



RENT MANAGEMENT SYSTEM

Developed By : AAYUSH SAHAY
Enrollement No : 21303334049
Session : 2021-2024
Institution : INDIAN INSTITUTE OF BUSINESS MANAGEMENT , PATNA
College Code : 334
Under Guidance OF: Prof. Shahbaz Shakeb
Submitted To ARYABHATTA KNOWLEDGE UNIVERSITY , PATNA in
Partial fulfillement of the Requirements for the Award of the Degree
Bachelor in Computer Applications (BCA)

Indian Institute of Business Management , Patna

Near Patna Museuem , Buddh Marg , Patna-800001

**CERTIFICATE**

This is to certify that , Aayush Sahay (21303334049) is a bonafide student of our college and has carried out a project entitled "**RENT MANAGEMENT SYSTEM**" under my guidance.

The project has been submitted during the academic year 2023-2024 in partial fulfillment of the requirement of the course of

**Degree in Bachelor of Computer Applications (BCA) by
Aryabhatta Knowledge University , Patna**

Guided By

Principal Stamp and Signature

External Examiners

1. _____

2. _____

DECLARATION

I am Aayush Sahay hereby declare that project entitled "Rent Management System" is bonafide work carried out by me under guidance of Prof. Shahbaz Shakeb The Project Presented in this report is my original work and have not been presented from any other University . This project has been submitted as a partial fulfillment of the requirement for the Course of Degree in Bachelor of Computer Applications of Aryabhatta Knowledge University , Patna

PLACE : PATNA

SIGNATURE OF THE STUDENT

DATE

ACKNOWLEDGEMENT

I thank God for privilege to pursue the Course of Bachelor in Computer Applications in the Institution of **Indian Institute of Business Management ,Patna.**

I extremely thank our Honorable Principal Mr. Ganesh Pandey Sir for providing good Learning Opportunities and Lab Facilities in our Institution.

I deem this opportunity to express my sincere thanks to Mr. for their encouragement throughout the Project.

I thank and appreciate all the respondents who spent their valuable time in responding to my questionnaire.

PLACE : PATNA

SIGNATURE OF THE STUDENT

DATE:

INDEX

SL. NO	CONTENTS	Page No.
1	Introduction	1-3
	1.1 Rent Management System	
	1.2 Aims and Objectives	
	1.2.1 Purpose of the Project	
	1.2.2 Scope of the Project	
	1.2.3 Goals of the Project	
2	System Analysis And Requirements Gathering	4-6
	2.1 Introduction	
	2.2 Existing System	
	2.3 Problems in the Existing System	
	2.4 Solutions aimed by Proposed System	
	2.5 Feasibility Studies	
	2.5.1 Types of Feasibility Studies	
3	Hardware and Software Requirements	7-7
	3.1 Hardware Requirements	
	3.1.1 Software Requirements for the Developers of the System	
	3.1.2 Software Requirements for the End Users of the System	
	3.1.3 Hardware Requirements for the Developers of the System	
	3.1.4 Hardware Requirements for the End Users of the System	
	System Design	8-32
	4.1 Introduction to S/w Engineering Style Followed	
	4.2 Data Flow Diagrams	
	4.3 SQL Tables	
	4.4 E-R Diagrams	
5	System Coding and Implementation	33-304
	5.1 Introduction To PHP	
	5.2 Introduction To MySQL RDBMS	
	5.3 Source Code	
	5.4 Documentation	
6	Software Testing	305-308
	6.1 Introduction	
	6.2 Strategic Approaches for Software Testing	
	6.3 Unit Testing	
7	Future Enhancement	309-310
8	Conclusion	311-311
9	Bibliograph & References	312-314

Project Report**Rent Management System**

10		SYNOPSIS : RENT MANAGEMENT SYSTEM	315-369
11		Installation Guide	370-371

CHAPTER - 1

INTRODUCTION

RENT MANAGEMENT SYSTEM

CHAPTER-1	Introduction
1.1	Rent Management System
1.2	Aims and Objectives
1.2.1	Purpose of the Project
1.2.2	Scope of the Project
1.2.3	Goals of the Project

1.1 Rent Management System

The Rent Management System is a web based application that follows the client-server architecture inorder to manage the business information and provide utility to the landlord. This is used in a room rental business where landlord could register , add rooms for display , could do the agreement , store details , the rent receipts and produce the reports in the form of pdf , also show the total rent received. It will also provide the flexibility to manually add the rent and electricity bill received such that the bills are made under the requirements of the landlord.

1.2 Aims and Objectives

The aims and objectives are carried out by the specific organs/modules of this application software. This software comprises of Seven main modules which are briefly described inorder to accomplish the objectives of the project .

(i) Login Module :

This is going to be the entry point for the landlord to the Software. This will allow the access of data necessary to the business point of view to be accessed by the owner only. This will prevent any 3rd party access , Thus protecting the data from malice attacks and unauthorised access.

(ii) Room Management Module :

The web based application allows the landlord to add the rooms , their specifications like interiors pictures , the size of the room inorder to present the consumers the availability for lease. The landlord is also capable of seeing the list of the current tenants occupying rooms. It would specify that is a room is under the state of being occupied or is ready to be leased . The landlord can also add or remove the rooms on the basis of the situation , but he will not be able to remove the currently occupied rooms, unless he removes the tenants who are currently occupying it.

(iii) Tenant Management Module :

This allows to register people who are interested or in need to occupy a room on rent. Incase they want to drop their plans they will get their details deleted by the landlord. The data of the communication details will be stored . Incase any tenant wants to vacate the room , he may pay the rent and get the room be vacated and after some time , that room will be available to be put for lease again.

(iv) Agreement Module :

This software is capable of making Electronic Agreements in presence of the Landlord and the consumer, that he / she wishes to occupy a room on rent , on the cost applicable to it , and by following the guidelines of the landlord . This software is built keeping in mind the rules to consider and protect the consumers rights. It allows the business to operate legally and safely. The agreement modules requires the name , the photo , the id of the people who tend to occupy the room abiding by the guidelines of the civil law and the landlord private property guidelines and finally making the pdf for the purpose of verification and for future references. This module will help to prevent the time taking paperwork involved in the legal procedures that are required to be followed to operate the small scale private businesses.

(v) Rent Collection Module:

This module lists the current tenants. This will show the status of the rent received till what date. This will list all the rooms and their tenants one by one , the landlord will be able to make the bill on the basis of the Monthly rent , Electricity bill , any dues in previous month. It will do the totalling and make the report in the form of a pdf.

(vi) Reports and Data Backup Module:

The Communication Details of the Tenants , Agreements , Monthly bill Receipts , Rent Received receipts are made in the form of the pdf , but also the database is also maintained in the future reference cases . The Data Backup module produces the copy of all the data and the pdf reports to the secure folders present in the drives that is going to increase the reliability , as this would come into handy when the application is subjected to the catastrophic failure , the data could still be retrieved easily as we tend to keep backing up the data and the reports at each step.

(vii) Password Reset Module

This is a web-application that features a login-logout system that is specifically tailored to meet the requirements of the client . In the situation of the client losing the password code , He may reset it according to the software key that is going to be different for each such software . This project is built considering the purpose of Learning and not for the commercial purposes , we have decided to keep the software key as 0000 , and entering that will enable the landlord to enter his email and set the new password and then will redirect the landlord to the login page .

1.2.1 The Purpose of the Project

1. The purpose is to manage the business of the landlord.
2. Its purpose is to provide all the facilities that automate the pen and paper procedure.
3. To make the data portable and standardise all the business operations using pdf reports
4. To make the management of the business carried out in a very reliable manner.

1.2.2 Scope of the Project

1. The scope of the web application is very vast, that are very less dependent on the Technology as compared to the Desktop Applications
2. Very Practical as to be used in commercial purposes / Rental Business management
3. Can be modified easily and redeployed incase the client encounters the need of more features than that are achievable by the above Software.
4. The Project is made keeping the flexibility to be provided at each steps to comply with the future Laws and Regulations.

1.2.3 Goals To Accomplish by the Proposed Software:

1. This application aims to provide the compactness of all procedures in the Room Rental Business.
2. The application's goal is to provide interface to easily maintain the records of Income.
3. This application's goal is to provide the convenience in the Business Operations.
4. The application's main goal is to bind the tenants to follow the rules and guidelines..

CHAPTER - 2

SYSTEM ANALYSIS

RENT MANAGEMENT SYSTEM

System Analysis And Requirement Gathering	
2.1	Introduction
2.2	Existing System
2.3	Problems in the Existing System
2.4	Solutions aimed by Proposed System
2.5	Feasibility Studies
2.5.1	Types of Feasibility Studies

2.1 Introduction to System Analysis

System analysis is used to examine a business problem . We identify its objectives and requirements and design the most optimal solution to fulfill the needs of the client. This is the very first step in the System Development where the developers come together to understand problems , the needs of the customer and determine the objectives of the Project.

The Key Aspects of System Analysis are as follows :

- (i) Identification of the Problem : It involves to identify the issues in the current system to shift the current system be automated to a software solution or when we tend to improve pre-existing software.
- (ii) Requirements Gathering : Once problems are identified , the next step is to gather and write down the requirements . This involves to communicate with the clients to gather the information about how is the system going to be designed
- (iii) Feasibility Study : Before the development is started , it is important to check the feasibility of the project , that includes evaluation of the technical , operational , financial and legal aspects to determine the feasibility of the proposed system.
- (iv)Analysis and Modelling : To get a very clear view of understanding , analysts develop various models such as data flow diagrams , use cases and Entity Relations diagrams . These models help the clients to visualise the system components and their functionalities.
- (v) Scope Definition : Defining the scope of the system is important to prevent adding excessive features to the system and to ensure that it stays within its limits. It identifies what is the part of the system and what is not. Thus helping to create an image of Compact Software while developing the software.

2.2 The Existing System

The Existing system is a process/procedure involving time-taking paperwork to keep a tenant/renter. Mr. Amit is a landlord and has to follow the below procedure to carry out his Private Room Rental Business .

- (i) Put up Posters for Advertising of the Rooms Available on Rent.
- (ii) Write down the communication details in the registers who are interested in the rooms.
- (iii) Give them a paper form to fill up the necessary details and appending the Identity Proof Like AADHAR CARD and Communication details.
- (iv) Then sending the copy to the local police authority and retain the main document for proof.
- (v) The form has an Agreement Section Stating to follow the rules and guidelines while occupying the room on rent on Mr. Amits Property.

2.3 The Problems in the Existing System :

Since the Work is the traditional pen and paper way to manage the Room Rental Business.

- (i) The process takes a lot of time . We can estimate it to 4 to 6 days.
- (ii) The renter may keep many people on the room without specifying earlier. According to the Tenant Rules : The landlord must be informed and must retain the id proof of the necessary people occupying the room on rent by their name .
- (iii) All paper work is time taking . One has to exercise extreme caution while handling the main documents to renter registration. Incase the main document is lost then it causes great trouble , and there are chances that everytime the operator may not manage them in digital formats in organised way.
- (iv) To manually feed and maintain the records in the register . Takes good amount of time in searching among the previous records. In modern world the records in the digital format is much preffered and manageable than handling paper reports.

2.4 The Solutions Aimed to be provided by the Proposed System :

- (i) The Software will make the work done in Registration and Agreement in 2-3 days.
- (ii) The renter will have to specify the no. of the people at the room , that would cause them to be liable for themselves legally while occupying the room on rent.
- (iii) The software will enable the creating of documents in Digital Formats (pdf) stating all the details and the transactions for the purpose of acknowledgement and future references.
- (iv) This will also provide the ease to manually feed in the records and produce the reports .

2.5 Feasibility Study

The term "Feasible" means "Is Possible ?" in English. This is the crucial step in the field of Software Analysis in which the feasibility of the proposed system is evaluated. This is also the measure of the software product in terms of how much beneficial and practical it would be in development of the Software for the Client's Organisation. The feasibility study concentrates down to Technical , Operational , Economic , Market , Resource and Legal Feasibility .

2.5.1 Different Feasibility Studies Carried out for the Proposed System :

- (i) Technical Feasibility : In Technical Feasibility current resources both in the form of Hardware and Software Technology are analysed to prepare to develop the project. It requires the skills for the operator and the maintenance staff to carry out the operations in the chosen Technology.
- (ii) Operational Feasibility : This requires to analyse the operative aspect of the software concentrating on the point of ease to operate and to maintain the software solution. This heavily relies on the Technical Aspects chosen to provide ease to the operator.
- (iii) Economic Feasibility : It is the study of the cost and benefit of the Project before the project is created and presented to the client party. A detailed analysis is carried out to develop the software considering the hardware and software resources required to design and develop the software . If the project in itself is very costly , so the client may not agree to proceed , this help us to Estimate the price of production before hand inorder to make the clients agree for the confirmation to carry out the further steps of development. For each functionality the optimum price is estimated by use of the rules in the Software Price Metrics.
- (iv) Legal Feasibility : The legal feasibility is analysis in the view of considering the barriers of the legal requirements like the legal implementations in producing the software considering the Consumer Rights and the Data Protection Act .It is carried out to point all the functionality whether do they comply with the legal and the ethical requirements.
- (v) Schedule Feasibility : We analyse the deadlines to figure out how much time to use to prepare and with what work force / team required to prevent the project being delivered after the timelines are elasped. For this we Carry out the Project Planning by the help of PERT Chart and Gannt Chart to identify the Processes and resolving them into functionalities and finally analysing what time to give to create the specific functionality.
- (vi) Market Feasibility : This refers to evaluate the presence of such similar software solutions present in the market . The Project : Rent Management System targets the Room Rental Business area of the Market. We also analyse the prevelant rival software and try to provide all the necessary features to the clients requirements. If we see the client is satisfied , we can also produce software targeting the other Rental Busineess and Business Management Software.
- (vii) Resource Feasibility This involves the Development Party to do an inventory check inorder to analyse the current resources they have , their team and figure out whether it is under or over sufficient inorder to produce the software fulfilling the requirements of the client. The Requirements to make the porject optimally is analysed in this phase

The Above feasibility studies were conducted prior to formulate the Development Process of the Rent Management System. This project is Technically , Operationally ,

Economically, Legally and Resourcefully Feasible project that is the Software Solution to counter the Business Problems that occur in the Room Rental Business.

CHAPTER - 3

HARDWARE AND SOFTWARE REQUIREMENTS RENT MANAGEMENT SYSTEM

Hardware and Software Requirements	
3.1	Hardware And Software Requirements
3.1.1	Software Requirements for the Developers of the System
3.1.2	Software Requirements for the End Users of the System
3.1.3	Hardware Requirements for the Developers of the System
3.1.4	Hardware Requirements for the End Users of the System

3.1 Hardware and Software Requirements

The Development of The Software begins on the Developers Computer Resources. After Finishing the project the Software product is delivered to the Client Party. The Requirements are Specified prehand to ensure that the delivered product runs easily on the client's computer . Or in other Case the Client party may prepare by arranging the resources inorder to run the System in their Organisation. This can be also negotiated with the Developer Party , by the Client sharing the information of current resources they have , to which keeping in mind the Software would be developed.

3.1.1 Software Requirements for the Developers of the System:

- (i) Xammp Server
- (ii) Vs Code
- (iii) Chrome and Microsoft Edge Browser
- (iv) A web layout designing tools , or an Image Editor

3.1.2 Software Requirements for the End Users of the System

- (i) Xammp Server and php support
- (ii) Latest Browser : Google Chrome or Microsoft Edge

3.1.3 Hardware Requirements for the Developers of the System:

- (i) Microsoft Windows 10 Operating System
- (ii) Microporcessor : 2.30 Ghz
- (iii) RAM : 6 Gb
- (iv) HDD : 100 Gb

3.1.4 Hardware Requirements for the End Users of the System :

- (i) Microsoft Windows 10 / 11 Operating System
- (ii) Microprocessor : 2.30 Ghz Minimum
- (iii) RAM : 8 Gb (4Gb Minimum)
- (iv) HDD : 100 Gb Minimum

The higher the level of H/w and S/w configuration , the More best quality of the output will be produced towards the End Users of the Rent Management System.

CHAPTER - 4

SYSTEM DESIGN

RENT MANAGEMENT SYSTEM

SYSTEM DESIGN	
4.1	Introduction of S/w Engineering Style
4.2	Data Flow Diagrams
4.3	SQL Tables
4.4	ER - Diagrams

4.1 Style of Software - Engineering Followed

The Iterative Waterfall model is a software development methodology that combines elements of the traditional Waterfall model with iterative refinement. Here's an explanation of the Iterative Waterfall model and its advantages:

Explanation of the Iterative Waterfall Model

The Iterative Waterfall model follows a sequential design process, similar to the classic Waterfall model, but introduces the concept of iteration. Instead of completing the entire project in one pass through the phases, the Iterative Waterfall model allows revisiting previous phases based on feedback and findings in subsequent phases. This process is repeated until the system is fully developed and meets the requirements.

Phases of the Iterative Waterfall Model:

1. Requirements Analysis:

- Gather and document the requirements from stakeholders.
- Identify the scope and objectives of the project.

2. System Design:

- Create the overall system architecture and design.
- Define system modules, components, and interfaces.

3. Implementation:

- Develop the code for the system based on the design.
- Integrate the components and modules.

4. Testing:

- Test the system to identify defects and issues.
- Perform unit testing, integration testing, system testing, and user acceptance testing.

5. Deployment:

- Deploy the system to the production environment.
- Ensure that the system is operational and accessible to users.

6. Maintenance:

- Perform ongoing maintenance to address issues and implement enhancements.
- Continuously monitor the system for performance and reliability.

In the Iterative Waterfall model, each phase is revisited iteratively based on feedback and findings from subsequent phases. This allows for continuous improvement and refinement of the system.

Advantages of the Iterative Waterfall Model

1. Improved Requirement Handling:

- Allows for refining and revisiting requirements based on feedback from later stages.
- Reduces the risk of missing or misunderstood requirements.

2. Early Detection of Issues:

- Facilitates early identification and resolution of defects and issues through iterative testing.
- Enhances the overall quality and reliability of the system.

3. Flexibility:

- Provides more flexibility than the traditional Waterfall model by allowing iterative refinement.
- Accommodates changes in requirements and design more effectively.

4. Better Risk Management:

- Iterative nature helps in identifying and addressing risks early in the development process.
- Reduces the impact of risks on the project timeline and budget.

5. Enhanced User Feedback:

- Involves stakeholders and users throughout the development process.
- Ensures that the final system meets user expectations and requirements.

6. Structured Approach:

- Maintains a structured and disciplined approach to software development.
- Clearly defined phases help in managing the project systematically.

Application in the Rent Management System Project

In your Rent Management System project, using the Iterative Waterfall model would involve:

1. Gathering initial requirements from stakeholders about the functionality needed, such as tenant management, billing, payment tracking, etc.
2. Designing the system architecture to meet these requirements, including the database schema and user interface design.
3. Implementing the features incrementally, such as creating the tenant registration module, followed by billing and payment modules.
4. Testing each module thoroughly before moving on to the next iteration, ensuring that each part of the system works correctly and integrates well with the existing components.
5. Deploying and gathering feedback from users, allowing for further iterations where necessary to refine and enhance the system based on real-world usage and feedback.
6. Maintaining the system to address any issues that arise and to implement new features or improvements based on ongoing user feedback.

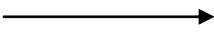
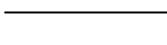
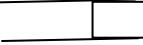
By adopting the Iterative Waterfall model, you would ensure that the Rent Management System is developed systematically while allowing for flexibility to adapt to changing requirements and feedback. This approach enhances the overall quality and usability of the system.

4.2 Dataflow Diagrams

Data Flow Diagram (DFDs):-

DFD stands for Data Flow Diagrams . This is also known as Bubble Charts through which we can represent the flow of data graphically in any information management system. We can easily understand the overall functionality of system because diagram represents the incoming dataflow and the outgoing dataflow and Stored data in the Graphical Form. It describes the flow of data in terms of input and output .

A DFD uses a number of notations or symbols that represent the flow of data in any System. These notations are illustrated as follows:

Serial No.	Name of Symbol	Figure/Symbol	Purpose
1	Entity		A person / Entity
2	Direction of Data Flow		Flow of Data from Entity/Process
3	Process or Bubble		Signify the process of a Module
4	Data Store	 OR 	Place of Storing Data

The Rules We Follow while preparing the DFD's are as follows :

1. Each Process Should have atleast one input and one output.
2. Each Data Store should have atleast one data flow in and one data flow out.
3. All Process in DFD either go to another process or to another Data Store.

0 th Level DFD :

This is the highest level DFD which provides overview of entire System. It shows the major processes , data flow and the data stores in the System. We also call it the Context Diagram. Here we can see the entire System working as a Complete Blackbox to which user provides input , processing is carried out by the system and the result is produced . Thus, Providing the Business the Desireable Advantages .

The 0th Level DFD illustrates the Rent Management System as :

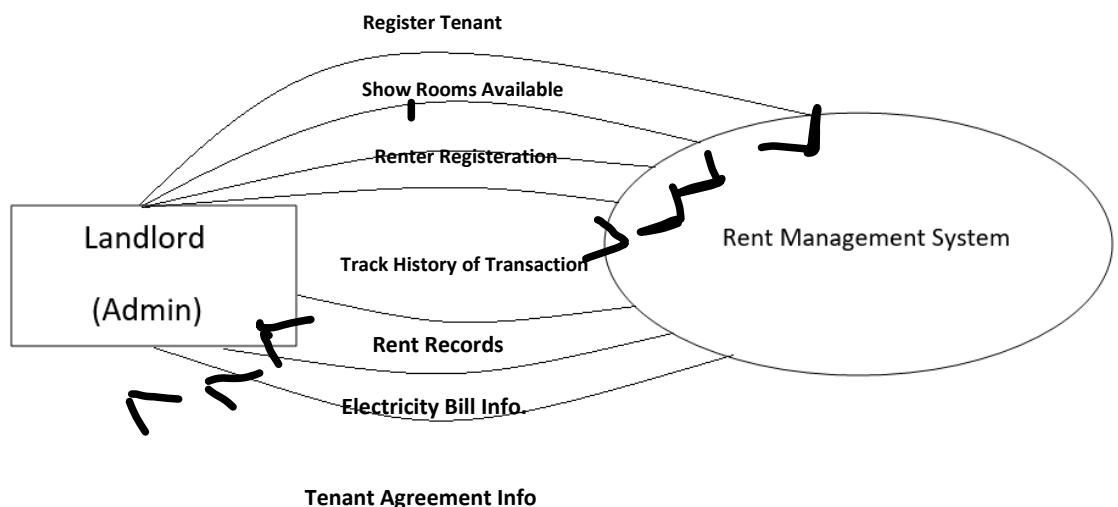


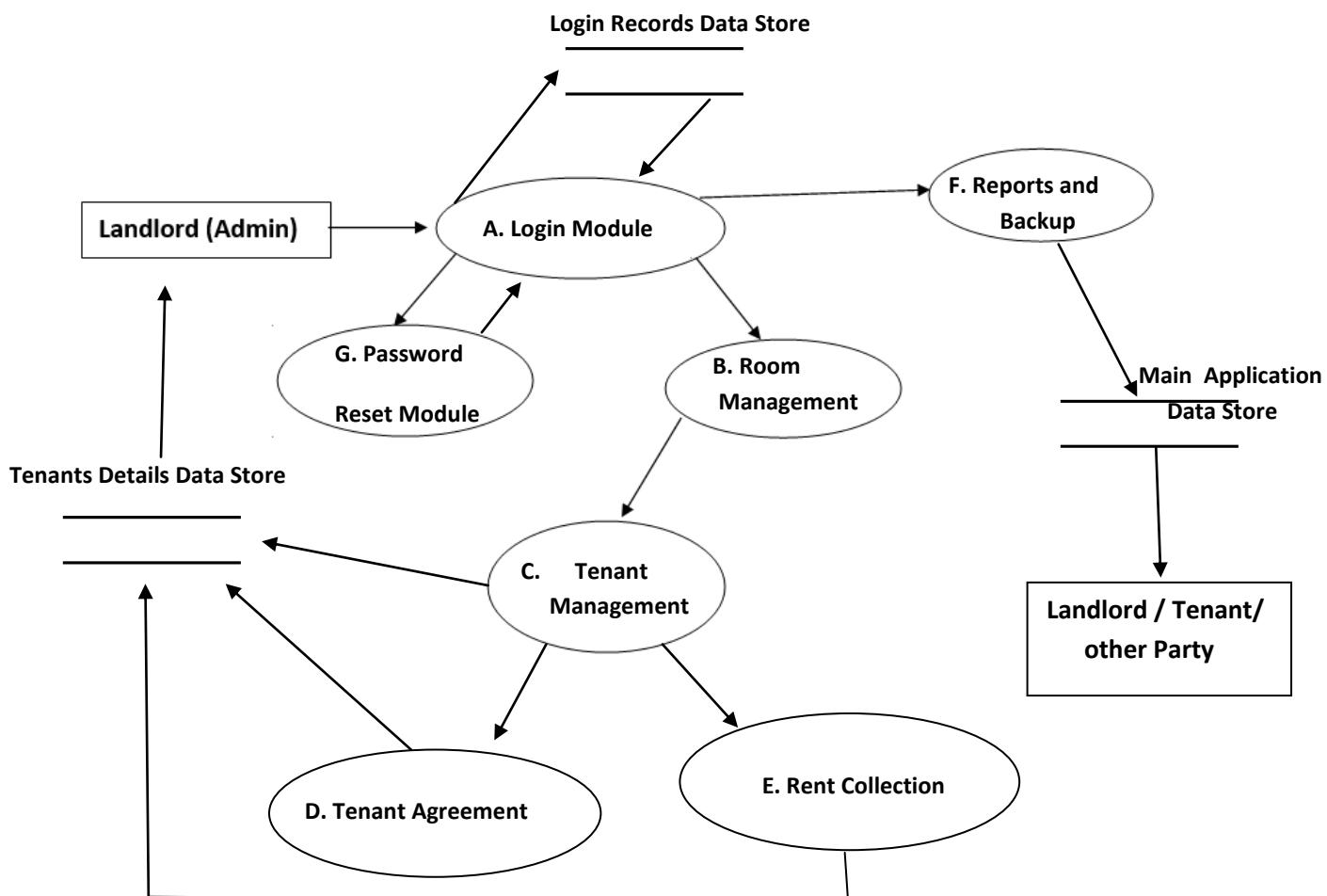
Fig : 0th Level DFD : Rent Management System

1 th Level DFD :

In 1 th Level DFD we decompose the Entire System or the Context Diagram That breaks into multiple branches of modules or sub processes . In this Level DFD all the Processes are rendered by the Modules that are present in the table of modules that is as follows :

Task	Name of the Task
A	Login Module
B	Room Management Module
C	Tenant Management Module
D	Agreement Module
E	Rent Collection Module
F	Reports and Data Backup Module
G	Change Password Module

The 1 th Level DFD illustrates the Rent Management System as :



2 th Level DFD :

In 2th Level DFD , We go one step deep into 1th Level DFD .

Thus , illustrating the Sub Processes and their dependencies in each Module. This is necessary as it is used to plan or record specific details about system functioning . Since , the Rent Management System Comprises of 7 Modules, Their individual DFD helps us to know their Internal Data Flow better.

The 2th Level DFD of Each Sub System / Module of Rent Management System are illustrated as follows:

Task	Name of the Task
A	Login Module
B	Room Management Module
C	Tenant Management Module
D	Agreement Module
E	Rent Collection Module
F	Reports and Data Backup Module
G	Change Password Module

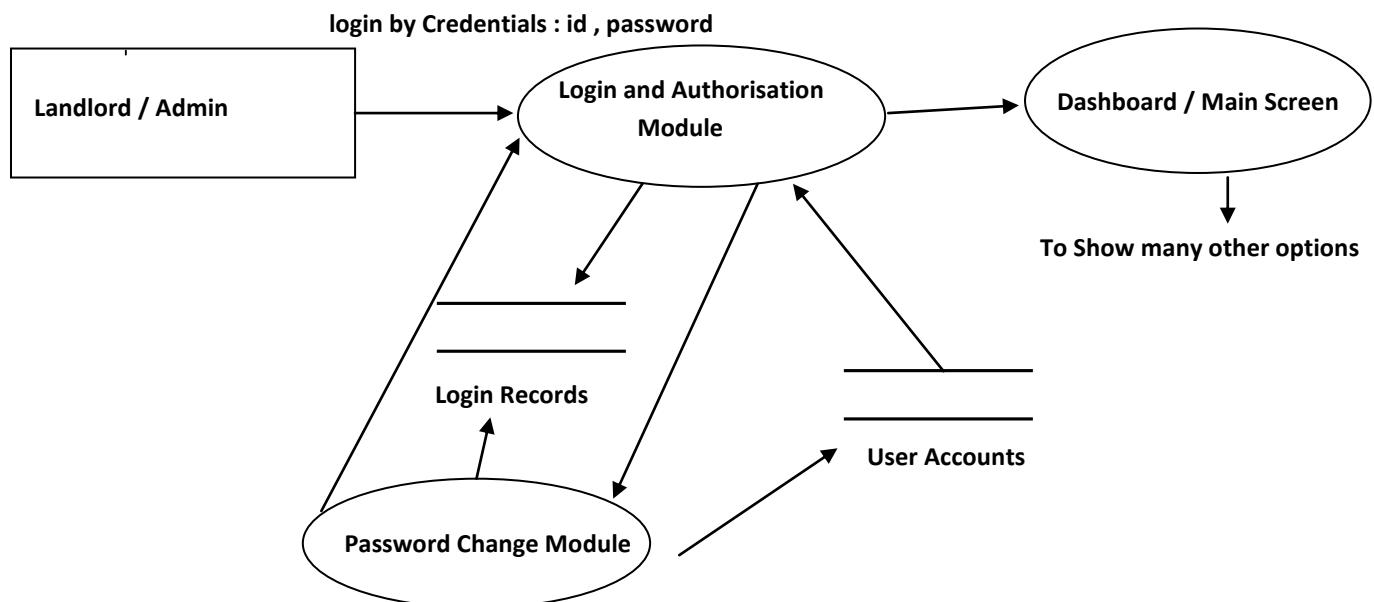


fig. Module A : illustration of the Login Module

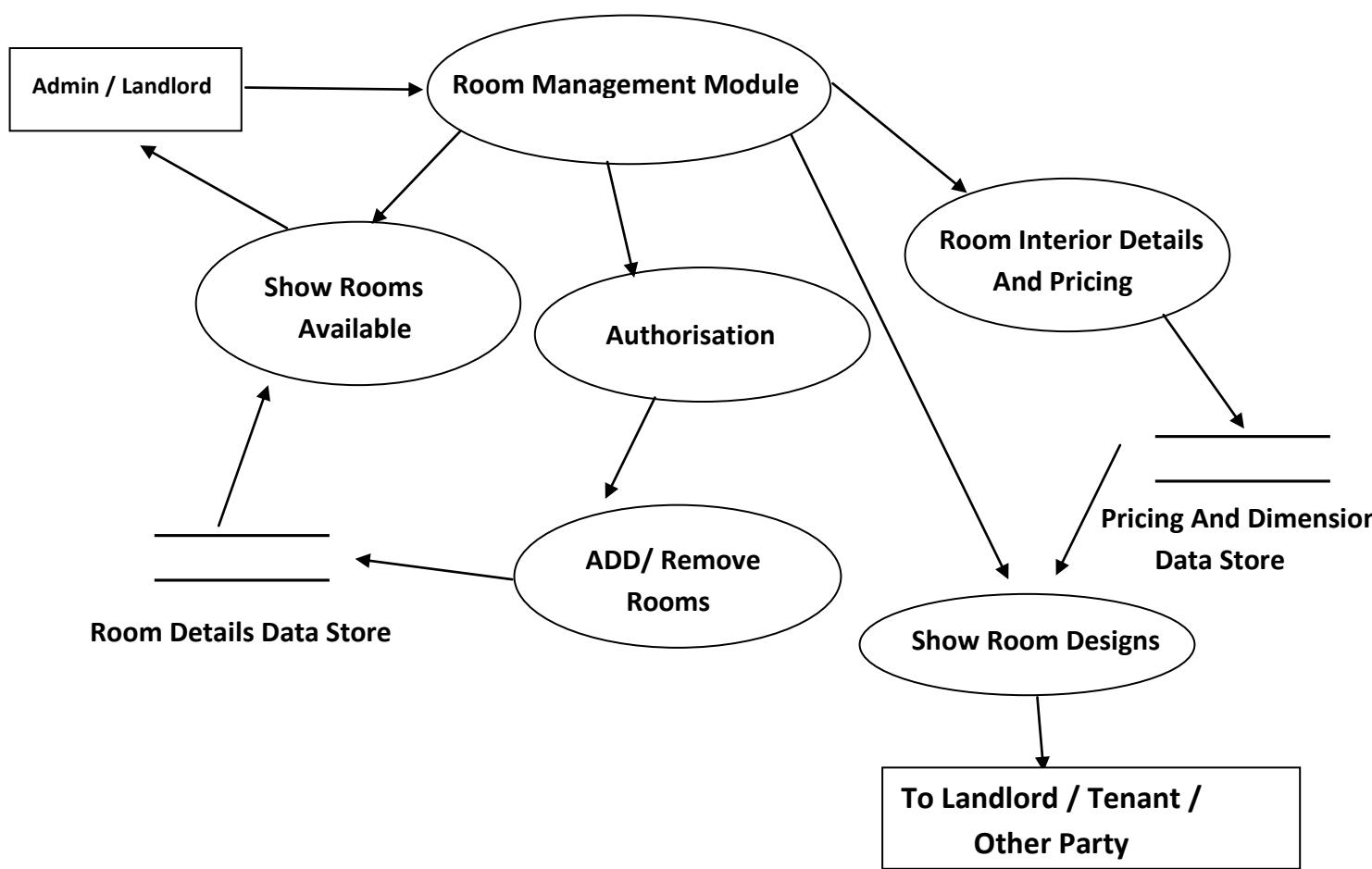


fig. Module B : illustration of the Room Management Module

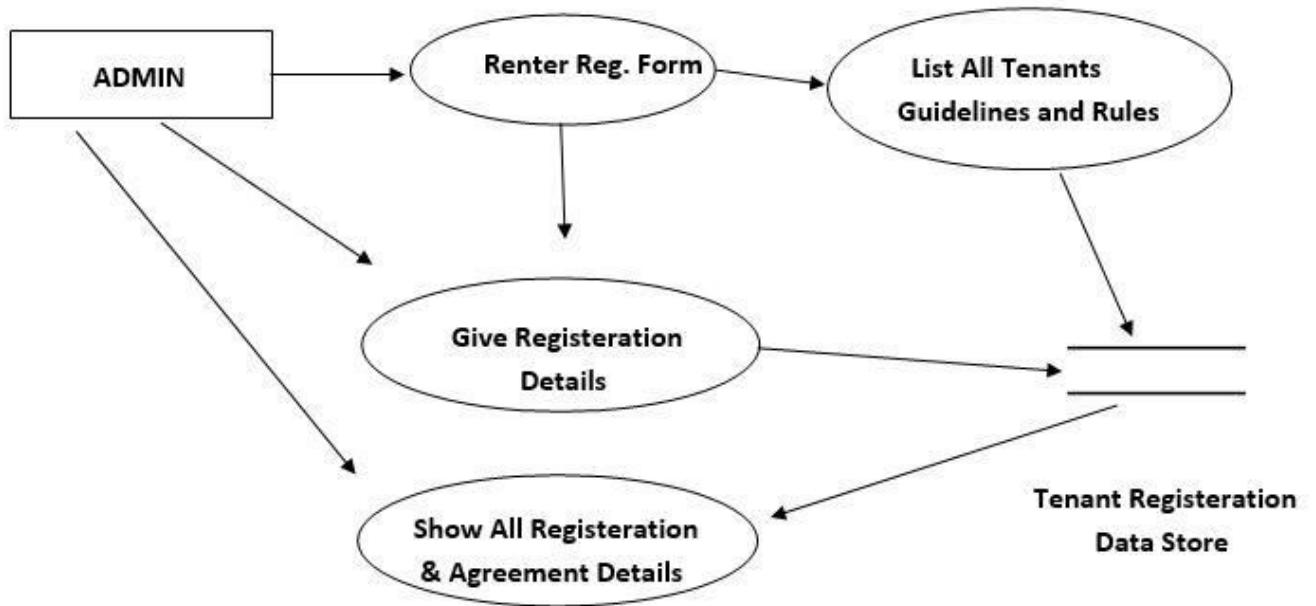
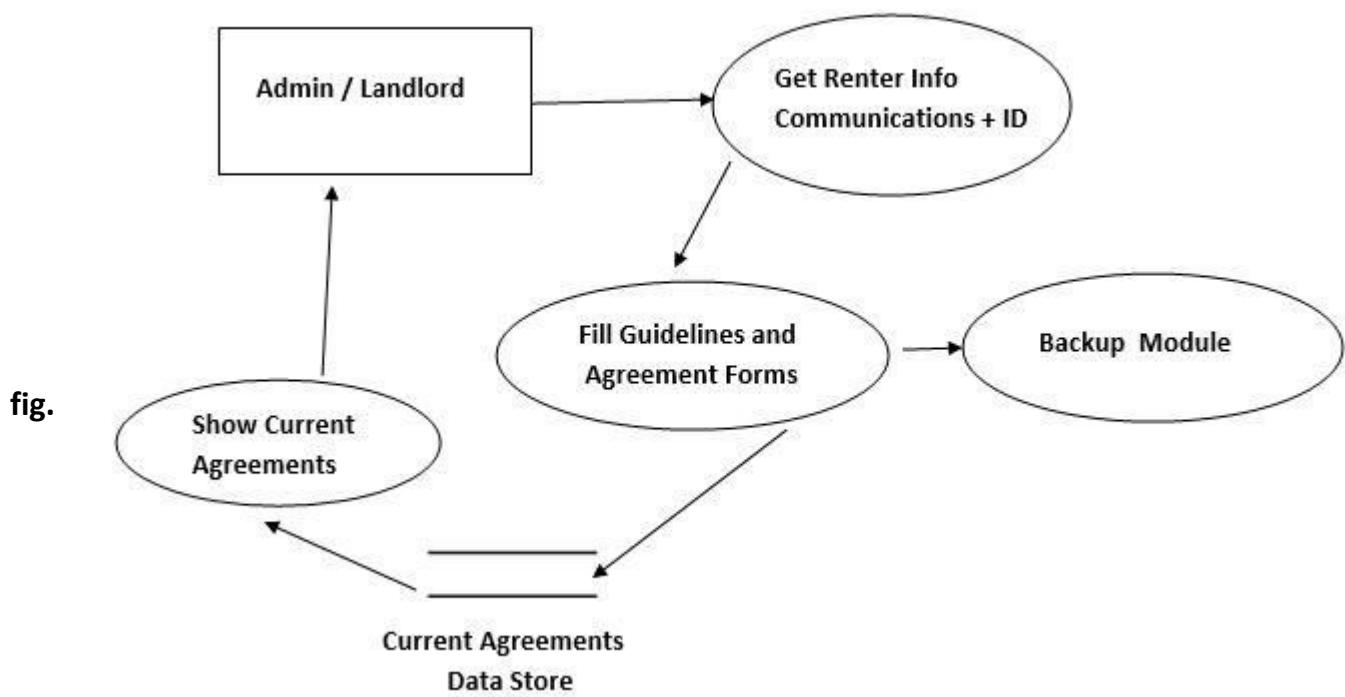


