Apartment Visitor Management System

Project Report

By

ACKNOWLEDGEMENT

I extend my deepest appreciation to my esteemed guide, Mr. XYZ for providing me with the possibility to complete this project with the right guidance and advice.

Special gratitude I give to my respected head of the division Mr.XYZ, for allowing me to use the facilities available and also help me to coordinate my project Furthermore, I would also like to acknowledge with much appreciation the crucial role of faculty members on this occasion.

Last but not least, I would like to thank friends who help me to assemble the parts and gave a suggestion about the project.

.

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.

Nowadays, 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.

Introduction

Introduction:-

Apartment Visitor Management system is a web-based technology that will revolutionize the way your society manages visitors and provide visitor pass to them. Visitor Management system is more important to security guards or security society. This web application provides a way to effectively control record & track society visitor traffic.

In AVMS we use PHP and MySQL database. This is the project which keeps records of visitors who visited in the Society. **AVMS has one module i.e. admin**

- **Dashboard:** In this section, admin can briefly view how many visitors visited in a particular period, total listed categories and total visitor pass created.
- **Categories:** In this section, admin can mange categories(Add/Delete).
- **New Visitors:** In this section, admin adds new visitors by filling their information in add visitors sections.
- **Manage Visitors:** In this section, admin can view and manage visitor's records. Admin also put visitors out time in the manage records section.
- **Entry Pass:** In this section, admin can manage entry pass(Create/View/Delete).
- **Search:** In this bar, admin can search a particular person by their name and phone number.
- **Visitor B/W Reports:** In this section admin can generate visitor's reports between two dates.
- **Visitor Pass B/W Reports:** In this section admin can generate visitor's pass reports between two dates.

Admin can also update his profile, change password and recover password.

Purpose:-

The purpose of developing apartment visitor management system is to computerized the tradition way of visitors. Another purpose for developing this application is to generate the report automatically.

The main aim of this application is to develop a system that effectively manages all the data related to apartment visitor which helps gate guards to maintain bulk records of visitors. The purpose is to maintain a centralized database of all visitor information. The goal is to support various functions and processes necessary to manage the data efficiently.

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.

Requirement Specification

Hardware Configuration :

Client Side:

RAM	512 MB
Hard disk	10 GB
Processor	1.0 GHz

Server side:

RAM	**1 GB**
Hard disk	**20 GB**
Processor	**2.0 GHz**

Software Requirement:

Client Side:

Web Browser	Google Chrome or any compatible browser
Operating System	Windows or any equivalent OS

Server Side:

Web Server	APACHE
Server side Language	PHP5.6 or above version
Database Server	MYSQL
Web Browser	Google Chrome or any compatible browser
Operating System	Windows or any equivalent OS

APACHE

The Apache HTTP Server Project is an effort to develop and maintain an open-source HTTP server for modern operating systems including UNIX and Windows. The goal of this project is to provide a secure, efficient and extensible server that provides HTTP services in sync with the current HTTP standards.

The Apache HTTP Server ("httpd") was launched in 1995 and it has been the most popular web server on the Internet since April 1996. It has celebrated its 20th birthday as a project in February 2015.

PHP

- PHP stands for PHP: Hypertext Preprocessor.
- PHP is a server-side scripting language, like ASP.

- PHP scripts are executed on the server.
- PHP supports many databases (MYSQL, Informix, Oracle, Sybase, Solid, Generic ODBC, etc.).
- PHP is open source software.
- PHP is free to download and use.

MYSQL

- MYSQL is a database server
- MYSQL is ideal for both small and large applications
- MYSQL supports standard SQL
- MYSQL compiles on a number of platforms
- MYSQL is free to download and use
- How to access MySQL:

  http://localhost/phpmyadmin

Analysis and Design

Analysis:

In present all visitor work done on the paper. The whole year visitor is stored in the registers. We can't generate reports as per our requirements because its take more time to calculate the visitors report.

Disadvantage of present system:

- **Not user friendly:** The present system not user friendly because data is not stored in structure and proper format.
- **Manual Control:** All report calculation is done manually so there is a chance of error.
- **Lots of paper work:** Visitors maintain in the register so lots of paper require storing details.
- **Time consuming**

Design Introduction:

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process or system in sufficient detail to permit its physical realization.

Once the software requirements have been analyzed and specified the software design involves three technical activities - design, coding, implementation and testing that are required to build and verify the software.

The design activities are of main importance in this phase, because in this activity, decisions ultimately affecting the success of the software implementation and its ease of maintenance are made. These decisions have the final bearing upon reliability and maintainability of the system. Design is the only way to accurately translate the customer's requirements into finished software or a system.

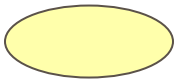
Design is the place where quality is fostered in development. Software design is a process through which requirements are translated into a representation of software. Software design is conducted in two steps. Preliminary design is concerned with the transformation of requirements into data

UML Diagrams:

Actor:

A coherent set of roles that users of use cases play when interacting with the use `cases.

Use case:A description of sequence of actions, including variants, that a system performs that yields an observable result of value of an actor.



UML stands for Unified Modeling Language. UML is a language for specifying, visualizing and documenting the system. This is the step while developing any product after analysis. The goal from this is to produce a model of the entities involved in the project which later need to be built. The representation of the entities that are to be used in the product being developed need to be designed.

USECASE DIAGRAMS:

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.

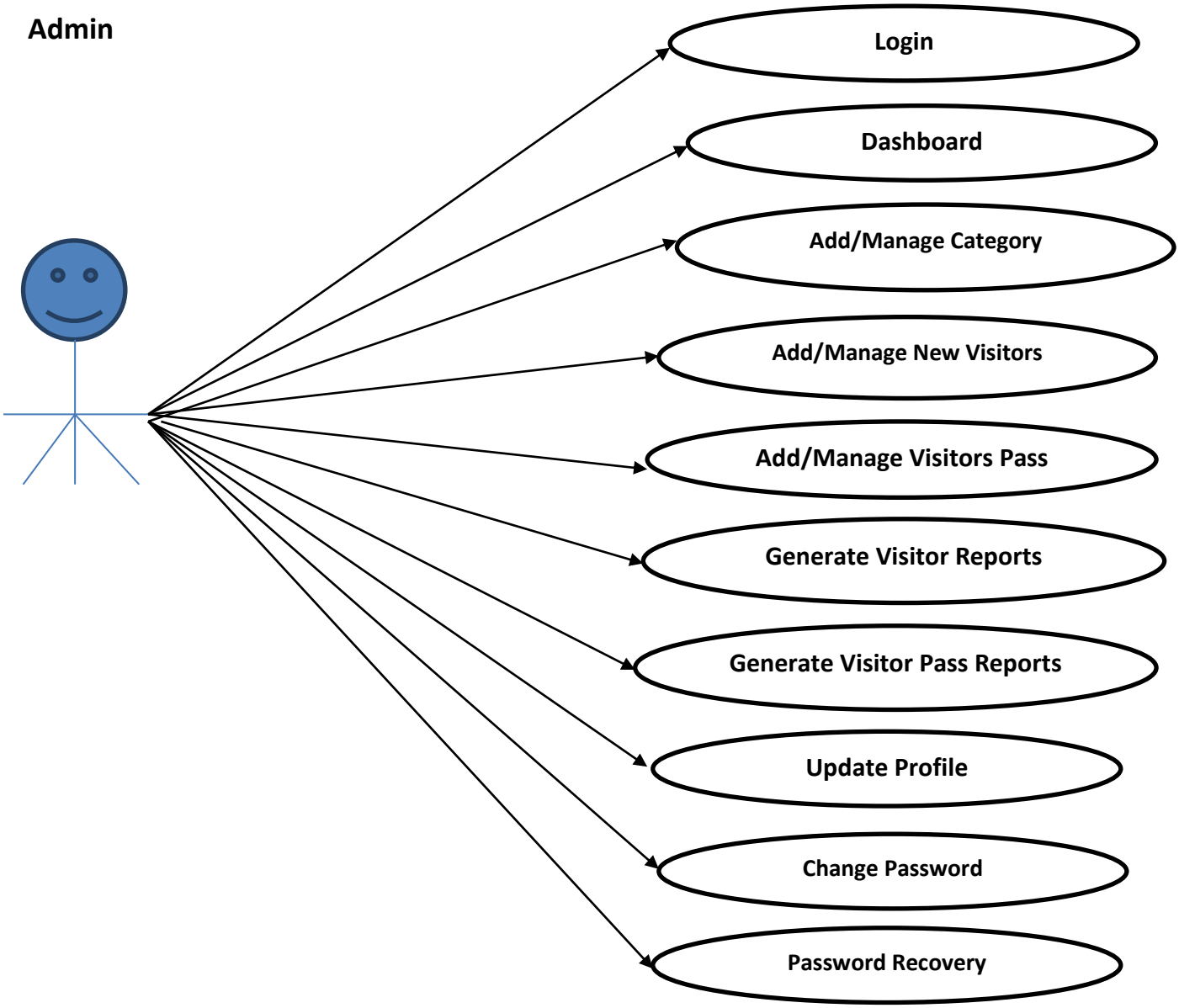
Use case diagram consists of use cases and actors and shows the interaction between the use case and actors.

- The purpose is to show the interactions between the use case and actor.
- To represent the system requirements from user's perspective.
- An actor could be the end-user of the system or an external system.

USECASE DIAGRAM: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.

