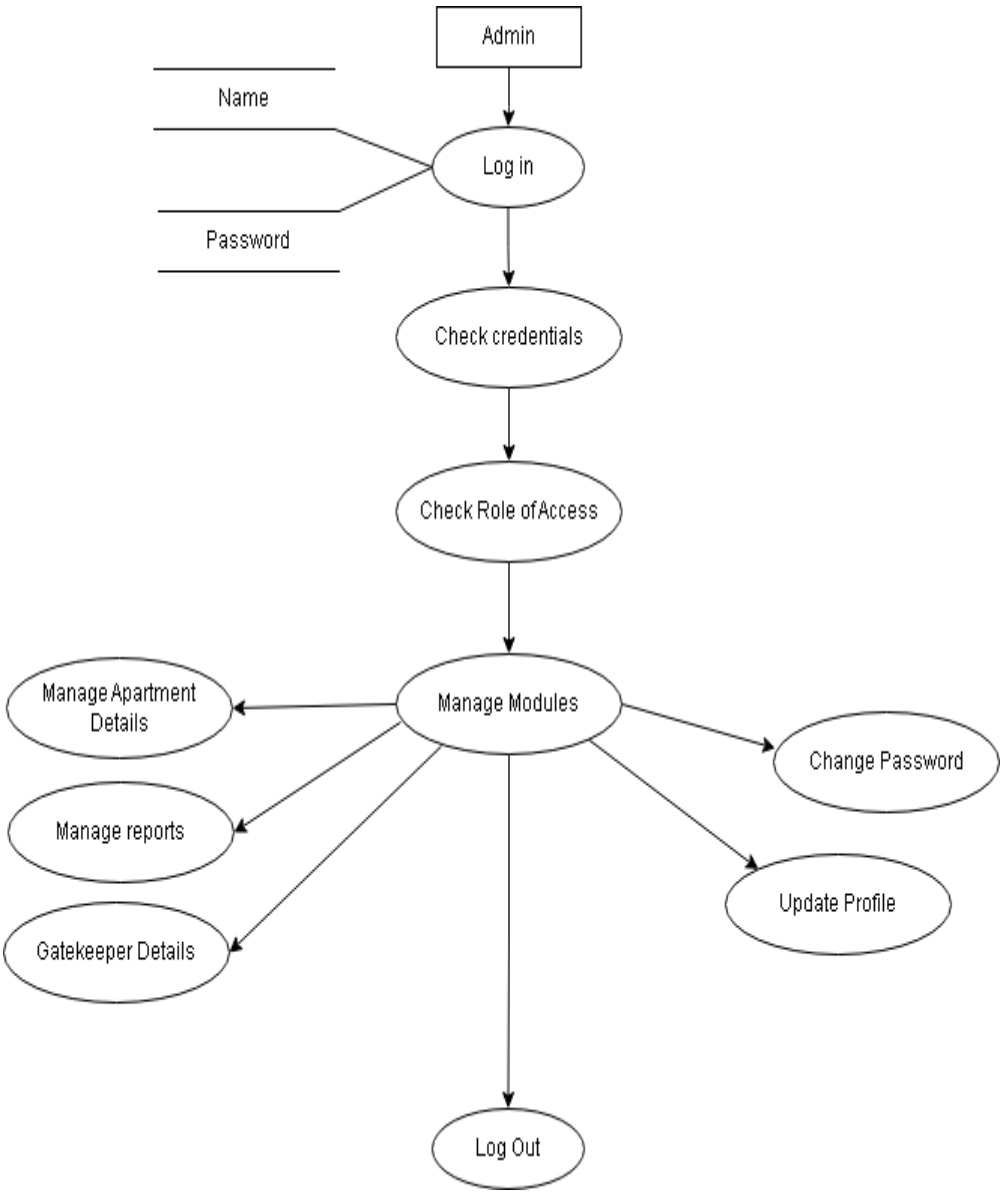
1. All names should be unique. This makes it easier to refer to elements in the DFD.
2. 2. Remember that DFD is not a flow chart. Arrows is a flow chart that represents the order of events; arrows in DFD represents flowing data. A DFD does not involve any order of events.
3. 3. Suppress logical decisions. If we ever have the urge to draw a diamond-shaped box in a DFD, suppress that urge! A diamondshaped box is used in flow charts to represents decision points with multiple exists paths of which the only one is taken. This implies an ordering of events, which makes no sense in a DFD.
4. 4. Do not become bogged down with details. Defer error conditions and error handling until the end of the analysis

- **Level 0 :**



Data Flow Diagram (Level 0)

- **Level 1 :**



Data Flow Diagram (Level 1)

Chapter 6: System Design

6.1 Database Schema design

- **Apartment Details**

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	**id**	int(255)			No	*None*		AUTO_INCREMENT	Change Drop More
2	**name**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
3	**phone_number**	bigint(255)			No	*None*			Change Drop More
4	**email**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
5	**house_number**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
6	**floor_wing**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
7	**owenership**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
8	**createat**	timestamp			No	current_timestamp()			Change Drop More
9	**updateat**	timestamp			Yes	*NULL*		ON UPDATE CURRENT_TIMESTAMP()	Change Drop More

- **Gatekeeper Details**

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	**id**	int(11)			No	*None*		AUTO_INCREMENT	Change Drop More
2	**name**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
3	**phone_number**	bigint(255)			No	*None*			Change Drop More
4	**address**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
5	**gate**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
6	**password**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
7	**createat**	timestamp			No	current_timestamp()			Change Drop More
8	**updateat**	timestamp			Yes	*NULL*		ON UPDATE CURRENT_TIMESTAMP()	Change Drop More
9	**roll**	int(255)			No	*None*			Change Drop More

- **Visitor Category**

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	id 🔑	int(11)			No	*None*		AUTO_INCREMENT	Change Drop More
☐	2	category_name	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	3	createat	timestamp			No	current_timestamp()			Change Drop More
☐	4	updateat	timestamp			Yes	*NULL*		ON UPDATE CURRENT_TIMESTAMP()	Change Drop More

- **Visitor Details**

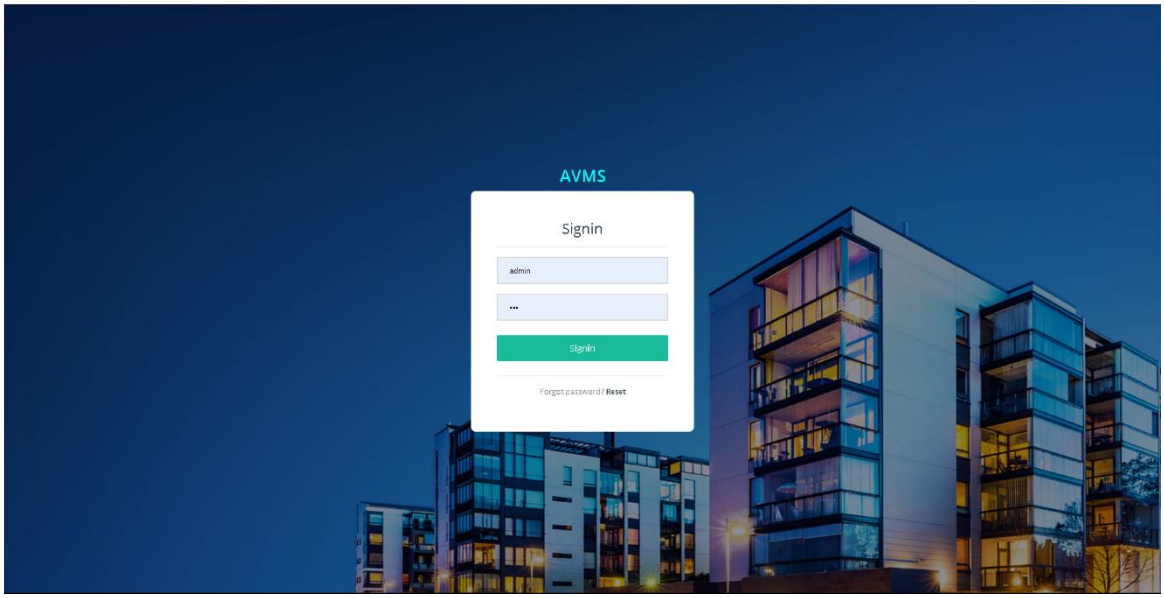
	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	id 🔑	int(11)			No	*None*		AUTO_INCREMENT	Change Drop More
☐	2	category_name	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	3	visitor_name	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	4	phone_number	bigint(255)			No	*None*			Change Drop More
☐	5	address	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	6	apartment	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	7	floor	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	8	whom_to_meet	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	9	reason_to_meet	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	10	createat	timestamp			No	current_timestamp()			Change Drop More
☐	11	out_remark	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change Drop More
☐	12	out_time	timestamp			Yes	*NULL*		ON UPDATE CURRENT_TIMESTAMP()	Change Drop More