fig. Module C : illustration of the Tenant Management Module



Module D : illustration of the Tenant Agreement Module

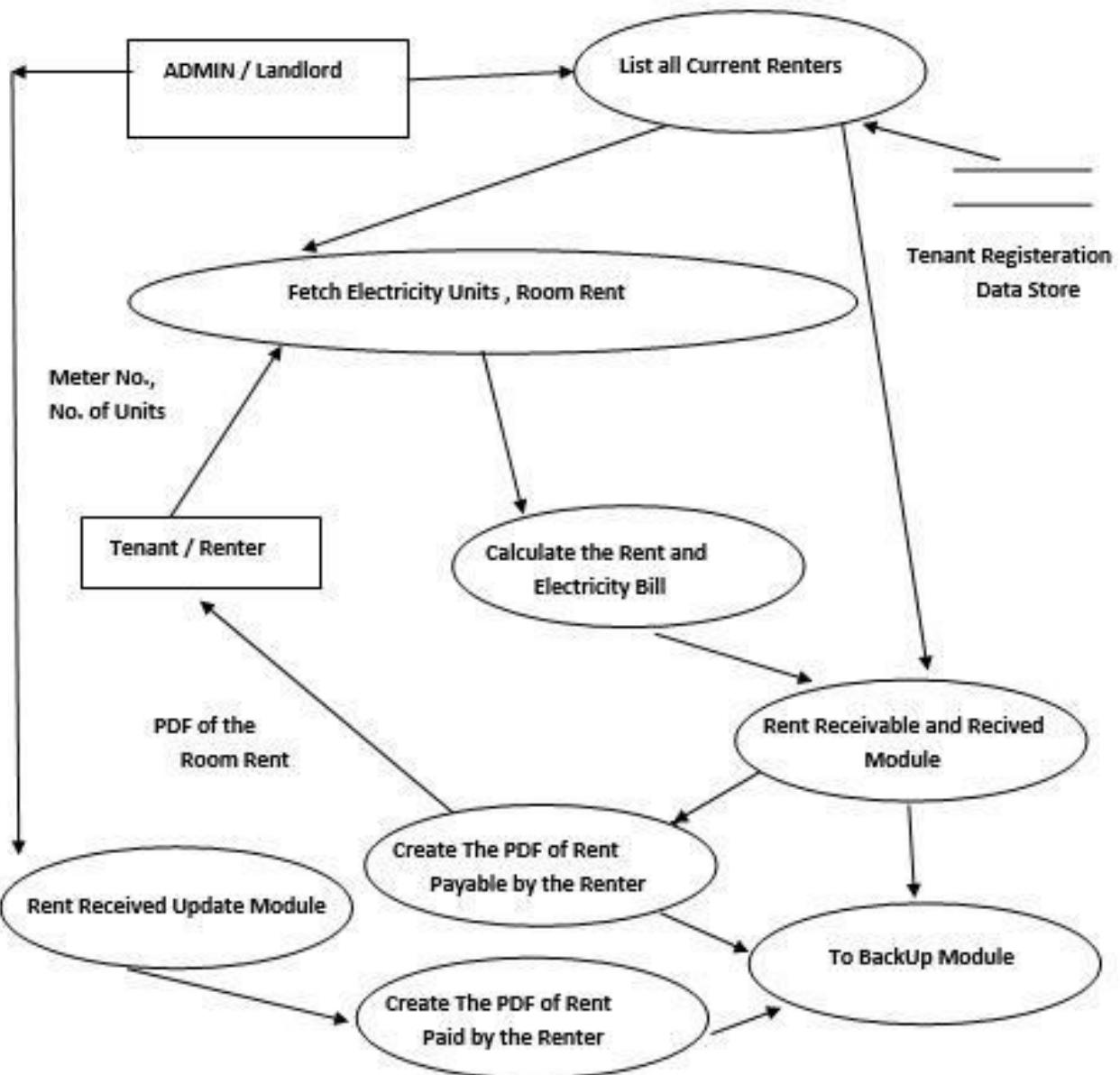


fig. Module E : illustration of the Rent Collection Module

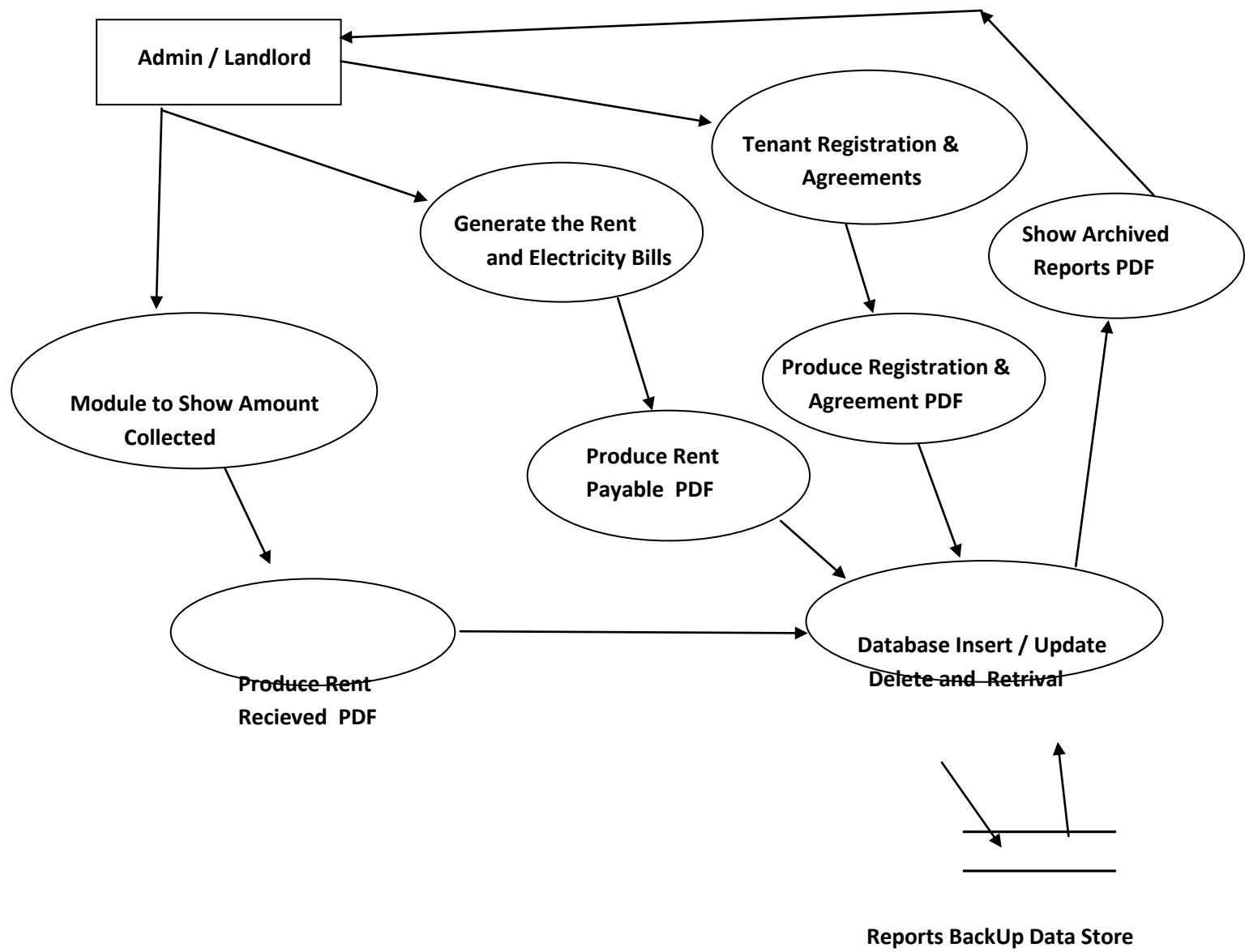


fig. Module F : illustration of the Rent Collection Module

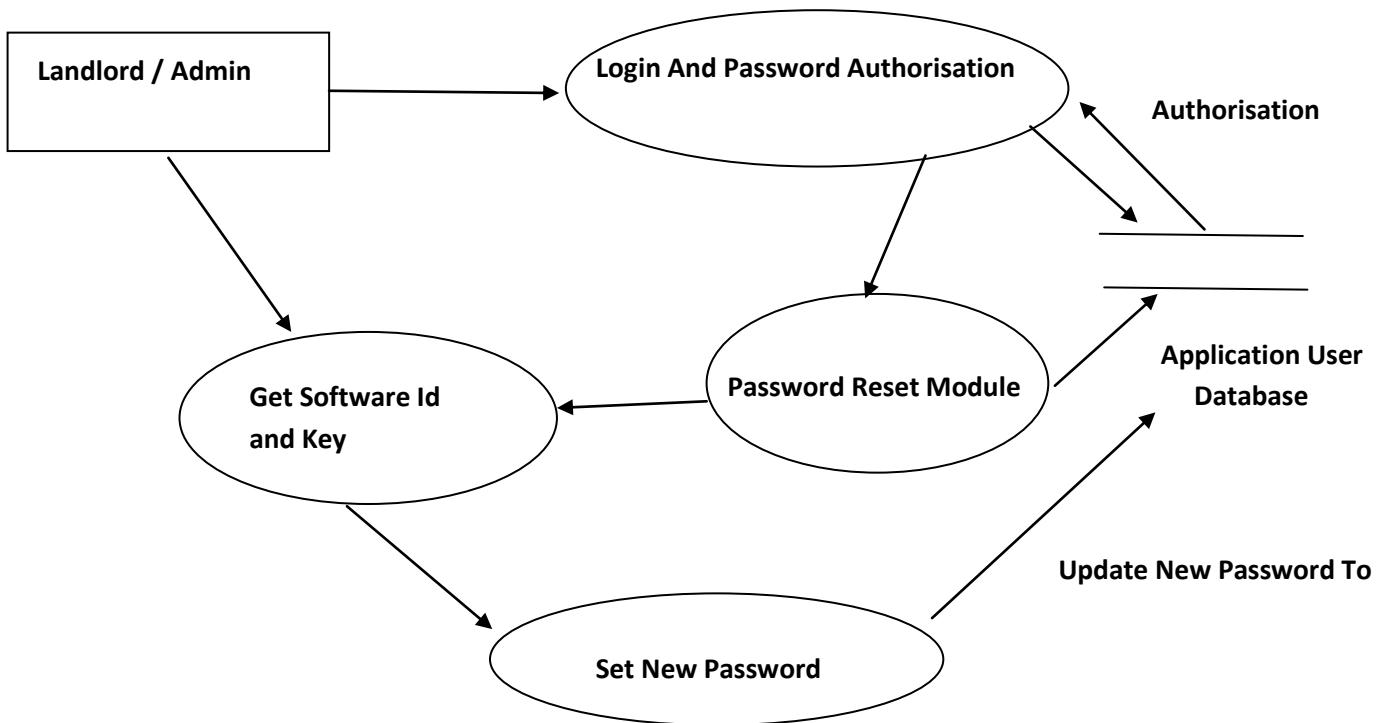
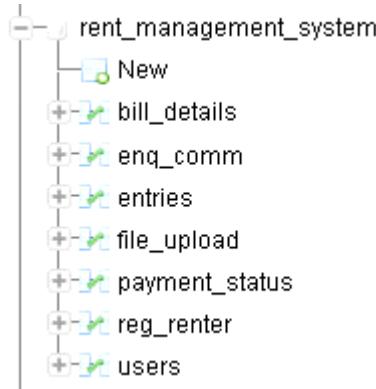


fig. Module G : illustration of the Rent Collection Module

4.3 SQL Tables in the Rent Management Table



The above is the representation of the list of tables that are created and utilised in the Rent Management System.

The Respective tables are made by their queries :

```
-- Table structure for table `bill_details`  
CREATE TABLE `bill_details` (  
    `id` int(11) NOT NULL,  
    `renter_id` int(11) NOT NULL,  
    `month` varchar(20) NOT NULL,  
    `year` int(11) NOT NULL,  
    `due_date` date NOT NULL,  
    `room_rent` decimal(10,2) NOT NULL,  
    `units_used` int(11) NOT NULL,  
    `electric_bill` decimal(10,2) NOT NULL,  
    `advance_paid` decimal(10,2) NOT NULL,  
    `amount_dues` decimal(10,2) NOT NULL,  
    `miscellaneous` text NOT NULL,  
    `total_amount` decimal(10,2) NOT NULL,  
    `created_at` timestamp NOT NULL DEFAULT current_timestamp()  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;  
  
-- Table structure for table `enq_comm`  
CREATE TABLE `enq_comm` (  
    `id` int(11) NOT NULL,  
    `name` varchar(50) NOT NULL,  
    `age` int(11) NOT NULL,  
    `gender` varchar(10) NOT NULL,  
    `address` varchar(100) NOT NULL,  
    `mobile` bigint(20) NOT NULL,  
    `time_of_saving` timestamp NOT NULL DEFAULT current_timestamp()  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;  
  
-- Table structure for table `entries`  
CREATE TABLE `entries` (
```

```

`id` int(11) NOT NULL,
`room_id` varchar(50) NOT NULL,
`photo_path` varchar(255) NOT NULL,
`sign_path` varchar(255) NOT NULL,
`aadhar_path` varchar(255) NOT NULL,
`created_at` timestamp NOT NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

-- Table structure for table `file_upload`
CREATE TABLE `file_upload` (
`id` int(11) NOT NULL,
`filename` varchar(50) NOT NULL,
`roomname` varchar(25) NOT NULL,
`status` varchar(10) NOT NULL,
`folder_path` varchar(100) NOT NULL,
`timestamp` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

-- Table structure for table `payment_status`
CREATE TABLE `payment_status` (
`id` int(11) NOT NULL,
`renter_id` varchar(50) NOT NULL,
`date_pay` date NOT NULL,
`month` varchar(50) NOT NULL,
`amount` decimal(10,2) NOT NULL,
`pay_status` varchar(50) NOT NULL,
`description` text DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

-- Table structure for table `reg_renter`
CREATE TABLE `reg_renter` (
`id` int(11) NOT NULL,
`name` varchar(255) NOT NULL,
`age` int(11) NOT NULL,
`gender` enum('Male','Female','Other') NOT NULL,
`datereg` date NOT NULL,
`aadhar` varchar(20) NOT NULL,
`mobile` varchar(15) NOT NULL,
`address` text NOT NULL,
`roomNo` varchar(50) NOT NULL,
`price` decimal(10,2) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

-- Table structure for table `users`
CREATE TABLE `users` (
`id` bigint(20) NOT NULL,
`user_id` bigint(20) NOT NULL,
`user_name` varchar(100) NOT NULL,

```

```

`password` varchar(100) NOT NULL,
`productkey` int(5) NOT NULL,
`date` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

```

These are the above lines of code that are the sql required to create the rent_management_system database tables.

4.4 ER Diagrams

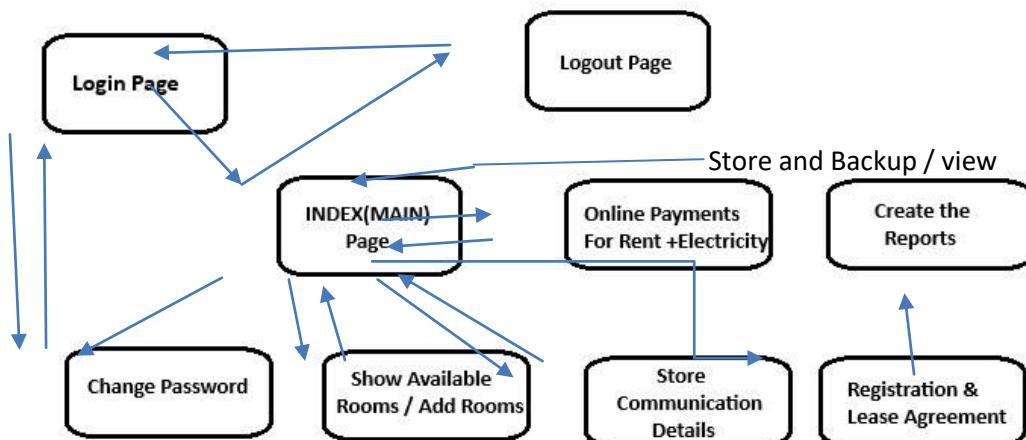
Introduction of Entity Relationship Diagrams of the Rent Management System

An Entity-Relationship (ER) diagram is a type of structural diagram for use in database designing for the Rent Management System. It provides graphical representation of the entities involved in a system and relations between them on the basis of the attributes they posses and need to be stored inside of the database.

The Database involved in the Rent Management System is the MySQL lite that comes along with the Xammp Server that provides us ease in clearly making the designs for the database that will be utilised to store the data in format with the special attributes that are involved within the entities of the System that can be the landlord , Tenant , Rooms , Bills etc.

For the preparation of the Required ER – Diagrams we follow the below Graphical Scheme inorder to produce the precise Designs for the Database for the Rent Management System. This Diagram serves as the blueprint ensuring that all necessary data relationships are efficiently Accounted.

Beginning to Design the ER model , we must consider the flow of the process inside the Rent Management System with the help of the Site map that is as follows.



The Site Map Depicts the Rent Management System's key pages that involve to deal with the main entities of the Rent Management System.

The Below Graphical Representations Identify the Main Entities involved in the System and also the Relations between them.

Entity

An Entity is defined as the object , being the part of the System has some attributes that are likeable to be stored in the database inorder to carry out the necessary tasks.

An Entity that is the part of the system is depicted as the square / rectangle shape



Fig. : Entity in ER Diagram

Attribute

Attributes are defined of an entity that are crucial points over which necessary actions are required to be carried out by the system. This is the clear feature or the set of values that can help the entity be identified and differentiated with the other entities involved inside the system. An attribute in the system is depicted as the oval shaped .

There are also some sub categories inside the attributes : that are :

Derived attributes : These attributes depend on the other attributes to make a decision.

Eg. : The Age is a Derived Attribute type that can be derived from the date of Birth.

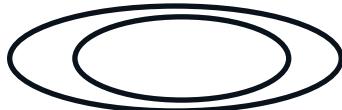
Multivalued Attributes : These category can have more than one types and that are legally accepted by te System .

Eg. : The Online Payment can be Done by various Online Payment Interfaces that will create the transactions with the System.

The Attributes and their sub types are depicted as follows:



Single valued / Simple



Multi Valued



Derived Attribute

Links between Two Entities

The Links are depicted by a single straight line , they define the relation between the two / more entities , they also bind the entity with their attributes and with the relationships that pair them respectively.

Fig . The Link between Two/ More Entity in the ER Diagram

Relation

The Relation is depicted by the shape of the rhombus that contains a keyword of English in the center of the Place.

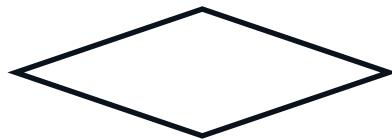


Fig.: The Relation Specifying Symbol in the ER Diagram

The Relations in the Real World Scenarios are of many types:

- (i) One to One Relation
- (ii) One to Many Relation
- (iii) Many to One Relation
- (iv) Many to Many Relation

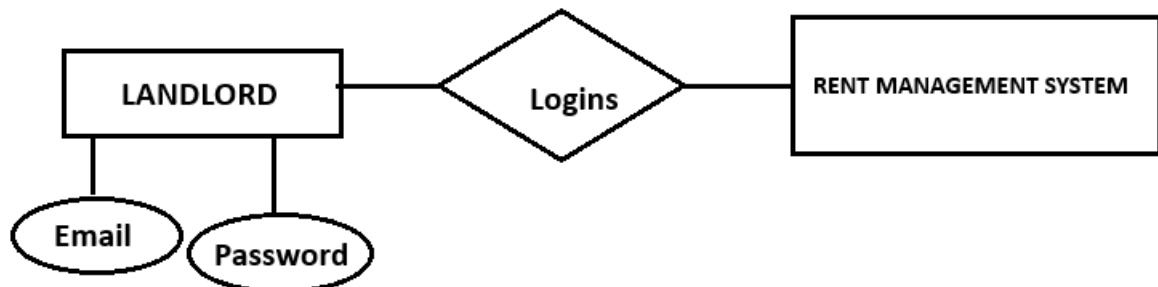
These are exemplified as follows:

- (i) One to One Relation : One Renter has One Identity Card for Registration.
- (ii) One To Many Relation : One landlord is Capable to manage many renters by the help of the Rent Management System.
- (iii) Many to One Relation : Many Customers use the Same Payments Gateway Interface for their online transactions.
- (iv) Many to Many Relation : A student can register for many classes, and a class can include many students.

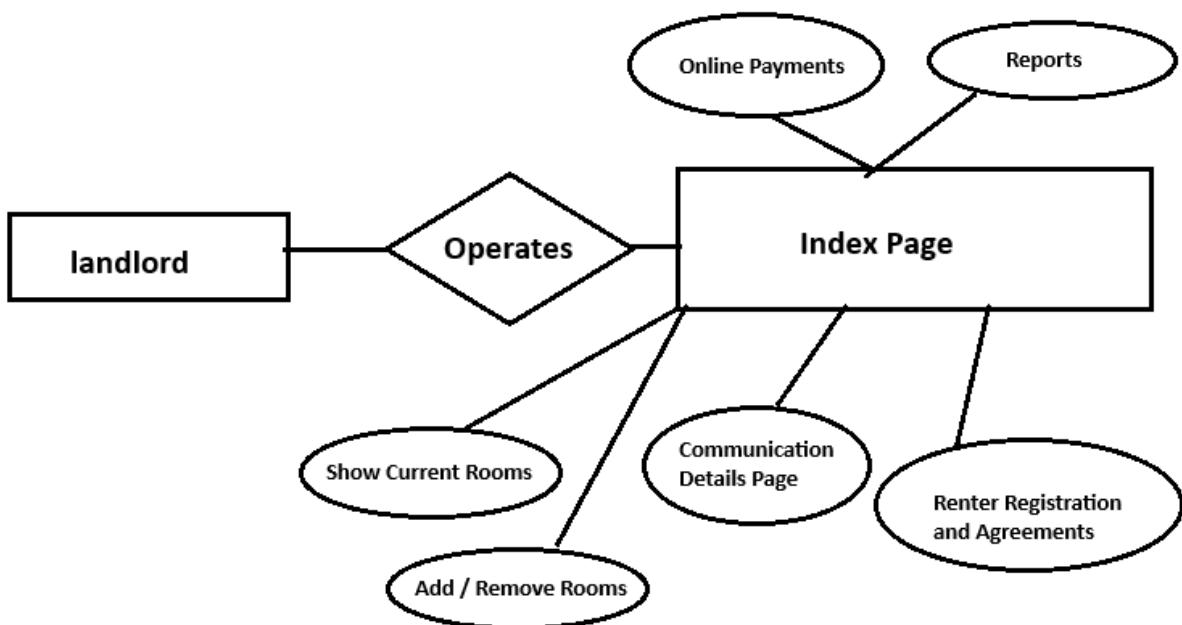
Login Page

The Landlord Logins using the Email and the Password into the system.

One LandLord has One Email and one Password.

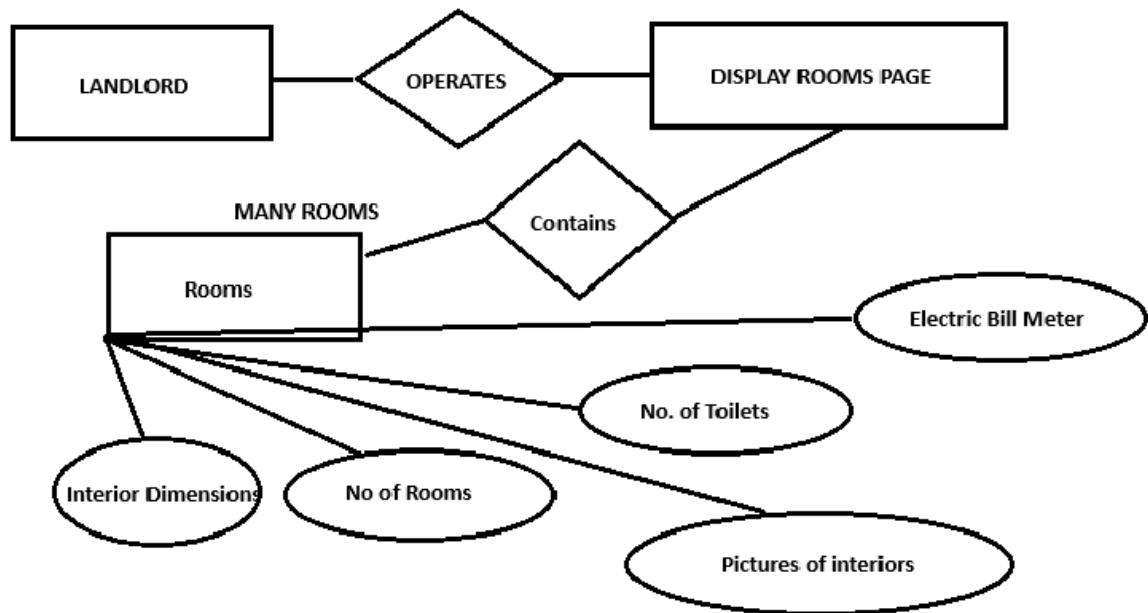


Index Page is the Main Page for the Rent Management System that Comprises of the various Pages that carry out the various tasks.



Display Rooms Page

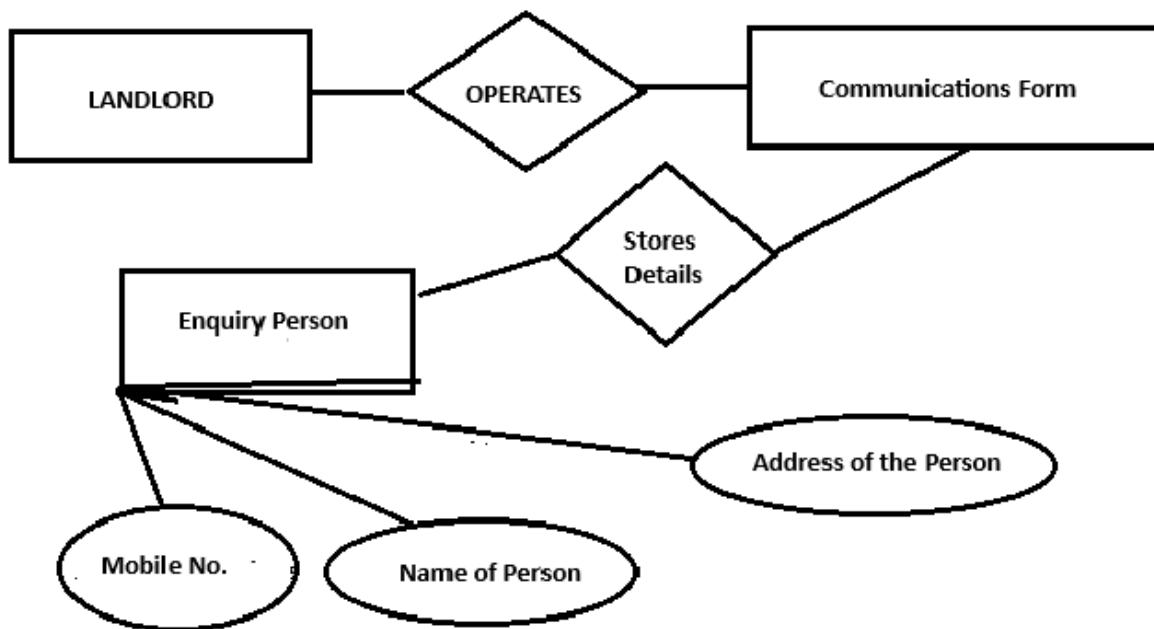
The Display Rooms page helps the Landlord to showcase the Rooms that are in his property that are available for the renting purpose , This will be beneficial for the landlord to diplay to the customers for enquiry and also share over the internet for the purpose of online advertisement.



Store Details Page:

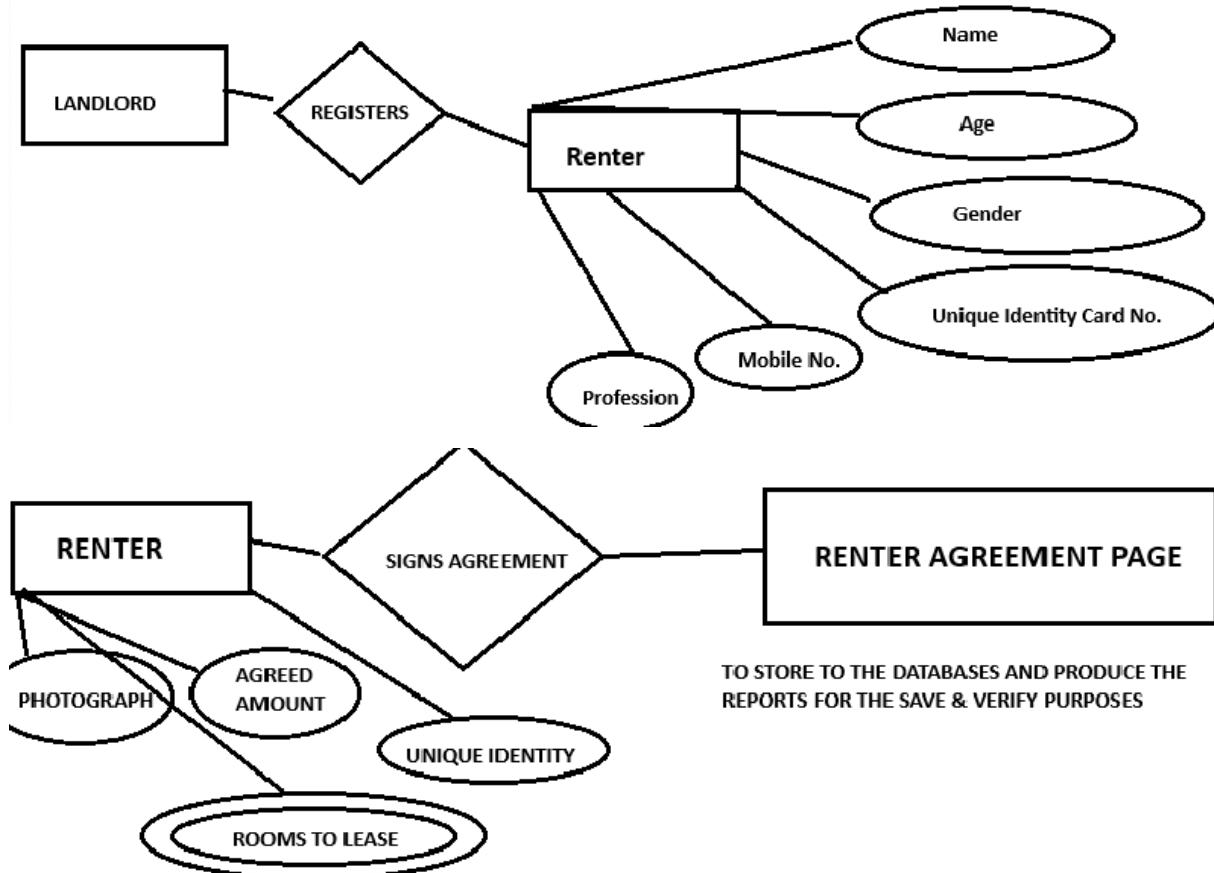
Sometimes , it is crucial to store the details of the consumers inorder to contact them in future , incase they are in need to find accomodations. There is always a chance that One who comes for enquiry , not be a customer , in case we may remind them that we have some rooms for renting, they may come to negotiate incase they are interested , or in dire need of.

This page is a form that will store the necessary Details like the Name and the Mobile no. , by which future communication may be established .



Renter Registration and Agreement Page

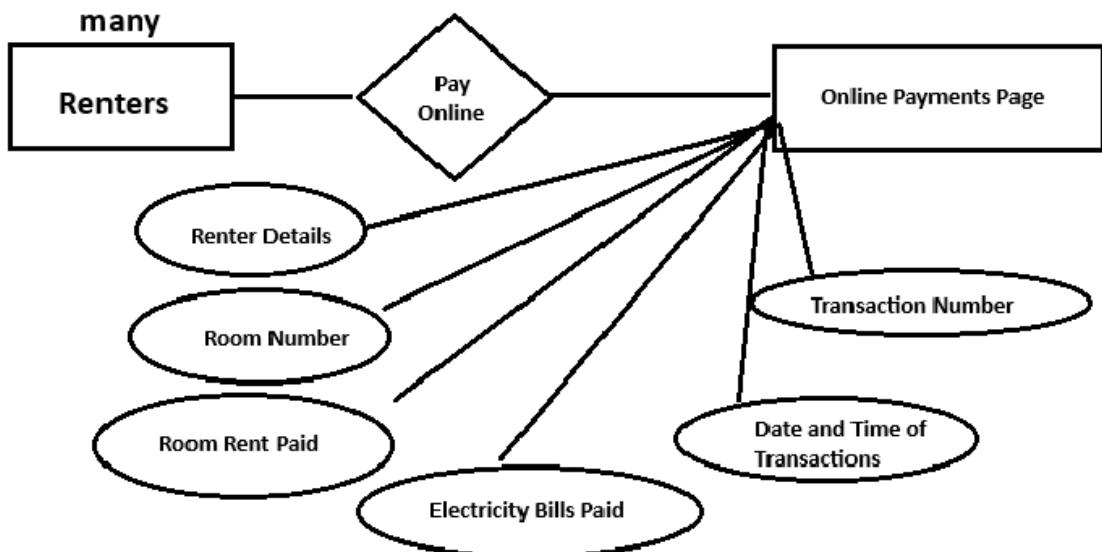
This is the most necessary and the crucial page for the purpose of legally storing the details of the consumer (Renter / Tenant) for the Rental Services that are provided by the Landlord(Service Provider). The Landlord Tends to Store the Necessary Details that must be Genuine for the purpose of occupying the room for the rent and for the safe future of the business.



Online Payments Page

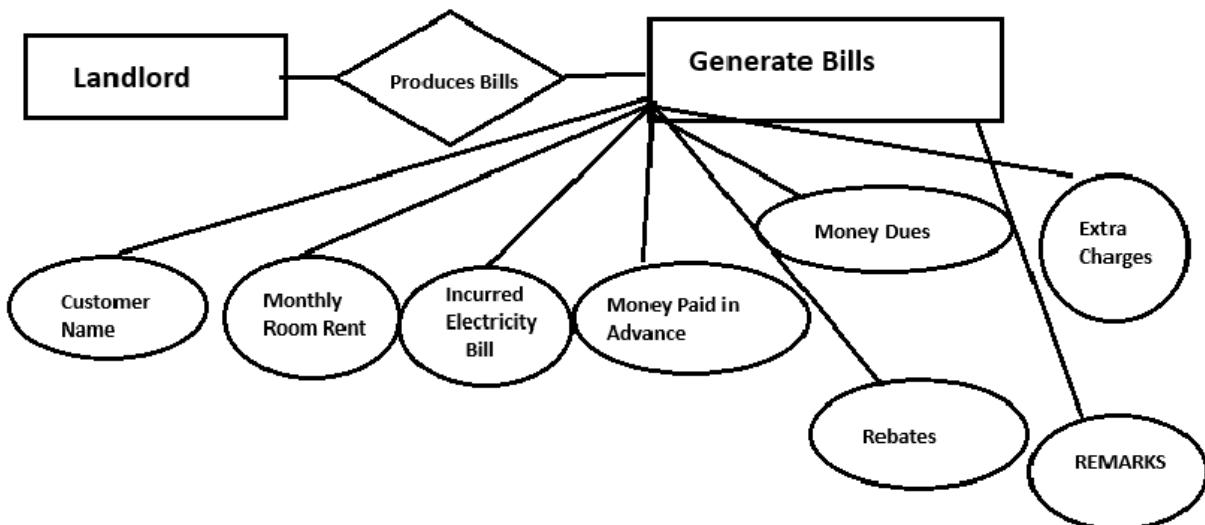
There will be many numbers of the renters to be managed by the System.