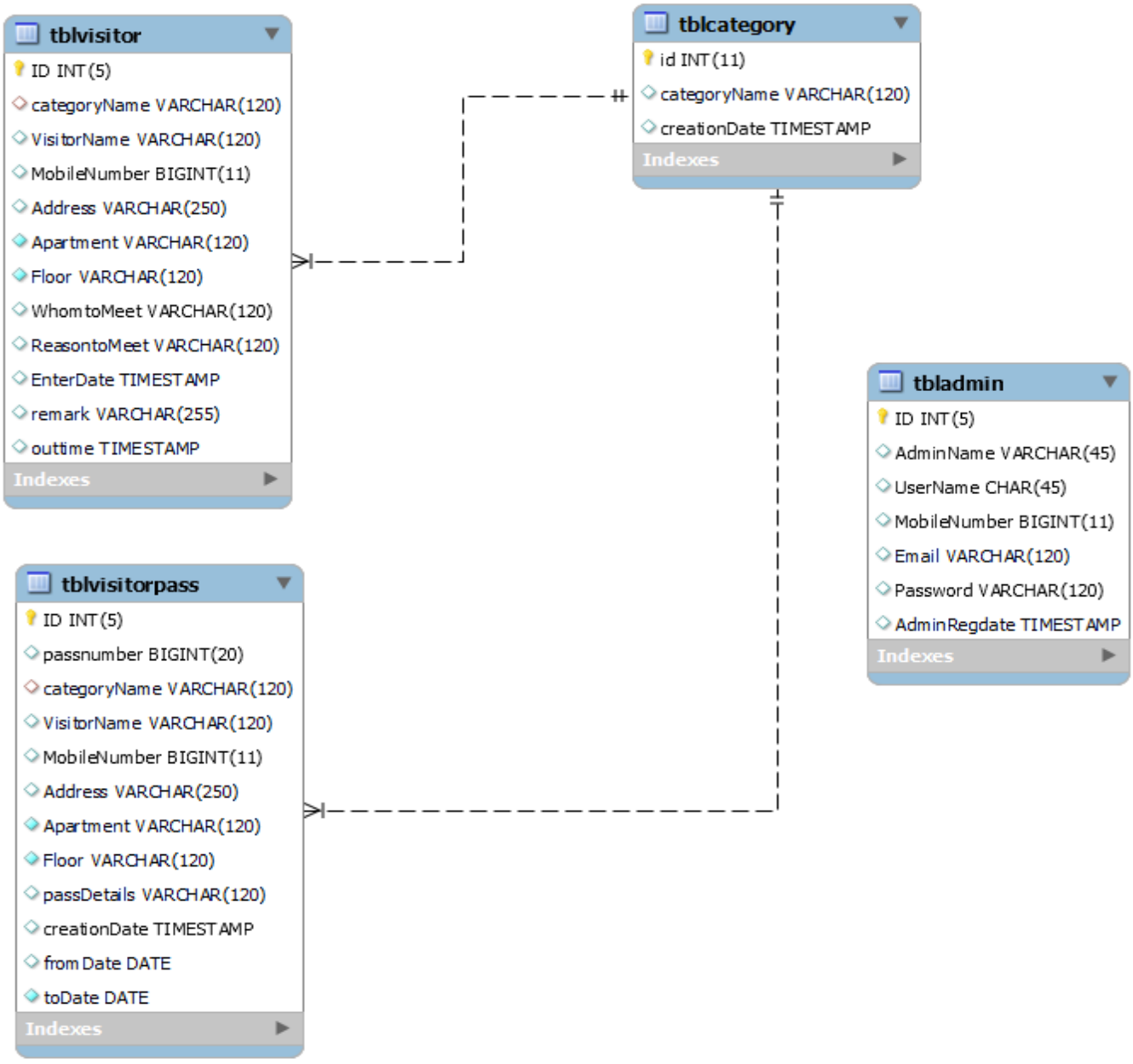
Use Case Diagrams:

Admin



Class Diagram:

A description of set of objects that share the same attributes operations, relationships, and semantics



ER 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.

ER Notation

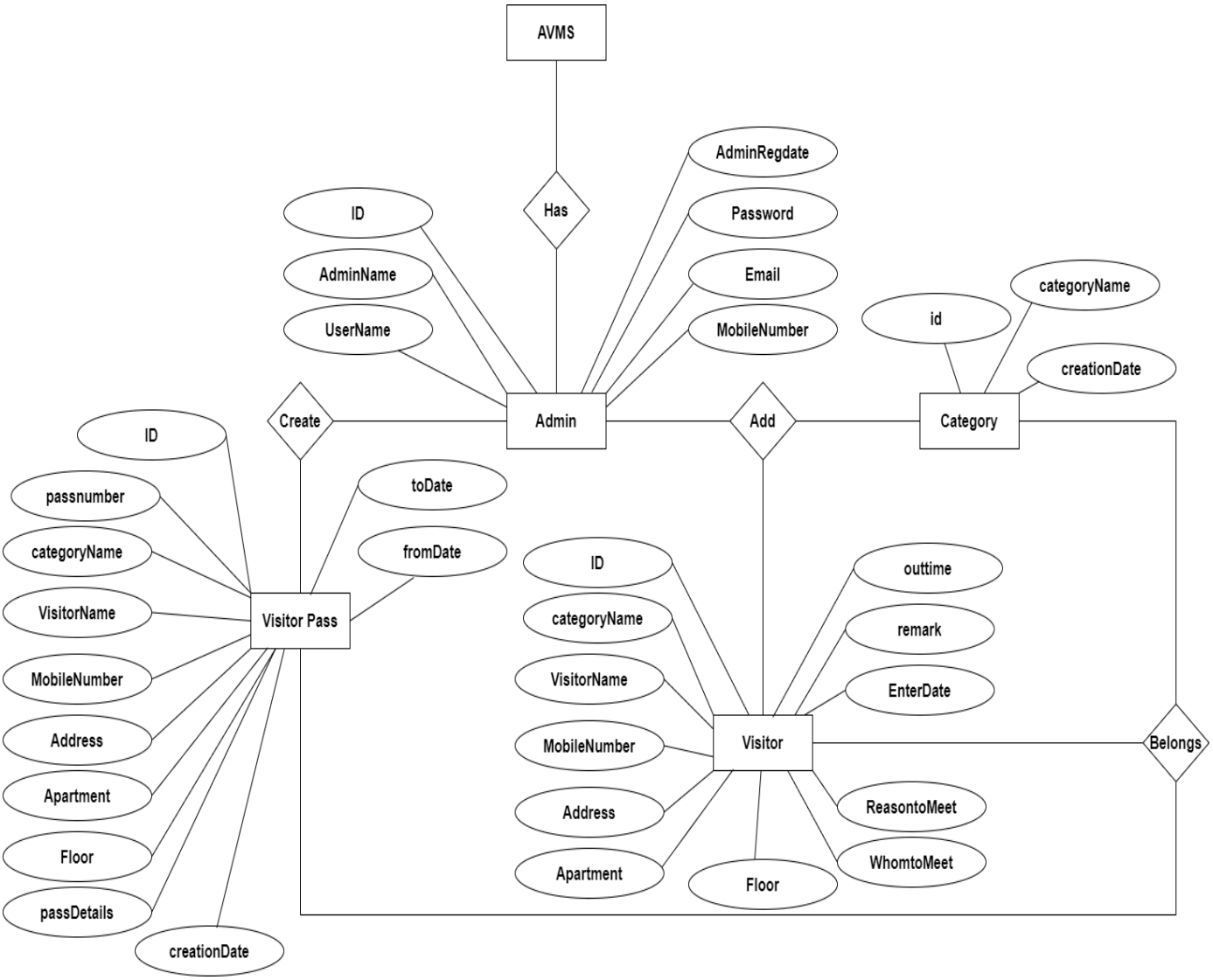
There is no standard for representing data objects in ER diagrams. Each modeling methodology uses its own notation. The original notation used by Chen is widely used in academics texts and journals but rarely seen in either CASE tools or publications by non-academics. Today, there are a number of notations used; among the more common are Bachman, crow's foot, and IDEFIX.

All notational styles represent entities as rectangular boxes and relationships as lines connecting boxes. Each style uses a special set of symbols to represent the

cardinality of a connection. The notation used in this document is from Martin. The symbols used for the basic ER constructs are:

- **Entities** are represented by labeled rectangles. The label is the name of the entity. Entity names 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.
- **Cardinality** of many is represented by a line ending in a crow's foot. If the crow's foot is omitted, the cardinality is one.

Existence is represented by placing a circle or a perpendicular bar on the line. Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required. Optional existence is shown by placing a circle next to the entity that is optional.


AVMS
Has
AdminRegdate
Password
Email
MobileNumber
ID
AdminName
UserName
Admin
Create
Add
id
categoryName
creationDate
Category
ID
passnumber
categoryName
VisitorName
MobileNumber
Address
Apartment
Floor
passDetails
creationDate
Visitor Pass
toDate
fromDate
ID
categoryName
VisitorName
MobileNumber
Address
Apartment
Floor
outtime
remark
EnterDate
Visitor
ReasontoMeet
WhomtoMeet
Belongs

Data Flow Diagrams

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.