- **Visitor Pass**

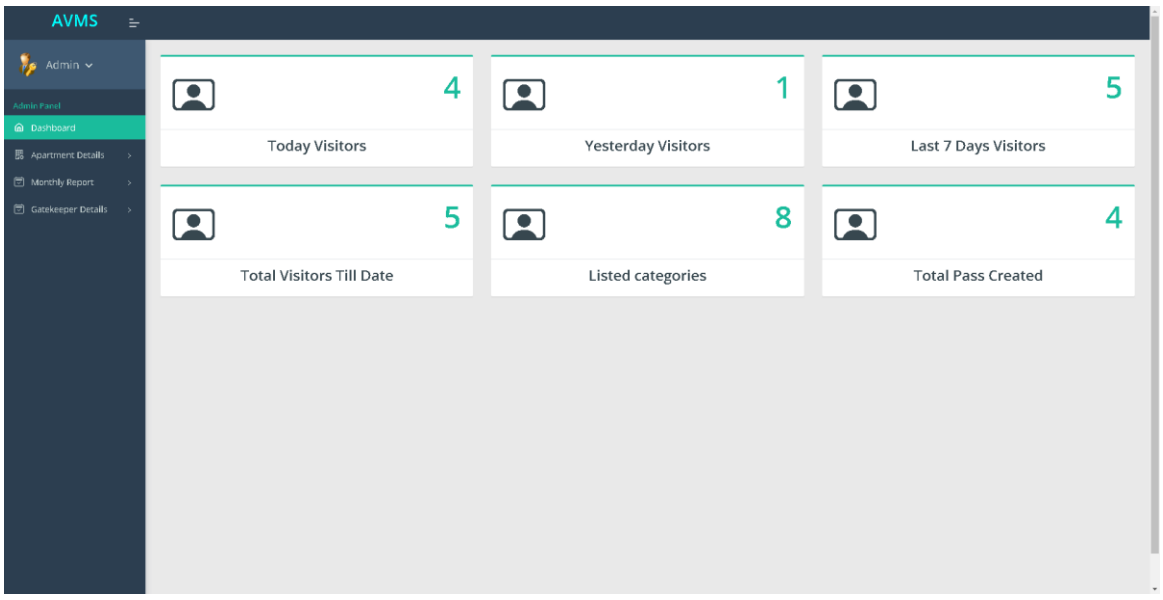
	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	**id**	int(11)			No	*None*		AUTO_INCREMENT	Change Drop More
☐	2	**passnumber**	bigint(255)			No	*None*			Change Drop More
☐	3	**category_name**	varchar(255)	latin1_swedish_ci		No	*None*			Change Drop More
☐	4	**visitor_name**	varchar(255)	latin1_swedish_ci		No	*None*			Change Drop More
☐	5	**phone_number**	bigint(255)			No	*None*			Change Drop More
☐	6	**address**	varchar(255)	latin1_swedish_ci		No	*None*			Change Drop More
☐	7	**apartment**	varchar(255)	latin1_swedish_ci		No	*None*			Change Drop More
☐	8	**floor**	varchar(255)	latin1_swedish_ci		No	*None*			Change Drop More
☐	9	**pass_description**	varchar(255)	latin1_swedish_ci		No	*None*			Change Drop More
☐	10	**from_date**	date			No	*None*			Change Drop More
☐	11	**to_date**	date			No	*None*			Change Drop More
☐	12	**createat**	timestamp			No	current_timestamp()			Change Drop More