This page is responsible for the task for receiving the payments if made in online mode by the Renters . This will contain a Page containing the QR-code of the Landlord . In the best cases , this can be linked to secure portal for the purpose of Online Transactions that is also known as the Payments Gateway Integration with the Software.



Produce Reports Page

This Page is important for the task to make the bills for the room renters and to send them , such that they get prepared about paying the monthly and the electricity bill keeping in mind the process of producing the bills.



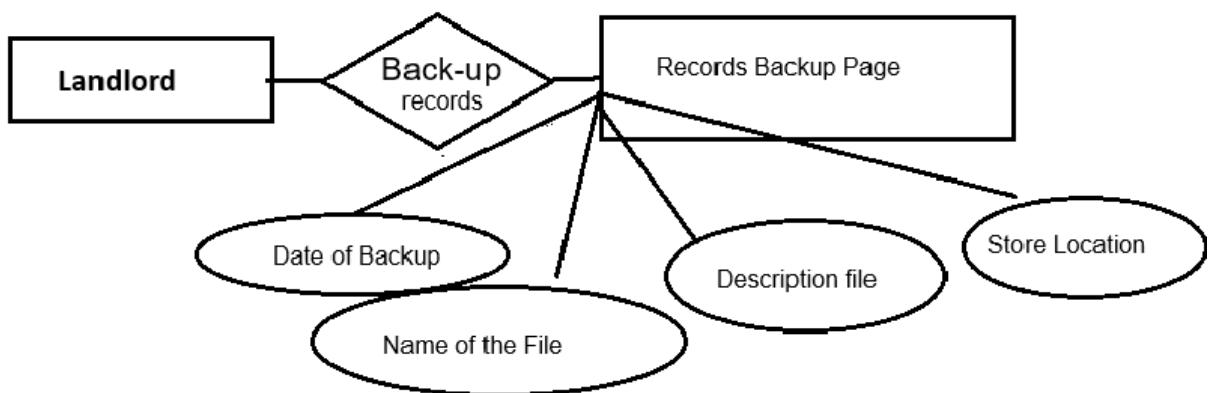
The Reports Backup Page :

This Page is the crucial component in the Rent Management System. The Reports are made stored in the computer , send a copy to the renter , and also , the main file is stored inside the main computer. There is always an uncertain risk that can lead to loss of those sensitive files.

The term ‘backup’ means to reliably make a copy . The Agreements contain the crucial data , incase a critical error may cause its loss , and in future the problems may be faced.

This will utilise the File Management System to Store the Report with a Details file . Since all the files are going to be made in .pdf format , it will store it to a good location , where the landlord may copy all of them in a disk / or in the cloud storage .

This is also going to maintain the database records , keeping the log of the events that took place , what file was made the copy of , what were the details it was stored with and which location its copy was made. This provides the convinience to the Landlord to produce separate copies inorder to store to the place he may feel convinient.



CHAPTER - 5

SYSTEM Coding & Implementation

RENT MANAGEMENT SYSTEM

SYSTEM DESIGN AND IMPLEMENTATION	
5.1	Introduction to PHP
5.2	Introduction to MySQL RDBMS
5.3	Source Code
5.4	Documentation

5.1 Introduction to PHP

. PHP (Hypertext Preprocessor) is a popular server-side scripting language designed for web development. Here are some of its key features:

1. Open Source

- PHP is open-source, meaning it is free to use and has a large community of developers who contribute to its continuous improvement.

2. Cross-Platform

- PHP runs on various platforms including Windows, Linux, Unix, and macOS, providing great flexibility for developers.

3. Easy to Learn and Use

- PHP has a straightforward syntax that is easy to understand for beginners, yet powerful enough for advanced developers.

4. Embeddable in HTML

- PHP code can be easily embedded within HTML, making it seamless to integrate server-side logic with client-side content.

5. Database Integration

- PHP supports a wide range of databases, including MySQL, PostgreSQL, Oracle, SQLite, and more. It provides a variety of functions and extensions to interact with these databases.

6. Wide Range of Frameworks

- There are numerous PHP frameworks available, such as Laravel, Symfony, CodeIgniter, and Zend, which streamline the development process by providing pre-built modules and components.

7. Rich Library of Functions

- PHP comes with an extensive standard library that includes numerous functions for handling strings, arrays, files, sessions, and more.

8. Object-Oriented Programming (OOP)

- PHP supports OOP, allowing for the creation of reusable code and modular programming. It includes features like classes, inheritance, interfaces, and namespaces.

9. Strong Community Support

- PHP has a large and active community, providing ample resources such as tutorials, forums, and documentation to help developers troubleshoot and learn.

10. Scalability

- PHP applications can be scaled easily, allowing for the development of both small websites and large enterprise applications.

11. Security

- PHP includes various built-in security features such as data encryption, SSL support, and functions to prevent SQL injection, XSS (Cross-Site Scripting), and CSRF (Cross-Site Request Forgery).

12. Session Management

- PHP provides robust session management capabilities, allowing developers to store and manage user session data effectively.

13. Error Reporting

- PHP offers a range of error reporting features that help developers identify and fix issues quickly. It includes various levels of error reporting, customizable error handling, and logging options.

14. Integration with Various Services

- PHP can integrate with different web services and APIs, allowing developers to build applications that interact with other systems and services.

15. Fast Performance

- PHP is known for its fast execution and performance, especially when running on a server with optimized configurations.

These features make PHP a versatile and powerful language for web development, suitable for a wide range of applications from small personal blogs to large-scale enterprise systems.

5.2 Introduction to MySQL RDBMS

Using MySQL in phpMyAdmin, which comes with the XAMPP server, offers several robust features for database management in a Relational Database Management System (RDBMS) environment. Here are the key features:

1. User-Friendly Interface

- phpMyAdmin: Provides a web-based interface for managing MySQL databases. It simplifies database administration tasks such as creating, modifying, and deleting databases, tables, fields, or rows.

2. Comprehensive Database Management

- Create and Drop Databases: Easily create and delete databases as per project requirements.
- Table Management: Create, alter, and drop tables. Define and manage table structures including setting primary keys, foreign keys, indexes, and constraints.
- Field Management: Add, modify, and delete fields/columns. Set data types, default values, and attributes like `AUTO_INCREMENT`.

3. Data Operations

- Data Insertion: Insert new records into tables.
- Data Update: Update existing records with new values.
- Data Deletion: Delete records from tables.
- Data Export/Import: Export databases or tables in various formats (e.g., SQL, CSV, XML) and import data from files.

4. Query Execution

- SQL Query Editor: Write and execute custom SQL queries. Provides syntax highlighting and error reporting.
- Stored Procedures and Triggers: Create and manage stored procedures, functions, and triggers to automate repetitive tasks and enforce business rules.

5. Database Design and Modeling

- ER Diagrams: Visualize database structure using Entity-Relationship diagrams.
- Normalization: Ensure database design follows normalization rules to minimize redundancy and improve data integrity.

6. User and Permission Management

- User Accounts: Create and manage user accounts with specific privileges.
- Permissions: Grant and revoke privileges on databases, tables, and other objects to control access and ensure security.

7. Backup and Restore

- Data Backup: Regularly back up databases to prevent data loss.
- Data Restore: Restore databases from backup files in case of data corruption or accidental deletion.

8. Performance Monitoring and Optimization

- Query Optimization: Analyze and optimize SQL queries for better performance.
- Indexing: Create and manage indexes to speed up data retrieval operations.
- Performance Metrics: Monitor server performance and track metrics like query execution time, memory usage, and CPU load.

9. Security Features

- SSL/TLS Support: Ensure secure connections between the database server and clients.
- Encryption: Use encryption to protect sensitive data at rest and in transit.
- SQL Injection Prevention: Implement best practices and use prepared statements to prevent SQL injection attacks.

10. Replication and High Availability

- Replication: Set up master-slave replication to ensure high availability and load balancing.
- Failover: Implement failover mechanisms to switch to a backup server in case of primary server failure.

11. Data Integrity and ACID Compliance

- ACID Properties: Ensure transactions are atomic, consistent, isolated, and durable to maintain data integrity.
- Foreign Keys: Use foreign keys to enforce referential integrity between tables.

12. Cross-Platform Compatibility

- Operating Systems: MySQL and phpMyAdmin run on various operating systems, including Windows, Linux, and macOS, providing flexibility in deployment.

13. Extensive Documentation and Community Support

- Documentation: Comprehensive documentation available for MySQL and phpMyAdmin.

- Community Support: Active community forums and support channels for troubleshooting and knowledge sharing.

These features make MySQL with phpMyAdmin in XAMPP a powerful combination for managing relational databases, suitable for a wide range of applications from small websites to large-scale enterprise systems.

5.3

SYSTEM CODING

RENT MANAGEMENT SYSTEM



RENT MANAGEMENT SYSTEM

Source Code

	Chapters	Page Number
Chapter 1	Register , login & Logout Code	38-41
Chapter 2	Add Rooms and Remove Rooms Code	52-62
Chapter 3	Add Communication Details Code	63-80
Chapter 4	Reset User Password Code	81-87
Chapter 5	Share Location Code	88-91
Chapter 6	Register Renter Code	92-100
Chapter 7	Agreement Renter Code	101-119
Chapter 8	View Renters Code	120-125
Chapter 9	Create Bills Code	126-141
Chapter 10	Search Bar Code	142-146
Chapter 11	Other Functions Code	147-159
Chapter 12	Payments System Code	160-187
Chapter 13	DataBackup System Code	188-200
Chapter 14	Homepage Code	201-207

Chapter : 1

Register , Login & Logout

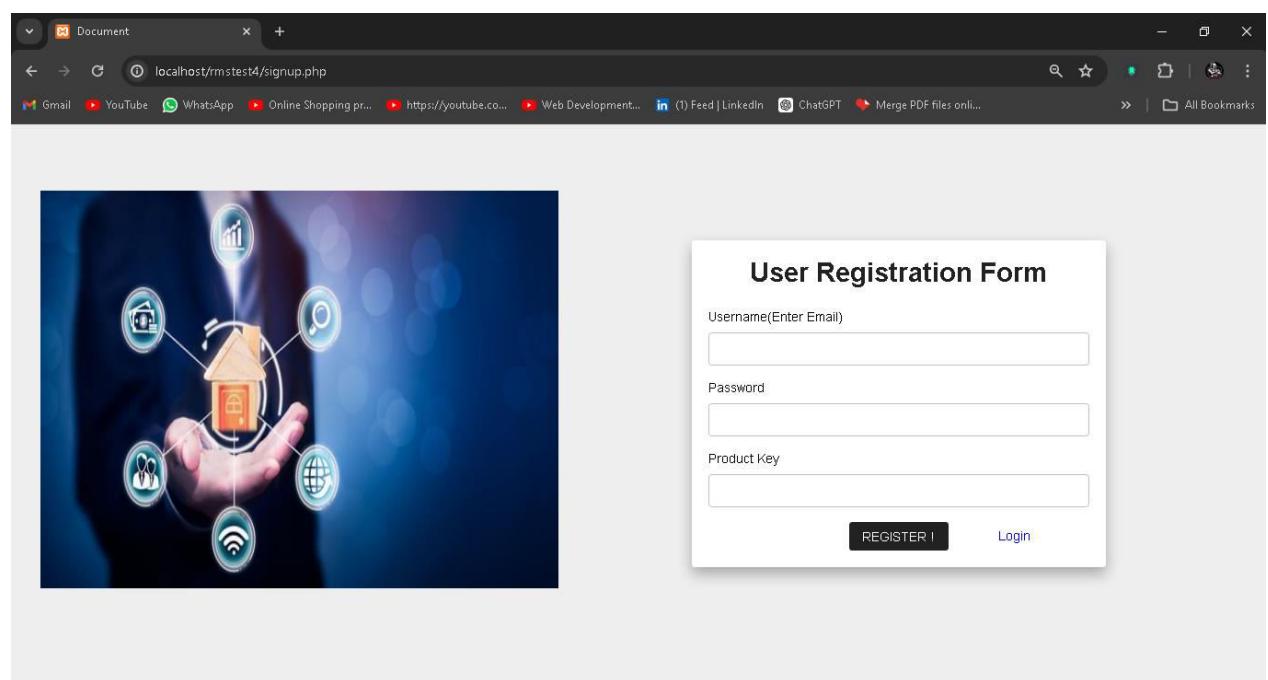
- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

Registration Page : Rent Management System

Description : This Page is used to register users on the Rent Management System. This requires Email and unique password , by help of which the user can successfully log-into the Rent Management System.

Snapshots : Registration Page.

name : signup.php



The field Product Key is set by Adminstrator , by help of which the landlord can reset the password of his system if the need arises in the future.

After the Details have been recorded and clicked upon Register button , the system redirects the user to the login page(login.php) where the user can log into by the use of the set email and the password.

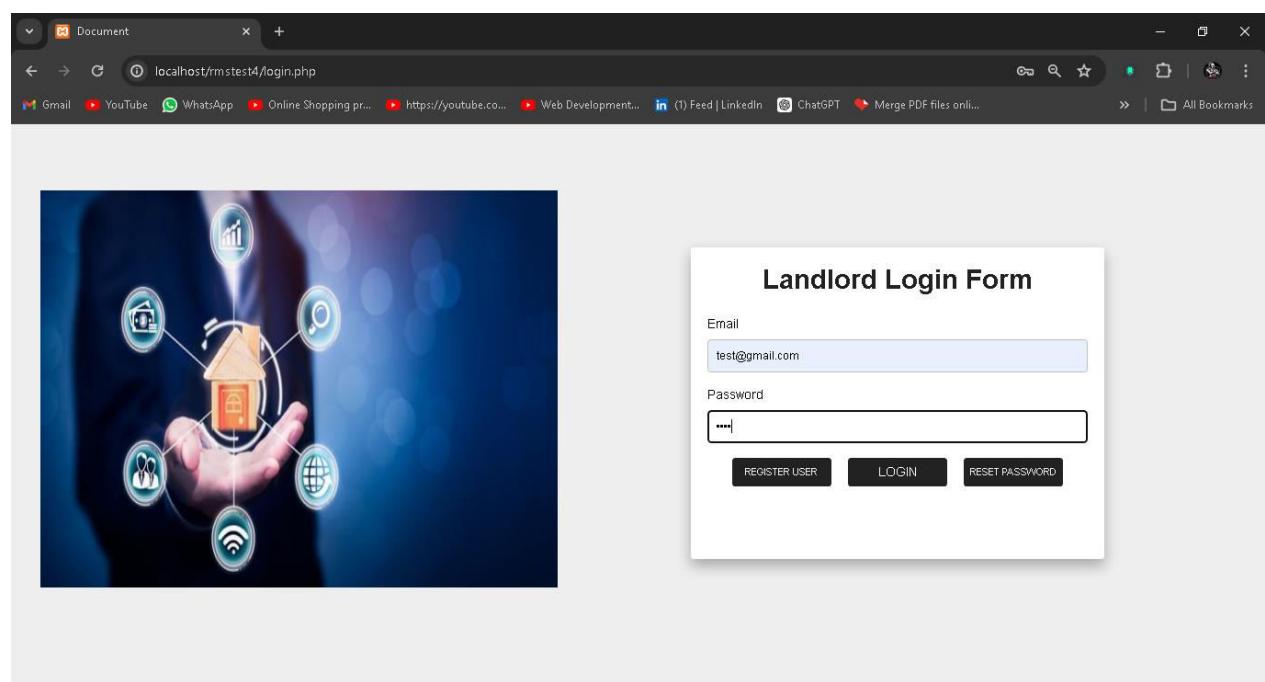
Log-in Page : Rent Management System

Description:

This Page plays the role of leading the user to the main content of the system. The system has many features that record and operate data , that are crucial and sensitive in terms of the business and their security must not be compromised , the Log-in page takes input of the credentials entered by the user and then redirects them to the main content , incase the Password is wrong the user gets the Option to reset the password .This page also notifies the incorrect password when entered by the user.

Email entered : test@gmail.com

Password entered : test

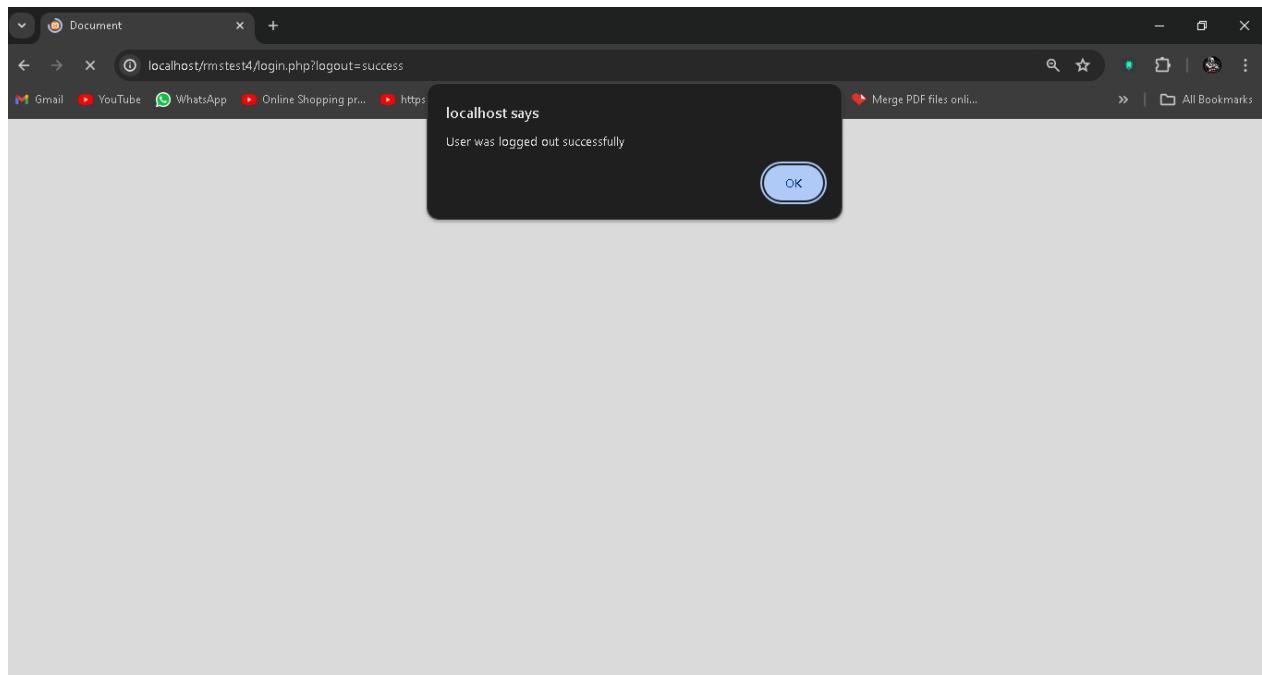


will lead us to the main content of the page. Also the main page can only be accessed when the login is successful , If any user tries to access the main page(index.php) via url of the user , he will automatically land on the login page

The Logout Page:

There is a logout page (logout.php) that serves the purpose of containing the main logic behind the logout operation , once the landlord is finished with his work.

The logout page is present on each pages and once logout button is pressed the user is redirected back to the login page with an alert that the user is logged-out Successfully.



Code : Registration (signup.php)

```
<?php

session_start();

include("connection.php");

include("functions.php");

if($_SERVER['REQUEST_METHOD']=="POST"){

    $user_name=$_POST['user_name'];

    $password=$_POST['password'];

    $prokey=$_POST['pk'];



    if(!empty($user_name) && !empty($password) && !is_numeric($user_name)){

        //save to the database

        $user_id=random_num(20);

        $query="insert      into      users(user_id,user_name,password,productkey)
values('$user_id','$user_name','$password','$prokey')";

        mysqli_query($con,$query);

        header("Location: login.php");

        die;

    }

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```

<title>Document</title>

<style>

    *{margin:0; padding:0; box-sizing:border-box; }

    body{min-height: 100vh; background:#eee; display:flex; font-family:sans-serif; }

    .container{margin:auto; width:500px; max-width:90%; }

    .container form{width:100%; height:100%; padding:20px; background:white;

border-radius:4px;box-shadow: 0 8px 16px rgba(0,0,0,0.3); }

    .container form h1{text-align:center; margin-bottom: 24px; color:#222; }

    .container form .form-control{width:100%; height:40px; background:white; border-radius:4px; border:1px solid silver; margin:10px 0 18px 0; padding: 0 10px; }

    .container form .btn{margin-left:50%; transform:translate(-50%); width:120px;

height:34px; border:none; outline:none; background:#222; cursor:pointer; font-size: 16px; text-transform:uppercase; color:white; border-radius: 4px; transition: 0.3s; }

    .container form .btn:hover{opacity:0.7; }

    #btn1 a{text-decoration: none; color: #eee; }

    #btn1{position:relative; top:-34px; left:140px; font-size: 12px; }

    #btn2 a{text-decoration: none; color: #eee; }

    #btn2{position:relative; top:-68px; left:-140px; font-size: 12px; }

```

```

    #logo{width:626px; position: relative; top:80px; height:480px;
left:35px; }

</style>

</head>

<body>



```

```
        <input class="btn" type="submit" value="Register !"><a  
        style="text-decoration: none; border-width: 10px; color:blue"  
        href="login.php">Login</a>  
  
    </form>  
  
  </div>  
  
</body>  
  
</html>
```

Code : Login Page (login.php)

```
<?php

session_start();

include("connection.php");

include("functions.php");

if($_SERVER['REQUEST_METHOD']=="POST"){

$user_name=$_POST['user_name'];

$password=$_POST['password'];



if(!empty($user_name) && !empty($password) && !is_numeric($user_name)) {

//read from the database

$query="select * from users where user_name='".$user_name' limit 1";

$result=mysqli_query($con,$query);

if($result){

if($result && mysqli_num_rows($result)>0){

$user_data=mysqli_fetch_assoc($result);

if($user_data['password']===$password){

$_SESSION['user_id']=$user_data['user_id'];

$_SESSION['productkey']=$user_data['productkey'];

header("Location: index.php");



} else{

echo"error in credentials";

die;

}

}

}

}

}
```

```

        header("Location: index.php");

        die;

    }

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

    <style>

        *{margin:0; padding:0; box-sizing:border-box; }

        body{min-height: 100vh; background:#eee; display:flex; font-family:
sans-serif; }

        .container{margin:auto; width:500px; max-width:90%; }

        .container form{width:100%; height:100%; padding:20px;
background:white;

        border-radius:4px;box-shadow: 0 8px 16px rgba(0,0,0,0.3) ; }

        .container form h1{text-align:center; margin-bottom: 24px;
color:#222; }

        .container form .form-control{width:100%; height:40px;
background:white; border-radius:4px; border:1px solid silver; margin:10px 0
18px 0; padding: 0 10px; }

        .container form .btn{margin-left:50%; transform:translate(-50%);
width:120px;

        height:34px; border:none; outline:none; background:#222;
cursor:pointer; font-size: 16px; text-transform:uppercase; color:white;
border-radius: 4px; transition: 0.3s; }

        .container form .btn:hover{opacity:0.7; }

#btn1 a{text-decoration: none; color: #eee; }

```

```

#btn1{position:relative;
      top:-34px;
      left:140px;
      font-size: 12px;}

#btn2 a{text-decoration: none; color: #eee; }

#btn2{position:relative;
      top:-68px;
      left:-140px;
      font-size: 12px; }

#logo{width:626px; position: relative; top:80px; height:480px;
left:35px; }


```

</style>

</head>

<body>

```

<?php

if (isset($_GET['logout']) && $_GET['logout'] == 'success') {
    echo "<script>alert('User was logged out successfully');</script>";
} else {
    echo "<script>console.log('Logout parameter not set or
incorrect');</script>";
}
?>


<div class="container">

<form action="login.php" method="post">
    <h1>Landlord Login Form</h1>

```

```
<div class="formgroup">

    <label for="">Email</label>

    <input type="email" class="form-control" name="user_name"
id="email" required>

</div>

<div class="formgroup">

    <label for="">Password</label>

    <input type="password" class="form-control"
name="password" id="password" required>

</div>

<input class="btn" type="submit" value="Login">

<button id="btn1" class="btn"><a href="resetpwd.php">Reset
Password</a></button>

<button id="btn2" class="btn"><a href="signup.php">Register
User</a></button>

</form>

</div>

</body>

</html>
```

Code : Logout Page(logout.php)

```
<?php  
  
session_start();  
  
if(isset($_SESSION['user_id'])){  
    unset($_SESSION['user_id']);  
}  
  
header("Location: login.php?logout=success");  
  
die;
```

Main Components Driving the Connections and Functionalitits:

There are two main components : connection.php and function.php that help the pages to perform their certain operations.

The Connection.php ensures the connectivity with MySql Lite (Innodb) is live , involving the flow of data inwards and outwards from the database.

The Function.php has two functions : this creates a random number for user id .

It also has a function that validates the connection , incase there is no login , then there is no way to access the main content of the system , the user gets automatically redirected to the main login page.

Their code are as follows :

Code : connection.php

```
<?php  
  
$dbhost="localhost";  
  
$dbuser="root";  
  
$dbpass="";  
  
$dbname="rent_management_system";
```

```

if(!$con = mysqli_connect($dbhost,$dbuser,$dbpass,$dbname)) {
    die("failed to connect");
}

```

Code : function.php

```

<?php

function check_login($con) {
    if(isset($_SESSION['user_id'])) {
        $id=$_SESSION['user_id'];

        $query="select * from users where user_id='$id' limit 1";
        $result = mysqli_query($con,$query);
        if($result && mysqli_num_rows($result)>0) {
            $user_data=mysqli_fetch_assoc($result);
            return $user_data;
        }
    }
    //redirect to login
    header("Location: login.php");
}

function random_num($length) {
    $text="";
    if($length<5) {
        $length=5;
    }
    $len=rand(4,$length);
    for($i=0; $i<=$length; $i++) {
        $text=rand(0,9);
    }
}

```

```

    }

    return $text;
}

}

```

Database Structure For Register , Login and Logout

The Database name is rent_management_system that has the table named users. This table is responsible to hold the key details such as the email , password , product key and time of registering, on the basis of that the login to main content is granted as well as the internal details of the landlord is handled.

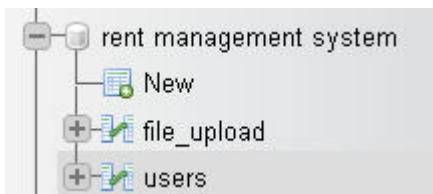


Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> users ★	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	96.0 KiB	-
1 table Sum		3	InnoDB	utf8mb4_general_ci	96.0 KiB	0 B

The structure is as follows :

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	id	bigint(20)			No	None	AUTO_INCREMENT		Change Drop More
<input type="checkbox"/> 2	user_id	bigint(20)			No	None			Change Drop More
<input type="checkbox"/> 3	user_name	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/> 4	password	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/> 5	productkey	int(5)			No	None			Change Drop More
<input type="checkbox"/> 6	date	timestamp			No	current_timestamp()	ON UPDATE CURRENT_TIMESTAMP()		Change Drop More

Resources Utilised : Images / Videos Used in Designing

Structure of the Folder : RentManagementSystem/rms/assets_login/



01. login-page-img.jpg

Chapter : 2

View ,Add & Remove Rooms

- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

The Add / Remove Module in Rent Management System

The Add/ Remove Functionality included in the Rent Management System Allows the Landlord to Add the rooms for Display . The Display of interiors of the room , also the blueprint will be able to be showcased , also the pdf form is fast to be shared to contacts if required.

To Access the Listing of the Rooms and the feature to add and remove the rooms gives the landlord the ability to showcase the rooms to those for enquiry and also for the purpose of the online advertisement .

The Landlord will be able to store the interior details and room display compactly as the form of pdf. The Pdf can be stored to the database , and through the help of it , it can also be viewed.

Snapshots :



Clicking on the above Option will lead the landlord to the next Pages whose site map is As follows .

Room listing -> View/Add Rooms /Remove Rooms ->User DashBoard

The Rooms Listing Module provides the View Rooms , View interiors of the Rooms , To Add Rooms and To remove the Rooms.

This will require the use of the File Structures and handling by the use of the PHP and MySql lite database.

The main pages present in the Room Listing Module are :

- (i) roomlisting.php
- (ii) addrooms.php
- (iii) removerooms.php

All the above files are the components of the room subfolder that is present inside the Rent Management System

These pages and the functionality they encompass are described in breif :

Room Listing Php contains the various links that interlink the above features of viewing , adding and removing the rooms features.

Add Rooms Php will help to add the rooms to the database , by following the rules followed to insert and update the values. The files are updated in the form of pdf for easy uploading and sharing.

Remove Rooms Php : this will help to remove the rooms from the listing , on the requirement of the Landlord.

Source Code of the Component pages for View / Add & Remove Rooms Module

Roomlistings Page (roomlisting.php)

```
<?php
$server="localhost";
$pass="";
$user="root";
$dbname="rent_management_system";
$conn=mysqli_connect($server,$user,$pass,$dbname);
$sql="select * from file_upload";
$result=$conn->query($sql);
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Room Listing</title>
    <link rel="stylesheet" href="roomlist.css">
</head>
<body>
```

```

<nav><a href="../index.php">Go to Home Page</a><br>
    <a href="../logout.php">Click to logout</a><br>
    <a href="addrrooms.php">Add Rooms</a><br>
    <a href="remove.php">Remove Rooms</a><br>
</nav>
<h1>List of Available Rooms</h1>
<table>
    <thead>
        <tr>
            <th>serial No.</th>
            <th>room id </th>
            <th>Rooms Name</th>
            <th>Status</th>
            <th>Room Layouts</th>
        </tr>
    </thead>
    <tbody>
        <?php
        $count=1;
        if($result->num_rows>0) {
            while($row=$result->fetch_assoc()) {
                echo"<tr>";
                echo"<td>".$count."</td>";
                echo"<td>".$row['id']."</td>";
                echo"<td>".$row['roomname']."</td>";
                echo"<td>".$row['status']."</td>";
                echo"<td><button id='openPdfBtn'>View
Rooms</button></td>";
                echo"</tr>";
                $count++;
            }
        }else{
            echo"<tr><td colspan=5>No records found</td></tr>";
        }
        ?>

    </tbody>
</table>

<script>
    document.addEventListener('DOMContentLoaded', () => {
const layoutButtons = document.querySelectorAll('.layout-btn');

layoutButtons.forEach(button => {
    button.addEventListener('click', () => {
        const roomNumber = button.getAttribute('data-room');
        alert(`Displaying layout for Room ${roomNumber}`);
    })
})

```

```

        // Here you can implement more complex logic, such as
fetching and displaying room layout details.
    });
});
});

document.getElementById('openPdfBtn').onclick = function() {
    window.open('uploads/room1.pdf', '_blank');
};

</script>
</body>
</html>

```

Add Rooms Page (addrooms.php)

```

<?php
$server="localhost";
$pass="";
$user="root";
$dbname="rent_management_system";
$conn=mysqli_connect($server,$user,$pass,$dbname);
$roomName=$_POST['roomname'];
$roomStatus=$_POST['roomstatus'];
if(isset($_POST['submit'])){
$targetDir='uploads/';
$targetFile=$targetDir.basename($_FILES['pdfFile']['name']);
$fileType=strtolower(pathinfo($targetFile,PATHINFO_EXTENSION));
if($fileType!="pdf" || $_FILES['pdfFile']['size']>6000000){
    echo"only pdf + 2mb less than";
}
else{
    //move it to the uploads folder
    if(move_uploaded_file($_FILES['pdfFile']['tmp_name'],$targetFile)){
        $filename=$_FILES['pdfFile']['name'];
        $folder_path=$targetDir;
        $timestamp=date('Y-m-d H:i:s');
        $sql="Insert into
file_upload(filename,roomname,status,folder_path,timestamp)
values('$filename','$roomName','$roomStatus','$folder_path','$timestamp')";
        if($conn->query($sql)==TRUE) {
            echo"success";
        }
        else{
            echo"Error".$sql."<br>".$conn->error;
        }
    }
}
}

```

```

        else{
            echo"Error upload!";
        }
    }
$conn->close();

?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Add Room Details</title>
    <link rel="stylesheet" href="roomlist.css">
</head>
<body>
    <h1>Add Rooms Details</h1><br>
    <form action="addrooms.php" method="post" enctype="multipart/form-
data">
        <label for="">Name of Room :</label>
        <input type="text" name="roomname"><br><br>
        <label for="">Status (Free/Occupied) :</label>
        <input type="text" name="roomstatus"><br><br>
        <label for="">Interiors and Layout (.pdf)</label>
        <input type="file" name='pdfFile' id=pdfFile>
        <button type="submit" name="submit">Submit</button>
    </form>
    <a href="roomlisting.php">Go to Room Listing</a>
</body>
</html>

```

Remove Rooms Page (remove.php)

```

<?php
$server="localhost";
$pass="";
$user="root";
$dbname="rent_management_system";
$conn=mysqli_connect($server,$user,$pass,$dbname);
$sql="select * from file_upload";
$result=$conn->query($sql);
$delId=$_POST['idtodelete'];
$sql1 = "DELETE FROM file_upload WHERE id = ?";

if ($stmt = $conn->prepare($sql1)) {
    $stmt->bind_param("i", $delId);
}

```

```

        }
        if ($stmt->execute()) {
            echo "Record deleted successfully";

        } else {
            echo "Error deleting record: " . $conn->error;
        }

$stmt->close();

?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Remove Rooms</title>
    <link rel="stylesheet" href="roomlist.css">
</head>
<body>
    <nav><a href="roomlisting.php">Go to Home Page</a><br>
        <a href="logout.php">Click to logout</a><br>
        <a href="addrrooms.php">Add Rooms</a><br>
        <a href="delete.php">Remove Rooms</a><br>
    </nav>
    <h1>List of Available Rooms</h1>
    <table>
        <thead>
            <tr>
                <th>serial No.</th>
                <th>room id </th>
                <th>Rooms Name</th>
                <th>Status</th>
                <th>Room Layouts</th>
            </tr>
        </thead>
        <tbody>
            <?php
                $count=1;
                if($result->num_rows>0) {
                    while($row=$result->fetch_assoc()) {
                        echo"<tr>";
                        echo"<td>".$count."</td>";
                        echo"<td>".$row['id']."'</td>";
                        echo"<td>".$row['roomname']."'</td>";
                    }
                }
            </tbody>
        </table>
    </body>
</html>

```

```

        echo"<td>".$row['status']."</td>";
        echo"<td><button id='openPdfBtn'>View
Rooms</button></td>";
        echo"</tr>";
        $count++;
    }
} else{
    echo"<tr><td colspan=5>No records found</td></tr>";
}
?>

</tbody>
</table>
<form action="remove.php" method="post">
<h1>Delete Rooms</h1>
<label for="">Enter the room id to Delete the room</label>
<input type="number" name="idtodelete" id="">
<button type="submit" name='submit'>Delete</button>
</form>
<script>
    document.addEventListener('DOMContentLoaded', () => {
const layoutButtons = document.querySelectorAll('.layout-btn');

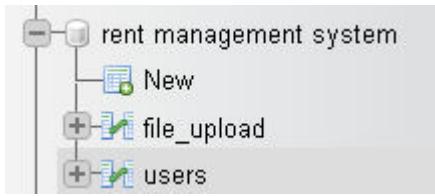
layoutButtons.forEach(button => {
    button.addEventListener('click', () => {
        const roomNumber = button.getAttribute('data-room');
        alert(`Displaying layout for Room ${roomNumber}`);
        // Here you can implement more complex logic, such as
fetching and displaying room layout details.
    });
});
});

document.getElementById('openPdfBtn').onclick = function() {
    window.open('uploads/pdf.pdf', '_blank');
};

</script>
</body>
</html>

```

Database Structures for the View / Add / Remove Rooms



The file_upload table is utilised to store the details of the rooms , it also contains the Room interiors and the basis of carrying out the rooms adding/removal will be provided by the table.