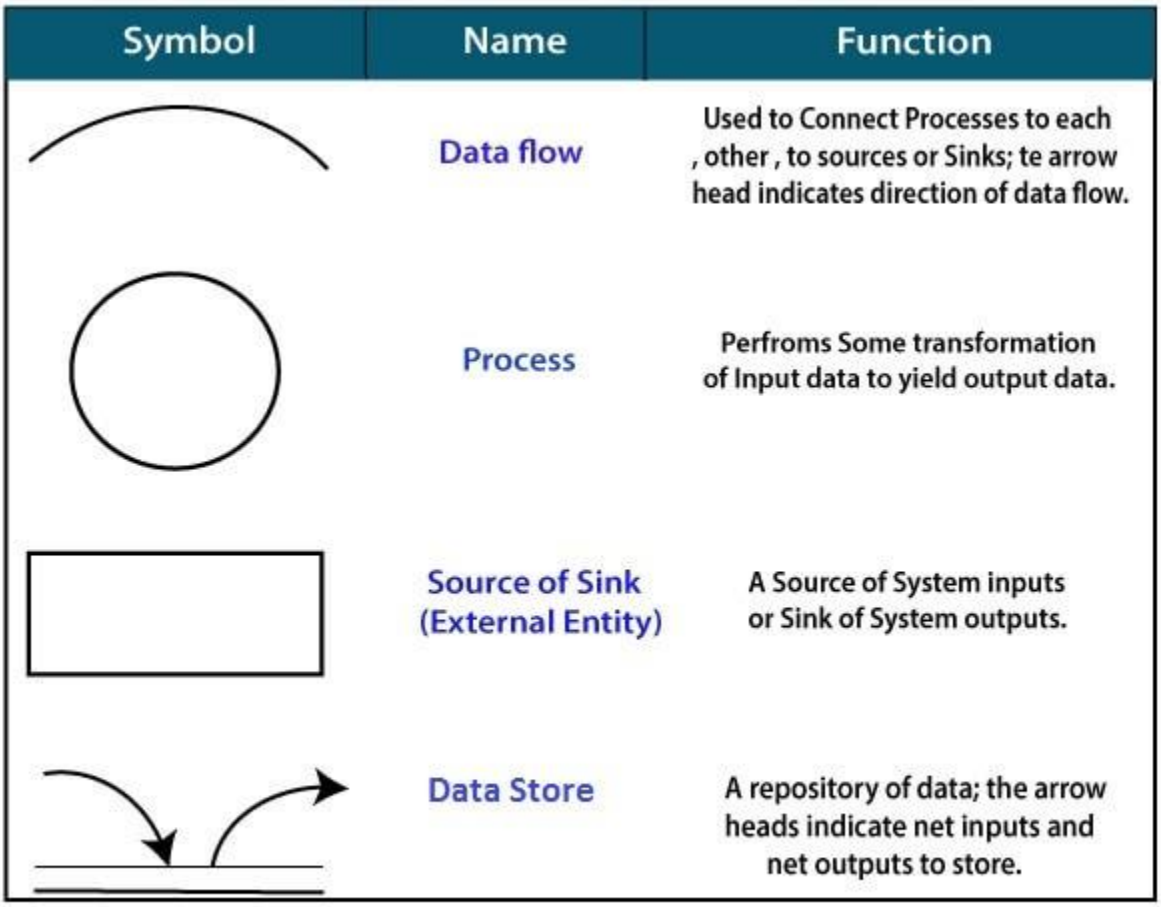
It shows how data enters and leaves the system, what changes the information, and where data is stored.

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. 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. Suppress logical decisions. If we ever have the urge to draw a diamond-shaped box in a DFD, suppress that urge! A diamond-shaped 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. Do not become bogged down with details. Defer error conditions and error handling until the end of the analysis.

Standard symbols for DFDs are derived from the electric circuit diagram analysis and are shown in fig:



Symbols for Data Flow Diagrams

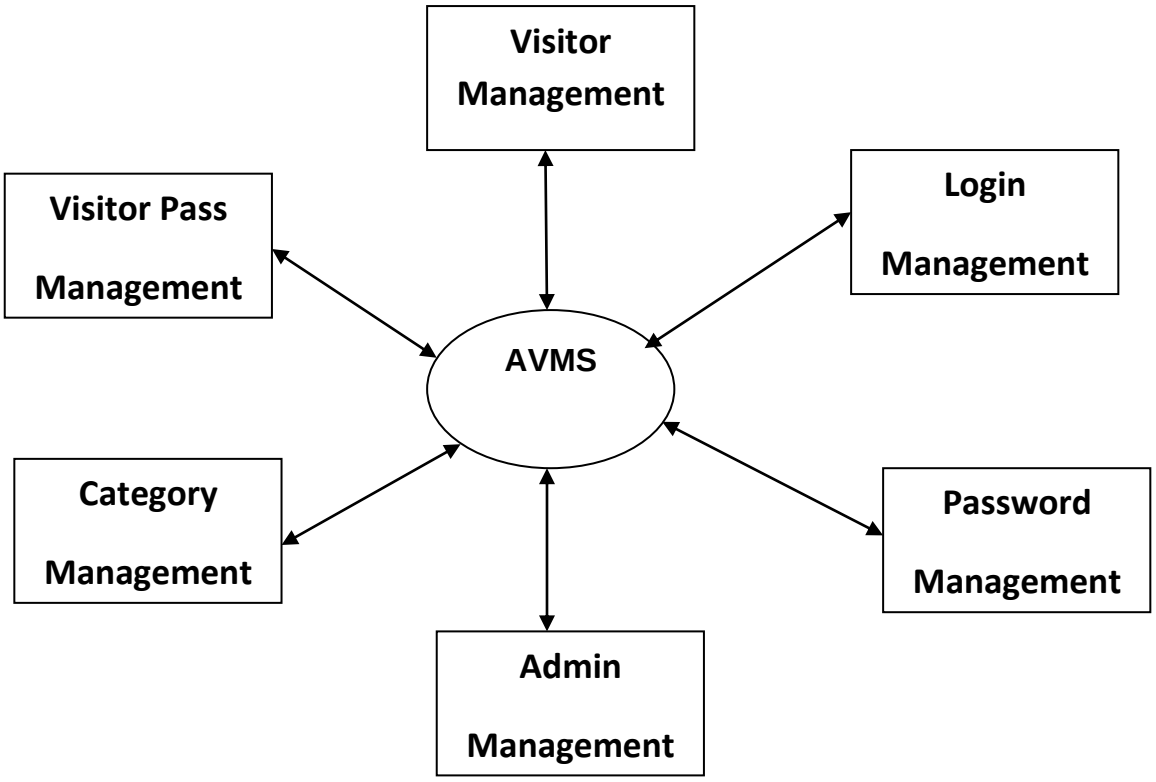
Circle: A circle (bubble) shows a process that transforms data inputs into data outputs.

Data Flow: A curved line shows the flow of data into or out of a process or data store.

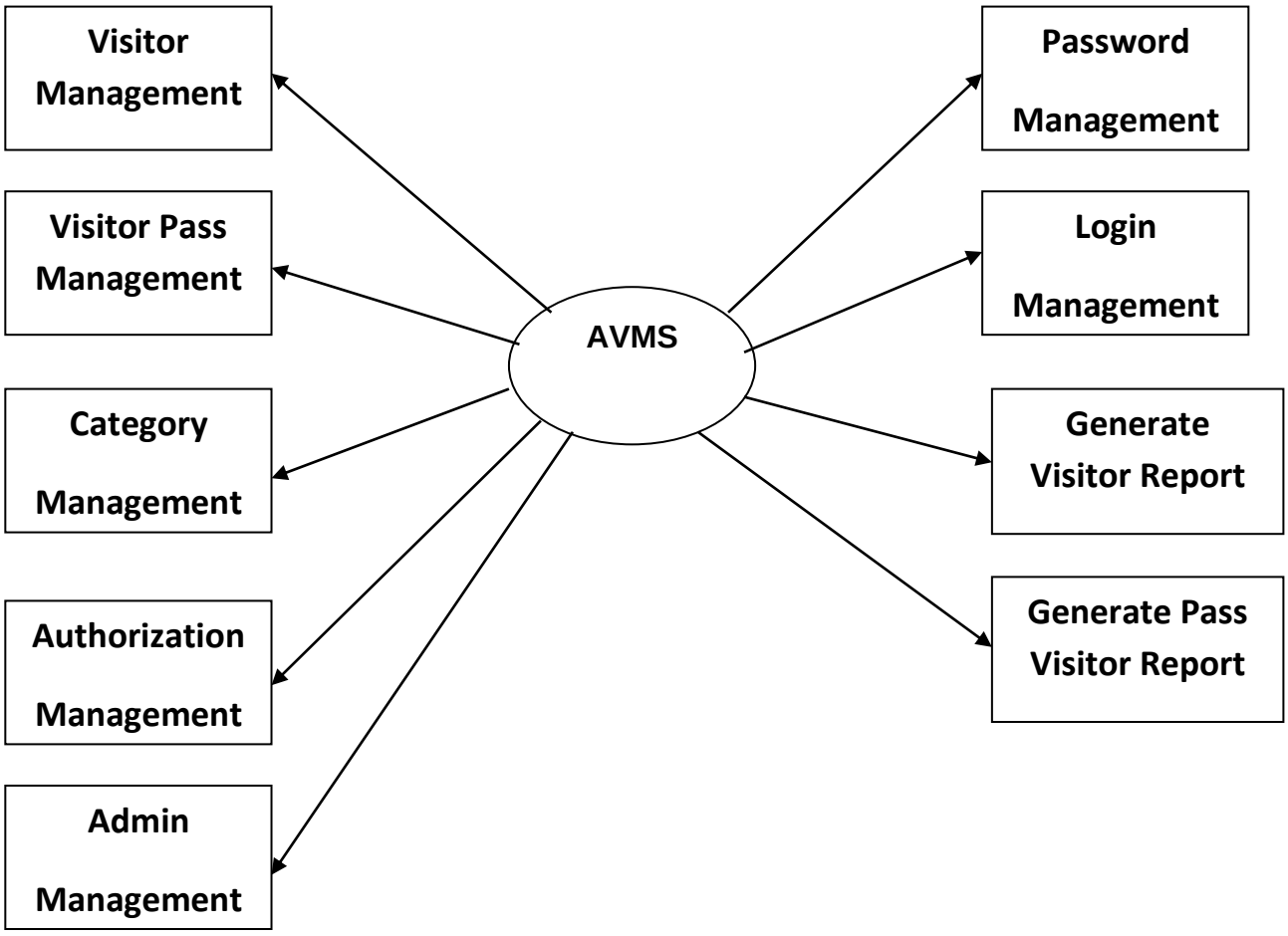
Data Store: A set of parallel lines shows a place for the collection of data items. A data store indicates that the data is stored which can be used at a later stage or by the other processes in a different order. The data store can have an element or group of elements.

Source or Sink: Source or Sink is an external entity and acts as a source of system inputs or sink of system outputs.