Chapter 7: Implementation

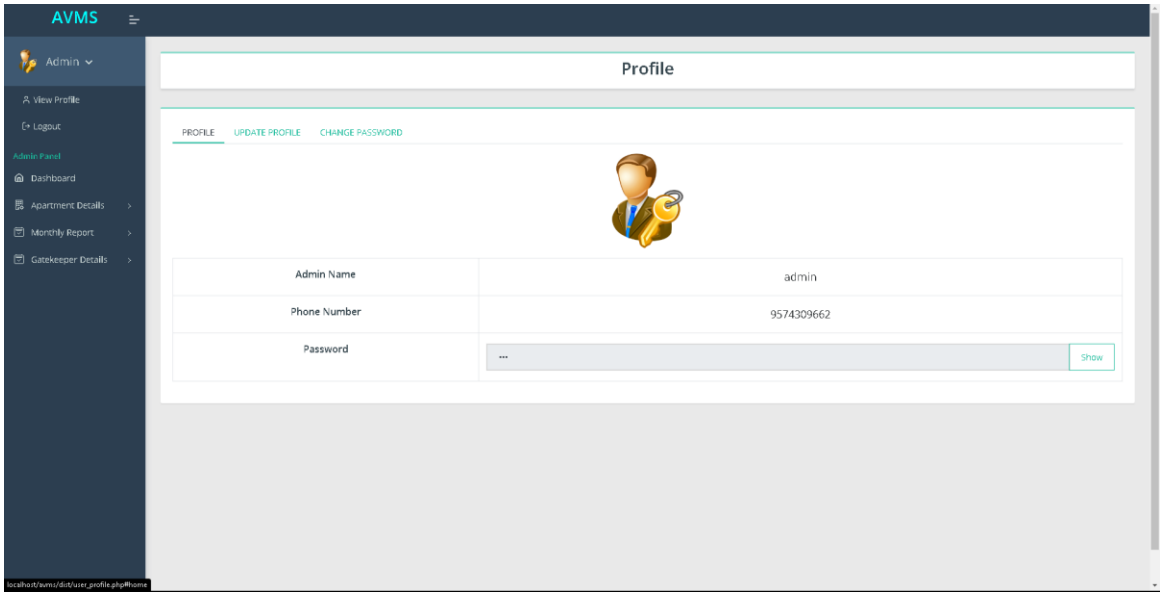
- **Admin :**
    - Admin Log in



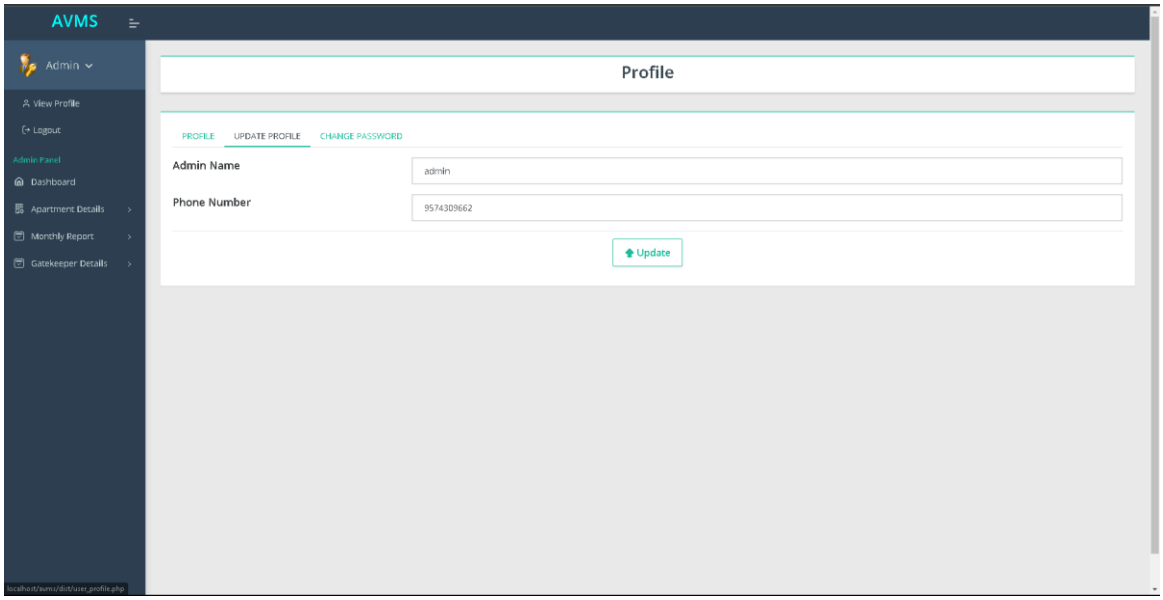
    - Dashboard :