The table structure is as follows :

	<input type="button" value="←"/>	<input type="button" value="→"/>		<input type="button" value="▼"/>	id	filename	roomname	status	folder_path	timestamp
<input type="checkbox"/>					1	room1.pdf	Room1	Free	uploads/	2024-06-20 11:29:10

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1 id	int(11)			No	None		AUTO_INCREMENT	More
<input type="checkbox"/>	2 filename	varchar(50)	utf8mb4_general_ci		No	None			More
<input type="checkbox"/>	3 roomname	varchar(25)	utf8mb4_general_ci		No	None			More
<input type="checkbox"/>	4 status	varchar(10)	utf8mb4_general_ci		No	None			More
<input type="checkbox"/>	5 folder_path	varchar(100)	utf8mb4_general_ci		No	None			More
<input type="checkbox"/>	6 timestamp	datetime			No	None			More

Snapshots of the Module

The screenshot shows a web application interface for managing rooms. At the top, there's a navigation bar with links like 'Go to Home Page', 'Click to logout', 'Add Rooms', and 'Remove Rooms'. Below this is a section titled 'List of Available Rooms' containing a table with the following data:

serial No.	room id	Rooms Name	Status	Room Layouts
1	1	Room1	Free	<input type="button" value="Open PDF"/>

The Picture is of the main page of the Module to View / Add / Remove rooms.

List of Available Rooms				
serial No.	room id	Rooms Name	Status	Room Layouts
1	1	Room1	Free	View rooms

clicking on view rooms help us to see the internal designs of the rooms that include the blueprint and the pictures of the Rooms.

Page To Add the Rooms

Add Rooms Details

Name of Room :

Status(Free/Occupied):

Interiors and Layout(.pdf)

No file chosen

The Room name , its current availability , and its blueprint can be uploaded and saved to it. Once adding the rooms the new view can be seen as below:

Add Rooms Details

Name of Room :

Status(Free/Occupied):

Interiors and Layout(.pdf)

 room2.pdf

<input type="checkbox"/>		Edit		Copy		Delete	1	room1.pdf	Room1	Free	uploads/	2024-06-20 11:29:10
<input type="checkbox"/>		Edit		Copy		Delete	2	room2.pdf	Room2	Free	uploads/	2024-06-20 12:18:45

[Go to Home Page](#)
[Click to logout](#)
[Add Rooms](#)
[Remove Rooms](#)

List of Available Rooms

serial No.	room id	Rooms Name	Status	Room Layouts
1	1	Room1	Free	view rooms
2	2	Room2	Free	view rooms

The Room was successfully added to the listing page .

To remove the room 2 , we will click on the option given above as Remove Rooms

serial No.	room id	Rooms Name	Status	Room Layouts
1	1	Room1	Free	<button>Open PDF</button>
2	2	Room2	Free	<button>Open PDF</button>

Delete Rooms

Enter the room id to Delete the room

Delete

Once The Delete Button is Clicked , the Alert appers on the user window stating that the room was successfully deleted from the room listing.

The screenshot shows a web browser window with the following details:

- Address Bar:** localhost/rmstest5/rooms/remove.php
- Message Bar:** Record deleted successfully
- Navigation Buttons:** Go to Home Page, Click to logout, Add Rooms, Remove Rooms
- Section Header:** List of Available Rooms

serial No.	room id	Rooms Name	Status	Room Layouts
1	1	Room1	Free	<button>Open PDF</button>

- Delete Room Form:** Enter the room id to Delete the room (with input field value: 2)

Chapter : 3

Add Communication Details

- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

Enquiry Details Registration

The Rent Management System also features the facility to record the details of the consumers that come for the enquiry purposes, This is providing a form that also stores the details and produces the pdf , of those records to share and store them in the pdf format.

This will provide the feature for the Landlord to contact those consumers in future , to invite them to see the rooms available for the rental purposes , if they require it .

Folder Structure : RMS / enq_comm/

- | -> enq_comm.php (enquiry - communication details form)
- | -> print.php

The above files hold the task to record and produce the printout of the records in 2 formats :

- (i) To Produce PDF of all the records all together.
- (ii) To Produce the PDF of the Specific Records only.

The Database Structure facilitating it is as follows:

The screenshot shows the MySQL Workbench interface with the following details:

- Server: 127.0.0.1
- Database: rent_management_system
- Table: enq_comm
- Table structure view is selected.
- Columns listed:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)	utf8mb4_general_ci		No	None		AUTO_INCREMENT	Change Drop More
2	name	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
3	age	int(11)			No	None			Change Drop More
4	gender	varchar(10)	utf8mb4_general_ci		No	None			Change Drop More
5	address	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
6	mobile	bigint(20)			No	None			Change Drop More
7	time_of_saving	timestamp			No	current_timestamp()			Change Drop More

Their respective source code are as follows:

enq_comm.php (Enquiry Details Form)

```
<?php  
// Database connection  
  
$servername = "localhost";  
$username = "root"; // Replace with your database username  
$password = ""; // Replace with your database password  
$dbname = "rent_management_system";  
$conn = new mysqli($servername, $username, $password, $dbname);  
if ($conn->connect_error) {  
    die("Connection failed: " . $conn->connect_error);  
}  
  
// Insert, Update, Delete operations  
if ($_SERVER["REQUEST_METHOD"] == "POST") {  
    if (isset($_POST['save'])) {  
        $name = $_POST['name'];  
        $age = $_POST['age'];  
        $gender = $_POST['gender'];  
        $address = $_POST['address'];  
        $mobile = $_POST['mobile'];  
        $id = $_POST['id'];  
  
        if ($id) {  
            // Update existing record  
            $sql = "UPDATE enq_comm SET name='$name', age='$age', gender='$gender', address='$address', mobile='$mobile' WHERE id=$id";  
        } else {  
            // Insert new record  
            $sql = "INSERT INTO enq_comm (name, age, gender, address, mobile)  
VALUES ('$name', '$age', '$gender', '$address', '$mobile')";  
        }  
    }  
}
```

```

$conn->query($sql);
}

if (isset($_POST['delete'])) {
    $id = $_POST['id'];
    $sql = "DELETE FROM enq_comm WHERE id=$id";
    $conn->query($sql);
}

if (isset($_POST['print'])) {
    $id = $_POST['id'];
    header("Location: print.php?id=$id");
    exit();
}

header("Location: enq_comm.php");
exit();
}

// Retrieve data
$sql = "SELECT * FROM enq_comm";
$result = $conn->query($sql);
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Details_Enquiry</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;

```

```
padding: 0;
background-color: #f4f4f4;
color: #333;
}

h1 {
    text-align: center;
    margin-top: 20px;
}

form {
    max-width: 600px;
    margin: 20px auto;
    padding: 20px;
    background: #fff;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

.form-group {
    margin: 10px 0;
}

.form-group label {
    display: block;
    margin-bottom: 5px;
    font-weight: bold;
}

.form-group input, .form-group select {
    width: 100%;
    padding: 8px;
    box-sizing: border-box;
    border: 1px solid #ccc;
    border-radius: 4px;
}

.form-actions {
    text-align: center;
    margin-top: 20px;
```

```
}

.form-actions button {
    padding: 10px 20px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    font-size: 16px;
    transition: background-color 0.3s ease;
}

.form-actions button[name="save"] {
    background-color: #28a745;
    color: #fff;
}

.form-actions button[name="save"]:hover {
    background-color: #218838;
}

table {
    width: 80%;
    margin: 20px auto;
    border-collapse: collapse;
    background: #fff;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

table, th, td {
    border: 1px solid #ddd;
}

th, td {
    padding: 12px;
    text-align: left;
}

th {
    background-color: #f8f9fa;
}

td {
```

```
background-color: #fff;
}

td button {
    padding: 5px 10px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    font-size: 14px;
    transition: background-color 0.3s ease;
}

td button[name="delete"] {
    background-color: #dc3545;
    color: #fff;
}

td button[name="delete"]:hover {
    background-color: #c82333;
}

td button[name="edit"] {
    background-color: #007bff;
    color: #fff;
}

td button[name="edit"]:hover {
    background-color: #0069d9;
}

.print-button {
    display: block;
    width: 150px;
    margin: 20px auto;
    padding: 10px 20px;
    background-color: #007bff;
    color: #fff;
    text-align: center;
    border-radius: 4px;
    cursor: pointer;
}
```

```

        transition: background-color 0.3s ease;
    }
    .print-button:hover {
        background-color: #0069d9;
    }

```

</style>

```

<script
src="https://cdnjs.cloudflare.com/ajax/libs/html2pdf.js/0.9.2/html2pdf.bundle.min.js"></script>
```

```

<script>

    function editRecord(id, name, age, gender, address) {
        document.getElementById('id').value = id;
        document.getElementById('name').value = name;
        document.getElementById('age').value = age;
        document.getElementById('gender').value = gender;
        document.getElementById('address').value = address;
    }

    function printPDF() {
        const element = document.getElementById('records');
        html2pdf().from(element).save();
    }

```

</script>

</head>

<body>

```

<h1>Enquiry / Communication Details</h1>
<form method="post" action="enq_comm.php">
    <input type="hidden" id="id" name="id">
    <div class="form-group">
        <label for="name">Name:</label>
        <input type="text" id="name" name="name" required>
    </div>
    <div class="form-group">
        <label for="age">Age:</label>
        <input type="number" id="age" name="age" required>

```

```

</div>

<div class="form-group">
    <label for="gender">Gender:</label>
    <select id="gender" name="gender" required>
        <option value="Male">Male</option>
        <option value="Female">Female</option>
    </select>
</div>

<div class="form-group">
    <label for="address">Address:</label>
    <input type="text" id="address" name="address" required>
</div>

<div class="form-group">
    <label for="mobile">Contact No.:</label>
    <input type="text" id="mobile" name="mobile" required>
</div>

<div class="form-actions">
    <button type="submit" name="save">Save</button>
</div>

</form>

<div id="records">
    <table>
        <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Age</th>
            <th>Gender</th>
            <th>Address</th>
            <th>Contact no</th>
            <th>Actions</th>
        </tr>
    <?php
        if ($result->num_rows > 0) {

```

```

        while($row = $result->fetch_assoc()) {
            echo "<tr>
                <td>{$row['id']}</td>
                <td>{$row['name']}</td>
                <td>{$row['age']}</td>
                <td>{$row['gender']}</td>
                <td>{$row['address']}</td>
                <td>{$row['mobile']}</td>
                <td>
                    <button name=\"edit\" onclick=\"editRecord('{$row['id']}', '{$row['name']}','{$row['age']}',
                    '{$row['gender']}','{$row['address']}')\">Edit</button>
                    <form method='post' action='enq_comm.php' style='display:inline;'>
                        <input type='hidden' name='id' value='{$row['id']}'>
                        <button type='submit' name='delete'>Delete</button>
                    </form>
                    <form method='post' action='enq_comm.php' style='display:inline;'>
                        <input type='hidden' name='id' value='{$row['id']}'>
                        <button type='submit' name='print'>Print</button>
                    </form>
                </td>
            </tr>";
        }
    } else {
        echo "<tr><td colspan='6'>No records found</td></tr>";
    }
    $conn->close();
?>
</table>
</div>

```

```
<button class="print-button" onclick="printPDF()">Print to PDF</button>
</body>
</html>
```

print.php (file to print the records to the PDF format)

```
<?php
// Database connection

$servername = "localhost";
$username = "root"; // Replace with your database username
$password = ""; // Replace with your database password
$dbname = "rent_management_system";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

if (isset($_GET['id'])) {
    $id = $_GET['id'];
    $sql = "SELECT * FROM enq_comm WHERE id=$id";
    $result = $conn->query($sql);

    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
    } else {
        echo "No record found";
        exit();
    }
} else {
    echo "No ID provided";
    exit();
}
?>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Print Record</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f4f4f4;
            color: #333;
            display: flex;
            flex-direction: column;
            align-items: center;
            height: 100vh;
            justify-content: center;
        }
        .record-container {
            background: #fff;
            padding: 20px;
            border-radius: 8px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            max-width: 600px;
            width: 100%;
            margin-bottom: 20px;
        }
        .record-container h1 {
            text-align: center;
            margin-bottom: 20px;
        }
        .record-container p {
```

```
    margin: 10px 0;
    font-size: 18px;
}

.print-button {
    display: block;
    width: 150px;
    margin: 20px auto;
    padding: 10px 20px;
    background-color: #007bff;
    color: #fff;
    text-align: center;
    border-radius: 4px;
    cursor: pointer;
    transition: background-color 0.3s ease;
}

.print-button:hover {
    background-color: #0069d9;
}

.back-link {
    display: block;
    text-align: center;
    margin: 20px auto;
    padding: 10px 20px;
    background-color: #007bff;
    color: #fff;
    text-decoration: none;
    border-radius: 4px;
    width: 200px;
    transition: background-color 0.3s ease;
}

.back-link:hover {
    background-color: #0069d9;
}

</style>
```

```

<script
src="https://cdnjs.cloudflare.com/ajax/libs/html2pdf.js/0.9.2/html2pdf.bundle.min.js"></script>

<script>
    function printPDF() {
        const element = document.querySelector('.record-container');
        html2pdf().from(element).save();
    }
</script>

</head>
<body>

<div class="record-container">
    <h1>Contact Details</h1>
    <p><strong>ID:</strong> <?php echo $row['id']; ?></p>
    <p><strong>Name:</strong> <?php echo $row['name']; ?></p>
    <p><strong>Age:</strong> <?php echo $row['age']; ?></p>
    <p><strong>Gender:</strong> <?php echo $row['gender']; ?></p>
    <p><strong>Address:</strong> <?php echo $row['address']; ?></p>
    <p><strong>Date of Enquiry:</strong> <?php echo $row['time_of_saving']; ?></p>
    <p><strong>Contact No.:</strong> <?php echo $row['mobile']; ?></p>
    <button class="print-button" onclick="printPDF()">Print to PDF</button>
</div>

<a href="enq_comm.php" class="back-link">Go Back to Records page</a>

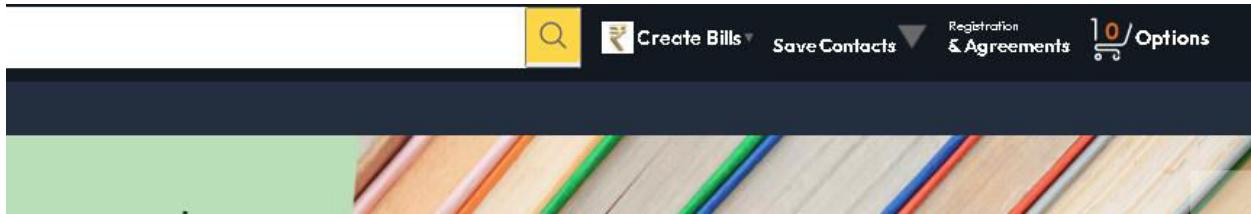
<script>
    document.querySelector('.back-link').addEventListener('mouseover',
function() {
    this.style.backgroundColor = '#0069d9';
});
    document.querySelector('.back-link').addEventListener('mouseout',
function() {
    this.style.backgroundColor = '#007bff';
});
</script>

```

```
</script>  
</body>  
</html>
```

Screenshots of the Enquiry Details Module

We click on the save contacts option :



This will lead us to the page that will record the consumers details who came for the purpose of enquiry.

Enquiry / Communication Details

Name:	<input type="text"/>
Age:	<input type="text"/>
Gender:	<input type="text"/> Male
Address:	<input type="text"/>
Contact No.:	<input type="text"/>

Save

ID	Name	Age	Gender	Address	Contact no	Actions
No records found						

Print to PDF

The current form is empty.

We will fill the details

Enquiry / Communication Details

Name: AAYUSH SAHAY

Age: 20

Gender: Male

Address: KADAMKUAN , PATNA , BIHAR

Contact No.: 9994445550

Save

ID	Name	Age	Gender	Address	Contact no	Actions
No records found						

Print to PDF

And then we click on the button to save it.

And in the below table , the details get automatic stored

ID	Name	Age	Gender	Address	Contact no	Actions
1	AAYUSH SAHAY	20	Male	KADAMKUAN , PATNA , BIHAR	9994445550	Edit Delete Print

Print to PDF

We have 2 choices , 1st we print the pdf , specifically as : clicking on the print button in the row of the id 1 .

We reach the page that looks like :

Contact Details

ID: 1
Name: AAYUSH SAHAY
Age: 20
Gender: Male
Address: KADAMKUAN , PATNA , BIHAR
Date of Enquiry: 2024-06-30 11:20:43
Contact No.: 9994445550

[Print to PDF](#)

[Go Back to Records page](#)

We click to print the pdf , the output is :

≡ file (7).pdf 1 / 1 | - 100% + | ☰ ⌂

Contact Details

ID: 1
Name: AAYUSH SAHAY
Age: 20
Gender: Male
Address: KADAMKUAN , PATNA , BIHAR
Date of Enquiry: 2024-06-30 11:20:43
Contact No.: 9994445550

[Print to PDF](#)

For printing multiple records in one single pdf document , we have added multiple records that are going to be displayed inside that pdf file :

ID	Name	Age	Gender	Address	Contact no	Actions		
1	AAYUSH SAHAY	20	Male	KADAMKUAN , PATNA , BIHAR	9994445550	Edit	Delete	Print
2	Akash Kumar	21	Male	Rajapur , Patna	1234512345	Edit	Delete	Print
3	Sneha Kumari	22	Female	Rajendra Nagar , Patna	9879874545	Edit	Delete	Print
Print to PDF								

The output is :

file (8).pdf

1 / 1 | - 100% + ⌂ ⌂

ID	Name	Age	Gender	Address	Contact no
1	AAYUSH SAHAY	20	Male	KADAMKUAN , BIHAR	9994445550
2	Akash Kumar	21	Male	Rajapur , Patna	1234512345
3	Sneha Kumari	22	Female	Rajendra Nagar , Patna	9879874545

Server: 127.0.0.1 > Database: rent_management_system > Table: enq_comm

[Browse](#) [Structure](#) [SQL](#) [Search](#) [Insert](#) [Export](#) [Import](#) [Privileges](#) [Operations](#) [Tracking](#)

Showing rows 0 - 2 (3 total, Query took 0.0004 seconds.)

```
SELECT * FROM `enq_comm`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

[Extra options](#)

	id	name	age	gender	address	mobile	time_of_saving
<input type="checkbox"/>	1	AAYUSH SAHAY	20	Male	KADAMKUAN , PATNA , BIHAR	9994445550	2024-06-30 11:20:43
<input type="checkbox"/>	2	Akash Kumar	21	Male	Rajapur , Patna	1234512345	2024-06-30 11:26:34
<input type="checkbox"/>	3	Sneha Kumari	22	Female	Rajendra Nagar , Patna	9879874545	2024-06-30 11:27:19

[↶](#) [Check all](#) [With selected](#) [Edit](#) [Copy](#) [Delete](#) [Export](#)

Chapter : 4

Reset Login Credentials / Reset User Password

- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

Password Reset

The Rent Management System aims to be robust as well as facility rich like the other software solutions in the market. The requirement of having the feature to rest the login credentials are more than a neccessity , The Data that is very precious for the business point of view., has risk to be accessed by other party incase they are known of the password.

This application Is a Web - Browser based , and when the web-browser is online , other services / malicious users may get password. So , it is a good practice to change the password . On the other hand , creating the backup of all the data is also crucial.

The Landlord can easily change the password by the use of the entery_email that is registered in the database. Then the product key that is going to be a different code for the other computers using the software . This Software is created to showcase the learning , therefore it is 55555.

And the Landlord can be able to reset / add a new password to the System.

Folder Structure : RMS / reset.php

reset.php (Page to reset the user password)

```
<?php  
session_start();  
include("connection.php");  
include("functions.php");  
if($_SERVER['REQUEST_METHOD']=="POST") {  
    $user_name=$_POST['user_name'];  
    $prokey=$_POST['prokey'];  
    $new_password = $_POST['new_password'];  
    if(!empty($user_name) && !empty($prokey) && !is_numeric($user_name)) {  
        // Read from the database  
        $query= "SELECT * FROM users WHERE user_name='$user_name' LIMIT 1";  
        $result1 = mysqli_query($con, $query);
```

```

if ($result1) {

    if ($result1 && mysqli_num_rows($result1) > 0) {

        $user_data = mysqli_fetch_assoc($result1);

        $update_query = "UPDATE users SET password='$new_password'
WHERE user_name='$user_name' and productkey='$prokey'";

        $update_result = mysqli_query($con, $update_query);

        if ($update_result) {

            // Password reset successful

            echo "<script>alert('Password      reset      successful');
window.location.href = 'login.php';</script>";

        } else {

            // Password reset failed

            echo "<script>alert('Password      reset      failed');
window.location.href = 'login.php';</script>";

        }

    } else {

        // User not found

        echo "<script>alert('User not found'); window.location.href =
'login.php';</script>";

    }

} else {

    // Query failed

    echo "<script>alert('Database query failed'); window.location.href
= 'login.php';</script>";

}

die;

```

```

    }

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Reset PWD</title>

<style>

 *{margin:0; padding:0; box-sizing:border-box; }

 body{min-height: 100vh; background:#eee; display:flex; font-family:sans-serif; }

 .container{margin:auto; width:500px; max-width:90%; }

 .container form{width:100%; height:100%; padding:20px; background:white;

 border-radius:4px; box-shadow: 0 8px 16px rgba(0,0,0,0.3); }

 .container form h1{text-align:center; margin-bottom: 24px; color:#222; }

 .container form .form-control{width:100%; height:40px; background:white; border-radius:4px; border:1px solid silver; margin:10px 0 18px 0; padding: 0 10px; }

 .container form .btn{margin-left:50%; transform:translate(-50%); width:120px;

 height:34px; border:none; outline:none; background:#222; cursor:pointer; font-size: 16px; text-transform:uppercase; color:white; border-radius: 4px; transition: 0.3s; }

 .container form .btn:hover{opacity:0.7; }

 #btn1 a{text-decoration: none; color: #eee; }

 #btn1{position:relative;

```

```

        top:-34px;
        left:140px;
        font-size: 12px; }

#btn2 a{text-decoration: none; color: #eee; }

#btn2{position:relative;
        top:-68px;
        left:-140px;
        font-size: 12px; }

#logo{width:626px; position: relative; top:80px; height:480px;
left:35px; }


```

</style>

</head>

<body>

```

<?php

if (isset($_GET['logout']) && $_GET['logout'] == 'success') {
    echo "<script>alert('User was logged out successfully');</script>";
} else {
    echo     "<script>console.log('Logout parameter not set or
incorrect');</script>";
}

?>


<div class="container">

        <form action="reset.php" method="post">
            <h1>Reset Password</h1>
            <div class="formgroup">

```

```

        <label for="">Email</label>

        <input type="email" class="form-control" name="user_name"
id="email" required>

    </div>

    <div class="formgroup">

        <label for="">Product Key</label>

        <input type="text" class="form-control" name="prokey"
id="prokey" required>

    </div>

    <div class="formgroup">

        <label for="">New Password</label>

        <input type="password" class="form-control"
name="new_password" id="password" required>

    </div>

    <input class="btn" type="submit" value="RESET">

    <button id="btn1" class="btn">
Login</button>

    <button id="btn2" class="btn"><a href="signup.php">Register
User</a></button>

</form>

</div>

</body>

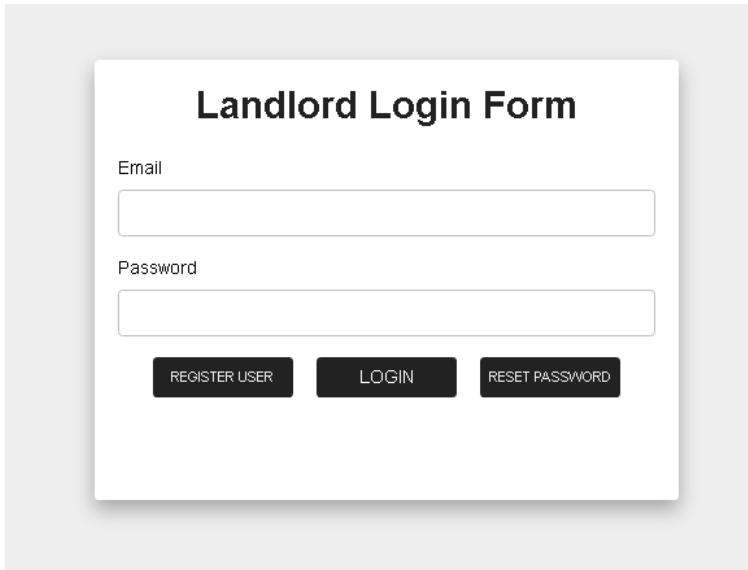
</html>

```

Features :

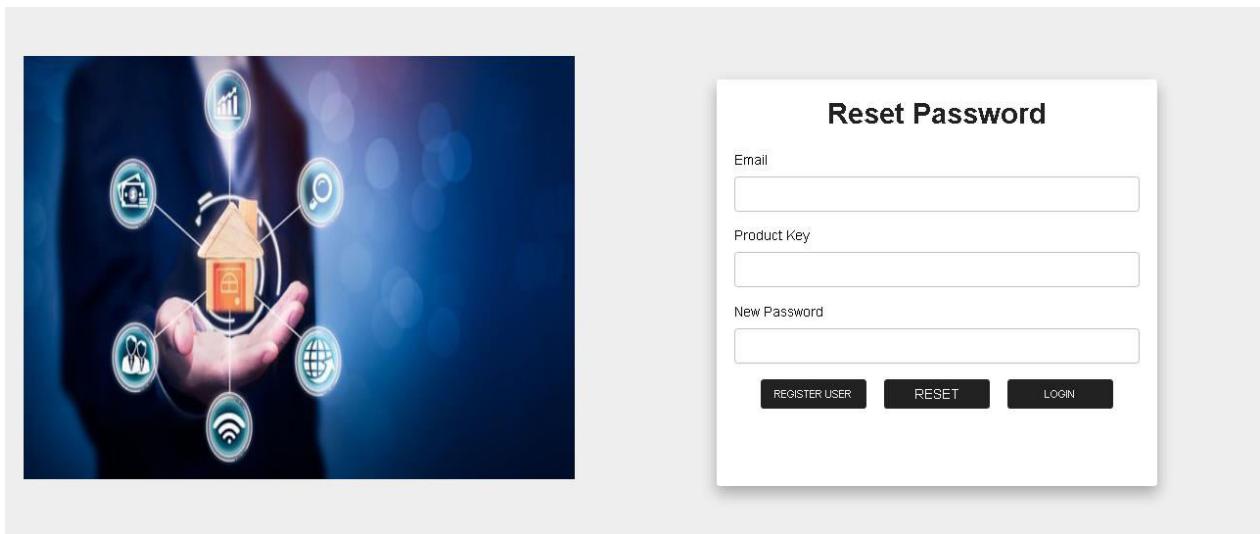
- (i) The user can reset the password from the main login and signup page.
- (ii) The user can reset while being logged in the System , Once the password is reset , it will redirect the user back to the login page , where he will be logging back to system using the new credentials

Screenshots



The screenshot shows a "Landlord Login Form" window. It has a title bar at the top with the text "Landlord Login Form". Below the title bar are two input fields: "Email" and "Password", each with a corresponding text input box. At the bottom of the form are three buttons: "REGISTER USER", "LOGIN", and "RESET PASSWORD".

We click on the reset password button.and fill in the details required to reset the password.



The screenshot shows a "Reset Password" window. On the left side, there is a decorative graphic of a hand holding a central house icon, which is surrounded by various circular icons representing different services or features like AI, security, connectivity, and more. To the right of the graphic is the "Reset Password" form. It includes fields for "Email" and "Product Key", both with text input boxes. Below these fields is a "New Password" field with a text input box. At the bottom of the form are three buttons: "REGISTER USER", "RESET", and "LOGIN".

We fill in the details as :

Reset Password

Email
test@gmail.com

Product Key
55555 : Password is test1

New Password
.....

[REGISTER USER](#) [RESET](#) [LOGIN](#)

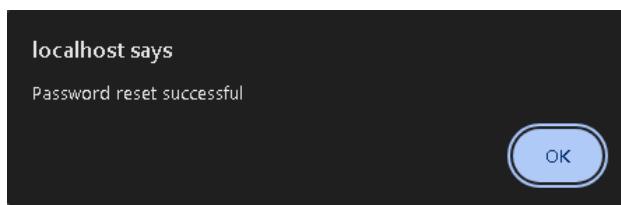
and then click on the reset button that will show us 4 results that can possibly happen:

(i) Error is Credentials like wrong username , or wrong product key.

(ii) Error lead to the password not change , caused by input of wrong / incorrect details

(iii) Internal Error of PHP services incase any exception encountered , leading to no run of the Query.

(iv) All credentials correct and the password was successfully changed.



	← T →	▼	id	user_id	user_name	password	productkey	date
<input type="checkbox"/>	Edit	Copy	Delete	1	7	test@gmail.com	test1	55555 2024-06-30 20:01:20

Chapter : 5

Share Location of the Office Page

- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

Share Location

The Rent Management System is a web based solution to manage the business of the room rental services. This is a web-browser operable application. It is also equipped with the feature to show the location of the office. Incase the landlord has property in a distributed manner / or he is featuring other plots of land on which rent is applicable , the location of the Google maps can be swiftly be shared. This features the location , depicted by the google maps and the share link by copy to the clipboard feature.

Folder Structure

Rent Management System

```
| -> loc_office
    | -> loc_office.php
    | -> styles11.css
    | -> script11.js
```

Source Code : loc_office.php

```
<?php
session_start();
include("../connection.php");
include("../functions.php");
$user_data=check_login($con);
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Location</title>
    <link rel="stylesheet" href="styles11.css">
</head>
<body>
    <div class="content">
        <div id="mapContainer">
```

```

        <iframe
src="https://www.google.com/maps/embed?pb=!1m26!1m12!1m3!1d224.871037742979!2d
85.1508438985214!3d25.607022778436473!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!
4m11!3e9!4m3!3m2!1d25.6068686!2d85.1510218999999!4m5!1s0x39ed588b1964d779%3A0
xd97cecb8666dc308!2sUpadhyay%20Ln%2C%20West%20Lohanipur%2C%20Lohanipur%2C%20Pa
tna%2C%20Bihar%20800003!3m2!1d25.6057955!2d85.1509610999999!5e0!3m2!1sen!2sin
!4v1719842826275!5m2!1sen!2sin" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy" referrerPolicy="no-referrer-when-downgrade"></iframe>

    </div>

    <button id="copyLink">Click to copy link</button>

    <button id="goback"><a href="../index.php">Go to home page</a></button>

</div>

<script src="script11.js"></script>

</body>
</html>

```

Source Code : styles11.css

```

body {
    font-family: Arial, sans-serif; display: flex; justify-content: center;
    align-items: center; height: 100vh; margin: 0;
    background-color: #f0f0f0;
}

.content {
    text-align: center;
}

#mapContainer {
    margin-bottom: 20px; border: 1px solid #ccc; border-radius: 10px;
    overflow: hidden; width: 600px; height: 450px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);
}

button {
    padding: 10px 20px; background-color: #007BFF; color: white;
}

```

```

border: none; border-radius: 5px; cursor: pointer;
}

button:hover {
    background-color: #0056b3;
}

#goback a{
    color: white;
    text-decoration: none;
}

```

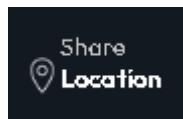
Source Code : script11.css

```

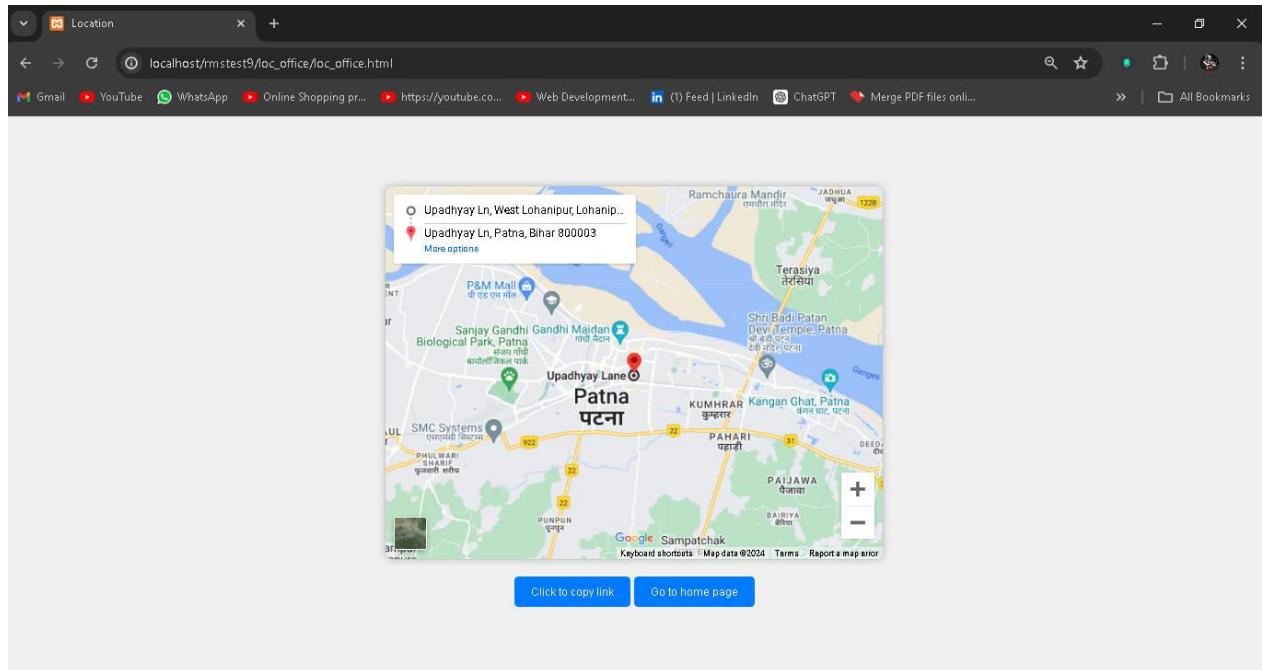
document.getElementById('copyLink').addEventListener('click', function() {
    const link = "https://maps.app.goo.gl/UBr1LwzngEu6pGmL7";
    navigator.clipboard.writeText(link).then(function() {
        alert('Link copied to clipboard');
    }).catch(function(error) {
        console.error('Error copying text: ', error);
    });
});

```

Screenshots



We click on the icon that redirects us to the loc_office.html that is responsible to show us the office location. That page looks like:



From that page , either we can share the location by copying the link to the clipboard by that copy to clipboard button , or we can go back to the main page .

This is ideal for the purpose of swift sharing of the location by use of google maps , also the consumer is not much technologically adept , the location can be communicated by use of the written format like the cards for contact purposes.

This page can be also be used for advertisements :" Rent A Room in Patna, Kadamkuan Region", as people are usually in need for shifting to new accomodations.

Chapter : 6

Store the Registration Details of the Renter

- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

Register Renter Details

The Rent Management System supports the preliminary purpose of registering the details of the consumers who have made their mind to have a room on rent. It is very crucial for the business to store the details. The details stored here will be then carried forward for the purpose of Agreement page. This page stores simple details .

The Agreement page will require submission of the crucial documents that include the adhar card , photo of family / photo of main member , signature.

This page will help to store to the database :rent_management_system

Folder Structure

Rent Management System

```
| -> reg_renter  
    | -> reg_renter.php
```

Source Code : reg_renter.php

```
<?php  
$servername = "localhost";  
$username = "root";  
$password = "";  
$database = "rent_management_system";  
  
// Create connection  
$conn = new mysqli($servername, $username, $password, $database);  
  
// Check connection  
if ($conn->connect_error) {  
    die("Connection failed: " . $conn->connect_error);  
}  
  
$message = '';  
  
if ($_SERVER["REQUEST_METHOD"] == "POST") {  
    $name = $_POST['name'];  
    $age = $_POST['age'];
```

```

$gender = $_POST['gender'];
$dreg = $_POST['dreg'];
$aadhar = $_POST['aadhar'];
$mobile = $_POST['mobile'];
$address = $_POST['address'];
$roomNo = $_POST['roomNo'];
$price = $_POST['price'];

// Prepare and bind
$stmt = $conn->prepare("INSERT INTO reg_renter (name, age, gender,
datereg, aadhar, mobile, address, roomNo, price) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)");
if ($stmt === false) {
    $message = "Prepare failed: " . $conn->error;
} else {
    $bind = $stmt->bind_param("sissssssd", $name, $age, $gender, $dreg,
$aadhar, $mobile, $address, $roomNo, $price);
    if ($bind === false) {
        $message = "Bind failed: " . $stmt->error;
    } else {
        $exec = $stmt->execute();
        if ($exec === false) {
            $message = "Execute failed: " . $stmt->error;
        } else {
            $message = "New record created successfully";
        }
    }
}

// Close the statement and connection
$stmt->close();
$conn->close();
}

?>

<!DOCTYPE html>
<html lang="en">
<head>

```

```
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Room Rental Registration Form</title>
<style>
    body {
        font-family: Arial, sans-serif;
        background-color: #f7f7f7;
        display: flex;
        justify-content: center;
        align-items: center;
        height: 100vh;
        margin: 0;
    }

    .container {
        position: relative;
        top: 95px;
        background: #fff;
        padding: 20px;
        border-radius: 10px;
        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        width: 100%;
        max-width: 600px;
    }

    h2 {
        text-align: center;
        margin-bottom: 20px;
    }

    .form-group {
        margin-bottom: 15px;
    }

    .form-group label {
        display: block;
        margin-bottom: 5px;
    }
```