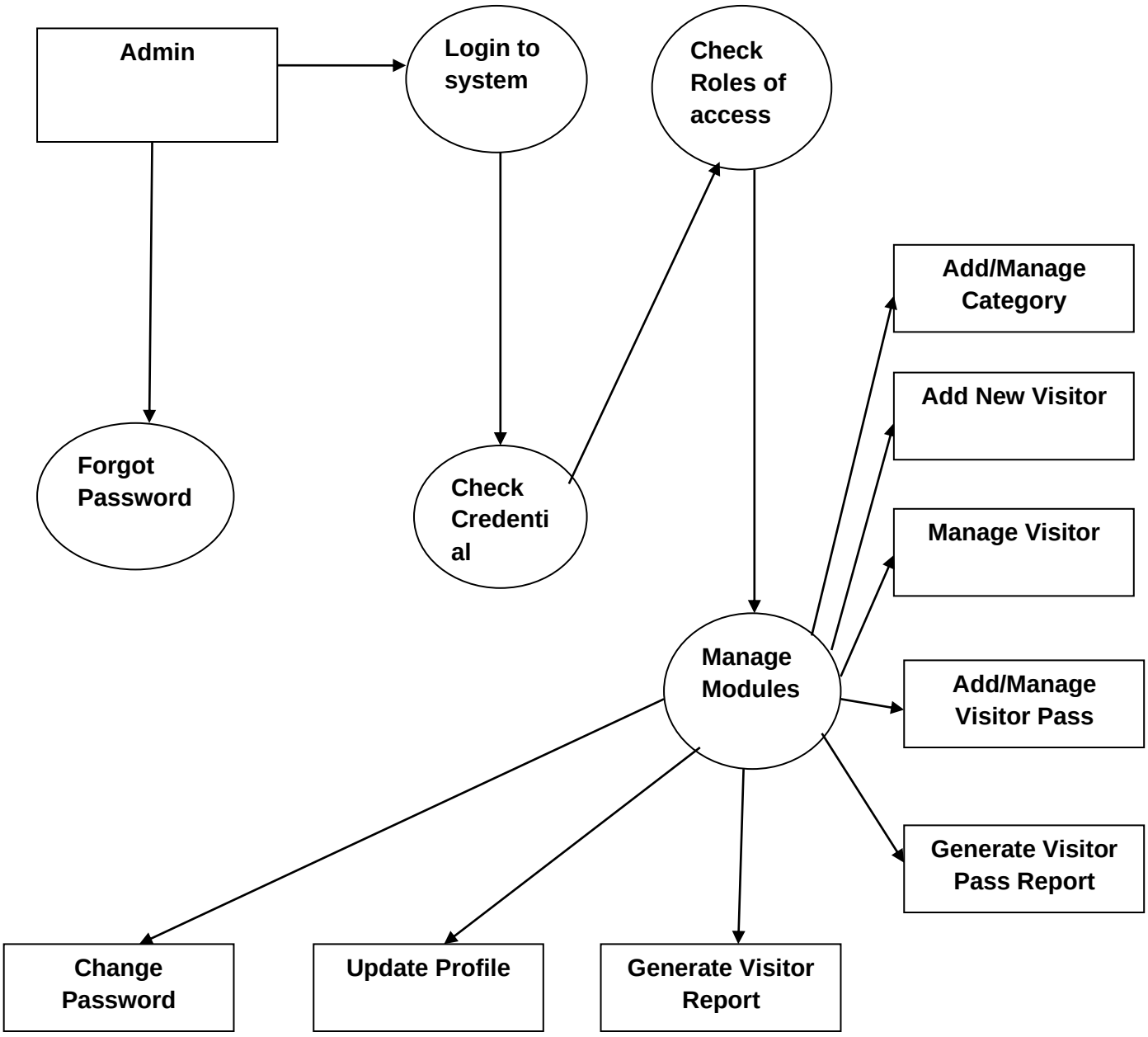
Zero Level DFD



First Level DFD



Second Level DFD



MySQL Data Tables:

Admin Table:(Table name is admin)

This table store admin personal and login details.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	ID	int(5)			No	*None*		AUTO_INCREMENT
2	AdminName	varchar(45)	latin1_swedish_ci		Yes	*NULL*		
3	UserName	char(45)	latin1_swedish_ci		Yes	*NULL*		
4	MobileNumber	bigint(11)			Yes	*NULL*		
5	Email	varchar(120)	latin1_swedish_ci		Yes	*NULL*		
6	Password	varchar(120)	latin1_swedish_ci		Yes	*NULL*		
7	AdminRegdate	timestamp			Yes	current_timestamp()		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	ID	1	A	No	

Visitor Table: (Table name is tblvisitor)

This table store the visitor details and admin remark

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	ID	int(5)			No	*None*		AUTO_INCREMENT
2	categoryName	varchar(120)	latin1_swedish_ci		Yes	*NULL*		
3	VisitorName	varchar(120)	latin1_swedish_ci		Yes	*NULL*		
4	MobileNumber	bigint(11)			Yes	*NULL*		
5	Address	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
6	Apartment	varchar(120)	latin1_swedish_ci		No	*None*		
7	Floor	varchar(120)	latin1_swedish_ci		No	*None*		
8	WhomtoMeet	varchar(120)	latin1_swedish_ci		Yes	*NULL*		
9	ReasontoMeet	varchar(120)	latin1_swedish_ci		Yes	*NULL*		
10	EnterDate	timestamp			Yes	current_timestamp()		
11	remark	varchar(255)	latin1_swedish_ci		Yes	*NULL*		
12	outtime	timestamp			Yes	*NULL*		ON UPDATE CURRENT_TIMESTAMP()

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	ID	3	A	No	

Category Table: (Table name is tblcategory)

This table stores the category of visitor.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**id**	int(11)			No	*None*		AUTO_INCREMENT
2	**categoryName**	varchar(120)	utf8mb4_general_ci		Yes	*NULL*		
3	**creationDate**	timestamp			Yes	current_timestamp()		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	9	A	No	

Visitor Pass Table: (Table name is tblvisitorpass)

This table stores the details of visitor pass.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**ID**	int(5)			No	*None*		AUTO_INCREMENT
2	**passnumber**	bigint(20)			Yes	*NULL*		
3	**categoryName**	varchar(120)	latin1_swedish_ci		Yes	*NULL*		
4	**VisitorName**	varchar(120)	latin1_swedish_ci		Yes	*NULL*		
5	**MobileNumber**	bigint(11)			Yes	*NULL*		
6	**Address**	varchar(250)	latin1_swedish_ci		Yes	*NULL*		
7	**Apartment**	varchar(120)	latin1_swedish_ci		No	*None*		
8	**Floor**	varchar(120)	latin1_swedish_ci		No	*None*		
9	**passDetails**	varchar(120)	latin1_swedish_ci		Yes	*NULL*		
10	**creationDate**	timestamp			Yes	current_timestamp()		
11	**fromDate**	date			Yes	*NULL*		
12	**toDate**	date			No	*None*		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	ID	2	A	No	
ID	BTREE	No	No	ID	2	A	No	

Implementation and System Testing

After all phase have been perfectly done, the system will be implemented to the server and the system can be used.

System Testing

The goal of the system testing process was to determine all faults in our project .The program was subjected to a set of test inputs and many explanations were made and based on these explanations it will be decided whether the program behaves as expected or not. Our Project went through two levels of testing

1. Unit testing

2. Integration testing

UNIT TESTING

Unit testing is commenced when a unit has been created and effectively reviewed .In order to test a single module we need to provide a complete environment i.e. besides the section we would require

- The procedures belonging to other units that the unit under test calls
- Non local data structures that module accesses
- A procedure to call the functions of the unit under test with appropriate parameters

1. Test for the admin module

- **Testing admin login form-**This form is used for log in of administrator of the system. In this form we enter the username and password if both are correct administration page will open otherwise if any of data is wrong it will get redirected back to the login page and again ask the details.

- **Report Generation:** admin can generate report from the main database.

INTEGRATION TESTING

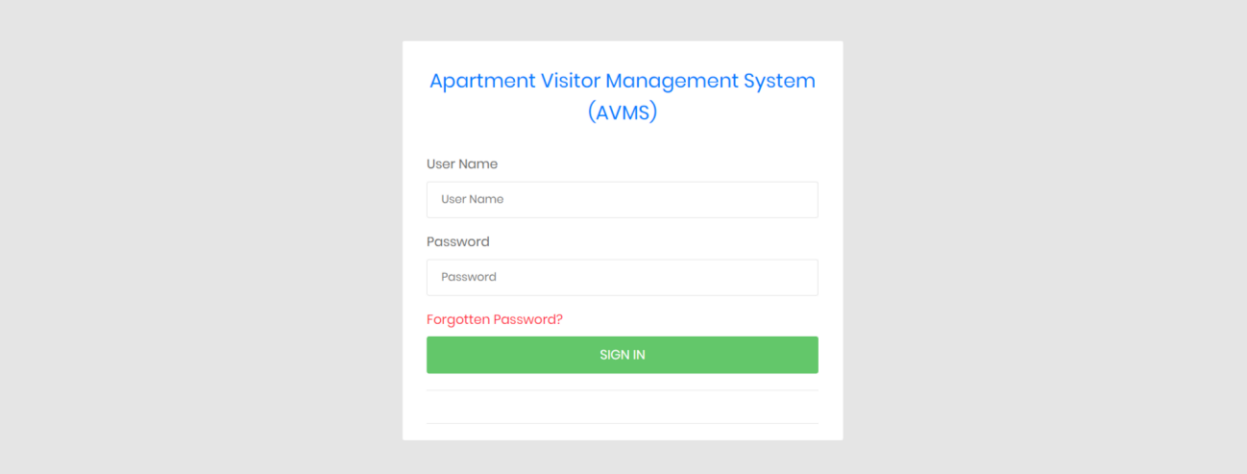
In the Integration testing we test various combination of the project module by providing the input.