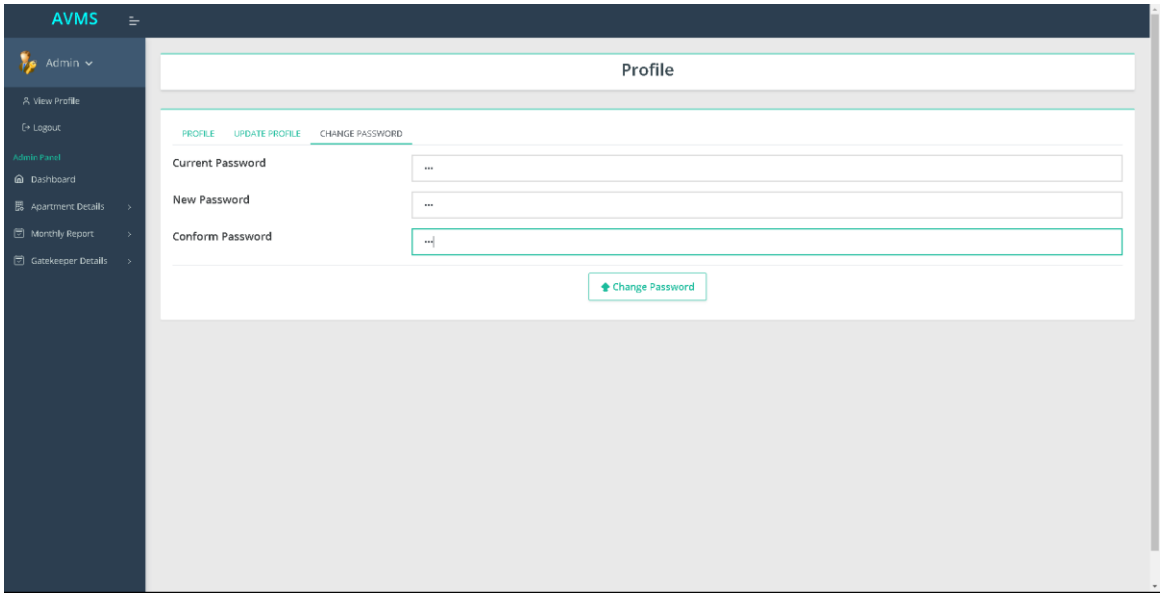
- Profile



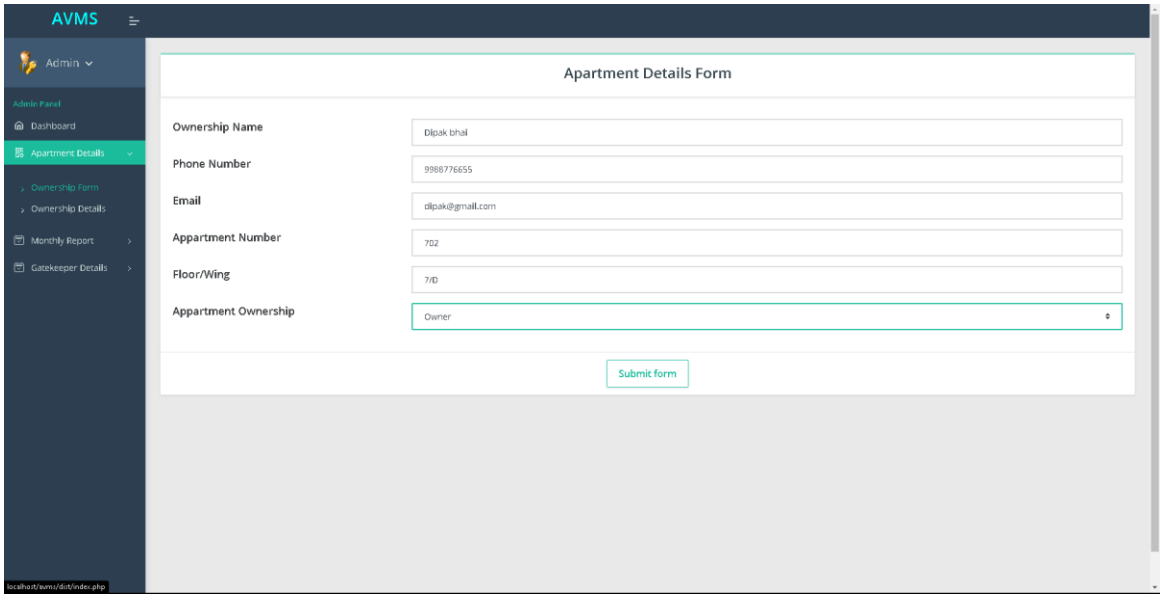
- UpdateProfile



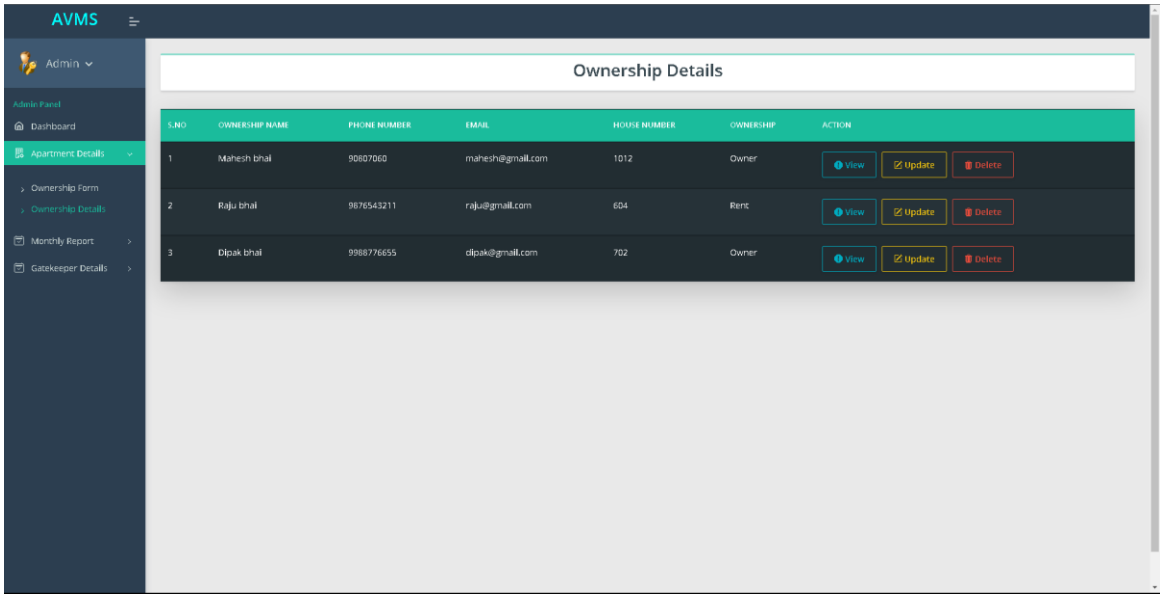
- Change Password



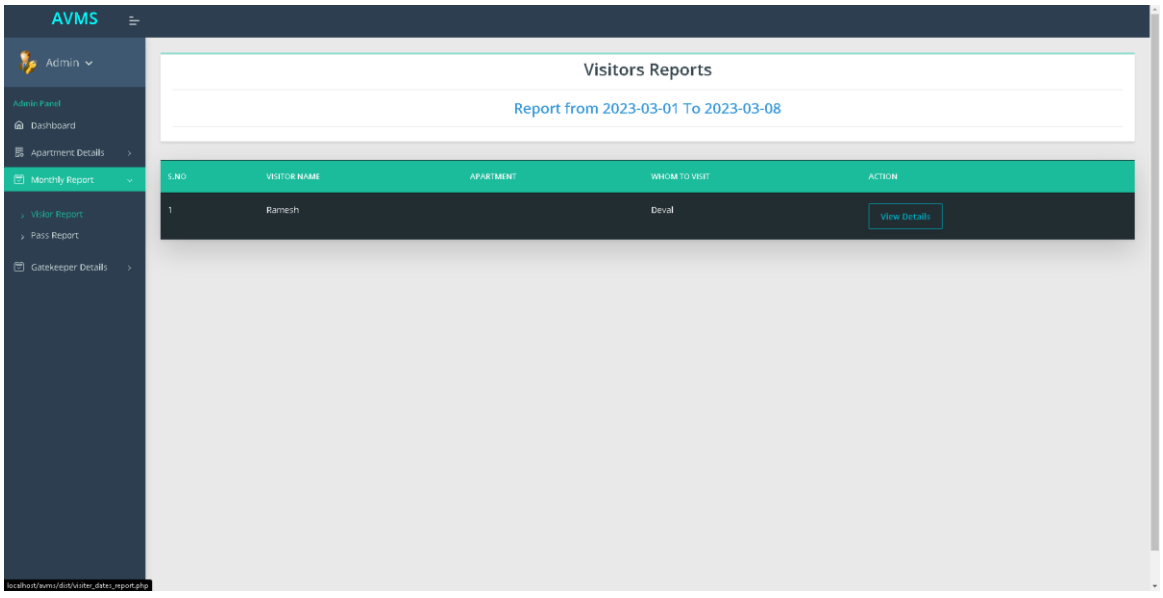
- Ownership Form



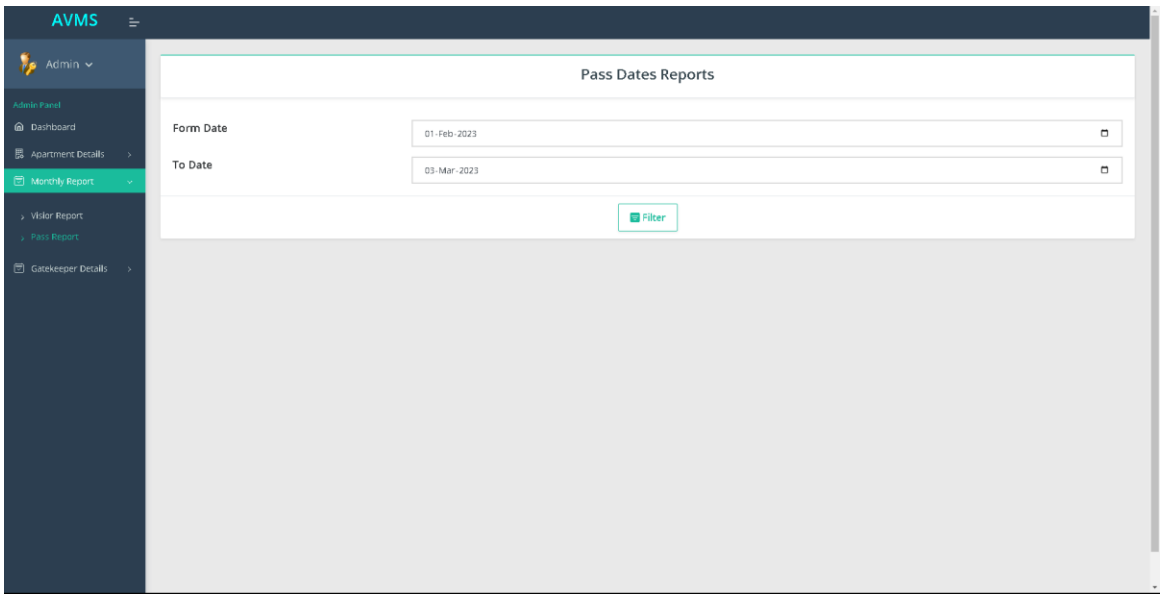