```
    font-weight: bold;
}

.form-group input,
.form-group select,
.form-group textarea {
    width: 100%;
    padding: 10px;
    border: 1px solid #ccc;
    border-radius: 5px;
}

textarea {
    resize: vertical;
}

button {
    width: 100%;
    padding: 10px;
    background-color: #007BFF;
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}

button:hover {
    background-color: #0056b3;
}

@media (max-width: 600px) {
    .container {
        padding: 15px;
    }
}

.formlinks {
```

```

        display: inline-block;
        margin: 10px 0;
        font-size: 16px;
        font-weight: bold;
    }

    .formlinks a {
        color: #007BFF;
        text-decoration: none;
    }

    .formlinks a:hover {
        color: #0056b3;
        text-decoration: underline;
    }
}

</style>
</head>
<body>

<div class="container">

    <h2>Room Rental Registration Form</h2>

    <h4><a href="#">Homepage</a>&ampnbsp&ampnbsp&ampnbsp<a href="#">Logout</a></h4>

    <form id="registrationForm" action="reg_renter.php" method="post">

        <div class="form-group">
            <label for="name">Name (Applying for Renting a room)</label>
            <input type="text" id="name" name="name" required>
        </div>

        <div class="form-group">
            <label for="age">Age:</label>
            <input type="number" id="age" name="age" required>
        </div>

        <div class="form-group">
            <label for="gender">Gender:</label>
            <select id="gender" name="gender" required>
                <option value="">Select</option>
                <option value="Male">Male</option>
                <option value="Female">Female</option>
            </select>
        </div>
    </form>
</div>

```

```

        <option value="Other">Other</option>
    </select>
</div>

<div class="form-group">
    <label for="dob">Date of Registration:</label>
    <input type="date" id="dreg" name="dreg" required>
</div>

<div class="form-group">
    <label for="aadhar">ID Details (Aadhar card):</label>
    <input type="text" id="aadhar" name="aadhar" required>
</div>

<div class="form-group">
    <label for="mobile">Mobile No. with WhatsApp:</label>
    <input type="tel" id="mobile" name="mobile" required>
</div>

<div class="form-group">
    <label for="address">Address as in Aadhar card:</label>
    <textarea id="address" name="address" required></textarea>
</div>

<div class="form-group">
    <label for="roomNo">Room No. selected:</label>
    <input type="text" id="roomNo" name="roomNo" required>
</div>

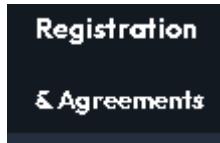
<div class="form-group">
    <label for="price">Negotiated Price to Rent the Room in
    ₹:</label>
    <input type="number" id="price" name="price" required>
</div>

    <button type="submit">Submit</button>
</form>

<?php if (!empty($message)) : ?>
<script>
    alert('<?php echo $message; ?>');
</script>
<?php endif; ?>
</div>
</body>
```

```
</html>
```

Screenshots (Renter Registration Page)



In the navigation bar of the index page , we see the above feature. We will click on Registration.This will lead us to the page that is reg_renter.php

Room Rental Registration Form

[Homepage](#) [Logout](#)

Name (Applying for Renting a room):

Age:

Gender:

Select

Date of Registration:

 mm/dd/yyyy

ID Details (Aadhar card):

Mobile No. with WhatsApp:

Address as in Aadhar card:

Room No. selected:

Negotiated Price to Rent the Room in ₹:

Submit

Currently it is empty , we will the details as :

Room Rental Registration Form

[Homepage](#) [Logout](#)

Name (Applying for Renting a room):

Mr. Amit Kumar

Age:

32

Gender:

Male

Date of Registration:

07/02/2024



ID Details (Aadhar card):

2323232323

Mobile No. with WhatsApp:

9090909090

Address as in Aadhar card:

Gaya , Bihar

Room No. selected:

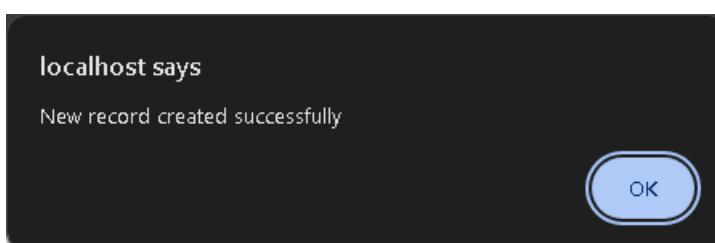
1

Negotiated Price to Rent the Room in ₹:

2500

Submit

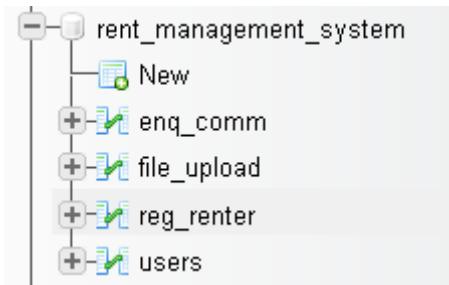
That on submission gives us the alert :



←→	▼	id	name	age	gender	date	aadhar	mobile	address	roomNo	price					
<input type="checkbox"/>		Edit		Copy		Delete	1	Mr.Amit Kumar	32	Male	2024	2323232323	9090909090	Gaya , Bihar	1	2500.00

Database Structure

The Register Renter page stores the details in the reg_renter table present in the rent_management_system database:



The structure of the reg_renter is as follows :

#		Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/>	2	name	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/>	3	age	int(11)			No	None			Change Drop More
<input type="checkbox"/>	4	gender	enum('Male', 'Female', 'Other')	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/>	5	datereg	date			No	None			Change Drop More
<input type="checkbox"/>	6	aadhar	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/>	7	mobile	varchar(15)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/>	8	address	text	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/>	9	roomNo	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/>	10	price	decimal(10,2)			No	None			Change Drop More

Chapter : 7

Agreements Page

- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

Signing Agreement with the Tenant

The Rent Management System automates the process of recording the details necessary for the verification purposes , before signing the contract. The Rental Business for rooms and accomodations has to comply with the legal regulations.

This is an electronic form that is responsible to collect and house the details , and produce a webpage that is printable to show that the agreement was signed between the consumer and the landlord.

Folder Structure

Rent Management System

```
| -> rent_agr
    | -> stores ( folder that stores all the files of respective entries)
    | ->ag_resc.php
    | ->process_agreement.php
    | ->print_agr.php
```

Source code : ag_resc.php(Agreement Resource)

This is going to collect the Picture , Id of the Registration of the user and the crucial documents like pictures , Adhar Card and sign . It will store that to the database. This is going to work with the logic present in the process_agreement.php

```
<?php
session_start();
include("../connection.php");
include("../functions.php");
$user_data=check_login($con);
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Agreement Form</title>
```

```
<style>

body {
    font-family: 'Arial', sans-serif;
    margin: 20px;
    background-color: #f4f4f4;
}

.container {
    max-width: 600px;
    margin: 0 auto;
    background-color: #fff;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0,0,0,0.1);
}

form {
    display: flex;
    flex-direction: column;
}

label {
    margin-bottom: 8px;
    font-weight: bold;
}

input[type="text"], input[type="file"] {
    padding: 8px;
    margin-bottom: 12px;
    border: 1px solid #ccc;
    border-radius: 4px;
    font-size: 14px;
}

input[type="file"] {
    cursor: pointer;
}
```

```
#preview {  
    display: flex;  
    justify-content: space-between;  
    flex-wrap: wrap;  
    margin-bottom: 20px;  
}  
  
#preview div {  
    width: calc(33.33% - 10px);  
    margin-bottom: 10px;  
}  
  
#preview img {  
    max-width: 100%;  
    height: auto;  
    border-radius: 4px;  
}  
  
input[type="submit"] {  
    background-color: #4CAF50;  
    color: white;  
    border: none;  
    padding: 12px 20px;  
    text-align: center;  
    text-decoration: none;  
    display: inline-block;  
    font-size: 16px;  
    border-radius: 4px;  
    cursor: pointer;  
    transition: background-color 0.3s;  
}  
  
input[type="submit"]:hover {  
    background-color: #45a049;  
}  
</style>
```

```

</head>
<body>
    <div class="container">
        <h2 style="text-align: center;">Agreement Form</h2>
        <form id="agreementForm" action="process_agreement.php"
method="post" enctype="multipart/form-data">
            <label for="name">Name:</label>
            <input type="text" name="name" id="name" required>

            <label for="room_id">Room ID:</label>
            <input type="text" name="room_id" id="room_id" required>

            <label for="photo">Upload Photograph:</label>
            <input type="file" name="photo" id="photo" accept="image/*"
required>

            <label for="signature">Upload Signature:</label>
            <input type="file" name="signature" id="signature"
accept="image/*" required>

            <label for="aadhar">Upload Aadhar Card:</label>
            <input type="file" name="aadhar" id="aadhar" accept="image/*"
required>

        <div id="preview">
            <div>
                <p>Photo Preview:</p>
                
            </div>
            <div>
                <p>Signature Preview:</p>
                
            </div>
            <div>
                <p>Aadhar Card Preview:</p>
                
            </div>
        </div>
    </div>

```

```

        <input type="submit" value="Submit">
    </form>
    <a href="../index.php">Go to Home Page</a>
    <a href="print_agr.php">Produce the Agreement PDF</a>
</div>

<script>
    document.getElementById('photo').onchange = function (event) {
        const [file] = event.target.files;
        if (file) {
            document.getElementById('photoPreview').src = URL.createObjectURL(file);
        }
    };

    document.getElementById('signature').onchange = function (event) {
        const [file] = event.target.files;
        if (file) {
            document.getElementById('signaturePreview').src = URL.createObjectURL(file);
        }
    };

    document.getElementById('aadhar').onchange = function (event) {
        const [file] = event.target.files;
        if (file) {
            document.getElementById('aadharPreview').src = URL.createObjectURL(file);
        }
    };
}

document.getElementById('agreementForm').onsubmit = function(event)
{
    event.preventDefault();
    var formData = new FormData(this);

    fetch('process_agreement.php', {

```

```

        method: 'POST',
        body: formData
    })
    .then(response => response.json())
    .then(data => {
        if (data.success) {
            alert(data.message);
        } else {
            alert(data.message);
        }
    })
    .catch(error => {
        console.error('Error:', error);
        alert("An error occurred while submitting the form. Please try again.");
    });
};

</script>
</body>
</html>

```

Source code : process_agreement.php

```

<?php
session_start();
include("../connection.php");
include("../functions.php");
$user_data=check_login($con);
// Database connection details
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "rent_management_system";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {

```

```

die("Connection failed: " . $conn->connect_error);
}

$response = array('success' => false, 'message' => 'File upload failed.');

if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $name = $_POST['name'];
    $room_id = $_POST['room_id'];
    $currentTime = time();
    $folderName = "entry" . $room_id . $name;

    $target_dir = "stores/" . $folderName . "/";

    // Ensure the target directory exists
    if (!is_dir($target_dir)) {
        mkdir($target_dir, 0777, true);
    }

    // File paths
    $photo_path = $target_dir . "photo.jpg";
    $sign_path = $target_dir . "sign.jpg";
    $aadhar_path = $target_dir . "aadhar.jpg";

    // Move uploaded files
    if (move_uploaded_file($_FILES["photo"]["tmp_name"], $photo_path) &&
        move_uploaded_file($_FILES["signature"]["tmp_name"], $sign_path) &&
        move_uploaded_file($_FILES["aadhar"]["tmp_name"], $aadhar_path)) {

        // Insert file paths into the database
        $stmt = $conn->prepare("INSERT INTO entries (room_id, photo_path,
sign_path, aadhar_path) VALUES (?, ?, ?, ?)");
        $stmt->bind_param("ssss", $room_id, $photo_path, $sign_path,
$aadhar_path);

        if ($stmt->execute()) {
            $response['success'] = true;
            $response['message'] = 'Files have been uploaded and stored
successfully.';
        }
    }
}

```

```

        }

$stmt->close();

}

$conn->close();
echo json_encode($response);
?>

```

Source code : print_agr.php (Produces the PDF of the Agreement)

```

<?php
session_start();
include("../connection.php");
include("../functions.php");
$user_data = check_login($con);
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Display Details and Files</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 20px;
        }
        .container {
            max-width: 800px;
            margin: 0 auto;
        }
        table {
            width: 100%;

```

```
border-collapse: collapse;
margin-top: 20px;
}

table, th, td {
    border: 1px solid #ccc;
    padding: 8px;
    text-align: left;
}

th {
    background-color: #f2f2f2;
}

.files {
    display: flex;
    justify-content: space-around;
    margin-top: 20px;
}

.files img {
    max-width: 300px;
    height: auto;
    border-radius: 8px;
    margin-bottom: 20px;
}

.form-container {
    margin-top: 20px;
    border: 1px solid #ccc;
    padding: 10px;
    width: fit-content;
}

.form-container input[type=text] {
    width: 100%;
    padding: 8px;
    margin-top: 8px;
    margin-bottom: 16px;
    box-sizing: border-box;
}

.form-container input[type=submit] {
    background-color: #4CAF50;
```

```
        color: white;
        padding: 10px 20px;
        border: none;
        border-radius: 4px;
        cursor: pointer;
        transition: background-color 0.3s;
    }

.form-container input[type=submit]:hover {
    background-color: #45a049;
}

.print-button {
    margin-top: 20px;
    text-align: center;
}

.print-button button {
    padding: 10px 20px;
    font-size: 16px;
    background-color: #4CAF50;
    color: white;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    transition: background-color 0.3s;
}

.print-button button:hover {
    background-color: #45a049;
}

#text-cont pre{
    font-family: Arial, Helvetica, sans-serif;
    font-size: 18px; font-weight: 300;
}

</style>

</head>
<body>

<div class="container">
    <div class="form-container">
        <div id="text-cont">
```

```

<pre>
    <h3>I Ensure to Follow the rules</h3>
    Rule 1: Timely Payments will be Made.
    Rule 2: 15 day Early Notice for Vacating.
    Rule 3: The Details will be given for verificationto the
nearest police station.
    Rule 4: To Ensure Good Conduct with other Tenants.
    Rule 5: To park the vehical , the tenant bears all risk.
    Rule 6: Any illegal work ,landlord is not liable. Will
inform Police as soon as possible.
    Rule 7: No vandalising of the Property / Electric Bill.
    Rule 8: Forbidden to keep other as tenant in the property.

```

```

<h3> Signature of Tenant Stamp</h3> Landlord
</pre>
</div>

<?php if (!isset($_GET['reg_id'])) || (isset($_GET['reg_id'])) &&
!$result1->num_rows > 0 && !$result2->num_rows > 0) : ?>
<h2>Enter Registration ID:</h2>
<form method="GET" action="<?php echo
htmlspecialchars($_SERVER["PHP_SELF"]); ?>">
    <input type="text" name="reg_id" placeholder="Enter
Registration ID" required>
    <input type="submit" value="Fetch Details">
</form>
<?php endif; ?>

<?php
// Database connection details
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "rent_management_system";

// Establish connection
$conn = new mysqli($servername, $username, $password, $dbname);

```

```

// Check connection

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Fetch details from reg_renter table based on registration ID
from form submission

if (isset($_GET['reg_id'])) {
    $reg_id = $_GET['reg_id'];

    $sql1 = "SELECT id, name, age, gender, datereg, aadhar,
mobile, address, roomNo, price FROM reg_renter WHERE id = ?";
    $stmt1 = $conn->prepare($sql1);
    $stmt1->bind_param("s", $reg_id);
    $stmt1->execute();
    $result1 = $stmt1->get_result();

    // Display details from reg_renter table
    if ($result1->num_rows > 0) {
        echo "<h2>Details for Registration ID: $reg_id</h2>";
        echo "<table>";
        echo
"<tr><th>ID</th><th>Name</th><th>Age</th><th>Gender</th><th>Date
Registered</th><th>Aadhar</th><th>Mobile</th><th>Address</th><th>Room
No</th><th>Price</th></tr>";
        while ($row1 = $result1->fetch_assoc()) {
            echo "<tr>";
            echo "<td>" . $row1["id"] . "</td>";
            echo "<td>" . $row1["name"] . "</td>";
            echo "<td>" . $row1["age"] . "</td>";
            echo "<td>" . $row1["gender"] . "</td>";
            echo "<td>" . $row1["datereg"] . "</td>";
            echo "<td>" . $row1["aadhar"] . "</td>";
            echo "<td>" . $row1["mobile"] . "</td>";
            echo "<td>" . $row1["address"] . "</td>";
            echo "<td>" . $row1["roomNo"] . "</td>";
            echo "<td>" . $row1["price"] . "</td>";
        }
    }
}

```

```

        echo "</tr>";
    }
    echo "</table>";
} else {
    echo "No details found for Registration ID:  

$reg_id</p>";
}

$stmt1->close();

// Fetch images from entries table based on ID from form
submission
$sql2 = "SELECT id, room_id, photo_path, sign_path,
aadhar_path, created_at FROM entries WHERE id = ?";
$stmt2 = $conn->prepare($sql2);
$stmt2->bind_param("s", $reg_id);
$stmt2->execute();
$result2 = $stmt2->get_result();

// Display images from entries table
if ($result2->num_rows > 0) {
    echo "<h2>Files for ID: $reg_id</h2>";
    echo "<div class='files'>";
    while ($row2 = $result2->fetch_assoc()) {
        echo "<div><img src='".$row2["photo_path"] . "' alt='Photograph'></div>";
        echo "<div><img src='".$row2["sign_path"] . "' alt='Signature'></div>";
        echo "<div><img src='".$row2["aadhar_path"] . "' alt='Aadhar Card'></div>";
    }
    echo "</div>";
} else {
    echo "<p style='text-align: center; margin-top: 20px;'>No files found for ID: $reg_id</p>";
}

$stmt2->close();
}

```

```

$conn->close();

?>

</div>

<?php if (isset($_GET['reg_id'])) && ($result1->num_rows > 0 || $result2->num_rows > 0) : ?>

<div class="print-button">
    <button onclick="window.print()">Print Page</button>
</div>

<?php endif; ?>
</div>

<a href="../index.php">Go to the Main Content</a>

</body>
</html>

```

Screenshots of the Process

1. Opening the page to start the process of Signing Agreements with the registered tenants.

We click on the

Registration&Agreements

This will lead us to the page , where we will enter the details as we filled in the previous form

We will see the new page as :

Agreement Form

Name:

Room ID:

Upload Photograph:

Choose File No file chosen

Upload Signature:

Choose File No file chosen

Upload Aadhar Card:

Choose File No file chosen

Photo Preview:



Signature Preview:



Aadhar Card Preview:



Submit

[Go to Home Page](#) [Produce the Agreement PDF](#)

We then fill the details as:

Agreement Form

Name:

Mr. Amit Kumar

Room ID:

1

Upload Photograph:

user image.jpg

Upload Signature:

user signature.png

Upload Aadhar Card:

user ac.JPG

Photo Preview:



Signature Preview:

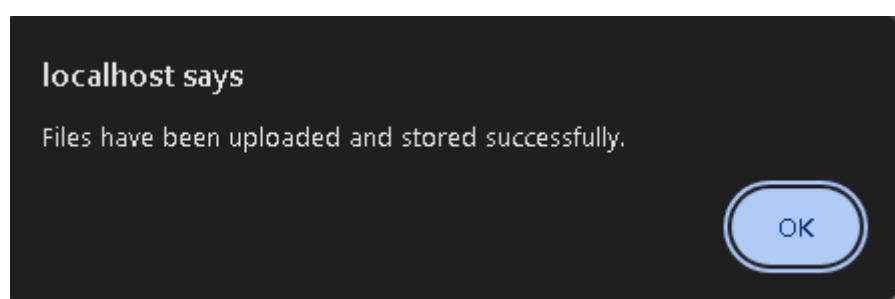
Signature

Aadhar Card Preview:



[Go to Home Page](#) [Produce the Agreement PDF](#)

Then we Click on the submit button , to see the alert that is provided



id	room_id	photo_path	sign_path	aadhar_path	created_at
1	1	stores/entry1Mr.Amit Kumar/photo.jpg	stores/entry1Mr.Amit Kumar/sign.jpg	stores/entry1Mr.Amit Kumar/aadhar.jpg	2024-07-02 16:27:04

Then we click on the below link to initiate making of the pdf. format of the agreement as :

Submit

[Go to Home Page](#) [Produce the Agreement PDF](#)

We click on the produce the agreement pdf link to see the new page as:

I Ensure to Follow the rules

- Rule 1: Timely Payments will be Made.
- Rule 2: 15 day Early Notice for Vacating.
- Rule 3: The Details will be given for verification to the nearest police station.
- Rule 4: To Ensure Good Conduct with other Tenants.
- Rule 5: To park the vehical , the tenant bears all risk.
- Rule 6: Any illegal work ,landlord is not liable. Will inform Police as soon as possible.
- Rule 7: No vandalising of the Property / Electric Bill.
- Rule 8: Forbidden to keep other as tenant in the property.

Signature of Tenant

Landlord Stamp

Enter Registration ID:

Enter Registration ID

Fetch Details

We then enter the registration id to retrive all the details regarding it

Rule 1: Timely Payments will be Made.
 Rule 2: 15 day Early Notice for Vacating.
 Rule 3: The Details will be given for verification to the nearest police station.
 Rule 4: To Ensure Good Conduct with other Tenants.
 Rule 5: To park the vehicle, the tenant bears all risk.
 Rule 6: Any illegal work, landlord is not liable. Will inform Police as soon as possible.
 Rule 7: No vandalising of the Property / Electric Bill.
 Rule 8: Forbidden to keep other as tenant in the property.

Signature of Tenant

Landlord Stamp

Enter Registration ID:

Details for Registration ID: 1

ID	Name	Age	Gender	Date Registered	Aadhar	Mobile	Address	Room No	Price
1	Mr. Amit Kumar	32	Male	2024-07-02	23232323	9090909090	gaga, blitar	1	2300.00

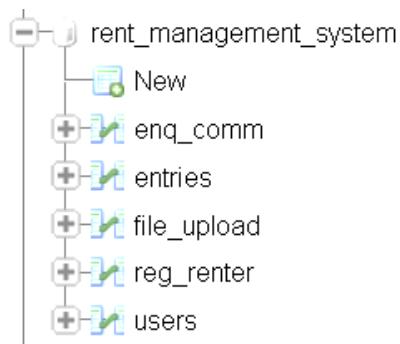
Files for ID: 1



The details are dynamically brought to the main page , the button will initiate to print the page to pdf , for swift sharing and printing as the hard copy.

The database Structure :

This Process utilises 2 tables in the database: rent_management_system.



The table named : `reg_renter` is going to store the details of the name , and the room selected for the accomodation.

The table named : `entries` will hold the images related to the signing of the agreement.

The table structure is as follows :

Table : `reg_renter`

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	<code>id</code>	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	<code>name</code>	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	<code>age</code>	int(11)			No	None			Change Drop More
4	<code>gender</code>	enum('Male','Female','Other')	utf8mb4_general_ci		No	None			Change Drop More
5	<code>datereg</code>	date			No	None			Change Drop More
6	<code>aadhar</code>	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
7	<code>mobile</code>	varchar(15)	utf8mb4_general_ci		No	None			Change Drop More
8	<code>address</code>	text	utf8mb4_general_ci		No	None			Change Drop More
9	<code>roomNo</code>	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
10	<code>price</code>	decimal(10,2)			No	None			Change Drop More

Table : `entries`

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	<code>id</code>	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	<code>room_id</code>	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
3	<code>photo_path</code>	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
4	<code>sign_path</code>	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
5	<code>aadhar_path</code>	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
6	<code>created_at</code>	timestamp			No	current_timestamp()			Change Drop More

There is also use of the file structures , Since there is no possible way to store the images in the database , while their location can always be stored. Thus its is a routine task to make the backup of the files that were utilised inorder to produce the agreement and also ensure that these parts don't get into security breeches.

Chapter : 8

View Renters

- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

Viewing Renters

The Rent Management System will show the registered renters , whom the agreement was signed with. Their Data is going to be displayed on the screen

Folder Structure

Rent Management System

```
| -> rentr_list  
|   | -> rentr_list.php (file that retrieves data to show on the page)
```

Source Code (rentr_list.php)

```
<?php  
  
session_start();  
  
include("../connection.php");  
  
include("../functions.php");  
  
$user_data=check_login($con);  
  
// Database connection  
  
$servername = "localhost";  
  
$username = "root";  
  
$password = "";  
  
$dbname = "rent_management_system";  
  
  
// Create connection  
  
$conn = new mysqli($servername, $username, $password, $dbname);  
  
  
// Check connection  
  
if ($conn->connect_error) {  
    die("Connection failed: " . $conn->connect_error);  
}  
  
  
// Fetch data from the reg_renter table  
  
$sql = "SELECT id, name, age, gender, datereg, aadhar, mobile, address,  
roomNo, price FROM reg_renter";  
  
$result = $conn->query($sql);  
  
?>  
  
<!DOCTYPE html>
```

```
<html>
<head>
    <title>Renter List</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f2f2f2;
            margin: 0;
            padding: 0;
        }
        .container {
            width: 80%;
            margin: auto;
            overflow: hidden;
        }
        .print-button {
            background-color: #008CBA;
            color: white;
            border: none;
            padding: 10px 20px;
            cursor: pointer;
            margin-bottom: 20px;
        }
        .print-button:hover {
            background-color: #007B9A;
        }
        table {
            width: 100%;
            margin: 20px 0;
            border-collapse: collapse;
        }
        table, th, td {
            border: 1px solid #ddd;
        }
        th, td {
            padding: 10px;
            text-align: left;
        }
    </style>
</head>
<body>
    <div class="container">
        <button class="print-button">Print</button>
        <table>
            <thead>
                <tr>
                    <th>Name</th>
                    <th>Address</th>
                    <th>Phone Number</th>
                </tr>
            </thead>
            <tbody>
                <tr>
                    <td>John Doe</td>
                    <td>123 Main Street</td>
                    <td>(555) 123-4567</td>
                </tr>
                <tr>
                    <td>Jane Smith</td>
                    <td>456 Elm Street</td>
                    <td>(555) 987-6543</td>
                </tr>
                <tr>
                    <td>Bob Johnson</td>
                    <td>789 Oak Street</td>
                    <td>(555) 543-2109</td>
                </tr>
            </tbody>
        </table>
    </div>
</body>
</html>
```

```
}

th {
    background-color: #f4b41a;
    color: white;
}

tr:nth-child(even) {
    background-color: #f2f2f2;
}

.details-button {
    background-color: #4CAF50;
    color: white;
    border: none;
    padding: 10px 20px;
    cursor: pointer;
}

.details-button:hover {
    background-color: #45a049;
}

.popup-panel {
    display: none;
    position: fixed;
    top: 20%;
    left: 50%;
    transform: translate(-50%, -50%);
    background-color: white;
    padding: 20px;
    border: 2px solid #888;
    box-shadow: 0px 0px 10px 0px #000;
    z-index: 1001;
}

.close-button {
    background-color: red;
    color: white;
    border: none;
    padding: 5px 10px;
    cursor: pointer;
    float: right;
}
```

```

        }

    </style>

</head>

<body>

    <div class="container">

        <h1>Renter List</h1>

        <button class="print-button" onclick="printRecords()">Print All Records</button>

        <table>

            <tr>
                <th>Name</th>
                <th>Room No.</th>
                <th>Mobile</th>
                <th>Price</th>
                <th>Action</th>
            </tr>

            <?php

                if ($result->num_rows > 0) {
                    while($row = $result->fetch_assoc()) {
                        echo "<tr>";
                        echo "<td>" . $row["name"] . "</td>";
                        echo "<td>" . $row["roomNo"] . "</td>";
                        echo "<td>" . $row["mobile"] . "</td>";
                        echo "<td>" . $row["price"] . "</td>";
                        echo "<td><button class='details-button' onclick='showDetails(" . json_encode($row) . ")'>View Details</button></td>";
                        echo "</tr>";
                    }
                } else {
                    echo "<tr><td colspan='5'>No records found</td></tr>";
                }
            ?>
        </table>
    </div>

    <div class="popup-panel" id="popupPanel">
```

```

        <button class="close-button"
        onclick="closeDetails()">Close</button>
        <div id="detailsContent"></div>
    </div>

<script>
    function showDetails(details) {
        var popupPanel = document.getElementById("popupPanel");
        var detailsContent = document.getElementById("detailsContent");

        detailsContent.innerHTML =
            <p><strong>Room No:</strong> ${details.roomNo}</p>
            <p><strong>ID:</strong> ${details.id}</p>
            <p><strong>Address:</strong> ${details.address}</p>
        `;

        popupPanel.style.display = "block";
    }

    function closeDetails() {
        var popupPanel = document.getElementById("popupPanel");
        popupPanel.style.display = "none";
    }

    function printRecords() {
        window.print();
    }
</script>
<h2><a href="../index.php">Go to Main Page</a></h2>
</body>
</html>

<?php
$conn->close();
?>

```

Screenshots of the page

We go to the main dashboard of the Rent Management System , and we click on the card that features to list the renters :



We click on "click to proceed" below and then it takes us to the page :

A screenshot of a web browser showing a table of renter records. The table has columns for Name, Room No., Mobile, Price, and Action. Two rows are listed: Mr. Amit Kumar (Room 1, Mobile 9090909090, Price 2300.00) and Mrs. Anita Devi (Room 2, Mobile 1234512345, Price 3200.00). Each row has a "View Details" button in the Action column. A "PrintAll Records" button is at the top left of the table.

Incase , you require to see how the registration is done on the System , as we can see the record : Mrs. Anita Devi was registered with the process that is illustrated by the video as : by the link that is contained inside the qr code . Kindly adjust its time to : 2:28 inorder to see how Mrs. Anita Devi Record was registered in the Rent Management System



Chapter : 9

Create Bills

- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

Creating Bills

The Rent Management System will provide the feature to the landlord to make the bills for the Renters of the property . The simple interface provides the ability to create bill , save the details to the database and moreover making the ability to make the pdf for the bill that can be saved for future references as well as a medium of communication that conveys the Bill information to the renters.

Rent Management System

```
| -> make_bill  
|   | -> make_bill.php (file with form to get the details)  
|   | -> bill_logic.php (file that transfers the details to the database)
```

Source Code (make_bill.php)

```
<?php  
session_start();  
include("../connection.php");  
include("../functions.php");  
$user_data=check_login($con);  
?  
  
<!DOCTYPE html>  
<html lang="en">  
<head>  
    <meta charset="UTF-8">  
    <meta name="viewport" content="width=device-width, initial-scale=1.0">  
    <title>Room Rent Bill</title>  
    <style>  
        body {  
            font-family: Arial, sans-serif;  
            background-color: #f2f2f2;  
            margin: 0;  
            padding: 0;  
        }  
    </style>
```

```
.container {
    width: 80%;
    margin: auto;
    overflow: hidden;
}

h1 {
    text-align: center;
    background-color: #4CAF50;
    color: white;
    padding: 10px;
}

.form-group {
    margin-bottom: 15px;
}

label {
    display: block;
    margin-bottom: 5px;
}

input, select, button {
    width: 100%;
    padding: 10px;
    margin-bottom: 10px;
}

table {
    width: 100%;
    margin-top: 20px;
    border-collapse: collapse;
}

table, th, td {
    border: 1px solid #ddd;
}

th, td {
    padding: 10px;
    text-align: left;
}

.print-button {
    background-color: #008CBA;
```

```

        color: white;
        border: none;
        padding: 10px 20px;
        cursor: pointer;
    }

.print-button:hover {
    background-color: #007B9A;
}

.add-row {
    background-color: #4CAF50;
    color: white;
    border: none;
    padding: 5px 10px;
    cursor: pointer;
}

.add-row:hover {
    background-color: #45a049;
}

.message {
    margin-top: 10px;
    font-style: italic;
}

</style>

</head>
<body>

<div class="container">
    <h1>Room Rent Bill</h1>
    <form method="post" action="bill_logic.php">
        <div class="form-group">
            <label for="renter">Tenant Name:</label>
            <select id="renter" name="renter" onchange="fetchRoomNo ()">
                <option value="">Select Tenant</option>
                <?php
                    // Database connection
                    $servername = "localhost";
                    $username = "root";
                    $password = "";

```

```

$dbname = "rent_management_system";

// Create connection
$conn = new mysqli($servername, $username, $password,
$dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Fetch data from the reg_renter table
$sql = "SELECT id, name, roomNo FROM reg_renter";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    while($row = $result->fetch_assoc()) {
        echo "<option value='" . $row["id"] . "' data-
roomNo=" . $row["roomNo"] . "'>" . $row["name"] . "</option>";
    }
}

$conn->close();
?>
</select>
</div>
<div class="form-group">
    <label>Room No: <span id="roomNoDisplay"></span></label>
</div>
<div class="form-group">
    <label for="month">Month:</label>
    <select id="month" name="month">
        <?php
            for ($i = 1; $i <= 12; $i++) {
                $month = date("F", mktime(0, 0, 0, $i, 10));
                echo "<option value='$month'>$month</option>";
            }
        </?php
    
```

```

    ?>

```

```

        </select>

```

```

    </div>

```

```

<div class="form-group">

```

```

    <label for="year">Year:</label>

```

```

    <select id="year" name="year">

```

```

        <?php

```

```

            $currentYear = date("Y");

```

```

            for ($i = $currentYear; $i <= $currentYear + 5; $i++) {

```

```

                echo "<option value='$i'>$i</option>";

```

```

            }

```

```

        ?>

```

```

    </select>

```

```

</div>

```

```

<div class="form-group">

```

```

    <label for="due_date">Rent To be cleared by date:</label>

```

```

    <input type="date" id="due_date" name="due_date" required>

```

```

</div>

```

```

<div class="form-group">

```

```

    <label for="room_rent">Room Rent of the Renter:</label>

```

```

    <input type="text" id="room_rent" name="room_rent" oninput="calculateTotal()" required>

```

```

</div>

```

```

<div class="form-group">

```

```

    <label id="electric_meter">Electric Bill of room: <span id="electricRoomNo"></span></label>

```

```

    <input type="text" id="units_used" name="units_used" oninput="calculateElectricBill()" required>

```

```

</div>

```

```

<div class="form-group">

```

```

    <label for="electric_bill">Electric Bill:</label>

```

```

    <input type="text" id="electric_bill" name="electric_bill" readonly required>

```

```

</div>

```

```

<div class="form-group">

```

```

    <label for="advance_paid">Amount paid in advance:</label>

```

```

    <input type="text" id="advance_paid" name="advance_paid" oninput="calculateTotal()" required>

```

```

</div>

<div class="form-group">
    <label for="amount_dues">Amount dues:</label>
    <input type="text" id="amount_dues" name="amount_dues"
    oninput="calculateTotal()" required>
</div>

<table id="miscTable">
    <thead>
        <tr>
            <th>Serial No</th>
            <th>Miscellaneous charges for</th>
            <th>Amount</th>
            <th><button type="button" class="add-row"
            onclick="addRow()">+</button></th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td>1</td>
            <td><input type="text"
            name="miscellaneous[0][description]" required></td>
            <td><input type="text"
            name="miscellaneous[0][amount]" oninput="calculateTotal()" required></td>
            <td></td>
        </tr>
    </tbody>
</table>

<div class="form-group">
    <label for="total_amount">Total Amount Payable:</label>
    <input type="text" id="total_amount" name="total_amount"
    readonly required>
</div>

<div class="form-group">
    <p class="message">* Room agreement Advance is never to be
    deducted.</p>
</div>

<button type="submit">Save Bill</button>
<button type="button" class="print-button"
    onclick="printBill()">Print Bill</button>

```

```

        </form>
    </div>

<script>

    function fetchRoomNo() {
        var renterSelect = document.getElementById("renter");
        var selectedOption = renterSelect.options[renterSelect.selectedIndex];
        var roomNo = selectedOption.getAttribute("data-roomNo");
        document.getElementById("roomNoDisplay").innerText = roomNo;
        document.getElementById("electricRoomNo").innerText = roomNo;
    }

    function calculateElectricBill() {
        var unitsUsed = document.getElementById("units_used").value;
        var electricBill = unitsUsed * 9; // Adjust this calculation
        based on your billing rate per unit
        document.getElementById("electric_bill").value = "₹" + electricBill.toFixed(2);
        calculateTotal();
    }

    function addRow() {
        var table = document.getElementById("miscTable").getElementsByTagName('tbody')[0];
        var rowCount = table.rows.length;
        var row = table.insertRow(rowCount);

        var cell1 = row.insertCell(0);
        var cell2 = row.insertCell(1);
        var cell3 = row.insertCell(2);
        var cell4 = row.insertCell(3);

        cell1.innerHTML = rowCount + 1;
        cell2.innerHTML = '<input type="text" name="miscellaneous[' + rowCount + '][description]" required>';
        cell3.innerHTML = '<input type="text" name="miscellaneous[' + rowCount + '][amount]" oninput="calculateTotal()" required>';
    }
}

```

```

        cell4.innerHTML = '';
    }

    function calculateTotal() {
        var roomRent = parseFloat(document.getElementById("room_rent").value) || 0;
        var electricBill = parseFloat(document.getElementById("electric_bill").value.substring(1)) || 0;
        var advancePaid = parseFloat(document.getElementById("advance_paid").value) || 0;
        var amountDues = parseFloat(document.getElementById("amount_dues").value) || 0;

        var miscTable = document.getElementById("miscTable").getElementsByTagName('tbody')[0];
        var rows = miscTable.rows;
        var miscTotal = 0;
        for (var i = 0; i < rows.length; i++) {
            var amount = parseFloat(rows[i].cells[2].getElementsByName('input')[0].value) || 0;
            miscTotal += amount;
        }

        var totalAmount = roomRent + electricBill + amountDues + miscTotal - advancePaid;
        document.getElementById("total_amount").value = totalAmount.toFixed(2);
    }

    function printBill() {
        window.print();
    }

```

</script>

<h2>Go to Main Page</h2>

</body>

</html>

Source Code (bill_logic.php)