The primary objective is to test the module interfaces in order to confirm that no errors are occurring when one module invokes the other module.

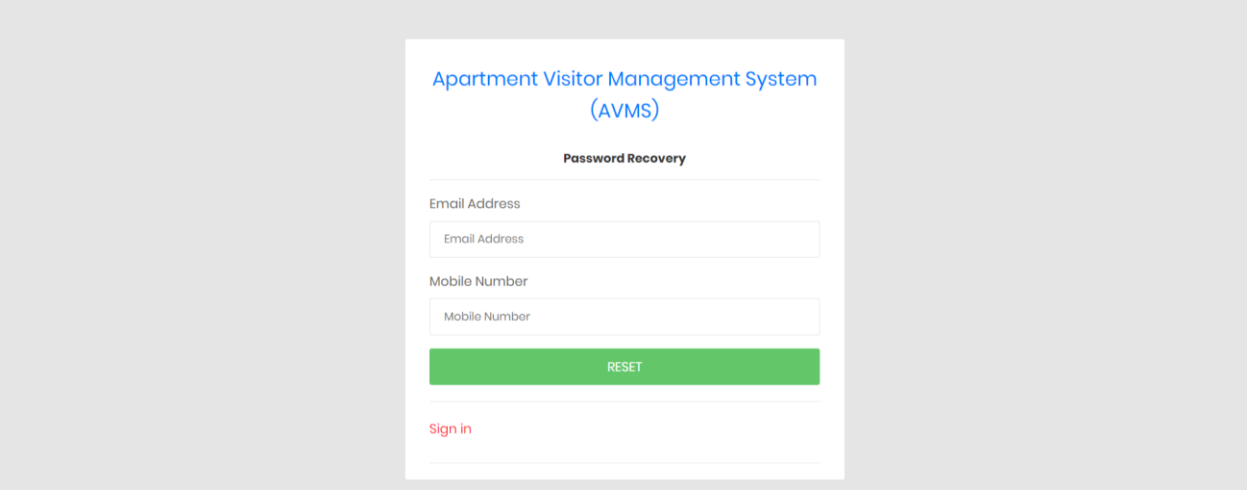
Evaluation (Project Outputs)