- Ownership Details



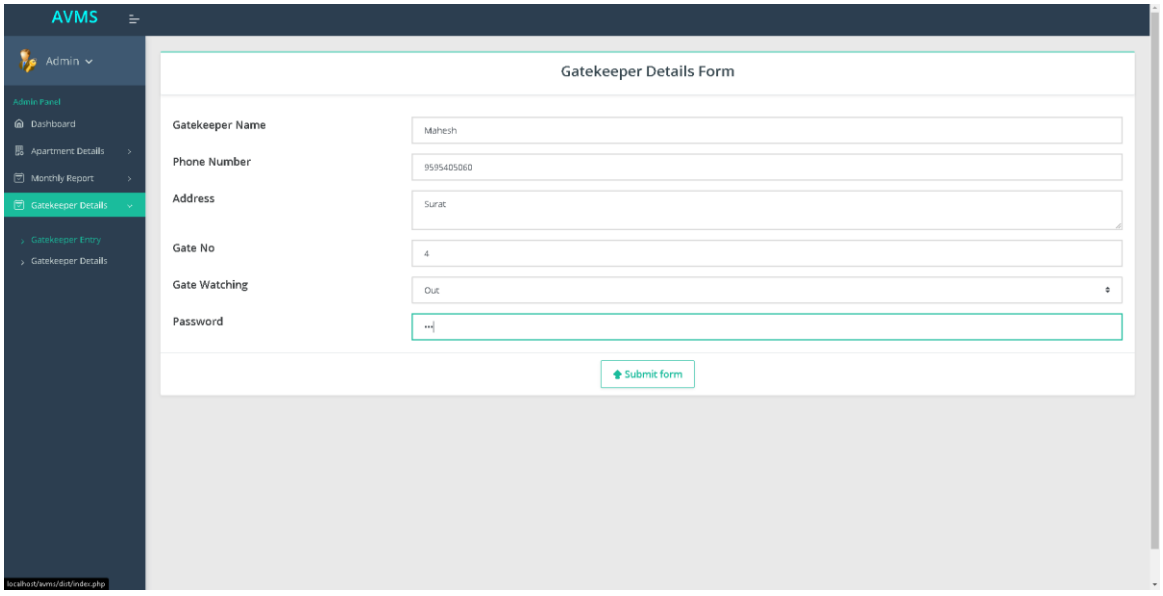
- Visitor Report



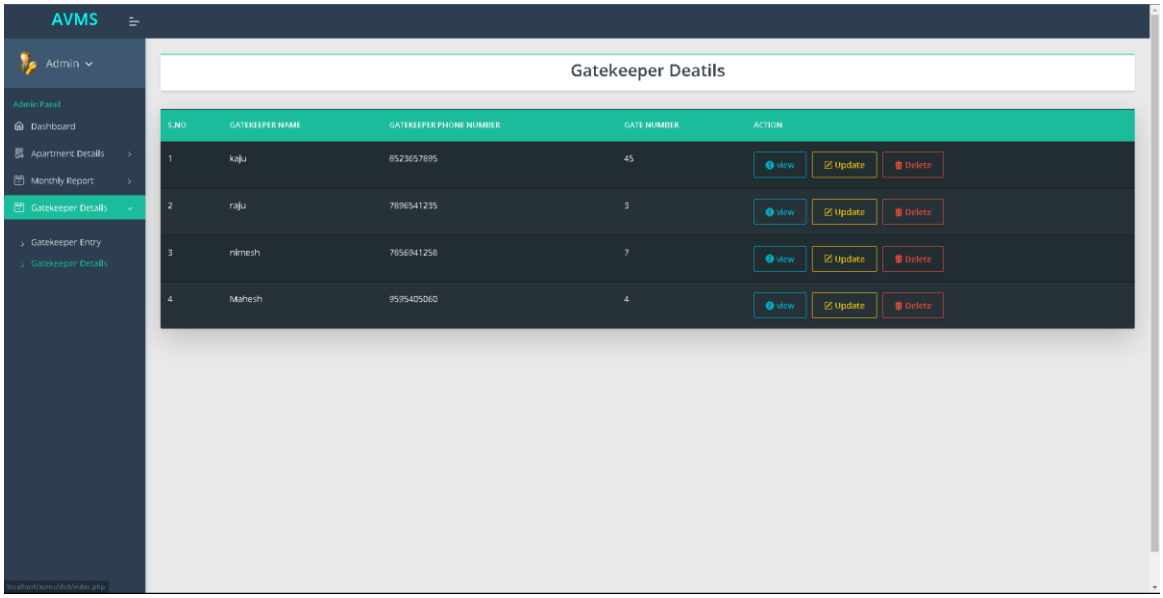
- Pass Report



- Gatekeeper Entry

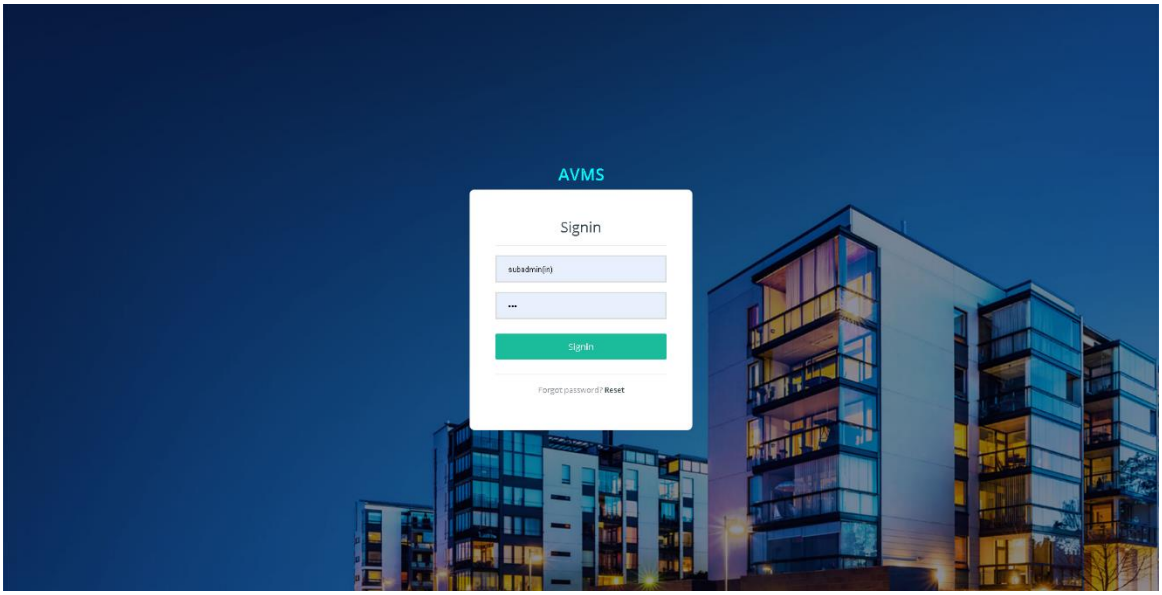


- Gatekeeper Details

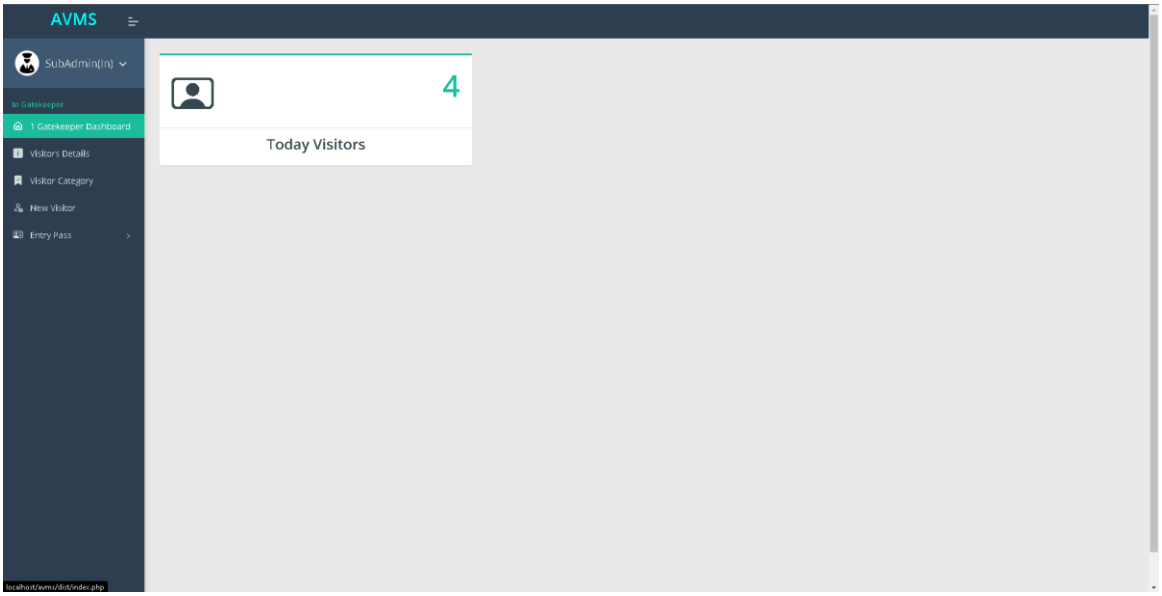


- Gatekeeper In :