```
<?php

session_start();
include("../connection.php");
include("../functions.php");
$user_data=check_login($con);
// Database connection
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "rent_management_system";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Process form data
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $renter_id = $_POST['renter'];
    $month = $_POST['month'];
    $year = $_POST['year'];
    $due_date = $_POST['due_date'];
    $room_rent = $_POST['room_rent'];
    $units_used = $_POST['units_used'];
    $electric_bill = $_POST['electric_bill'];
    $advance_paid = $_POST['advance_paid'];
    $amount_dues = $_POST['amount_dues'];
    $miscellaneous = $_POST['miscellaneous'];
    $total_amount = $_POST['total_amount'];
}
```

```

// Insert into bill_details table

$sql = "INSERT INTO bill_details (renter_id, month, year, due_date,
room_rent, units_used, electric_bill, advance_paid, amount_dues,
miscellaneous, total_amount)

VALUES      ('$renter_id', '$month', '$year', '$due_date',
'$room_rent', '$units_used', '$electric_bill', '$advance_paid',
'$amount_dues', '" . json_encode($miscellaneous) . "', '$total_amount')";

if ($conn->query($sql) === TRUE) {
    echo "Bill details saved successfully.";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

}

$conn->close();
?>

```

Screenshots of the Page (make_bill.php)

We go to the main dashboard of the Rent Management System . In the Top of the navigation bar , we click the Create Bills feature :



Clicking on that will lead us to the page that looks like:

Room Rent Bill

Tenant Name:	<input type="text" value="Select Tenant"/>
Room No:	
Month:	<input type="text" value="January"/>
Year:	<input type="text" value="2024"/>
Rent To be cleared by date:	<input type="text" value="mm/dd/yyyy"/>
Room Rent of the Renter:	<input type="text"/>
Electric Bill of room:	<input type="text"/>
Electric Bill:	<input type="text"/>

Amount paid in advance:	<input type="text"/>		
Amount dues:	<input type="text"/>		
Serial No	Miscellaneous charges for	Amount	[+]
1	<input type="text"/>	<input type="text"/>	
Total Amount Payable:	<input type="text"/>		
<small>* Room agreement Advance is never to be deducted.</small>			
<input type="button" value="Save Bill"/> <input type="button" value="Print Bill"/>			

[Go to Main Page](#)

The above form is currently empty we fill the details in it . It is retriving the details from the reg_renter details , which means that it will make you produce the bills for those who are registered.

The form is very versatile in nature as the details can be flexibly be added or removed as per the requirement.

The Create Bill feature , produced the bill in pdf format also stores the details to the table : bill_details in the database : rent_management system.

The procedure of filling the details in order to make the bill is :

Step 1 : we select the Renter by the name and fill the details

Room Rent Bill

Tenant Name:	<input type="text" value="Mr. Amit Kumar"/>
Room No:	<input type="text"/>
Month:	<input type="text" value="July"/>
Year:	<input type="text" value="2024"/>
Rent To be cleared by date:	<input type="text" value="07/15/2024"/>
Room Rent of the Renter:	<input type="text" value="2300"/>
Electric Bill of room: 1	<input type="text" value="11"/>
Electric Bill:	<input type="text" value="₹99.00"/>

The name is Mr. Amit Kumar , his bill is for the july 2024 , he will pay 2300 Rs. as rent amount , 11 units of electricity cost him 99 Rs. He will pay the rent till the date of 15th July 2024.

Amount paid in advance:	200	
Amount dues:	0	
Serial No	Miscellaneous charges for	Amount
1	Car Parking fee	800
Total Amount Payable:	2999.00	
<small>* Room agreement Advance is never to be deducted.</small>		
<input type="button" value="Save Bill"/> <input type="button" value="Print Bill"/>		

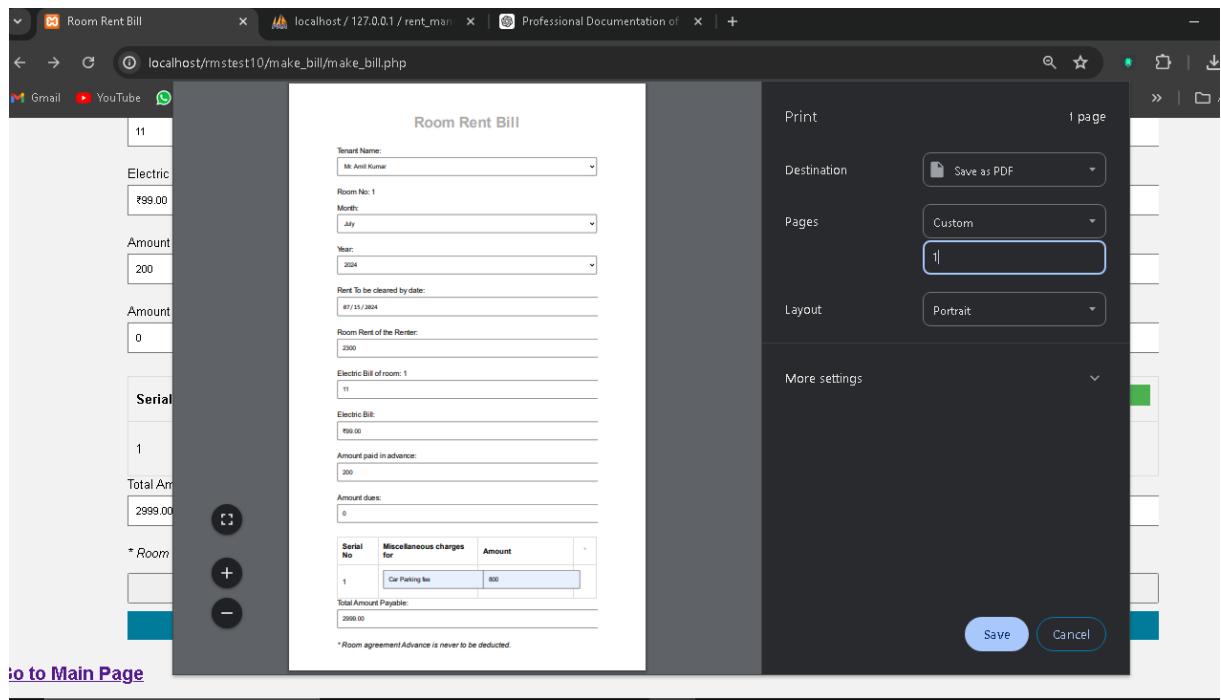
[Go to Main Page](#)

We can see that he has paid 200 Rs in advance , so that will be deducted. He has 0 Rs. in dues amount. Currently a car parking fee is charged against him for Rs. 800. , We click on the Save Bill to store the details to the database.

Bill details saved successfully.

We get the response that all the necessary details were stored in the database.

We can go back <- to the back page in the browser to see the option to print the bill that helps us to store it in the form of pdf in the suitable location on the disk of computer . Also it can be shared to the renter.



Room Rent Bill

Tenant Name:

Mr. Amit Kumar

Room No: 1

Month:

July

Year:

2024

Rent To be cleared by date:

07 / 15 / 2024

Room Rent of the Renter:

2300

Electric Bill of room: 1

11

Electric Bill:

₹99.00

Amount paid in advance:

200

Amount dues:

0

Serial No	Miscellaneous charges for	Amount	+
1	Car Parking fee	800	

Total Amount Payable:

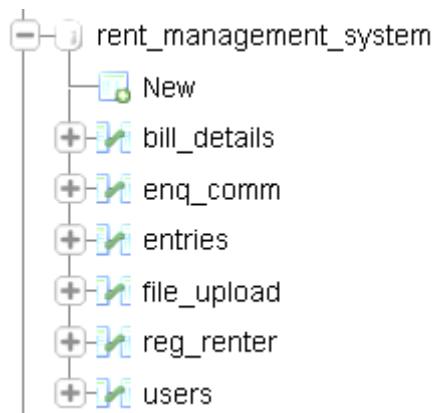
2999.00

* Room agreement Advance is never to be deducted.

This is format of pdf format bill produced by the Rent Management System

Database Structure

We make the tables are in the database like :



The structure of bill Details is :

Server: 127.0.0.1 > Database: rent_management_system > Table: bill_details

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None	AUTO_INCREMENT	Change Drop More	
2	renter_id	int(11)			No	None		Change Drop More	
3	month	varchar(20)	utf8mb4_general_ci		No	None		Change Drop More	
4	year	int(11)			No	None		Change Drop More	
5	due_date	date			No	None		Change Drop More	
6	room_rent	decimal(10,2)			No	None		Change Drop More	
7	units_used	int(11)			No	None		Change Drop More	
8	electric_bill	decimal(10,2)			No	None		Change Drop More	
9	advance_paid	decimal(10,2)			No	None		Change Drop More	
10	amount_dues	decimal(10,2)			No	None		Change Drop More	
11	miscellaneous	text	utf8mb4_general_ci		No	None		Change Drop More	
12	total_amount	decimal(10,2)			No	None		Change Drop More	
13	created_at	timestamp			No	current_timestamp()		Change Drop More	

The details are stored in the table as below , after we had created the bill for Mr. Amit Kumar:

id	renter_id	month	year	due_date	room_rent	units_used	electric_bill	advance_paid	amount_dues	miscellaneous	total_amount	created_at
1	1	July	2024	2024-07-15	2300.00	11	99.00	200.00	0.00	[{"description": "Car Parking fee", "amount": "800"}]	2999.00	2024-07-04 15:38:54

Chapter : 10

Search Bar Functionality

- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

Search Bar

The Rent Management System will provide the feature to the landlord to swiftly search the tenant records and details by entering either the room id or the name of the tenant.

Rent Management System

```
| -> index.php  
| -> searchAction.php
```

Source Code (index.php)

The Index page (index.php) is the place where we can see the search box.

The search box is very handy tool when it comes to search the records of the statement , incase the landlord is quite busy to search in the system in more detail.

The source code for the search box is below :

```
<form action="searchAction.php" method="GET" id="form1" style="width:75px; height:10px; padding-top:none;">  
    <input type="text" class="nav-search-input" name="query" placeholder="Search @ Rent Management System" style="width:310px; height:16px;">  
    <button type="submit" style="position:relative; top:-37px; left:549px;">  
          
    </button>  
</form>
```

The form in the index.php as a search bar will transfer the values to the searchAction.php that contains the logic to fetch the details.

Source Code (searchAction.php)

```
<?php  
session_start();  
include("connection.php");  
include("functions.php");  
$user_data=check_login($con);  
// Database connection  
$servername = "localhost";  
$username = "root";
```

```

$password = "";
$dbname = "rent_management_system";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Get the search query
$searchQuery = isset($_GET['query']) ? $_GET['query'] : '';

// Sanitize the input
$searchQuery = $conn->real_escape_string($searchQuery);

// SQL query to search by roomNo or partial name match
$sql = "SELECT * FROM reg_renter WHERE roomNo = '$searchQuery' OR name LIKE
'%$searchQuery%';

$result = $conn->query($sql);
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Search Results</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f2f2f2;
            padding: 20px;
        }
        table {

```

```

width: 100%;

border-collapse: collapse;
margin-top: 20px;
}

table, th, td {
border: 1px solid #ddd;
}

th, td {
padding: 10px;
text-align: left;
}

th {
background-color: #007bff;
color: white;
}

</style>
</head>
<body>

<h1>Search Results</h1>

<?php if ($result->num_rows > 0): ?>

<table>

<tr>
<th>Serial No</th>
<th>Date Registered</th>
<th>Name</th>
<th>Room No</th>
<th>Price</th>
</tr>

<?php
$serialNo = 1;
while($row = $result->fetch_assoc()):
?>

<tr>
<td><?php echo $serialNo++; ?></td>
<td><?php echo $row['datereg']; ?></td>
<td><?php echo $row['name']; ?></td>
<td><?php echo $row['roomNo']; ?></td>

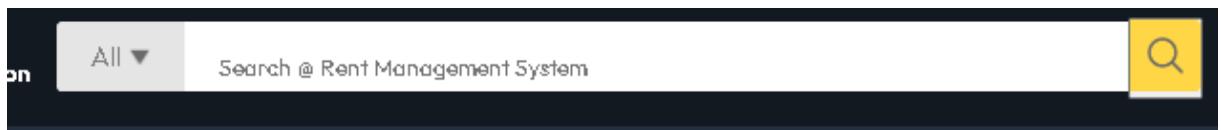
```

```

<td><?php echo $row['price']; ?></td>
</tr>
<?php endwhile; ?>
</table>
<?php else: ?>
<p>No results found for "<?php echo htmlspecialchars($searchQuery); ?>"</p>
<?php endif; ?>
<?php $conn->close(); ?>
</body>
</html>

```

Screenshots



Suppose we enter Amit



Then click on the search button.

Search Results				
Serial No	Date Registered	Name	Room No	Price
1	2024-07-02	Mr. Amit Kumar	1	2300.00

The search results are there .

Suppose we enter the renter_id/ room Number then we will be also be able to see the results



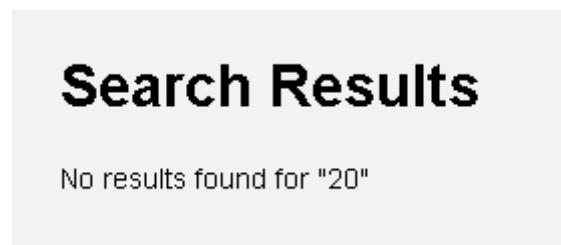
The output was:

Serial No	Date Registered	Name	Room No	Price
1	2024-07-03	Mrs. Anita Devi	2	3200.00

This Search bar also provides the error output. If we enter the room id = 20 that currently does not exists , it will display us the error as :



The output we get is :



Chapter : 11

Other Functions

- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

Other Functions

The Rent Management System will provide some necessary features , like the details that are not used much frequent but serve to be a necessary component of the Software.

These pages are : the IT Rules followed , Developer Contacts , Landlord Details , Reset Password and the Logout page. These are the supplementary components.

The Reset Password is the Ch 4 . of the Source code and the documentation.

Rent Management System

| -> other_functions

| -> othr_func_resc (folder that is containing the image files)

| -> dev_details.php (developer details)

| -> it_rules.php (IT Rules kept in mind)

| -> ll_regist.php (Landlord's Business Registration Details)

| -> logout.php (page to log out of the system)

Source Code (dev_details.php)

```
<?php
session_start();
include("../connection.php");
include("../functions.php");
$user_data=check_login($con);
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Developer Details</title>
    <style>
        body {
            font-family: Arial, sans-serif;
```

```
margin: 0;
padding: 0;
background-color: #f2f2f2;
display: flex;
justify-content: center;
align-items: center;
height: 100vh;
}

.container {
background-color: #fff;
padding: 20px;
border-radius: 10px;
box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
max-width: 600px;
text-align: center;
}

.container img {
border-radius: 10px;
}

.details {
margin-top: 20px;
}

.details h1 {
margin: 0;
font-size: 24px;
color: #333;
}

.details p {
margin: 5px 0;
font-size: 18px;
color: #555;
}

.print-button {
margin-top: 20px;
padding: 10px 20px;
font-size: 16px;
color: #fff;
```

```

        background-color: #007bff;
        border: none;
        border-radius: 5px;
        cursor: pointer;
    }
    .print-button:hover {
        background-color: #0056b3;
    }
#idc{height:315px;}
</style>
</head>
<body>
<div class="container">
    
    <div class="details">
        <h1>Aayush Sahay</h1>
        <p>Student of: Indian Institute of Business Management, Patna</p>
        <p>Course: BCA</p>
        <p>Roll No.: AKU/31</p>
        <p>Session: 2021-2024</p>
        <p>Enrollment No.: 21303334049</p>
    </div>
    <br>
    <button class="print-button" onclick="window.print()">Print Details</button>
</div>
</body>
</html>

```

Source Code (it_rules.php)

```

<?php
session_start();
include("../connection.php");
include("../functions.php");
$user_data=check_login($con);
?>

```

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>IT Rules</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            line-height: 1.6;
            margin: 0;
            padding: 0;
            background-color: #f4f4f4;
            color: #333;
        }
        .container {
            width: 80%;
            margin: auto;
            overflow: hidden;
        }
        header {
            background: #50b3a2;
            color: #fff;
            padding-top: 30px;
            min-height: 70px;
            border-bottom: #e8491d 3px solid;
        }
        header a {
            color: #fff;
            text-decoration: none;
            text-transform: uppercase;
            font-size: 16px;
        }
        header ul {
            padding: 0;
            list-style: none;
        }
    </style>

```

```
}

header li {
    float: right;
    display: inline;
    padding: 0 20px 0 20px;
}

header .highlight, header .current a {
    color: #e8491d;
    font-weight: bold;
}

header #branding {
    float: left;
}

header #branding h1 {
    margin: 0;
}

.button {
    height: 40px;
    background: #50b3a2;
    border: none;
    padding: 0 20px;
    color: #fff;
    text-transform: uppercase;
    font-size: 18px;
    cursor: pointer;
    border-radius: 5px;
}

.content {
    padding: 20px;
    background: #fff;
    border-radius: 5px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    margin-top: 20px;
}

.content h2 {
    border-bottom: 2px solid #e8491d;
    padding-bottom: 10px;
```

```

        }

    .content ul {
        padding: 0;
        list-style: none;
    }

    .content li {
        margin-bottom: 10px;
        padding-left: 20px;
        position: relative;
    }

    .content li:before {
        content: "✓";
        position: absolute;
        left: 0;
        color: #50b3a2;
    }

</style>

</head>

<body>

    <header>

        <div class="container">
            <div id="branding">
                <h1>IT Rules & Best Practices</h1>
            </div>
        </div>

    </header>

    <div class="container">
        <div class="content">
            <h2>Security Measures</h2>
            <ul>
                <li>Data Validation and Sanitization</li>
                <li>Authentication and Authorization</li>
                <li>Secure Password Storage</li>
                <li>Session Management</li>
                <li>Encryption</li>
                <li>Prevent Cross-Site Scripting (XSS)</li>
            </ul>
        </div>
    </div>

```

```
</ul>

<h2>Data Integrity</h2>
<ul>
    <li>Database Normalization</li>
    <li>Transactions</li>
</ul>

<h2>Performance Optimization</h2>
<ul>
    <li>Efficient Queries</li>
    <li>Caching</li>
    <li>Pagination</li>
</ul>

<h2>Error Handling</h2>
<ul>
    <li>Logging</li>
    <li>User-Friendly Messages</li>
</ul>

<h2>Scalability</h2>
<ul>
    <li>Modular Code</li>
    <li>Database Design</li>
</ul>

<h2>Compliance and Data Protection</h2>
<ul>
    <li>GDPR Compliance</li>
    <li>Audit Trails</li>
</ul>

<h2>Backup and Recovery</h2>
<ul>
    <li>Regular Backups</li>
    <li>Disaster Recovery Plan</li>
```

```

        </ul>

        <h2>Usability</h2>
        <ul>
            <li>User Interface Design</li>
            <li>Accessibility</li>
        </ul>

        <h2>Documentation and Testing</h2>
        <ul>
            <li>Comprehensive Documentation</li>
            <li>Testing</li>
        </ul>

        <h2>Compliance with Coding Standards</h2>
        <ul>
            <li>Coding Standards</li>
        </ul>

        <h2>Version Control</h2>
        <ul>
            <li>Source Control</li>
        </ul>
    </div>
    <button class="button" onclick="printPage()">Go Through Our IT Considerations</button>
</div>

<script>
    function printPage() {
        window.print();
    }
</script>
</body>
</html>

```

Source Code (ll_regist.php)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Company Registration Details</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f2f2f2;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
        }
        .container {
            background-color: #fff;
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            max-width: 800px;
            text-align: center;
        }
        .container img {
            max-width: 100%;
            height: auto;
            border-radius: 10px;
            margin-bottom: 20px;
        }
        .details {
            margin-top: 20px;
        }
        .details h1 {
```

```

        margin: 0;
        font-size: 28px;
        color: #333;
    }

.details p {
    margin: 5px 0;
    font-size: 18px;
    color: #555;
}

.print-button {
    margin-top: 20px;
    padding: 10px 20px;
    font-size: 16px;
    color: #fff;
    background-color: #007bff;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}

.print-button:hover {
    background-color: #0056b3;
}

</style>

</head>
<body>

<div class="container">
    <h1>Company Registration Details</h1>
    
    <div class="details">
        <p><strong>Company Name:</strong> ABC Apartments and Housings PVT LTD</p>
        <p><strong>Owner:</strong> Mr. ABC Kumar</p>
        <p><strong>Registered In:</strong> Bihar, India</p>
        <p><strong>Year of Registration:</strong> 2008</p>
        <p><strong>Trademark Purchased:</strong> 2010</p>
    </div>
    

```

```

        <button class="print-button" onclick="window.print()">Print Details</button>
    </div>
</body>
</html>

```

Source Code (logout.php)

```

<?php

session_start();

if(isset($_SESSION['user_id'])){

    unset($_SESSION['user_id']);

}

header("Location: login.php?logout=success");

die;

```

Screenshots



We can click to see the IT rules considered to prepare the software solution.

The screenshot displays a section titled "IT Rules & Best Practices" with a teal header. Below the header, there are three main sections: "Security Measures", "Data Integrity", and "Performance Optimization", each with a list of checked items.

- Security Measures**
 - ✓ Data Validation and Sanitization
 - ✓ Authentication and Authorization
 - ✓ Secure Password Storage
 - ✓ Session Management
 - ✓ Encryption
 - ✓ Prevent Cross-Site Scripting (XSS)
- Data Integrity**
 - ✓ Database Normalization
 - ✓ Transactions
- Performance Optimization**
 - ✓ Efficient Queries
 - ✓ Caching
 - ✓ Pagination

Next , we can see the details of the Developer making the system , the Landlord may contact the developers to fix confusion , do the mainatainence of the software , or to add more functionality in the future to provide the convineance to the landlord.



आर्यमट्ट ज्ञान विश्वविद्यालय
ARYABHATTA KNOWLEDGE UNIVERSITY

Aayush Sahay

Student of: Indian Institute of Business Management, Patna

Course: BCA

Roll No.: AKU/31

Session: 2021-2024

Enrollment No.: 21303334049



[Print Details](#)

We can also set up the Business Registration of the Landlord , inorder to make the software aligned to legally work with the business of his Room Rental services:

Company Registration Details

ABC APARTMENTS & HOUSING PVT LTD

Company Name: ABC Apartments and Housings PVT LTD

Owner: Mr. ABC Kumar

Registered In: Bihar, India

Year of Registration: 2008

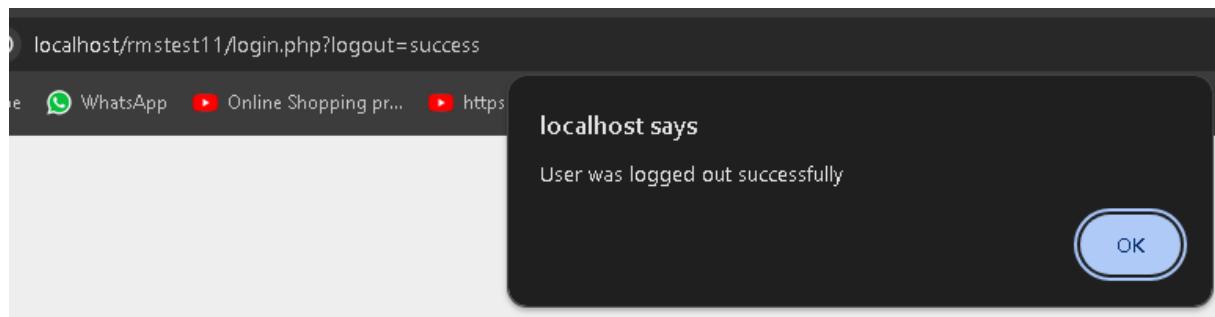
Trademark Purchased: 2010



[Print Details](#)

Finally we can logout of the Rent Management System , once we have done our work.

The logout page , signs out the user out of the session and redirecting the user to the login page. This page is also efficient as it can be accessed along most of the main pages in the system.



Chapter : 12

Payment System

- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

Online Payments Interface

The Rent Management System will provide online payments interface

Rent Management System

```
| -> pay_online
|   | -> assets_ptm (folder that is containing the image files for the app)
|   | -> ptm.php (Main payments page)
|   | -> ptmAction.php (handler of the transactions )
|   | -> pay_records.php (Landlord's interface to manage cash transactions )
|   | -> delete_record.php (delete details of specific records (for correction))
| -> see_records
|   | -> see_records.php (page to see the transaction details of all kind)
```

The above files are responsible for carrying out the online / offline transactions of the Business. The source code of the respective files are as follows :

Source code (ptm.php)

```
<?php
session_start();
include("../connection.php");
include("../functions.php");
$user_data=check_login($con);
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Paytm Payment Gateway Clone</title>
    <style>
        * { margin: 0; padding: 0; box-sizing: border-box; }
        body { font-family: Arial, sans-serif; }
```

```

        header { display: flex; justify-content: center; align-items: center; padding: 20px; background-color: #002970; }

        .logo img { height: 40px; }

        nav ul { display: flex; list-style: none; }

        nav ul li { margin-left: 20px; }

        nav ul li a { color: #fff; text-decoration: none; font-weight: bold; }

        .btn-primary { background-color: #00baf2; padding: 10px 20px; border-radius: 5px; }

        .btn-secondary { background-color: #004c97; padding: 10px 20px; border-radius: 5px; }

        .hero { display: flex; justify-content: center; padding: 50px; background-color: #f5f7fa; position: relative; }

        .hero-content { max-width: 50%; }

        .hero-content h1 { font-size: 3em; color: #002970; margin-bottom: 20px; }

        .hero-content p { font-size: 1.2em; color: #666; margin-bottom: 20px; }

        .hero-image { justify-content: space-between; width: 100%; }

        .qr-panel { position: relative; top: -26px; left: 9px; width: 300px; padding: 20px; background-color: #fff; border: 2px solid #ccc; border-radius: 15px; box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1); text-align: center; margin-right: 20px; }

        .qr-panel img { width: 200px; height: 200px; margin-bottom: 20px; }

        .form-panel { position: relative; top: -313px; left: 333px; width: 300px; padding: 20px; background-color: #fff; border: 2px solid #ccc; border-radius: 15px; box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1); text-align: center; }

        .form-panel form { display: flex; flex-direction: column; }

        .form-panel input { margin-bottom: 10px; padding: 10px; border: 1px solid #ccc; border-radius: 5px; }

        .form-panel button { padding: 10px; background-color: #00baf2; border: none; border-radius: 5px; color: #fff; font-weight: bold; cursor: pointer; }

        .features { padding: 50px; background-color: #fff; text-align: center; }

        .features h2 { font-size: 2.5em; color: #002970; margin-bottom: 40px; }

        .feature-item { display: inline-block; width: 30%; padding: 20px; text-align: center; }

        .feature-item img { height: 60px; margin-bottom: 20px; }

        .feature-item h3 { font-size: 1.5em; color: #002970; margin-bottom: 10px; }

```

```

        .feature-item p { font-size: 1em; color: #666; }

    footer { padding: 20px; background-color: #002970; text-align: center; color: #fff; }

</style>
</head>
<body>

<header>

    <div class="logo">
        
    </div>

    <nav>
        <ul>
            <li><a href="../index.php">Home Page</a></li>
            <li><a href="../index.php">Logout</a></li>
            <li><a href="#">Products</a></li>
            <li><a href="#">Solutions</a></li>
            <li><a href="#">Pricing</a></li>
            <li><a href="#">Resources</a></li>
            <li><a href="#">About Us</a></li>
            <li><a href="#">Contact Us</a></li>
            <li><a href="#" class="btn-primary">Sign Up</a></li>
            <li><a href="#" class="btn-secondary">Login</a></li>
        </ul>
    </nav>
</header>

<main>

    <section class="hero">
        <div class="hero-content">
            <h3>AAYUSH SAHAY</h3>
            <h1>Accept Payments with Ease</h1>
            <p>Start accepting payments instantly with Paytm Payment Gateway</p>
        </div>
        <div class="hero-image">
            <div class="qr-panel">
                
            </div>
        </div>
    </section>
</main>

```

```

        <p>Scan QR To Pay</p>
    </div>
    <div class="form-panel">
        <form action="ptmAction.php" method="post">
            <input type="text" name="name" placeholder="Name" required>
            <input type="text" name="mobile" placeholder="Mobile Number" required>
            <input type="text" name="renter_id" placeholder="Renter ID" required>
            <input type="text" name="amount" placeholder="Payment Amount" required>
            <button type="submit">Pay</button>
        </form>
    </div>
</div>
</section>

<section class="features">
    <h2>Why Choose Paytm Payment Gateway?</h2>
    <div class="feature-item">
        
        <h3>Secure</h3>
        <p>Advanced security features to protect your transactions.</p>
    </div>
    <div class="feature-item">
        
        <h3>Easy Integration</h3>
        <p>Simple and quick integration with your website or app.</p>
    </div>
    <div class="feature-item">
        
        <h3>24/7 Support</h3>
        <p>Dedicated support team available around the clock.</p>
    </div>
</section>

```

```

        </div>
    </section>
</main>

<footer>
    <div class="footer-content">
        <p>&copy; 2024 Paytm. All rights reserved.</p>
    </div>
</footer>

<script>
    document.addEventListener('keydown', function(event) {
        if (event.ctrlKey && event.key === '/') {
            alert('Transaction was successful.');
        } else if (event.ctrlKey && event.key === ';') {
            alert('Transaction was failure.');
        }
    });
</script>
</body>
</html>

```

Source code (ptmAction.php)

```

<?php
session_start();
include("../connection.php");
include("../functions.php");
$user_data=check_login($con);
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Payment Transaction</title>
    <style>
        body {

```

```
font-family: Arial, sans-serif;
background-color: #f4f4f9;
margin: 0;
padding: 0;
}

.container {
    width: 90%;
    margin: auto;
    overflow: hidden;
}

.section {
    margin: 20px 0;
    padding: 20px;
    background: #fff;
    border-radius: 10px;
    box-shadow: 0 0 10px rgba(0,0,0,0.1);
}

.section h2 {
    margin-top: 0;
    color: #00b3b3;
}

table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 10px;
}

table, th, td {
    border: 1px solid #ddd;
}

th, td {
    padding: 12px;
    text-align: left;
}

th {
    background-color: #00b3b3;
    color: #fff;
}
```

```

        tr:nth-child(even) {
            background-color: #f9f9f9;
        }

        .print-button {
            background-color: #0070ba;
            color: white;
            padding: 10px 20px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
        }

        .print-button:hover {
            background-color: #005f9e;
        }
    
```

</style>

<script>

```

        function printPage() {
            window.print();
        }
    
```

</script>

</head>

<body>

```

        <div class="container">
            <div class="section">
                <h2>Payment Transaction Details</h2>
                <?php
                    // Function to generate random 6-character UPI ID
                    function generateUPIID() {
                        $characters = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789';
                        $upi_id = 'UPI';
                        for ($i = 0; $i < 6; $i++) {
                            $upi_id .= $characters[rand(0, strlen($characters) - 1)];
                        }
                        return $upi_id;
                    }
                
```

```

        // Handling form submission

        if ($_SERVER["REQUEST_METHOD"] == "POST") {

            // Extract form data

            $name = $_POST['name'];

            $mobile = $_POST['mobile'];

            $renter_id = $_POST['renter_id'];

            $amount = $_POST['amount'];



            // Generate random UPI ID

            $upi_id = generateUPIID();



            // Current date and time

            $date_pay = date("Y-m-d H:i:s");



            // Database connection

            $conn = new mysqli('localhost', 'root', '',

'rent_management_system');





            // Check connection

            if ($conn->connect_error) {

                die("Connection failed: " . $conn->connect_error);

            }



            // Prepare SQL statement for inserting data into

            payment_status table

            $sql = "INSERT INTO payment_status (renter_id, date_pay,

month, amount, pay_status, description)

VALUES ('$renter_id', '$date_pay', '', '$amount',

'online_trxn_success', '$name$mobile$upi_id')";





            // Execute SQL statement

            if ($conn->query($sql) === TRUE) {

                echo "<script>alert('Transaction successful. UPI ID:

$upi_id');</script>";

            } else {

                echo "<script>alert('Transaction failed. Error: " .