Project URL: http://localhost/avms

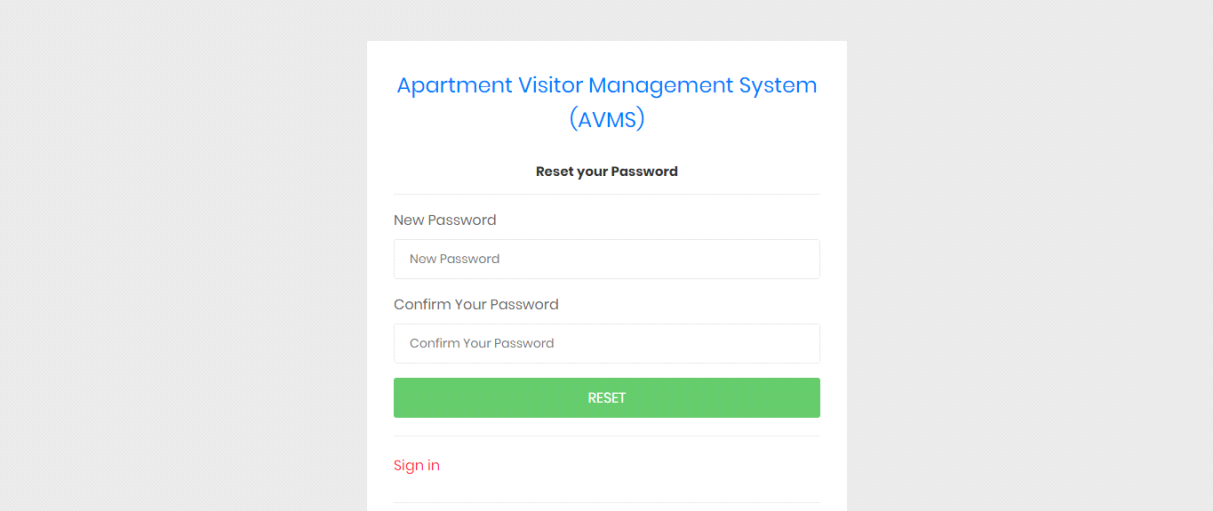
Login Page



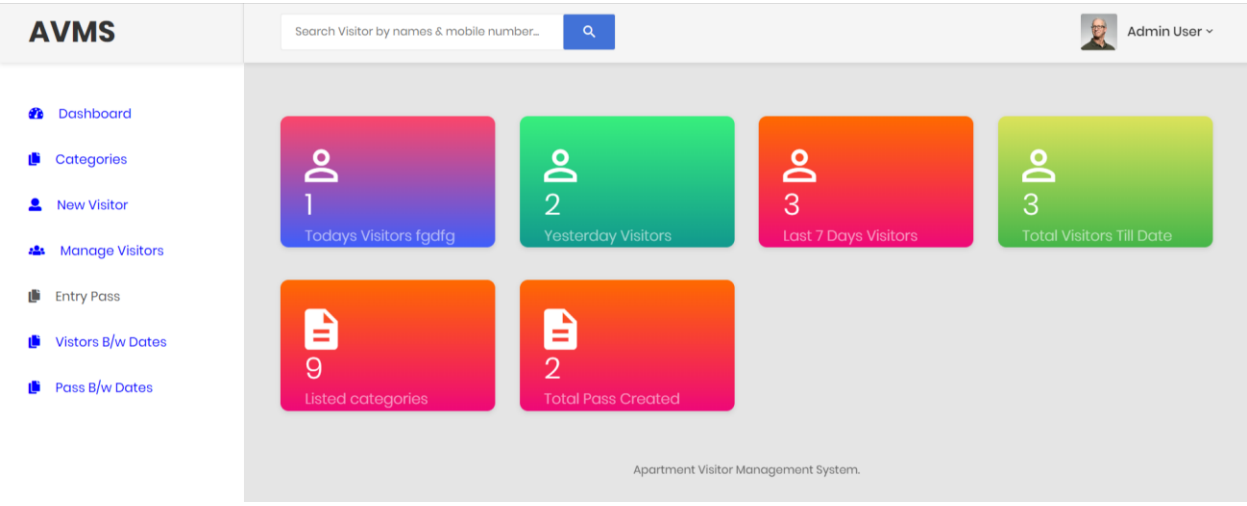
Forgot Password



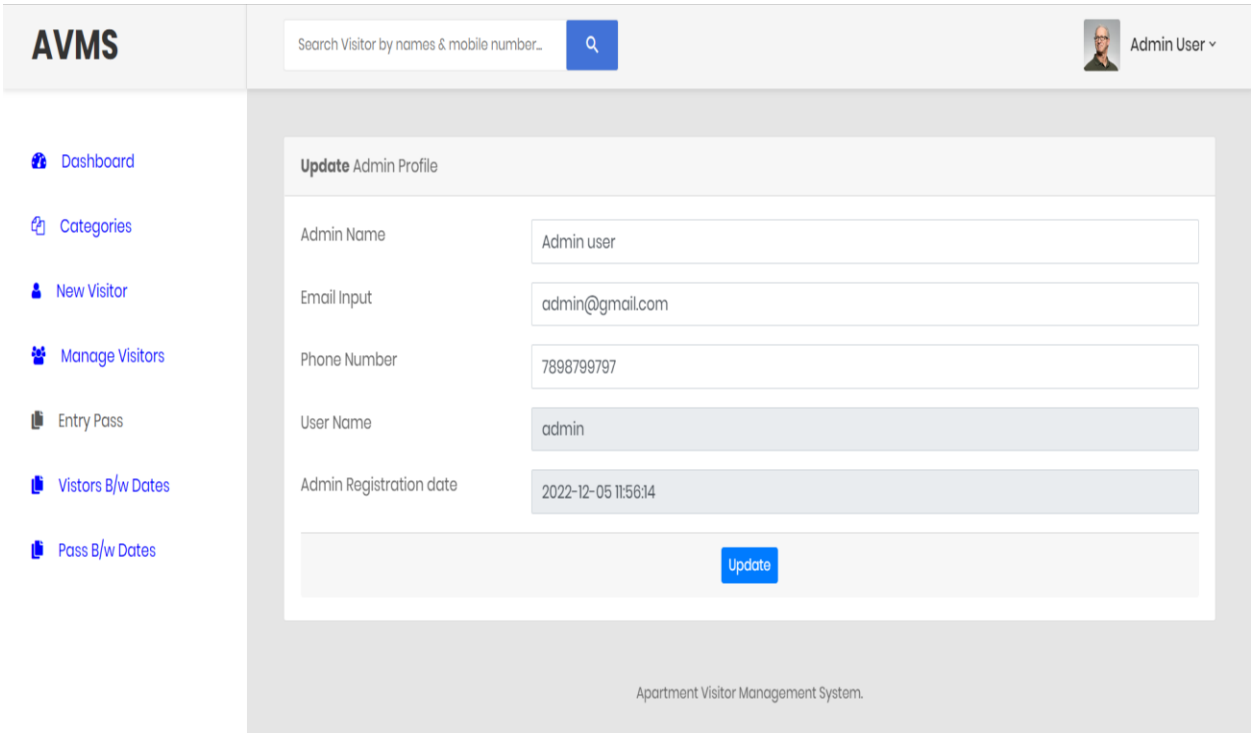
Reset Password



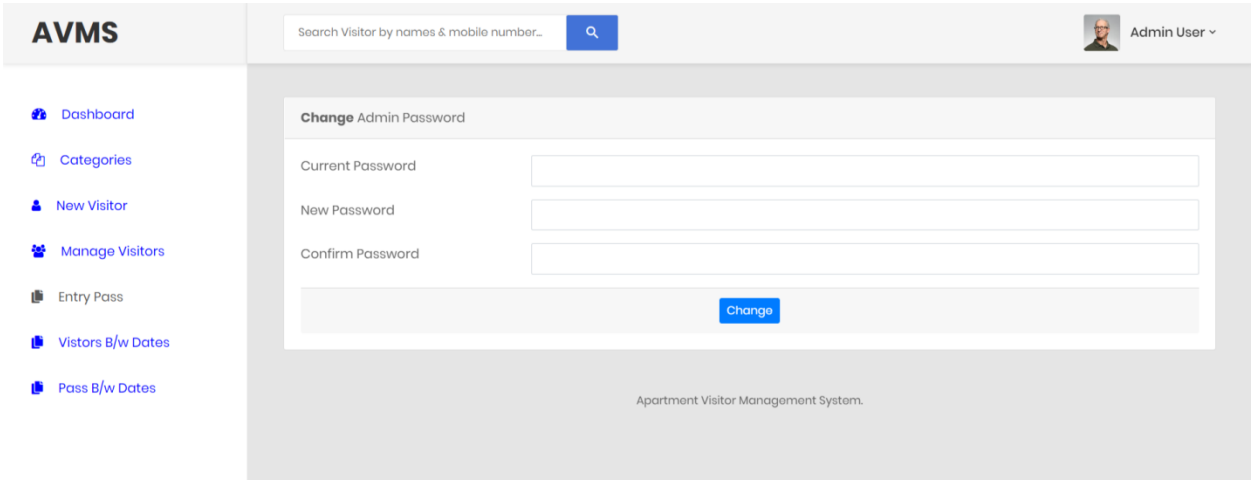
Dashboard



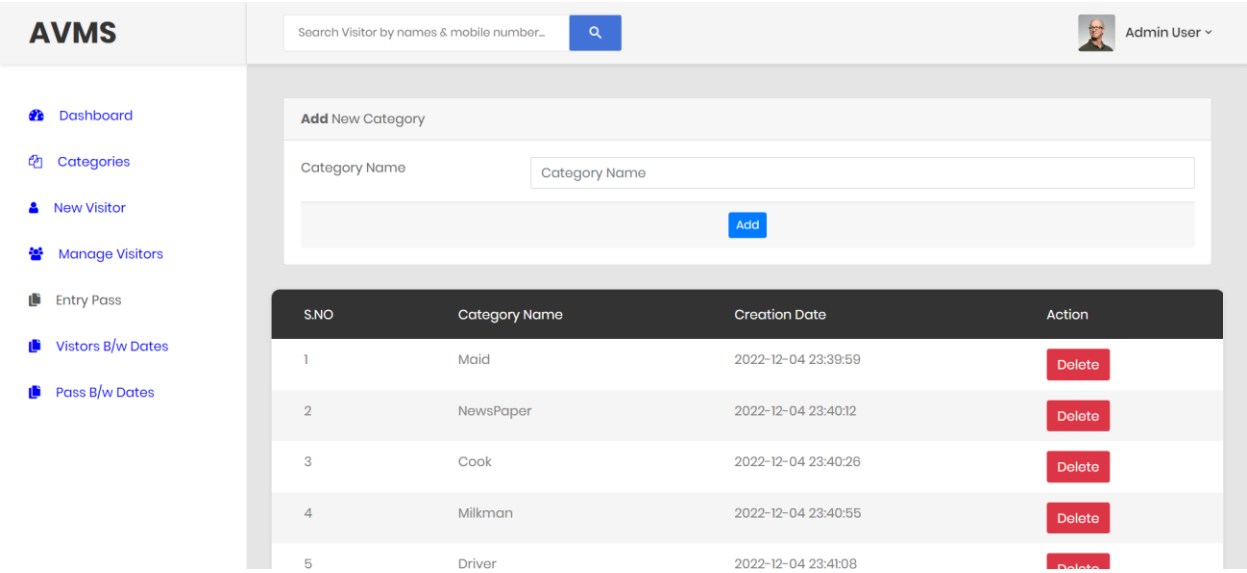
Admin Profile



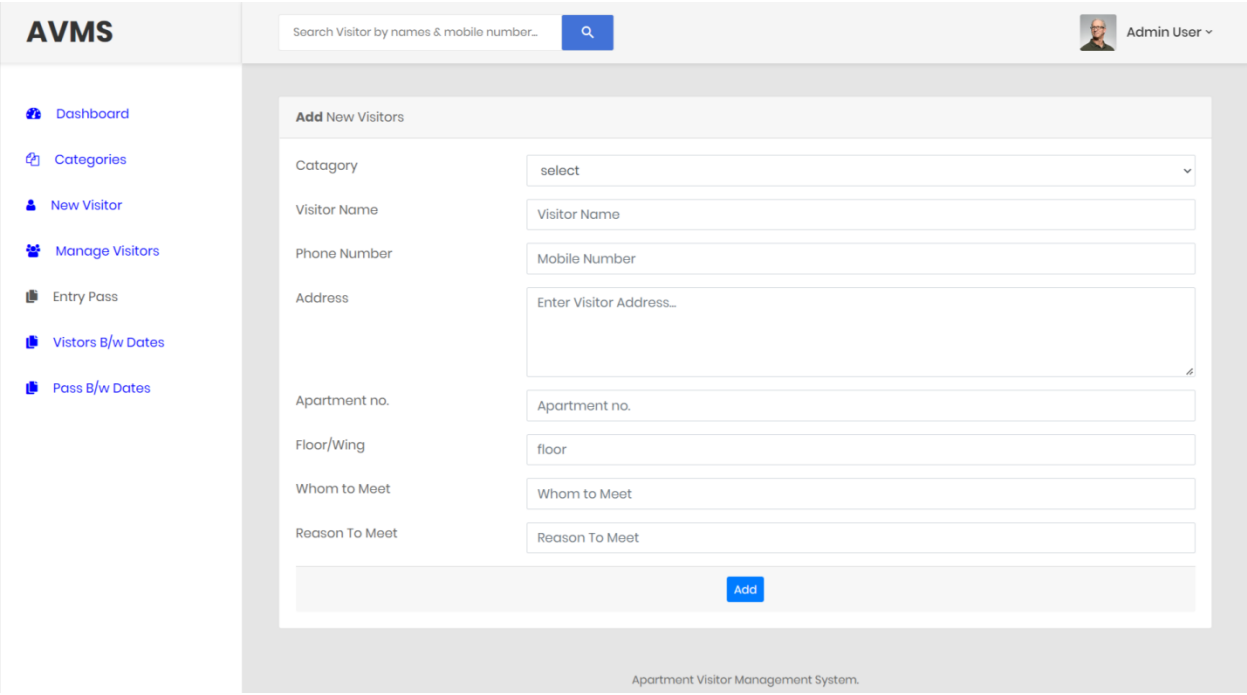
Change Password



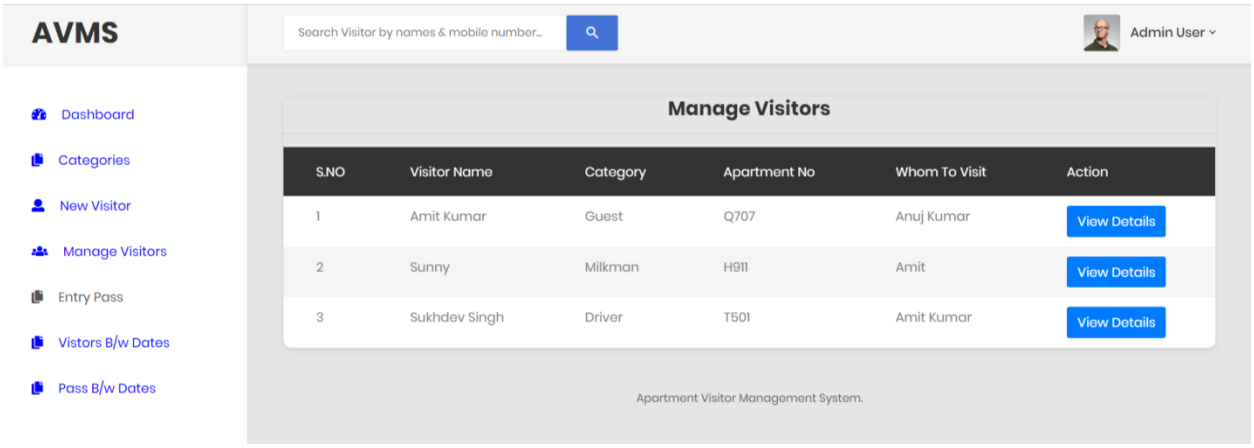
Manage Category



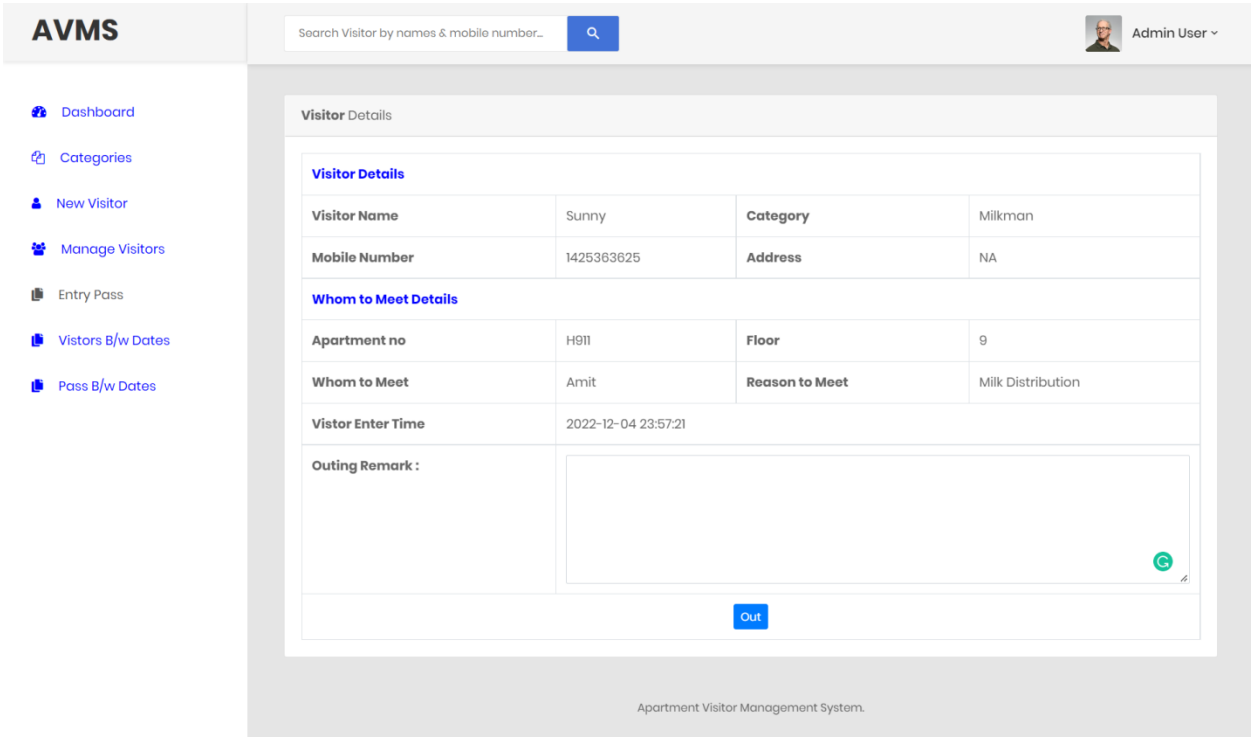
Add New Visitor



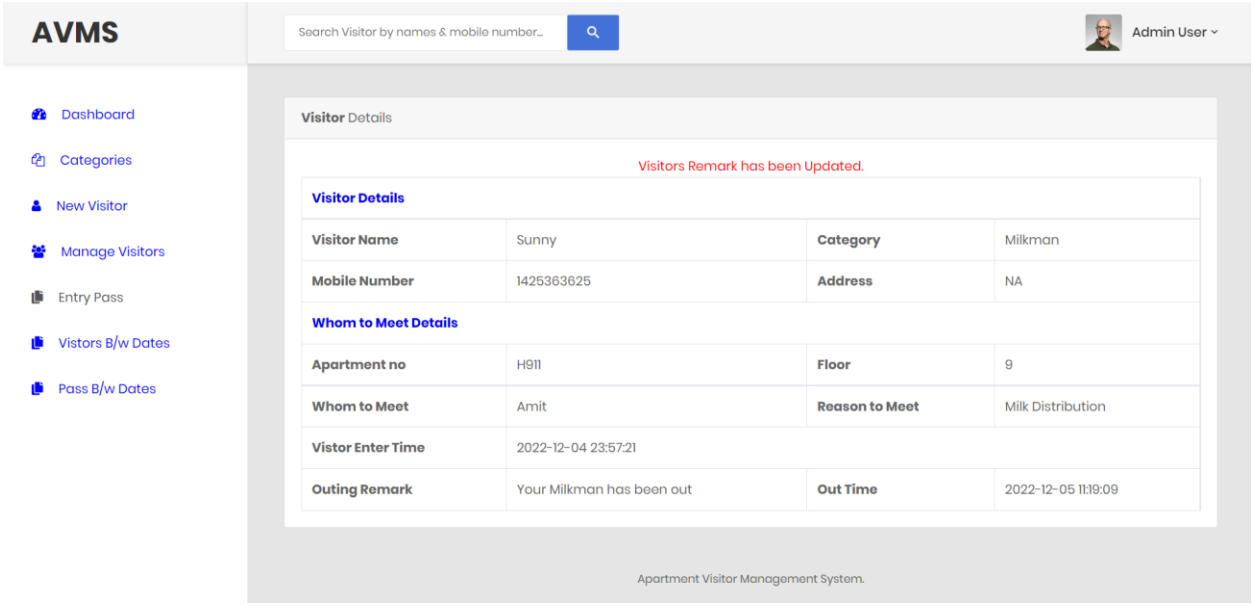
Manage Visitor



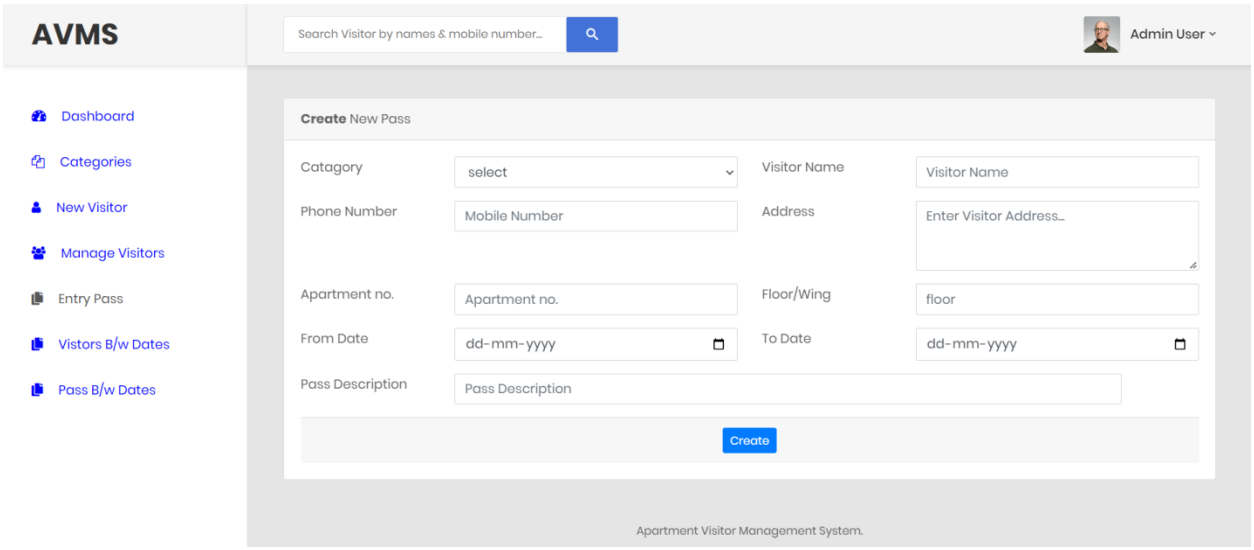
Visitor Detail



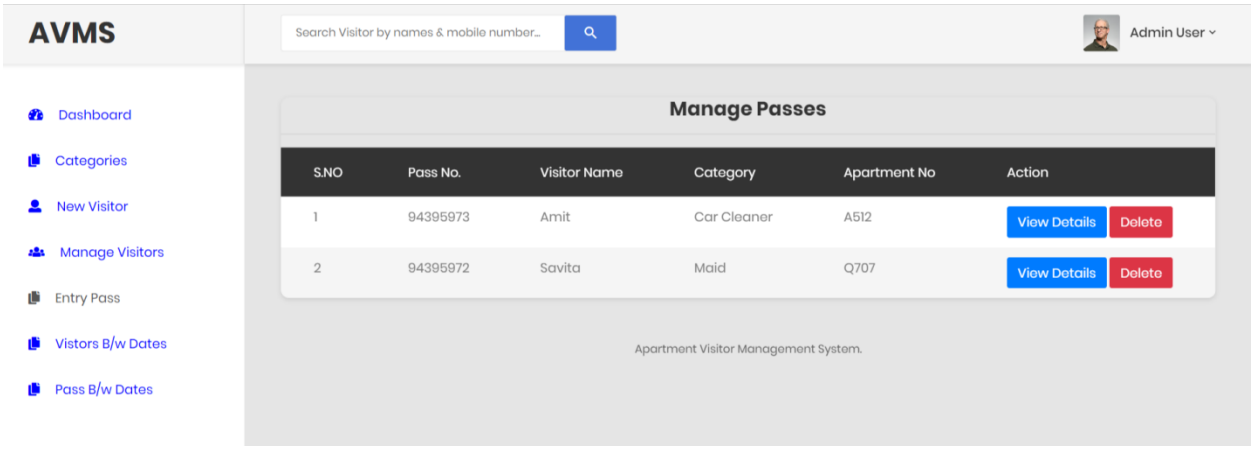
Visitor Detail after Update



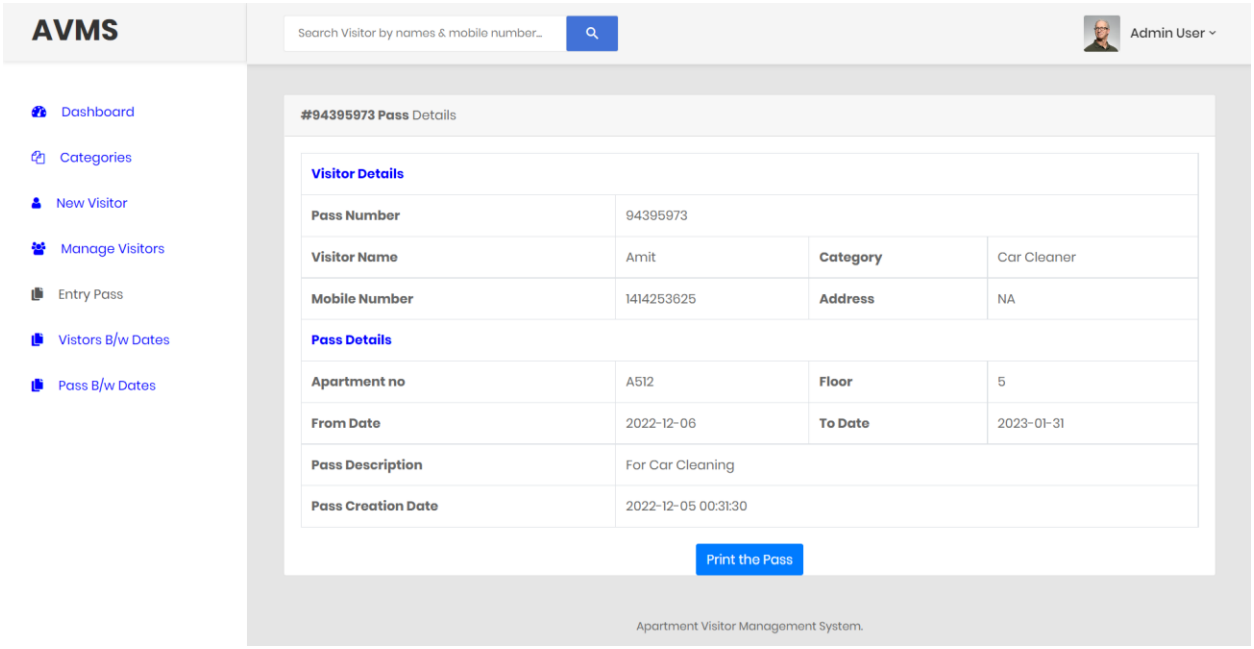
Create Entry Pass



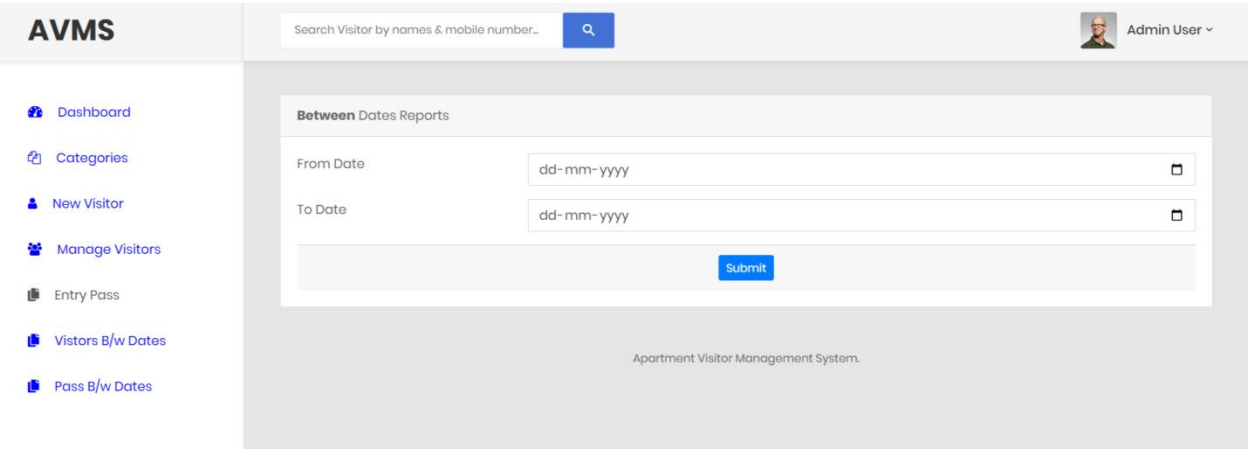
Manage Entry Pass