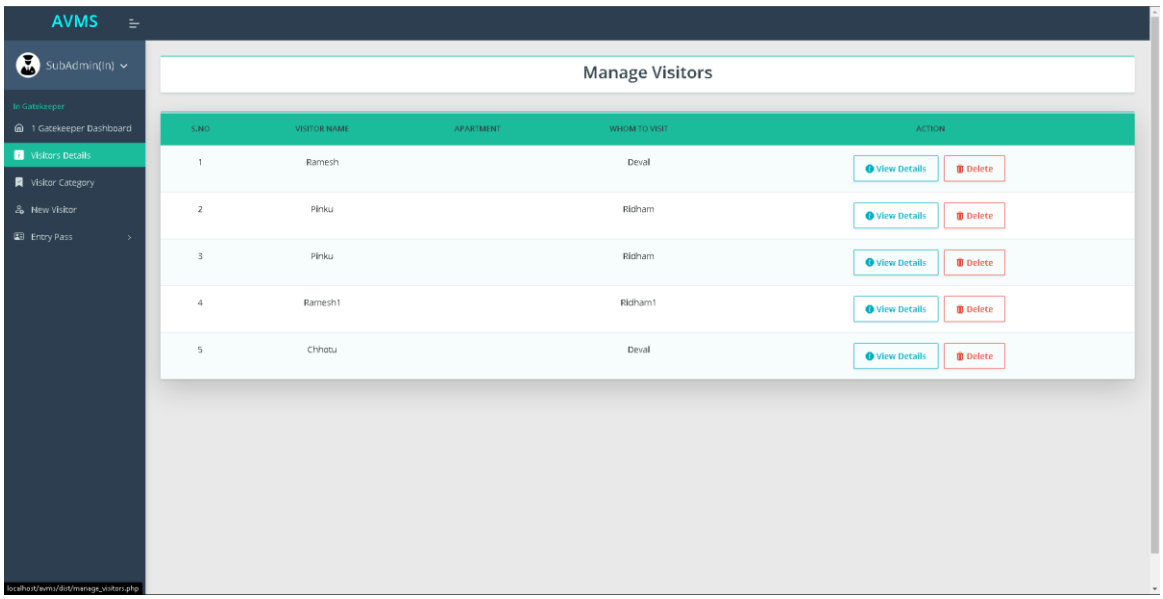
  - Log in:



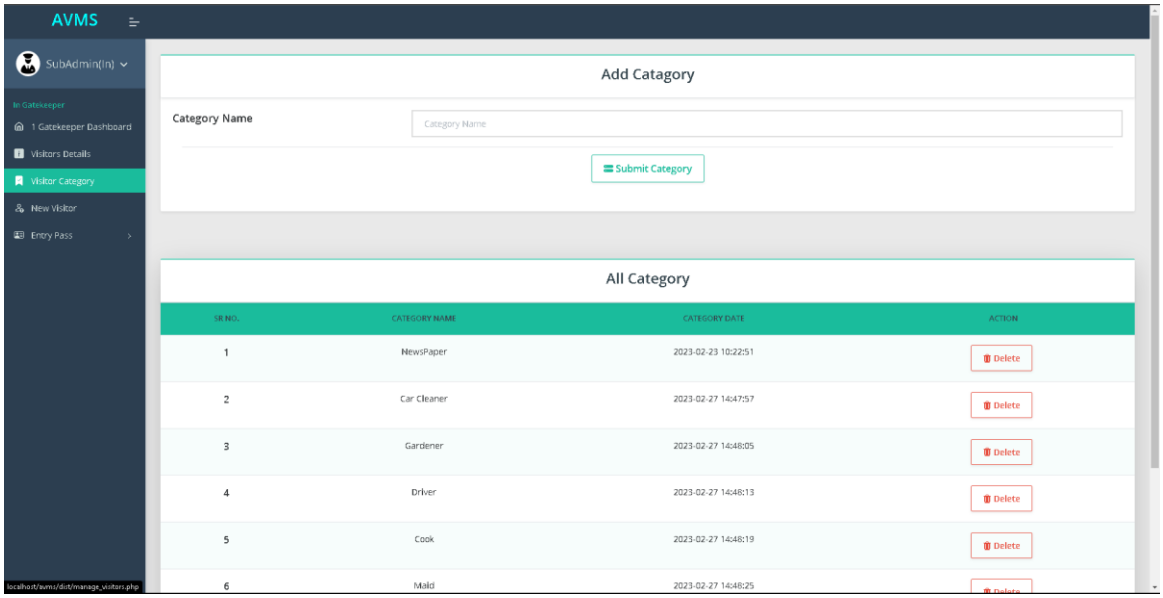
  - Dashboard



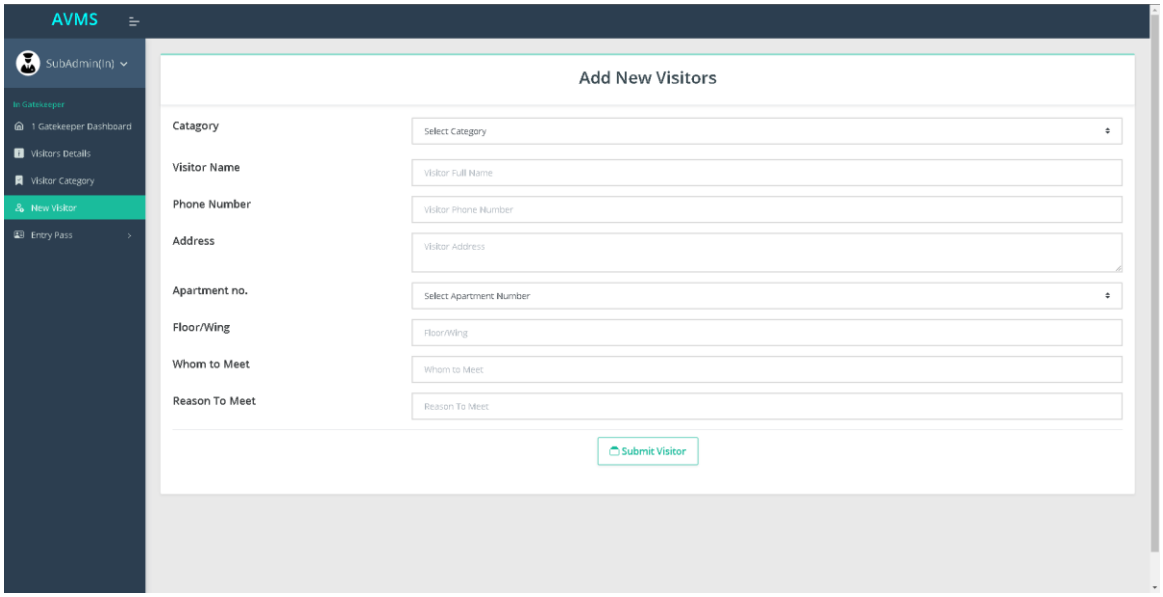
- Visitor Details



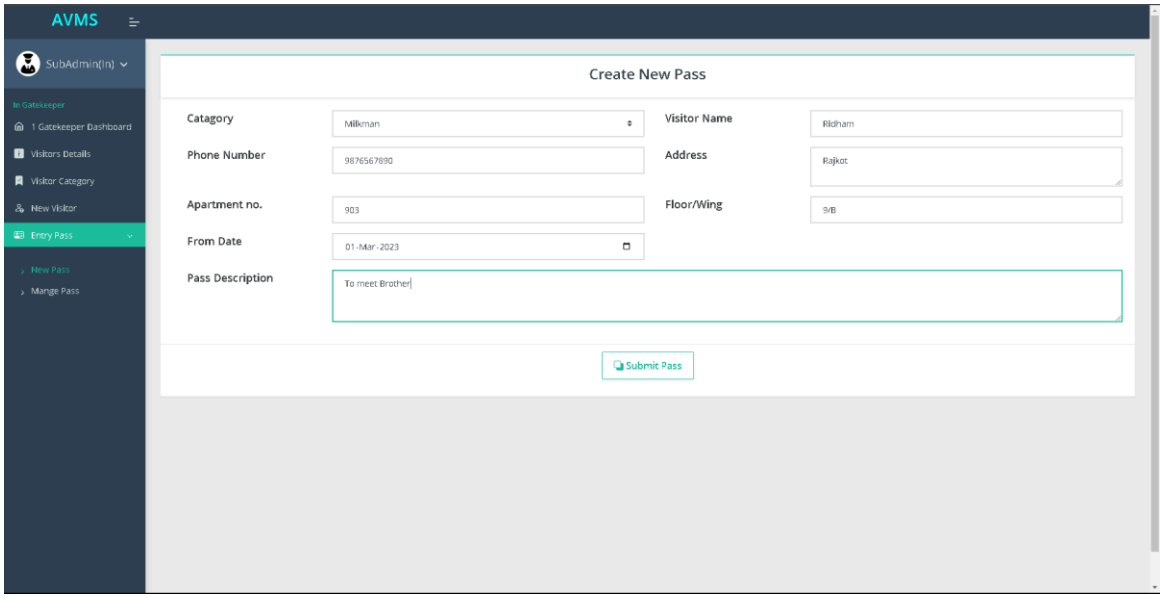
- Visitor Category



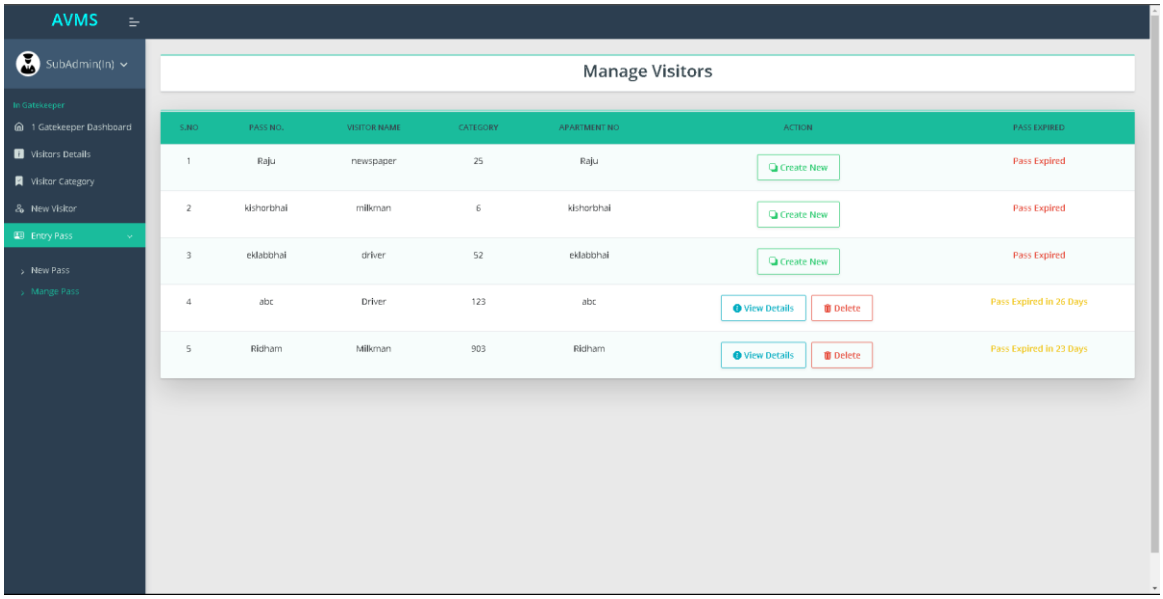
- New Visitor



- New Pass

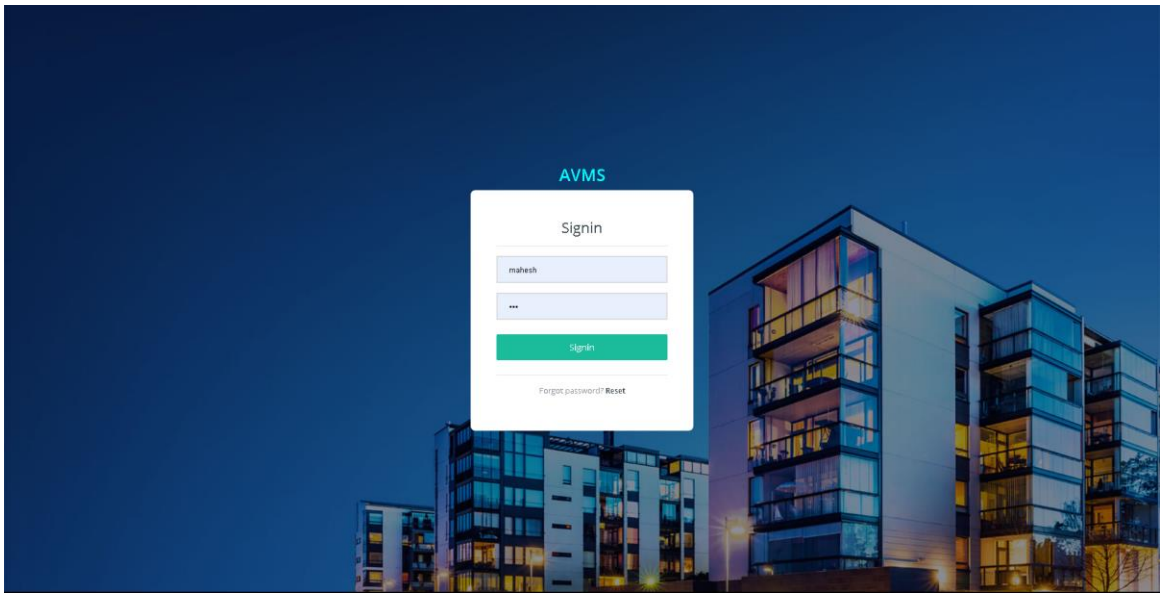


- Manage Pass

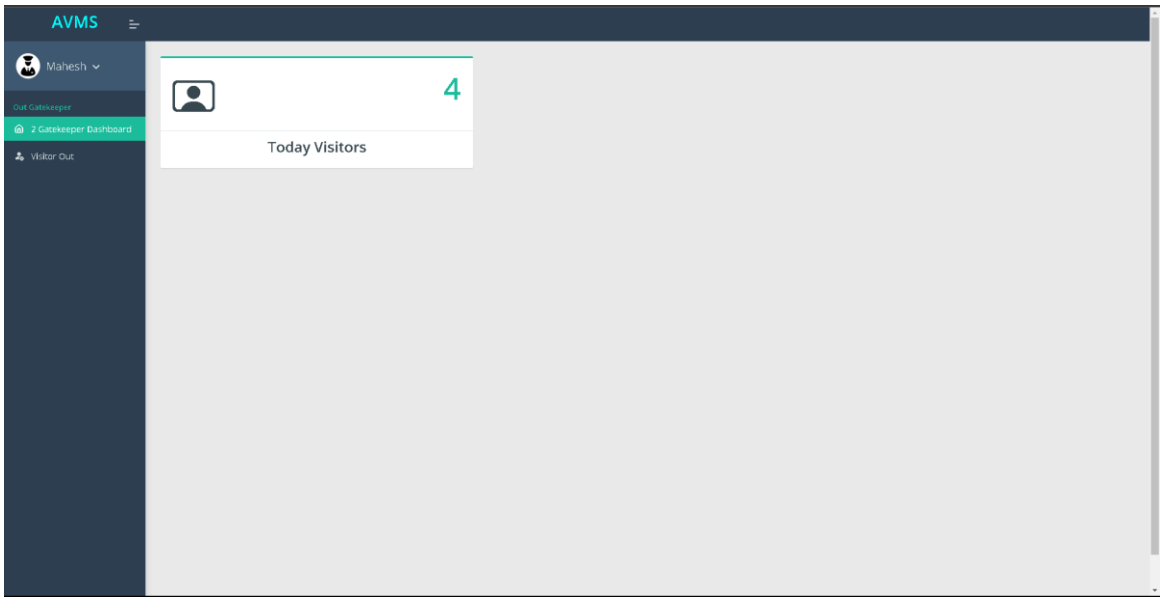

AVMS
SubAdmin(In)
In Gatekeeper
1 Gatekeeper Dashboard
Visitors Details
Visitor Category
New Visitor
Entry Pass
New Pass
Manage Pass
Manage Visitors
S.NO
PASS NO.
VISITOR NAME
CATEGORY
APARTMENT NO
ACTION
PASS EXPIRED
1 Raju newspaper 25 Raju Create New Pass Expired
2 kishorbhai milkman 6 kishorbhai Create New Pass Expired
3 eklabbhai driver 52 eklabbhai Create New Pass Expired
4 abc Driver 123 abc View Details Delete Pass Expired in 26 Days
5 Ridham Milkman 903 Ridham View Details Delete Pass Expired in 23 Days

- Gatekeeper(Out)

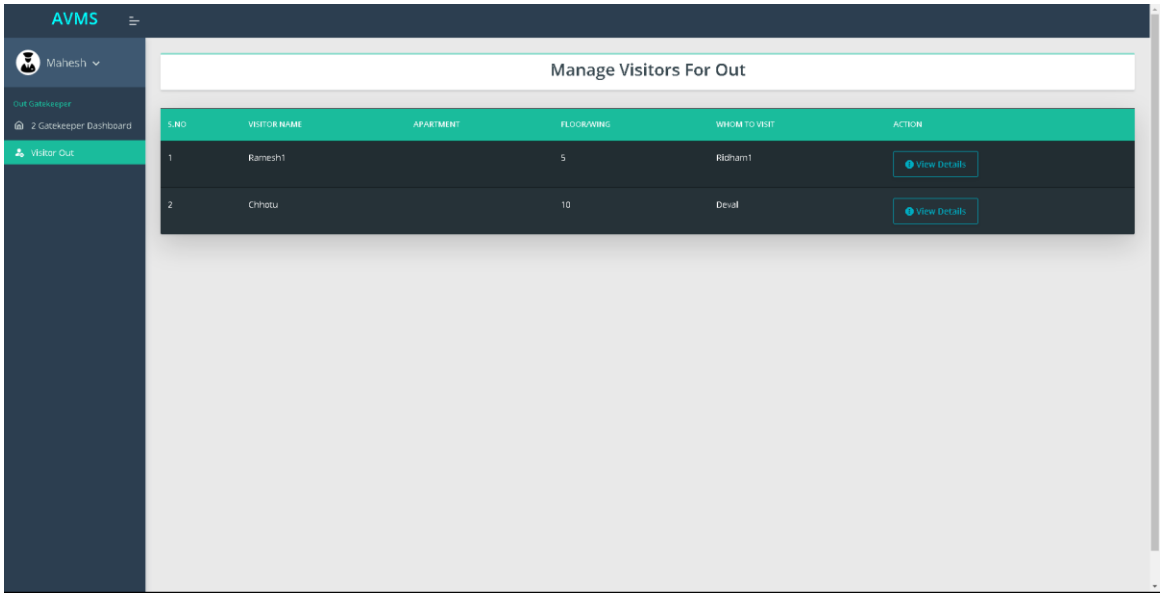
  - Log in



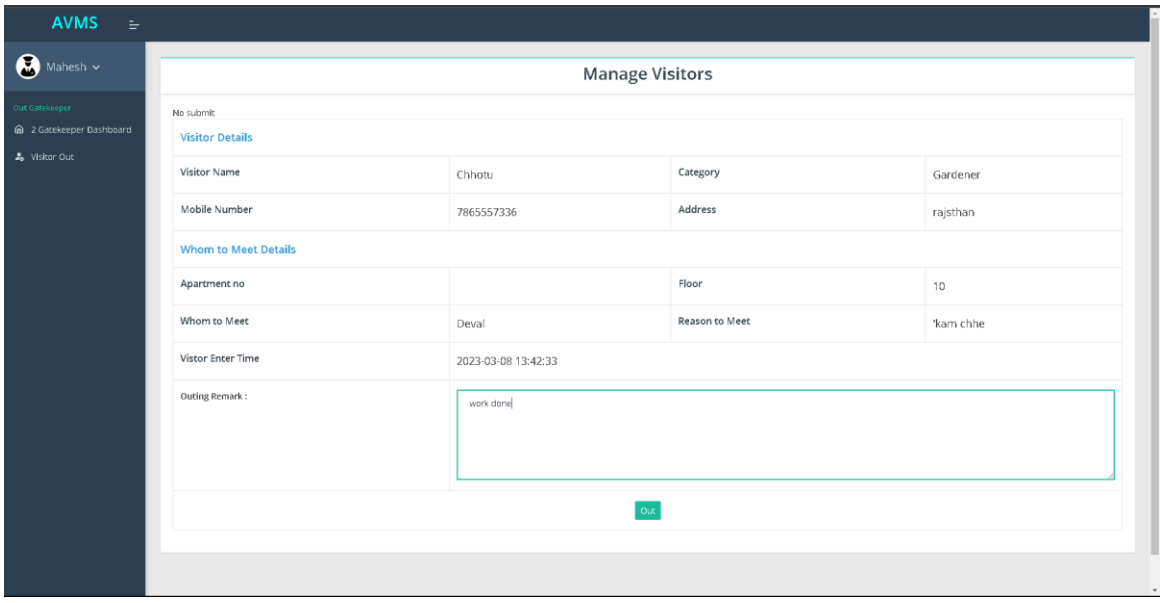
  - Dashboard



- Manage Visitor Out



- Visitor Out



Chapter 8: Canclusion and Future work

Appendices :

User manual: A detailed guide for how to use the visitor management system. This can include step-by-step instructions, screenshots, and other relevant information.

Sample visitor logs: Copies of visitor logs that were generated by the system during testing or implementation. This can show how the system records visitor information and how it can be used to track visitor activity.

Technical specifications: A detailed list of the hardware and software requirements needed to run the visitor management system. This can help property managers or IT professionals determine if the system is compatible with their existing infrastructure.

Training materials: Materials that were used to train property managers or other staff members on how to use the visitor management system. This can include presentation slides, handouts, or other resources.

Security protocols: A description of the security measures that are in place to protect visitor information and prevent unauthorized access. This can help property managers or other stakeholders understand how the system handles sensitive data.

Case studies: Examples of how the visitor management system has been used successfully in other apartment complexes or similar settings. This can provide context for how the system can be used and its potential benefits.

References :

- www.w3schools.com
- www.devdocs.com

Any work may not always be perfect :

There may be some defects or errors. We have taken enough care to make the project user friendly and more interactive.

Major focus is to save time, because time is money.

THANK YOU...