$conn->error . "');</script>";

            }

        }

    }

}

```

```

// Close connection
$conn->close();

// Display transaction details
echo "
<table>
    <tr>
        <th>Name</th>
        <td>$name</td>
    </tr>
    <tr>
        <th>Mobile</th>
        <td>$mobile</td>
    </tr>
    <tr>
        <th>Renter ID</th>
        <td>$renter_id</td>
    </tr>
    <tr>
        <th>Amount</th>
        <td>$amount</td>
    </tr>
    <tr>
        <th>UPI ID</th>
        <td>$upi_id</td>
    </tr>
    <tr>
        <th>Date & Time</th>
        <td>$date_pay</td>
    </tr>
</table>
";
}

?>
</div>

```

```

        <button class="print-button" onclick="printPage () ">>Produce the
Copies</button>
    </div>
</body>
</html>

```

Source code (pay_records.php)

```

<?php
session_start();
include("../connection.php");
include("../functions.php");
$user_data=check_login($con);
?>
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Payment Records</title>
<style>
body {
    font-family: Arial, sans-serif;
    background-color: #f4f4f9;
    margin: 0;
    padding: 0;
}
.container {
    width: 80%;
    margin: auto;
    overflow: hidden;
}
.section {
    margin: 20px 0;
    padding: 20px;
    background: #fff;
    border-radius: 10px;
    box-shadow: 0 0 10px rgba(0,0,0,0.1);
}

```

```
.section h2 {
    margin-top: 0;
    padding-bottom: 10px;
    border-bottom: 1px solid #ddd;
}

table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 10px;
}

table, th, td {
    border: 1px solid #ddd;
}

th, td {
    padding: 12px;
    text-align: left;
}

th {
    background-color: #f4f4f4;
}

tr:nth-child(even) {
    background-color: #f9f9f9;
}

.form-container {
    margin-top: 20px;
    padding: 20px;
    background: #f9f9f9;
    border-radius: 5px;
    box-shadow: 0 0 5px rgba(0,0,0,0.1);
}

.form-container label, .form-container input {
    display: block;
    margin-bottom: 10px;
}

.form-container input[type="text"],
.form-container input[type="number"],
.form-container input[type="date"] {
```

```

width: calc(100% - 22px); /* Adjusted width for inputs */
padding: 8px;
font-size: 14px;
border: 1px solid #ddd;
border-radius: 4px;
}

.form-container input[type="submit"] {
width: 100%;
padding: 10px;
background-color: #4CAF50;
color: white;
border: none;
cursor: pointer;
border-radius: 4px;
}

.form-container input[type="submit"]:hover {
background-color: #45a049;
}

.delete-btn {
padding: 5px 10px;
background-color: #f44336;
color: white;
border: none;
cursor: pointer;
border-radius: 4px;
margin-left: 10px;
}

.delete-btn:hover {
background-color: #da190b;
}

</style>
<script>

function confirmDelete(id) {
var pin = prompt("Enter PIN to delete this record:");
if (pin === "55555") {
window.location.href = "delete_record.php?id=" + id;
} else {
}
}

```

```

        alert("Incorrect PIN. Record not deleted.");
    }
}

</script>
</head>
<body>

<div class="container">
    <div class="section">
        <h2>Add Payment Record</h2>
        <div class="form-container">
            <form action="pay_records.php" method="post">
                <label for="date_pay">Date of Payment:</label>
                <input type="date" id="date_pay" name="date_pay" required>

                <label for="renter_id">Renter ID:</label>
                <input type="text" id="renter_id" name="renter_id" required>

                <label for="month">Month:</label>
                <input type="text" id="month" name="month" required>

                <label for="amount">Amount:</label>
                <input type="number" id="amount" name="amount" required>

                <label for="pay_status">Status:</label>
                <input type="text" id="pay_status" name="pay_status" required>

                <label for="description">Payment Description:</label>
                <input type="text" id="description" name="description" required>

                <input type="submit" value="Submit">
            </form>
        </div>
    </div>

```

```

<div class="section">
    <h2>Payment Records</h2>
    <table>
        <tr>
            <th>Renter ID</th>
            <th>Date of Payment</th>
            <th>Month</th>
            <th>Amount</th>
            <th>Status</th>
            <th>Payment Description</th>
            <th>Action</th>
        </tr>
        <?php
            // Database connection
            $conn = new mysqli('localhost', 'root', '',
            'rent_management_system');

            // Check connection
            if ($conn->connect_error) {
                die("Connection failed: " . $conn->connect_error);
            }

            // Insert record into payment_status table
            if ($_SERVER["REQUEST_METHOD"] == "POST") {
                $date_pay = $_POST['date_pay'];
                $renter_id = $_POST['renter_id'];
                $month = $_POST['month'];
                $amount = $_POST['amount'];
                $pay_status = $_POST['pay_status'];
                $description = $_POST['description'];

                $sql = "INSERT INTO payment_status (renter_id,
                date_pay, month, amount, pay_status, description)
                VALUES ('$renter_id', '$date_pay', '$month',
                '$amount', '$pay_status', '$description')";

                if ($conn->query($sql) === TRUE) {

```

```

        echo "<meta http-equiv='refresh' content='0'>";
    } else {
        echo "Error: " . $sql . "<br>" . $conn->error;
    }
}

// Fetch data from payment_status table
$sql = "SELECT * FROM payment_status";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    while ($row = $result->fetch_assoc()) {
        echo "<tr>
            <td>{$row['renter_id']}

```

Source code (delete_record.php)

```
<?php

session_start();
include("../connection.php");
include("../functions.php");
$user_data=check_login($con);

// Check if ID is provided
if (isset($_GET['id'])) {
    // Get the ID from the URL parameter
    $id = $_GET['id'];

    // Database connection
    $conn = new mysqli('localhost', 'root', '', 'rent_management_system');

    // Check connection
    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error);
    }

    // SQL to delete record from payment_status table
    $sql = "DELETE FROM payment_status WHERE id = $id";

    if ($conn->query($sql) === TRUE) {
        // Redirect back to pay_records.php after successful deletion
        header("Location: pay_records.php");
        exit();
    } else {
        echo "Error deleting record: " . $conn->error;
    }
}

// Close connection
$conn->close();
} else {
    // Redirect to pay_records.php if ID parameter is not provided
}
```

```

        header("Location: pay_records.php");
        exit();
    }
?>

```

Source code (see_records.php)

```

<?php
session_start();
include("../connection.php");
include("../functions.php");
$user_data=check_login($con);
?>
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>See Records</title>
<style>
body {
    font-family: Arial, sans-serif;
    background-color: #f4f4f9;
    margin: 0;
    padding: 0;
}
.container {
    width: 90%;
    margin: auto;
    overflow: hidden;
}
.section {
    margin: 20px 0;
    padding: 20px;
    background: #fff;
    border-radius: 10px;
    box-shadow: 0 0 10px rgba(0,0,0,0.1);
}

```

```
.section h2 {
    margin-top: 0;
}

table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 10px;
}

table, th, td {
    border: 1px solid #ddd;
}

th, td {
    padding: 12px;
    text-align: left;
}

th {
    background-color: #f4f4f4;
}

tr:nth-child(even) {
    background-color: #f9f9f9;
}

.print-button {
    margin-top: 20px;
    padding: 10px 20px;
    background-color: #4CAF50;
    color: white;
    border: none;
    cursor: pointer;
}

.print-button:hover {
    background-color: #45a049;
}

</style>

</head>
<body>

<div class="container">
    <div class="section">
```

```

<h2>Bill Details</h2>
<table>
    <tr>
        <th>Serial Number</th>
        <th>ID</th>
        <th>Renter ID</th>
        <th>Month</th>
        <th>Year</th>
        <th>Due Date</th>
        <th>Room Rent</th>
        <th>Units Used</th>
        <th>Electric Bill</th>
        <th>Advance Paid</th>
        <th>Amount Dues</th>
        <th>Miscellaneous</th>
        <th>Total Amount</th>
        <th>Created At</th>
    </tr>
    <?php
        // Database connection
        $conn = new mysqli('localhost', 'root', '',
                           'rent_management_system');

        // Check connection
        if ($conn->connect_error) {
            die("Connection failed: " . $conn->connect_error);
        }

        // Fetch data from bill_details table
        $sql = "SELECT * FROM bill_details";
        $result = $conn->query($sql);

        if ($result->num_rows > 0) {
            // Output data of each row
            $serial_number = 1;
            while ($row = $result->fetch_assoc()) {
                echo "<tr>

```

```

        <td>{$serial_number}</td>
        <td>{$row['id']}</td>
        <td>{$row['renter_id']}</td>
        <td>{$row['month']}</td>
        <td>{$row['year']}</td>
        <td>{$row['due_date']}</td>
        <td>{$row['room_rent']}</td>
        <td>{$row['units_used']}</td>
        <td>{$row['electric_bill']}</td>
        <td>{$row['advance_paid']}</td>
        <td>{$row['amount_dues']}</td>
        <td>{$row['miscellaneous']}</td>
        <td>{$row['total_amount']}</td>
        <td>{$row['created_at']}</td>
    </tr>" ;
$serial_number++;
}
} else {
echo "<tr><td colspan='14'>No records found</td></tr>";
}

// Close connection
$conn->close();
?>
</table>
</div>

<div class="section">
<h2>Payment Records</h2>
<table>
<tr>
<th>ID</th>
<th>Renter ID</th>
<th>Date of Payment</th>
<th>Month</th>
<th>Amount</th>
<th>Status</th>

```

```

<th>Description</th>
</tr>
<?php
// Database connection
$conn      =      new      mysqli('localhost',      'root',      '',
'rent_management_system');

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Fetch data from payment_status table
$sql = "SELECT * FROM payment_status";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // Output data of each row
    while ($row = $result->fetch_assoc()) {
        echo "<tr>
 {$row['id']} | {$row['renter_id']} | {$row['date_pay']} | {$row['month']} | {$row['amount']} | {$row['pay_status']} | {$row['description']} |
</tr>";
    }
} else {
    echo "<tr><td colspan='7'>No records found</td></tr>";
}

// Close connection
$conn->close();
?>
</table>

```

```

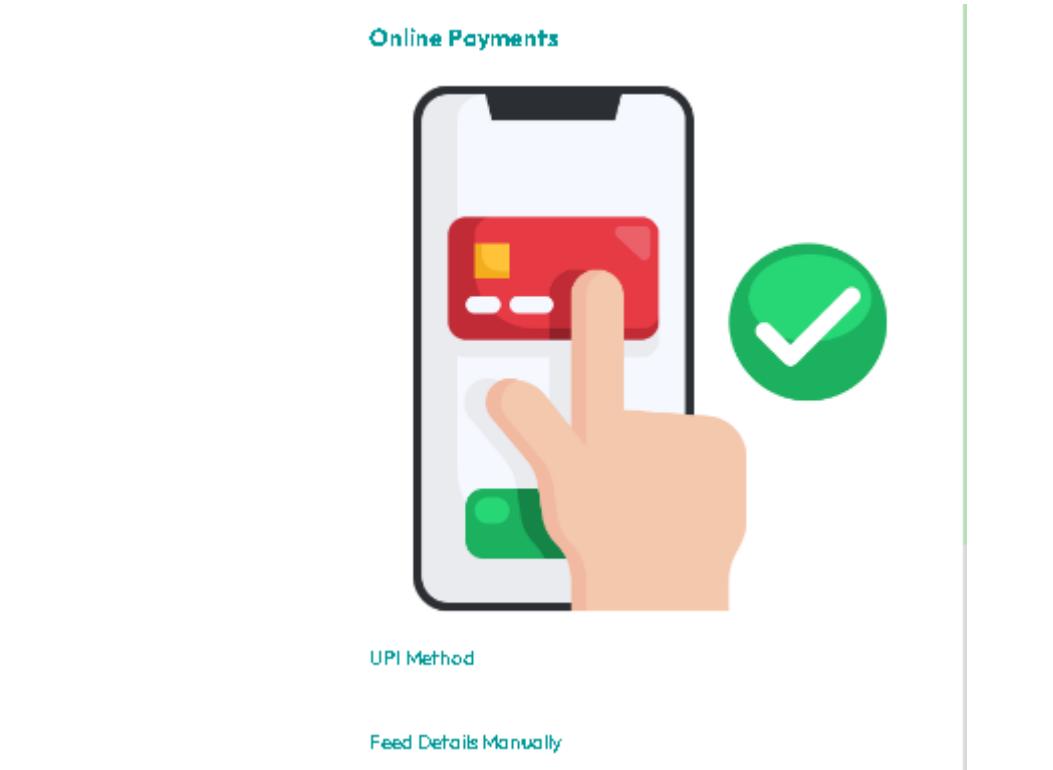
        </div>

        <button class="print-button" onclick="window.print()">Produce
Copies</button>
    </div>
</body>
</html>

```

Screenshots

We will open the Dashboard and go for the below option:

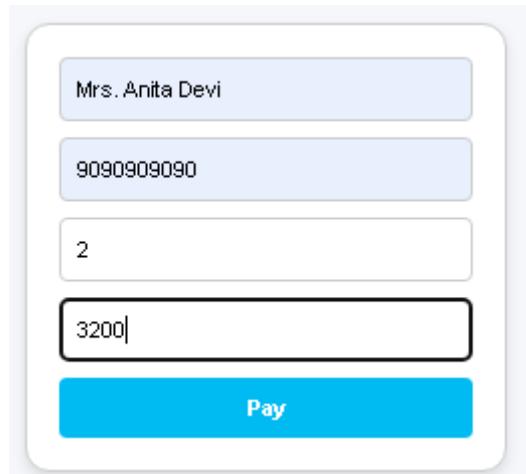


We can Do the payments by the online or manually feed in the transaction details manually.

We click on the upi Method

The image shows a screenshot of the Paytm Payment Gateway dashboard. At the top, there is a dark blue header bar with the Paytm logo and navigation links: Home Page, Logout, Products, Solutions, Pricing, Resources, About Us, Contact Us, Sign Up, and Login. Below the header, the user's name "AYUSH SAHAY" is displayed. The main section features the heading "Accept Payments with Ease". Below this, a sub-instruction reads "Start accepting payments instantly with Paytm Payment Gateway". To the right of the text is a large QR code with the instruction "Scan QR To Pay" underneath. To the right of the QR code is a form with four input fields: "Name", "Mobile Number", "Renter ID", and "Payment Amount", followed by a blue "Pay" button.

The tenant can scan the qr to make the payments , the Landlord can also make the Online Transaction Invoice after filling the details



Mrs. Anita Devi
9090909090
2
3200|
Pay

We will click on to Pay , we can make the invoice , once the transaction was successful.



Payment Transaction Details	
Name	Mrs. Anita Devi
Mobile	9090909090
Renter ID	2
Amount	3200
UPI ID	UPIEBZ3XS
Date & Time	2024-07-06 11:26:43

Produce the Copies

The transaction will be having the details that can be stored for the future occurrences.

Also the Landlord can feed the transaction details by the help of the form in the manual fashion.

The pay_records.php has the feature to fill in the transaction details manually. Also it will also show the details of the payments made by the tenants.

Add Payment Record

Date of Payment:



Renter ID:

Month:

Amount:

Status:

Payment Description:

Submit

Add Payment Record

Date of Payment:



Renter ID:

Month:

Amount:

Status:

Payment Description:

Submit

And when we submit it , we can also see the other details on the same page.

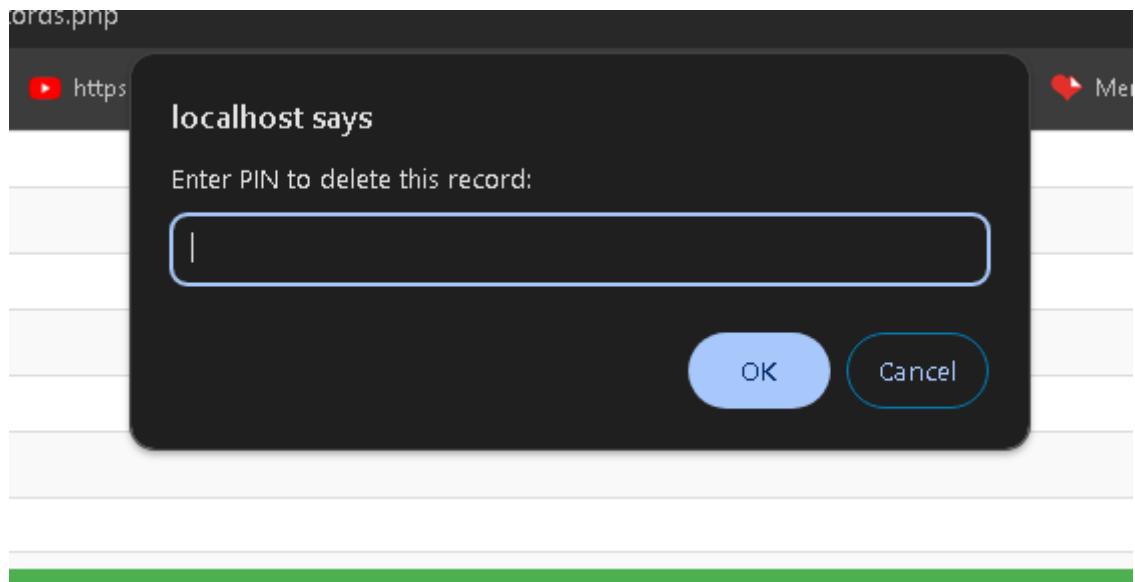
Payment Records

Renter ID	Date of Payment	Month	Amount	Status	Payment Description	Action
1	2024-07-06	JULY	2999.00	July Full Paid	-	<button>Delete</button>
1	2024-07-06		2999.00	online_trxn_success	Mr. Amit Kumar1234512345UPI2FBPGI	<button>Delete</button>
2	2024-07-06		3200.00	online_trxn_success	Mrs. Anita Dev9090909090UPIEBZ3XS	<button>Delete</button>
2	2024-07-06	JULY	3200.00	July Full Paid	Paid By Online	<button>Delete</button>

We see the details are stored in the tabular format , and they have a delete button near to them . The landlord cannot mistakenly delete any record , this will require the pin for the deleting that . The pin is the Software product key which is 55555.

The delete option is a feature when there is a requirement to delete the redundant entries.

We have Entered the payment details twice , once by the UPI , other by the manually way. We can delete that manually entry to remove the ambiguity.The manually section is required to be utilised when the payments to the landlord are made in cash.



Payment Records

Renter ID	Date of Payment	Month	Amount	Status	Payment Description	Action
1	2024-07-06	JULY	2999.00	July Full Paid	-	<button>Delete</button>
1	2024-07-06		2999.00	online_trxn_success	Mr. Amit Kumar1234512345UPI2FBPGI	<button>Delete</button>
2	2024-07-06		3200.00	online_trxn_success	Mrs. Anita Devi9090909090UPIEBZ3XS	<button>Delete</button>

The entry was removed easily.

When we are required to see the bill and the respective entries of the system , for all the transaction , we will go to open this option:

Records & Reports



[Click to Open Early Records](#)

[Export Reports\(in pdf\)](#)

We will click on to open the Records , 1st link , we will open the page that lists all the bills and their payment status that leads us to the see_records.php

Bill Details

Serial Number	ID	Renter ID	Month	Year	Due Date	Room Rent	Units Used	Electric Bill	Advance Paid	Amount Dues	Miscellaneous	Total Amount	Created At
1	1	1	July	2024	2024-07-15	2300.00	11	99.00	200.00	0.00	[{"description": "Car Parking fee", "amount": "600"}]	2999.00	2024-07-04 15:38:54

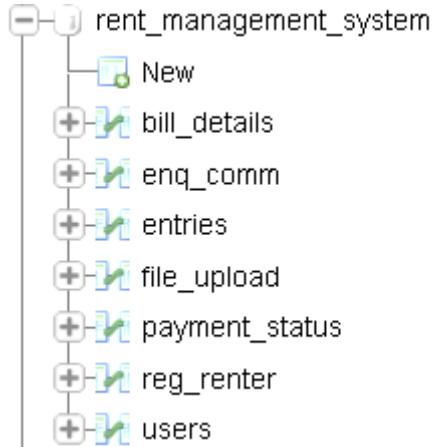
Payment Records

ID	Renter ID	Date of Payment	Month	Amount	Status	Description
1	1	2024-07-06	JULY	2999.00	July Full Paid	-
2	1	2024-07-06		2999.00	online_trxn_success	Mr. Amit Kumar1234512345UPI2FBPGI
5	2	2024-07-06		3200.00	online_trxn_success	Mrs. Anita Devi9090909090UPIEBZ3XS

[Produce Copies](#)

We have added to print to pdf functionality for the task of Easy sharing and storing.

The Database Structures utilised for the Payment Interface in the Rent Management System are :



We have utilised 2 tables in the interface that are :

- (i) bill_details.php
- (ii) payment_status

bill_details

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id 📃	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	renter_id	int(11)			No	None			Change Drop More
3	month	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
4	year	int(11)			No	None			Change Drop More
5	due_date	date			No	None			Change Drop More
6	room_rent	decimal(10,2)			No	None			Change Drop More
7	units_used	int(11)			No	None			Change Drop More
8	electric_bill	decimal(10,2)			No	None			Change Drop More
9	advance_paid	decimal(10,2)			No	None			Change Drop More
10	amount_dues	decimal(10,2)			No	None			Change Drop More
11	miscellaneous	text	utf8mb4_general_ci		No	None			Change Drop More
12	total_amount	decimal(10,2)			No	None			Change Drop More
13	created_at	timestamp			No	current_timestamp()			Change Drop More

payments_status

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id 📃	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	renter_id	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
3	date_pay	date			No	None			Change Drop More
4	month	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
5	amount	decimal(10,2)			No	None			Change Drop More
6	pay_status	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
7	description	text	utf8mb4_general_ci		Yes	NULL			Change Drop More

The data present in the bill_details table

id	renter_id	month	year	due_date	room_rent	units_used	electric_bill	advance_paid	amount_dues	miscellaneous	total_amount	created_at
1	1	July	2024	2024-07-15	2300.00	11	99.00	200.00	0.00	{"description": "Car fee", "amount": "800"}	2999.00	2024-07-04 15:38:54

The data present inside the payments_status table

id	renter_id	date_pay	month	amount	pay_status	description
1	1	2024-07-06	JULY	2999.00	July Full Paid	-
2	1	2024-07-06		2999.00	online_trxn_success	Mr. Amit Kumar1234512345UPI2FBPGI
5	2	2024-07-06		3200.00	online_trxn_success	Mrs. Anita Devi9090909090UPIEBZ3XS

Chapter : 13

Data - Backup System

- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

Data - Backup System

The Rent Management System will provide the feature to make the folder to make the data backup , that can be used to utilised in future occurrences , when we face a data failure.

Data Failure can only occur in the below consequences :

- (i) The internal files can get deleted by any data cleanup software.
- (ii) the data can be mistakenly be deleted.

To overcome above consequences to prevent and overcome the problems , we have added the Data-backup system

Rent Management System

```
| -> backup_proc  
| -> backup.php
```

The responsibility of the above file is to produce all the documents submitted to make the agreement , and all the data stored inside the database tables.

This will produce the folder that will contain all the files and the data in the form of the html , that can be viewed and printed to suitable page format

Source code (backup.php)

```
<?php  
// Function to generate random 6-character UPI ID  
function generateUPIID() {  
    $characters = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789';  
    $upi_id = 'UPI';  
    for ($i = 0; $i < 6; $i++) {  
        $upi_id .= $characters[rand(0, strlen($characters) - 1)];  
    }  
    return $upi_id;  
}  
  
// Handling form submission  
if ($_SERVER["REQUEST_METHOD"] == "POST") {  
    // Verify the PIN  
    $pin = $_POST['pin'];
```

```

if ($pin != '55555') {
    echo "<script>alert('Incorrect PIN');</script>";
    exit();
}

// Set up the backup directory on the Desktop
$backup_dir = getenv("HOMEDRIVE") . getenv("HOMEPATH") .
'\Desktop\Backup_' . date('Ymd');

if (!file_exists($backup_dir)) {
    mkdir($backup_dir, 0777, true);
}

// Function to copy directory
function copyDirectory($src, $dst) {
    $dir = opendir($src);
    @mkdir($dst);
    while (false !== ($file = readdir($dir))) {
        if (($file != '.') && ($file != '..')) {
            if (is_dir($src . '/' . $file)) {
                copyDirectory($src . '/' . $file, $dst . '/' . $file);
            } else {
                copy($src . '/' . $file, $dst . '/' . $file);
            }
        }
    }
    closedir($dir);
}

// Copy the stores directory to the backup folder
$source_dir = 'C:/xampp/htdocs/rmstest11/rent_agr/stores';
$destination_dir = $backup_dir . '/stores';
copyDirectory($source_dir, $destination_dir);

// Database connection
$conn = new mysqli('localhost', 'root', '', 'rent_management_system');

// Check connection

```

```

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Function to fetch data from a table and return as HTML
function fetchTableData($conn, $tableName, $fields) {
    $sql = "SELECT " . implode(", ", $fields) . " FROM " . $tableName;
    $result = $conn->query($sql);

    if ($result->num_rows > 0) {
        $html = "<h2>$tableName</h2>";
        $html .= "<table border='1'><tr>";
        foreach ($fields as $field) {
            $html .= "<th>$field</th>";
        }
        $html .= "</tr>";

        while ($row = $result->fetch_assoc()) {
            $html .= "<tr>";
            foreach ($fields as $field) {
                $html .= "<td>" . $row[$field] . "</td>";
            }
            $html .= "</tr>";
        }
        $html .= "</table>";
        return $html;
    } else {
        return "<h2>$tableName</h2><p>No records found</p>";
    }
}

// Fetch data from each table
$users_fields = ['id', 'user_id', 'user_name', 'password',
'productkey', 'date'];
$reg_renter_fields = ['id', 'name', 'age', 'gender', 'datereg',
'aadhar', 'mobile', 'address', 'roomNo', 'price'];
$entries_fields = ['id', 'room_id', 'photo_path', 'sign_path',
'aadhar_path', 'created_at'];

```

```

    $enq_comm_fields = ['id', 'name', 'age', 'gender', 'address', 'mobile',
'time_of_saving'];

    $bill_details_fields = ['id', 'renter_id', 'month', 'year', 'due_date',
'room_rent', 'units_used', 'electric_bill', 'advance_paid', 'amount_dues',
'miscellaneous', 'total_amount', 'created_at'];

    $payment_status_fields = ['id', 'renter_id', 'date_pay', 'month',
'amount', 'pay_status', 'description'];



$data_html = "";
$data_html .= fetchTableData($conn, 'users', $users_fields);
$data_html .= fetchTableData($conn, 'reg_renter', $reg_renter_fields);
$data_html .= fetchTableData($conn, 'entries', $entries_fields);
$data_html .= fetchTableData($conn, 'enq_comm', $enq_comm_fields);

$data_html .= fetchTableData($conn, 'bill_details',
$bill_details_fields);
$data_html .= fetchTableData($conn, 'payment_status',
$payment_status_fields);

// Close connection
$conn->close();

// Save the data HTML to a file
$data_file = $backup_dir . '/data.html';
file_put_contents($data_file, $data_html);

echo "<script>alert('Backup created successfully');</script>";
echo "<script>window.location.href='backup.php';</script>";
}

?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Backup Module</title>
    <style>
        body {

```

```
font-family: Arial, sans-serif;
background-color: #f4f4f9;
margin: 0;
padding: 0;
}

.container {
    width: 90%;
    margin: auto;
    overflow: hidden;
}

.section {
    margin: 20px 0;
    padding: 20px;
    background: #fff;
    border-radius: 10px;
    box-shadow: 0 0 10px rgba(0,0,0,0.1);
}

.section h2 {
    margin-top: 0;
    color: #00b3b3;
}

table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 10px;
}

table, th, td {
    border: 1px solid #ddd;
}

th, td {
    padding: 12px;
    text-align: left;
}

th {
    background-color: #00b3b3;
    color: #fff;
}
```

```

        tr:nth-child(even) {
            background-color: #f9f9f9;
        }

        .backup-button {
            background-color: #0070ba;
            color: white;
            padding: 10px 20px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
        }

        .backup-button:hover {
            background-color: #005f9e;
        }
    
```

</style>

</head>

<body>

```

    <div class="container">
        <div class="section">
            <h2>Data Backup</h2>
            <button class="backup-button" onclick="createBackup()">Create
            Backup</button>
            <div id="data-container">
                <!-- Data will be loaded here by JavaScript -->
            </div>
        </div>
    </div>

```

<script>

```

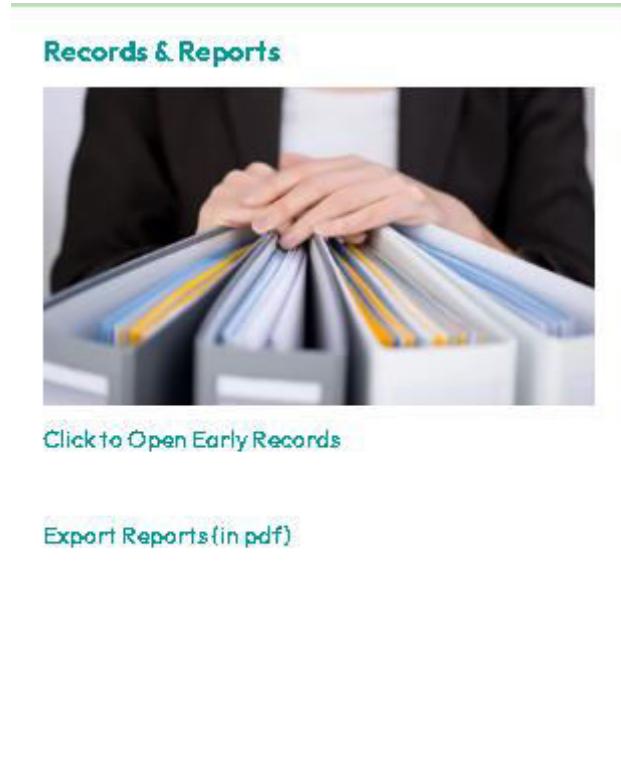
        function createBackup() {
            var pin = prompt("Enter Pin:");
            if (pin !== "55555") {
                alert("Incorrect PIN");
                return;
            }
            document.forms["backupForm"].submit();
        }
    
```

```
</script>

<form id="backupForm" method="POST" style="display: none;">
    <input type="hidden" name="pin" value="55555">
</form>
</body>
</html>
```

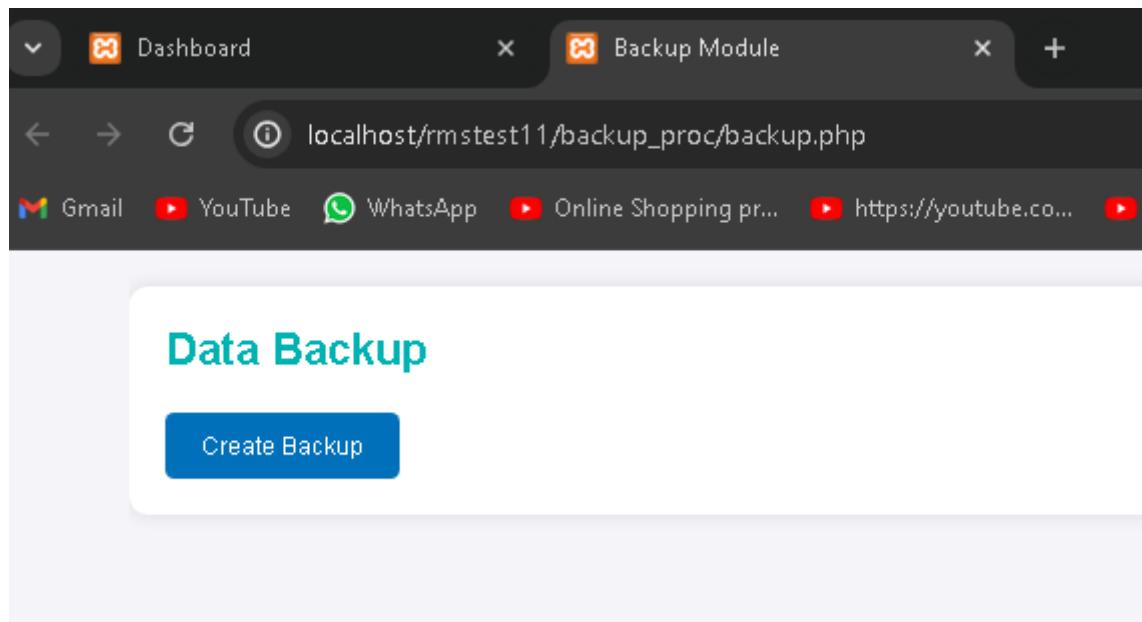
Screenshots of the Backup Process

We will go to the dashboard of the Rent Management System

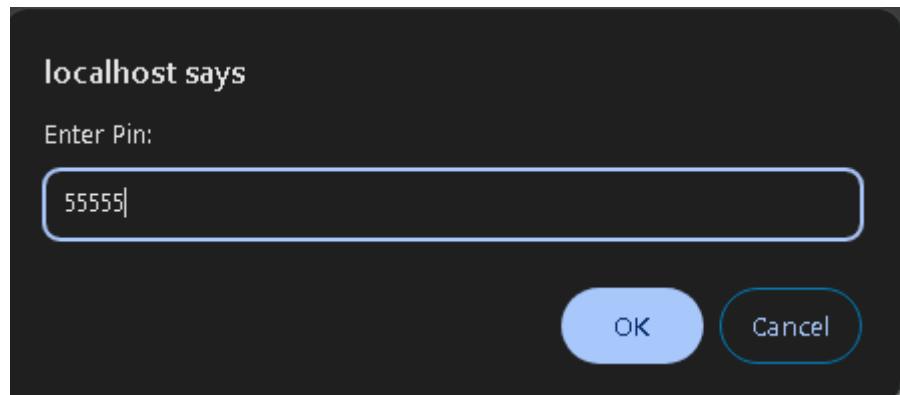


We will click on the below option : Export Reports in pdf.

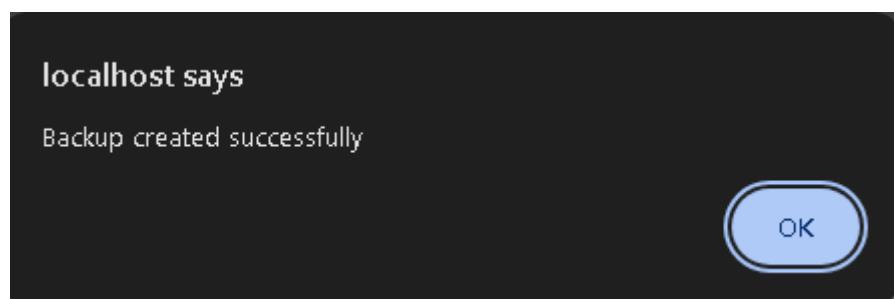
Then in the next step we are redirected to the page that looks like:



We will click on the Create Backup button. That will ask us the pin which is the software product key -55555



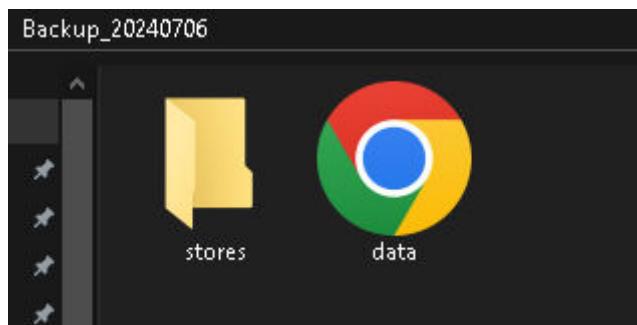
On the correct entry , we are notified as :



The backup is made in the Desktop Location of the landlord.

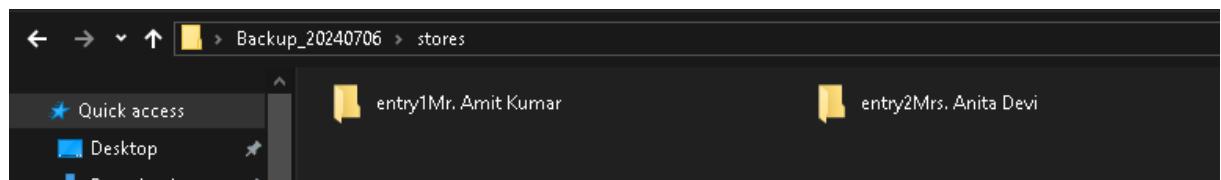


We double click to open the folder to see its contents :

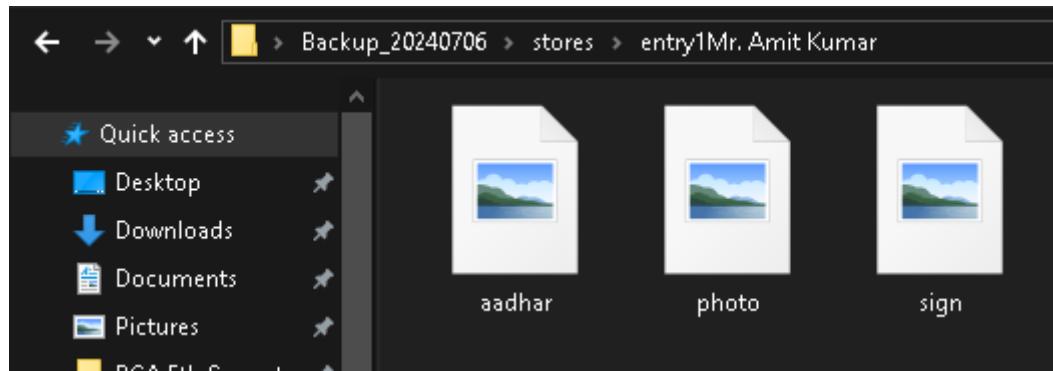


We will open the stores folder. This folder is going to contain the necessary files / documents that were submitted to the software to produce the agreements for the tenants.

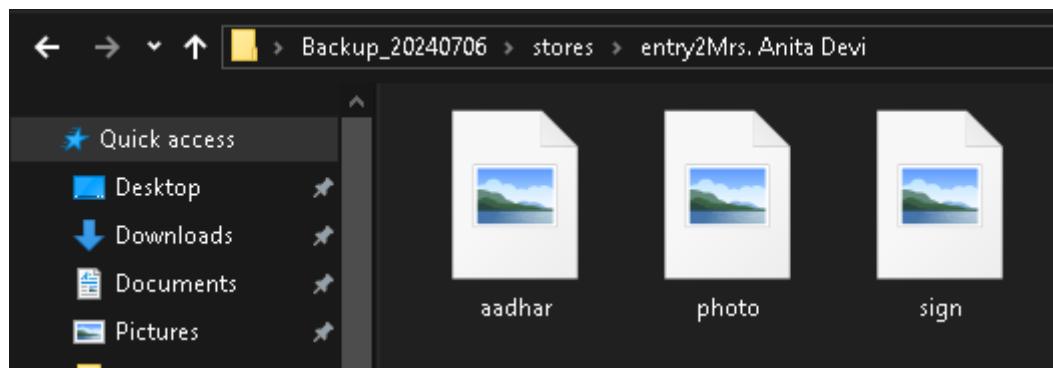
We open the stores folder to see:



Each folder is opened to verify that all the contents are present in them or not.



and for the second folder :



Now we will open the data that is present in the data.html

users

id	user_id	user_name	password	productkey	date
1	7	test@gmail.com	test	55555	2024-07-01 11:04:19

reg_renter

id	name	age	gender	datereg	aadhar	mobile	address	roomNo	price
1	Mr. Amit Kumar	32	Male	2024-07-02	23232323	9090909090	gaya , bihar	1	2300.00
2	Mrs. Anita Devi	47	Female	2024-07-03	5656565656	1234512345	Rajendra Nagar Patna	2	3200.00

entries

id	room_id	photo_path	sign_path	aadhar_path	created_at
1	1	stores/entry1Mr. Amit Kumar/photo.jpg	stores/entry1Mr. Amit Kumar/sign.jpg	stores/entry1Mr. Amit Kumar/aadhar.jpg	2024-07-02 16:27:04
2	2	stores/entry2Mrs. Anita Devi/photo.jpg	stores/entry2Mrs. Anita Devi/sign.jpg	stores/entry2Mrs. Anita Devi/aadhar.jpg	2024-07-03 18:35:12

enq_comm

id	name	age	gender	address	mobile	time_of_saving
1	AAYUSH SAHAY	20	Male	KADAMKUAN , PATNA , BIHAR	9994445550	2024-06-30 11:20:43
2	Akash Kumar	21	Male	Rajapur , Patna	1234512345	2024-06-30 11:26:34
3	Sneha Kumari	22	Female	Rajendra Nagar , Patna	9879874545	2024-06-30 11:27:19
4	Mrs. Anita Devi	48	Female	Rajendra Nagar , Patna	1234512345	2024-07-03 18:31:56

bill_details

id	renter_id	month	year	due_date	room_rent	units_used	electric_bill	advance_paid	amount_dues	miscellaneous	total_amount	created_at
1	1	July	2024	2024-07-15	2300.00	11	99.00	200.00	0.00	[{"description": "Car Parking fee", "amount": "800"}]	2999.00	2024-07-04 15:38:54

payment_status

id	renter_id	date_pay	month	amount	pay_status	description
1	1	2024-07-06	JULY	2999.00	July Full Paid	-
2	1	2024-07-06		2999.00	online_txrn_success	Mr. Amit Kumar1234512345UPI2FBPGI
5	2	2024-07-06		3200.00	online_txrn_success	Mrs. Anita Devi9090909090UPIEBZ3KS

Thus all the data present in the databases is also produced to the pdf by just clicking CTRL+P to get the interface to print to pdf entire web page according to the convenience

users					
id	user_id	user_name	password	productkey	date
1	7	test@gmail.com	test	55555	2024-07-01 11:04:19

reg_renter

id	name	age	gender	datereg	aadhar	mobile	address	roomNo	price
1	Mr. Amit Kumar	32	Male	2024-07-02	23232323	9090909090	Gaya , Bihar	1	2300.00
2	Mrs. Anita Devi	47	Female	2024-07-03	5656565656	1234512345	Rajendra Nagar Patna	2	3200.00

entries

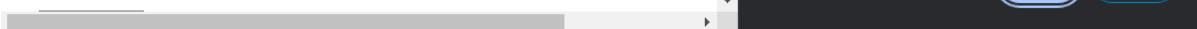
id	room_id	photo_path	sign_path	aadhar_path
1	1	stores/entry1Mr. Amit Kumar/photo.jpg	stores/entry1Mr. Amit Kumar/sign.jpg	stores/entry1Mr. Amit Kumar/aadhar.jpg
2	2	stores/entry2Mrs. Anita Devi/photo.jpg	stores/entry2Mrs. Anita Devi/sign.jpg	stores/entry2Mrs. Anita Devi/aadhar.jpg

enq_comm

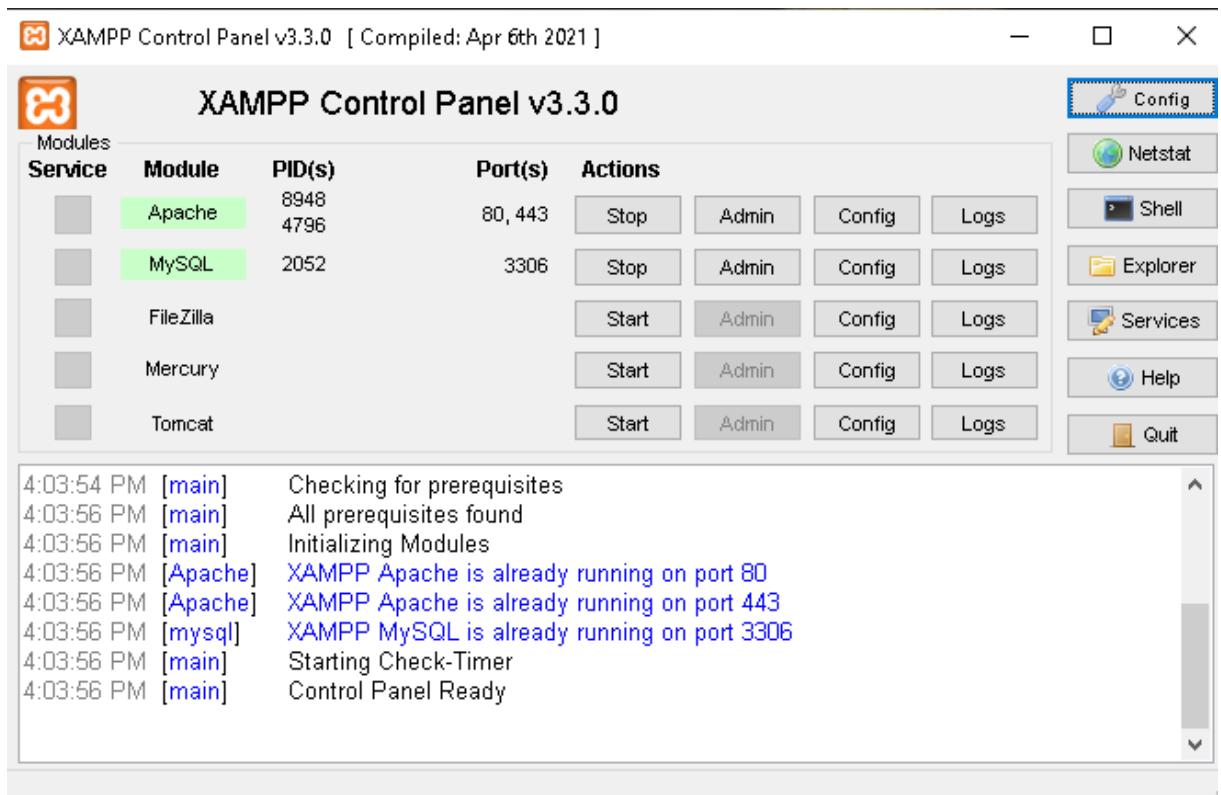
id	name	age	gender	address	mobile	time_of_saving
1	AAYUSH SAHAY	20	Male	KADAMKUAN , PATNA , BIHAR	9994445550	2024-06-30 11:20:43
2	Akash Kumar	21	Male	Rajapur , Patna	1234512345	2024-06-30 11:26:34
3	Sneha Kumari	22	Female	Rajendra Nagar , Patna	9879874545	2024-06-30 11:27:19
4	Mrs. Anita Devi	48	Female	Rajendra Nagar , Patna	1234512345	2024-07-03 18:31:56

bill_details

id	renter_id	month	year	due_date	room_rent	units_used	electric_bill	advance_paid	amount_dues	miscellaneous	total
1	1	July	2024	2024-07-15	2300.00	11	99.00	200.00	0.00	[{"description": "Car Parking fee", "amount": "800"}]	29

payment_status

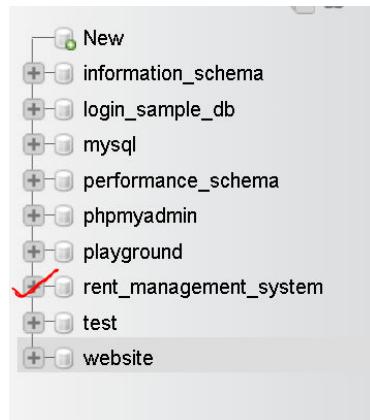
Also to backup the data from the database , another most efficient way is to open the xammp control panel.



We will open the Admin button along the right side of stop button next to the MySQL services. The Admin button opens the phpmyadmin on the localhost , and it is accessed by the help of the browser.

The screenshot shows the phpMyAdmin interface. The left sidebar lists databases: New, information_schema, login_sample_db, mysql, performance_schema, phpmyadmin, playground, rent_management_system, test, and website. The main panel shows the 'General settings' section with 'Server connection collation: utf8mb4_unicode_ci'. It also includes 'Appearance settings' (Language: English, Theme: pmahomme) and sections for 'Database server', 'Web server', and 'phpMyAdmin' (version information: 5.2.1 up to date). The 'Database server' section lists the server configuration: Server: 127.0.0.1 via TCP/IP, Server type: MariaDB, Server connection: SSL is not being used, Server version: 10.4.32-MariaDB - mariadb.org binary distribution, Protocol version: 10, User root@localhost, and Server charset: UTF-8 Unicode (utf8mb4).

We will see on the left hand side and select the required database that is the rent_management_system. We will click to open that.



In the next page we will click on the export button

Exporting tables from "rent_management_system" database

New template:

Template name Create

Existing templates:

Template: -- Select a template -- Update Delete

Export method:

Quick - display only the minimal options
 Custom - display all possible options

Format:

SQL

Export

We can select various types of format to save the export. It is more preferred to save as : Export to Sql , we will export it to the Backup Folder that we made. This sql file can be able to be imported , in the same admin page.

We can use the sql file incase we tend to shift the entire software system to another working computer. Its much preferred to make the sql export and databackup each time a new Tenant is Signed the contract with.

Chapter : 14

Homepage : Rent Management System

- (i) list of Pages
- (ii) Code of Pages
- (iii) Database Structure
- (iv) Resources Utilised

Homepage

The Rent Management System , features are accessed only through the homepage.

The login page will bring us to the homepage of the Rent Management System

Rent Management System

- | - > index.php (homepage of the system)
- | - > style.css (for styling the homepage)
- | - >script1.js (providing interactivity to the system)

Source Code (index.php)