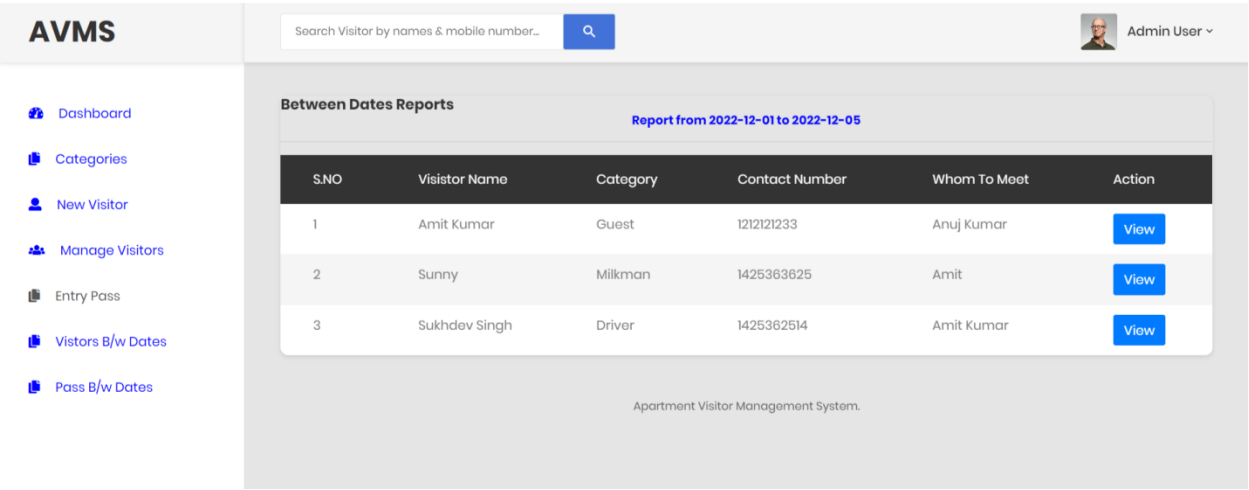
View detail of entry pass



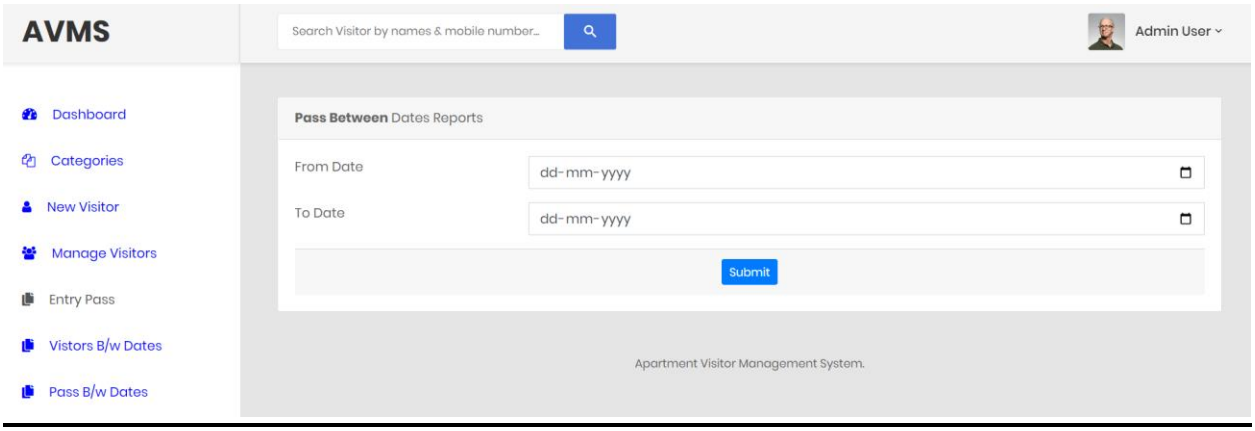
Between Dates Reports



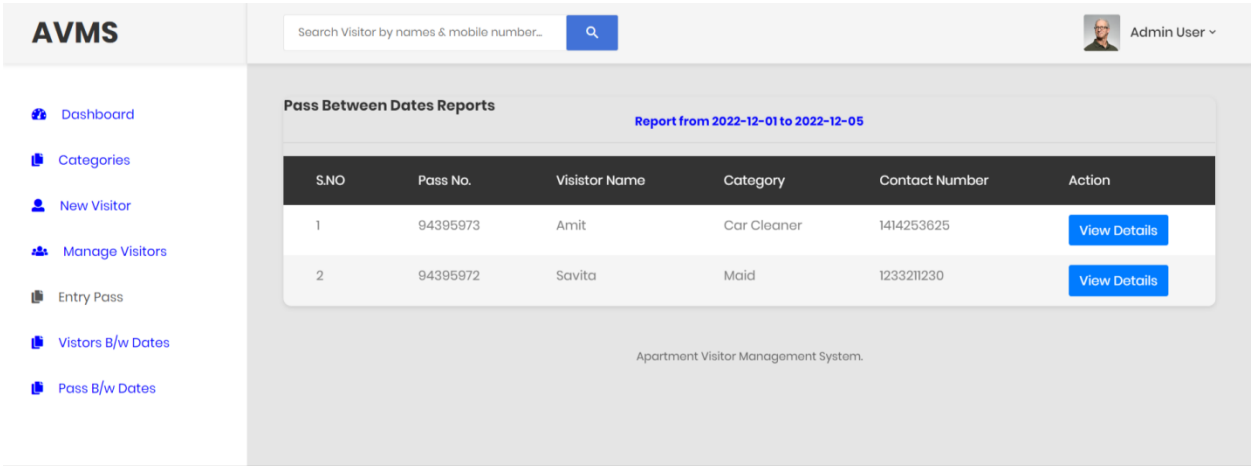
View Details of Between Dates Report



Between Dates Report of Entry Pass



View Details Between Dates Report of Entry Pass



Conclusion

This Application provides a computerized version of Apartment Visitor Management System which will benefit the society of gate guards who have to maintain a bulky and very hard to maintain record books for all visit who visit in the society for various reasons.

It makes entire process online and can generate reports. It has a facility of staff's login where staff can fill the visitor details and generate report.

The Application was designed in such a way that future changes can be done easily. The following conclusions can be deduced from the development of the project.

- Automation of the entire system improves the productivity.
- It provides a friendly graphical user interface which proves to be better when compared to the existing system.
- It gives appropriate access to the authorized users depending on their permissions.
- It effectively overcomes the delay in communications.
- Updating of information becomes so easier.
- System security, data security and reliability are the striking features.
- The System has adequate scope for modification in future if it is necessary.

References

For PHP

- https://www.w3schools.com/php/default.asp
- https://www.sitepoint.com/php/
- https://www.php.net/

For MySQL

- https://www.mysql.com/
- http://www.mysqltutorial.org

For XAMPP

- https://www.apachefriends.org/download.html