```
<?php
session_start();
include("connection.php");
include("functions.php");
$user_data=check_login($con);
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Dashboard</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <nav>
        <a href="/"></a>
        <div class="nav-country">
            
            <div>
                <p>Share</p>
            </div>
        </div>
    </nav>
    <div class="content">
        <h1>Welcome to Rent Management System</h1>
        <h2>Dashboard</h2>
        <div>
            <table border="1">
                <thead>
                    <tr>
                        <th>Category</th>
                        <th>Count</th>
                    </tr>
                </thead>
                <tbody>
                    <tr>
                        <td>Properties</td>
                        <td>100</td>
                    </tr>
                    <tr>
                        <td>Tenants</td>
                        <td>50</td>
                    </tr>
                    <tr>
                        <td>Payments</td>
                        <td>200</td>
                    </tr>
                </tbody>
            </table>
        </div>
    </div>
</body>
</html>
```

```

        <h1><a href="loc_office/loc_office.php">Location</a></h1>
        </div>
    </div>

    <div class="nav-search">
        <div class="nav-search-category">
            <p>All</p>
            
        </div>
        <form action="searchAction.php" method="GET" id="form1" style="width:75px; height:10px; padding-top:none;">
            <input type="text" class="nav-search-input" name="query" placeholder="Search @ Rent Management System" style="width:310px; height:16px;">
            <button type="submit" style="position:relative; top:-37px; left:549px;">
                
            </button>
        </form>
    </div>

    <div class="nav-language">
        
        <a href="make_bill/make_bill.php"><p>Create Bills</p></a>
        
    </div>

    <div class="nav-text">
        <p></p>
        <h1><a href="enq_comm/enq_comm.php">Save Contacts</a></h1>
    </div>
    <div class="nav-text">
        <p><a style="font-size: 16px; font-weight:bold;" href="reg_renter/reg_renter.php">Registration</a></p><br>
        <h1><a style="font-size: 16px; font-weight:bold;" href="rent_agr/ag_resc.php">Registration& Agreements </a></h1>
    </div>

```

```

        <a href="#" class="nav-cart">
            <h4>Options</h4></a>
        </nav>

<div class="nav-bottom">
    <div>

        <p><a href="other_functions/it_rules.php" target="_blank">IT-Rules Followed</a></p>&ampnbsp&ampnbsp&ampnbsp
        <p><a href="other_functions/dev_details.php">Developer Contacts</a></p>&ampnbsp&ampnbsp&ampnbsp
        <p></p>&ampnbsp&ampnbsp&ampnbsp
        <p><a href="other_functions/l1_regist.php">Landlord Details</a></p>&ampnbsp&ampnbsp&ampnbsp
        <p><a href="reset.php">Reset Password</a></p>&ampnbsp&ampnbsp&ampnbsp
        <p><a href="logout.php">Logout</a></p>&ampnbsp&ampnbsp&ampnbsp
    </div>
</div>

<div class="header-slider">
    <a href="#" class="control_prev">#129144</a>
    <a href="#" class="control_next">#129146</a>
    <ul>
        
        
        
        
        
        
    </ul>
</div>

<div class="box-row header-box">
    <div class="box-col">
        <h3>Online Payments</h3>
        

```

```

<a href="pay_online(ptm.php">UPI Method</a><br>
    <a href="pay_online/pay_records.php" target="_blank">Feed
Details Manually</a>
</div>
<div class="box-col">
    <h3>Records & Reports</h3>
    
    <a href="see_records/see_records.php" target="_blank"><p>Click
to Open Early Records</p></a><br>
        <p><a href="backup_proc/backup.php"
target="_blank">Export Reports (in pdf)</a></p>
</div>
<div class="box-col">
    <h3>List of Renters</h3>
    
    <a href="rentr_list/rentr_list.php" target="_blank">Click To
Proceed</a>
</div>
<div class="box-col">
    <h3>Rooms Available</h3>
    
    <a href="rooms/roomlisting.php">View List of Available Rooms
<br>+ Add Rooms <br>+Remove Rooms</a>
</div>
</div>
<?php echo "{$user_data['user_name']} &password
:{$user_data['password']}?">
<script src="script1.js"></script>
</body>
</html>

```

Source Code (style.css)

```

@import
url('https://fonts.googleapis.com/css2?family=Outfit:wght@100..900&display=
swap');

*{padding:0; margin:0; box-sizing:border-box; font-family: Outfit;}

body{background:#dadada;}

```

```
a{text-decoration:none; color:inherit;}

nav{display:flex; align-items:center; justify-content: space-between;
background:#131921;padding:10px 20px; color:#fff; }

.nav-country{display:flex; align-items:end; margin-left:15px; font-size:13px; color:#c4c4c4;}

.nav-country h1{color:#fff; font-size: 14px; }

.nav-search{flex:1; display:flex; align-items:center; background:white;
color:gray; max-width:1000px; border-radius:4px; margin-left:15px;
}

.nav-search-category{display:flex; align-items:center; padding:10px 20px;
gap:5px; background:#e5e5e5; border-radius:4px 0 0 4px; }

.nav-search-input{border:none; outline:none; padding-left:20px; width:100%;
height:8px; padding-bottom: 0px; }

.nav-search-icon{max-width:41px; padding:8px; background:#ffd646; border-radius:0 4px 4px 0; }

.nav-search-icon{max-width:41px; padding:8px; background: fixed3ffd64f;
border-radius:0 4px 4px 0; }

.nav-language{display:flex; align-items:center; gap:2px; font-weight: 600;
margin-left:15px; }

.nav-text{margin-left:15px; }

.nav-text p{font-size:10px; }

.nav-text h1{font-size:14px; }

.nav-cart{display:flex; align-items:center; margin :0 15px; }

.nav-bottom{display:flex; align-items:center; gap:20px; padding:8px 20px;
background:#232f3e; color:white; font-size:15px; }

.nav-bottom div{display:flex; align-items:center; font-weight:500;
gap:5px; }

.header-slider ul{display:flex; overflow-y: hidden; }

.header-slider img{max-width:100%; mask-image: linear-gradient(to bottom,
#000000 50%, transparent 50%);}
```

```

.header-slider a{position:absolute; top:20%; z-index:1;padding:5vh 1vw;
background:#fffffff4f; color:#0000007b; text-decoration:none;font-
weight:600; font-size:18px; cursor:pointer; }

.control_next{right:0; }

.box-row{display:flex; flex-wrap:wrap; row-gap:20px; justify-content:space-
between; margin:20px 30px; }

.box-col{display:flex; flex-direction:column; gap:10px; padding:15px 20px;
background:#fff; max-width:24%; min-width:24%; min-height:200px; z-
index:1; }

.box-col{font-size:13px; color:#009999; font-weight:500; }

.header-box{margin-top:-20vw; }

```

Source Code (script1.js)

```

const imgs = document.querySelectorAll('.header-slider ul img');

const prev_btn = document.querySelector('.control_prev');

const next_btn = document.querySelector('.control_next');

let n = 0;

function changeSlide() {
    for (let i = 0; i < imgs.length; i++) {
        imgs[i].style.display = 'none';
    }
    imgs[n].style.display = 'block';
}

changeSlide();

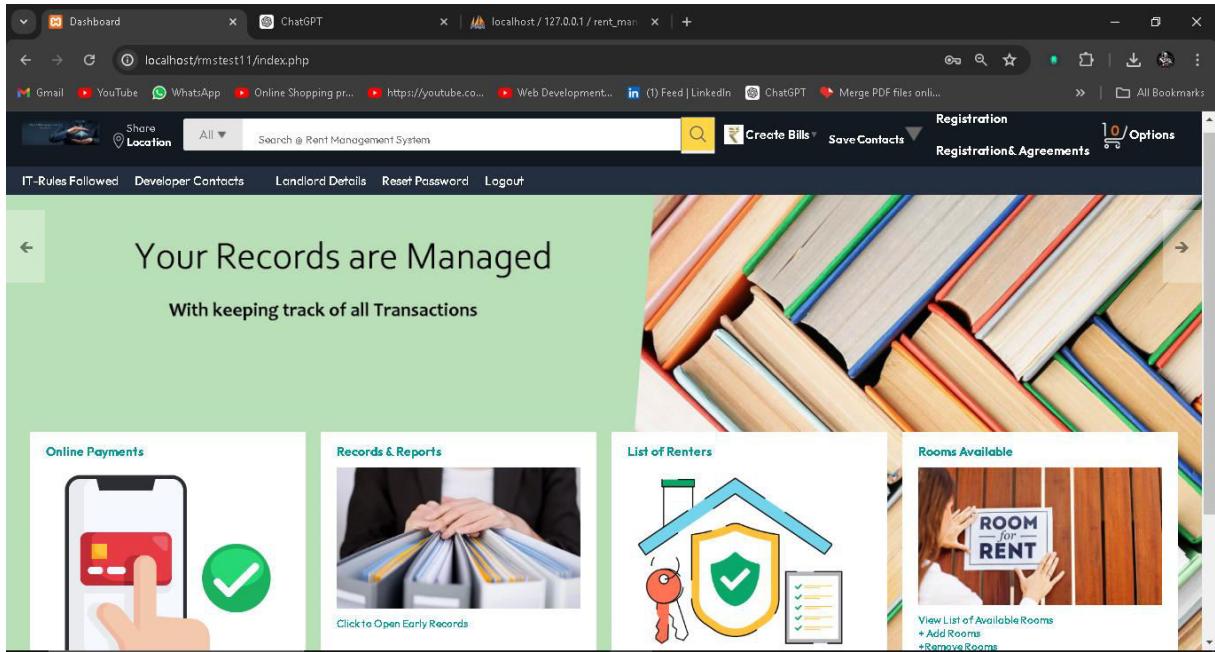
prev_btn.addEventListener('click', (e) => {
    if (n > 0) {
        n--;
    } else {
        n = imgs.length - 1;
    }
    changeSlide();
});

next_btn.addEventListener('click', (e) => {
    if (n < imgs.length - 1) {

```

```
n++;  
} else {  
    n = 0;  
}  
changeSlide();  
});
```

Screenshots



5.4

SOURCE CODE DOCUMENTATION

RENT MANAGEMENT SYSTEM



RENT MANAGEMENT SYSTEM

Documentation of Source Code

	Chapters	Page Number
Chapter 1	Register , login & Logout Documentation	208-221
Chapter 2	Add Rooms and Remove Rooms Documentation	222-228
Chapter 3	Add Communication Details Documentation	229-230
Chapter 4	Reset User Password Documentation	231-237
Chapter 5	Share Location Documentation	238-241
Chapter 6	Register Renter Documentation	242-243
Chapter 7	Agreement Renter Documentation	244-245
Chapter 8	View Renters Documentation	246-251
Chapter 9	Create Bills Documentation	252-258
Chapter 10	Search Bar Documentation	259-267
Chapter 11	Other Functions Documentation	268-282
Chapter 12	Payments System Documentation	283-290
Chapter 13	DataBackup System Documentation	291-300
Chapter 14	Homepage Documentation	301-304

Chapter : 1

Register , Login & Logout

Documentation

Documentation : Registration Page (signup.php)

PHP Code Documentation:

- Start a new session or resume the existing session
- Include the database connection file
- Include the functions file
- Check if the request method is POST
- Get the username from the form
- Get the password from the form
- Get the product key from the form
- Validate inputs
- Generate a random user ID
- Insert the new user into the database
- Redirect to the login page and terminate the script

HTML and CSS Documentation:

- Define the document type and HTML language
- Set character encoding to UTF-8
- Set the viewport to ensure proper scaling on different devices
- Set the title of the document
- Define CSS styles for the page and form elements
- Create a container div to center the form
- Add an image with ID "logo"
- Create the form with action "signup.php" and method "post"
- Add a form group for the username input
- Add a form group for the password input
- Add a form group for the product key input
- Add a submit button and a link to the login page

Documentation : In Detail

PHP Code Documentation:

- **Start a new session or resume the existing session**

```
php  
session_start();
```

Initiates a session or resumes the current one to store session variables.

- **Include the database connection file**

```
php  
include("connection.php");
```

Connects to the database by including the connection settings from `connection.php`.

- **Include the functions file**

```
php  
include("functions.php");
```

Includes utility functions from `functions.php` for use in this script.

- **Check if the request method is POST**

```
php  
if ($_SERVER['REQUEST_METHOD'] == "POST") {
```

Determines if the form was submitted via a POST request.

- **Get the username from the form**

```
php  
$user_name = $_POST['user_name'];
```

Retrieves the username input from the POST request.

- **Get the password from the form**

```
php
```

```
$password = $_POST['password'];
```

Retrieves the password input from the POST request.

- **Get the product key from the form**

```
php
```

```
$prokey = $_POST['pk'];
```

Retrieves the product key input from the POST request.

- **Validate inputs**

```
php
```

```
if (!empty($user_name) && !empty($password) && !is_numeric($user_name)) {
```

Checks that the username and password are not empty and that the username is not numeric.

- **Generate a random user ID**

```
php
```

```
$user_id = random_num(20);
```

Generates a random 20-character user ID using a function from the included functions.php.

- **Insert the new user into the database**

```
php
```

```
$query = "insert into users(user_id, user_name, password, productkey) values('$user_id', '$user_name', '$password', '$prokey')";  
mysqli_query($con, $query);
```

Executes an SQL query to insert the new user's details into the users table.

- **Redirect to the login page and terminate the script**

```
php
```

```
header("Location: login.php");  
die;
```

Redirects the user to login.php after successful registration and stops further script execution.

HTML and CSS Documentation:

- **Define the document type and HTML language**

```
html

<!DOCTYPE html>
<html lang="en">
```

- **Set character encoding to UTF-8**

```
html

<meta charset="UTF-8">
```

- **Set the viewport to ensure proper scaling on different devices**

```
html

<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

- **Set the title of the document**

```
html

<title>Document</title>
```

- **Define CSS styles for the page and form elements**

```
html

<style>
    * { margin: 0; padding: 0; box-sizing: border-box; }
    body { min-height: 100vh; background: #eee; display: flex; font-
family: sans-serif; }
    .container { margin: auto; width: 500px; max-width: 90%; }
    .container form { width: 100%; height: 100%; padding: 20px;
background: white; border-radius: 4px; box-shadow: 0 8px 16px
rgba(0,0,0,0.3); }
    .container form h1 { text-align: center; margin-bottom: 24px; color:
#222; }
    .container form .form-control { width: 100%; height: 40px;
background: white; border-radius: 4px; border: 1px solid silver; margin:
10px 0 18px 0; padding: 0 10px; }
    .container form .btn { margin-left: 50%; transform: translate(-50%);
width: 120px; height: 34px; border: none; outline: none; background:
#222; cursor: pointer; font-size: 16px; text-transform: uppercase;
color: white; border-radius: 4px; transition: 0.3s; }
    .container form .btn:hover { opacity: 0.7; }
    #btn1 a { text-decoration: none; color: #eee; }
    #btn1 { position: relative; top: -34px; left: 140px; font-size:
12px; }
    #btn2 a { text-decoration: none; color: #eee; }
```

```
#btn2 { position: relative; top: -68px; left: -140px; font-size: 12px; }
#logo { width: 626px; position: relative; top: 80px; height: 480px; left: 35px; }
</style>
```

- **Create a container div to center the form**

html

```
<div class="container">
```

- **Add an image with ID "logo"**

html

```

```

- **Create the form with action "signup.php" and method "post"**

html

```
<form action="signup.php" method="post">
```

- **Add a form group for the username input**

html

```
<div class="formgroup">
    <label for="">Username (Enter Email)</label>
    <input type="text" class="form-control" name="user_name" id="" required>
</div>
```

- **Add a form group for the password input**

html

```
<div class="formgroup">
    <label for="">Password</label>
    <input type="password" class="form-control" name="password" id="" required>
</div>
```

- **Add a form group for the product key input**

html

```
<div class="formgroup">
    <label for="">Product Key</label>
    <input type="number" class="form-control" name="pk" id="" required>
</div>
```

- **Add a submit button and a link to the login page**

```
html

<input class="btn" type="submit" value="Register!">
<a style="text-decoration: none; border-width: 10px; color: blue"
href="login.php">Login</a>
```

- **Close the form and container div**

```
html

</form>
</div>
```

- **Close the body and html tags**

```
html

</body>
</html>
```

Documentation : Login Page (login.php)

PHP Code Documentation:

1. Start a new session or resume the existing session

```
php

session_start();
```

2. Include the database connection file

```
php

include("connection.php");
```

3. Include the functions file

```
php

include("functions.php");
```

4. Check if the request method is POST

```
php

if ($_SERVER['REQUEST_METHOD'] == "POST") {
```

5. Get the username from the form

```
php  
$user_name = $_POST['user_name'];
```

6. Get the password from the form

```
php  
$password = $_POST['password'];
```

7. Validate inputs

```
php  
if (!empty($user_name) && !empty($password) && !is_numeric($user_name))  
{
```

8. Read from the database

```
php  
$query = "select * from users where user_name=' $user_name' limit 1";  
$result = mysqli_query($con, $query);
```

9. Check if the query returned any results

```
php  
if ($result && mysqli_num_rows($result) > 0) {
```

10. Fetch user data from the result set

```
php  
$user_data = mysqli_fetch_assoc($result);
```

11. Check if the password matches

```
php  
if ($user_data['password'] === $password) {
```

12. Set session variables for the user

```
php  
$_SESSION['user_id'] = $user_data['user_id'];  
$_SESSION['productkey'] = $user_data['productkey'];
```

13. Redirect to the index page

```
php

header("Location: index.php");
die;
```

14. Handle incorrect credentials

```
php

echo "error in credentials";
die;
```

HTML and CSS Documentation:

1. Define the document type and HTML language

```
html

<!DOCTYPE html>
<html lang="en">
```

2. Set character encoding to UTF-8

```
html

<meta charset="UTF-8">
```

3. Set the viewport to ensure proper scaling on different devices

```
html

<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

4. Set the title of the document

```
html

<title>Document</title>
```

5. Define CSS styles for the page and form elements

```
html

<style>
    * { margin: 0; padding: 0; box-sizing: border-box; }
    body { min-height: 100vh; background: #eee; display: flex; font-
family: sans-serif; }
    .container { margin: auto; width: 500px; max-width: 90%; }
    .container form { width: 100%; height: 100%; padding: 20px;
background: white; border-radius: 4px; box-shadow: 0 8px 16px
rgba(0,0,0,0.3); }
```

```

    .container form h1 { text-align: center; margin-bottom: 24px; color: #222; }
    .container form .form-control { width: 100%; height: 40px; background: white; border-radius: 4px; border: 1px solid silver; margin: 10px 0 18px 0; padding: 0 10px; }
    .container form .btn { margin-left: 50%; transform: translate(-50%); width: 120px; height: 34px; border: none; outline: none; background: #222; cursor: pointer; font-size: 16px; text-transform: uppercase; color: white; border-radius: 4px; transition: 0.3s; }
    .container form .btn:hover { opacity: 0.7; }
    #btn1 a { text-decoration: none; color: #eee; }
    #btn1 { position: relative; top: -34px; left: 140px; font-size: 12px; }
    #btn2 a { text-decoration: none; color: #eee; }
    #btn2 { position: relative; top: -68px; left: -140px; font-size: 12px; }
    #logo { width: 626px; position: relative; top: 80px; height: 480px; left: 35px; }
</style>

```

6. Create a container div to center the form

```

html

<div class="container">

```

7. Add an image with ID "logo"

```

html



```

8. Create the form with action "login.php" and method "post"

```

html

<form action="login.php" method="post">

```

9. Add a form group for the email input

```

html

<div class="formgroup">
    <label for="">Email</label>
    <input type="email" class="form-control" name="user_name" id="email" required>
</div>

```

10. Add a form group for the password input

```

html

<div class="formgroup">

```

```
<label for="">Password</label>
<input type="password" class="form-control" name="password"
id="password" required>
</div>
```

11. Add a submit button

```
html

<input class="btn" type="submit" value="Login">
```

12. Add a button to reset the password

```
html

<button id="btn1" class="btn"><a href="resetpwd.php">Reset
Password</a></button>
```

13. Add a button to register a new user

```
html

<button id="btn2" class="btn"><a href="signup.php">Register
User</a></button>
```

14. Close the form and container div

```
html

</form>
</div>
```

15. Close the body and html tags

```
html

</body>
</html>
```

16. Display an alert if the user was logged out successfully

```
html

<?php
if (isset($_GET['logout'])) && $_GET['logout'] == 'success') {
    echo "<script>alert('User was logged out
successfully');</script>";
} else {
    echo "<script>console.log('Logout parameter not set or
incorrect');</script>";
}
?>
```

Documentation : logout Page (logout.php)

PHP Code Documentation:

1. Start a new session or resume the existing session

```
php  
session_start();
```

2. Check if the session variable 'user_id' is set

```
php  
if (isset($_SESSION['user_id'])) {
```

3. Unset the 'user_id' session variable

```
php  
unset($_SESSION['user_id']);
```

4. Redirect to the login page with a logout success parameter

```
php  
header("Location: login.php?logout=success");
```

5. Terminate the script

```
php  
die;
```

Documentation : connection.php

- Define the database host

```
php  
$dbhost = "localhost";
```

- Define the database user

```
php  
$dbuser = "root";
```

- Define the database password

```
php
```

```
$dbpass = "";
```

- Define the database name

```
php
```

```
$dbname = "rent_management_system";
```

- Attempt to establish a connection to the MySQL database using the defined parameters

```
php
```

```
if (!$con = mysqli_connect($dbhost, $dbuser, $dbpass, $dbname)) {
```

- Terminate the script with an error message if the connection fails

```
php
```

```
die("failed to connect");
```

Documentation : functions.php

PHP Code Documentation:

1. Define the `check_login` function

```
php
```

```
function check_login($con) {
```

2. Check if the `user_id` session variable is set

```
php
```

```
if (isset($_SESSION['user_id'])) {
```

3. Get the `user_id` from the session

```
php
```

```
$id = $_SESSION['user_id'];
```

4. Query the database for the user with the given `user_id`

```
php
```

```
$query = "select * from users where user_id='\$id' limit 1";
```

5. Execute the query and store the result

```
php  
$result = mysqli_query($con, $query);
```

6. Check if the query returned any results

```
php  
if ($result && mysqli_num_rows($result) > 0) {
```

7. Fetch user data from the result set

```
php  
$user_data = mysqli_fetch_assoc($result);
```

8. Return the user data

```
php  
return $user_data;
```

9. Redirect to the login page if user_id is not set or user is not found

```
php  
header("Location: login.php");
```

10. Close the check_login function

```
php  
}
```

11. Define the random_num function

```
php  
function random_num($length) {
```

12. Initialize an empty string text

```
php  
$text = "";
```

13. Ensure the length is at least 5

```
php  
if ($length < 5) {  
    $length = 5;  
}
```

14. Generate a random length between 4 and the specified length

```
php  
$len = rand(4, $length);
```

15. Generate a random number string of the specified length

```
php  
for ($i = 0; $i <= $length; $i++) {  
    $text = rand(0, 9);  
}
```

16. Return the generated random number string

```
php  
return $text;
```

17. Close the random_num function

```
php  
}
```

Chapter : 2

View ,Add & Remove Rooms

Sorce Code Documentation

Room Listings PHP Script Documentation

PHP Section

Description This section establishes a connection to the database and retrieves data from the `file_upload` table.

Code

```
<?php
$server = "localhost";
$pass = "";
$user = "root";
$dbname = "rent_management_system";
$conn = mysqli_connect($server, $user, $pass, $dbname);
$sql = "select * from file_upload";
$result = $conn->query($sql);
?>
```

HTML Section

Description Defines the structure of the HTML document, including metadata, navigation links, headings, a table for displaying room listings, and a script for handling user interactions.

Code

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Room Listing</title>
    <link rel="stylesheet" href="roomlist.css">
</head>
<body>
    <nav>
        <a href="../index.php">Go to Home Page</a><br>
        <a href="../logout.php">Click to logout</a><br>
        <a href="addrooms.php">Add Rooms</a><br>
        <a href="remove.php">Remove Rooms</a><br>
    </nav>
    <h1>List of Available Rooms</h1>
    <table>
        <thead>
            <tr>
                <th>Serial No.</th>
                <th>Room ID</th>
                <th>Room Name</th>
                <th>Status</th>
                <th>Room Layouts</th>
            </tr>
        </thead>
```

```

<tbody>
<?php
$count = 1;
if ($result->num_rows > 0) {
    while ($row = $result->fetch_assoc()) {
        echo "<tr>";
        echo "<td>" . $count . "</td>";
        echo "<td>" . $row['id'] . "</td>";
        echo "<td>" . $row['roomname'] . "</td>";
        echo "<td>" . $row['status'] . "</td>";
        echo "<td><button id='openPdfBtn'>View Rooms</button></td>";
        echo "</tr>";
        $count++;
    }
} else {
    echo "<tr><td colspan='5'>No records found</td></tr>";
}
?>
</tbody>
</table>
<script>
    document.addEventListener('DOMContentLoaded', () => {
        const layoutButtons = document.querySelectorAll('.layout-btn');

        layoutButtons.forEach(button => {
            button.addEventListener('click', () => {
                const roomNumber = button.getAttribute('data-room');
                alert(`Displaying layout for Room ${roomNumber}`);
                // Here you can implement more complex logic, such as
                fetching and displaying room layout details.
            });
        });
    });

    document.getElementById('openPdfBtn').onclick = function() {
        window.open('uploads/room1.pdf', '_blank');
    };
</script>
</body>
</html>

```

Add Rooms PHP Script Documentation

PHP Section

Description This section handles the addition of room details into the database, including validation for PDF uploads and storing file information.

Code

```
<?php
$server = "localhost";
$pass = "";
$user = "root";
$dbname = "";
$conn = mysqli_connect($server, $user, $pass, $dbname);

$roomName = $_POST['roomname'];
$roomStatus = $_POST['roomstatus'];

if(isset($_POST['submit'])) {
    $targetDir = 'uploads/';
    $targetFile = $targetDir . basename($_FILES['pdfFile']['name']);
    $fileType = strtolower(pathinfo($targetFile, PATHINFO_EXTENSION));

    if ($fileType != "pdf" || $_FILES['pdfFile']['size'] > 6000000) {
        echo "Only PDF files less than 2MB are allowed.";
    } else {
        // Move uploaded file to the uploads directory
        if (move_uploaded_file($_FILES['pdfFile']['tmp_name'], $targetFile)) {
            $filename = $_FILES['pdfFile']['name'];
            $folder_path = $targetDir;
            $timestamp = date('Y-m-d H:i:s');

            $sql = "INSERT INTO file_upload (filename, roomname, status,
            folder_path, timestamp)
                    VALUES ('$filename', '$roomName', '$roomStatus',
                    '$folder_path', '$timestamp')";

            if ($conn->query($sql) === TRUE) {
                echo "Room details added successfully.";
            } else {
                echo "Error: " . $sql . "<br>" . $conn->error;
            }
        } else {
            echo "Error uploading file.";
        }
    }
}

$conn->close();
?>
```

HTML Section

Description Defines the HTML structure for adding room details, including form elements for room name, status, and PDF upload, along with submission handling.

Code

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Add Room Details</title>
    <link rel="stylesheet" href="roomlist.css">
</head>
<body>
    <h1>Add Rooms Details</h1><br>
    <form action="addrrooms.php" method="post" enctype="multipart/form-data">
        <label for="roomname">Name of Room :</label>
        <input type="text" id="roomname" name="roomname"><br><br>
        <label for="roomstatus">Status (Free/Occupied):</label>
        <input type="text" id="roomstatus" name="roomstatus"><br><br>
        <label for="pdfFile">Interiors and Layout (.pdf)</label>
        <input type="file" id="pdfFile" name="pdfFile"><br><br>
        <button type="submit" name="submit">Submit</button>
    </form>
    <a href="roomlisting.php">Go to Room Listing</a>
</body>
</html>
```

Explanation

- **PHP Section**
 - Establishes a connection to the database (`$conn`) using `mysqli_connect`.
 - Retrieves form inputs (`$roomName`, `$roomStatus`) via `$_POST`.
 - Validates and handles file uploads (`$_FILES['pdfFile']`).
 - Inserts room details into the `file_upload` table upon successful upload.
- **HTML Section**
 - Displays a form (`<form>`) for users to input room details.
 - Includes input fields for room name, status, and PDF file upload.
 - Provides a submit button (`<button>`) to submit room details.
 - Includes a link (`<a>`) to navigate back to the room listing page (`roomlisting.php`).

Remove Rooms PHP Script Documentation

PHP Section

Description Handles the deletion of room records from the database based on user input.

Code

```
<?php
$server = "localhost";
$pass = "";
$user = "root";
$dbname = "rent_management_system";
$conn = mysqli_connect($server, $user, $pass, $dbname);

$sql = "SELECT * FROM file_upload";
$result = $conn->query($sql);

$delId = $_POST['idtodelete'];
$sql1 = "DELETE FROM file_upload WHERE id = ?";

if ($stmt = $conn->prepare($sql1)) {
    $stmt->bind_param("i", $delId);
}

if ($stmt->execute()) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record: " . $conn->error;
}

$stmt->close();
$conn->close();
?>
```

HTML Section

Description Defines the HTML structure for displaying room listings and providing a form for deleting rooms based on room ID.

Code

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Remove Rooms</title>
    <link rel="stylesheet" href="roomlist.css">
</head>
<body>
    <nav>
```

```

<a href="roomlisting.php">Go to Home Page</a><br>
<a href="logout.php">Click to logout</a><br>
<a href="addrooms.php">Add Rooms</a><br>
<a href="delete.php">Remove Rooms</a><br>
</nav>
<h1>List of Available Rooms</h1>
<table>
    <thead>
        <tr>
            <th>Serial No.</th>
            <th>Room ID</th>
            <th>Room Name</th>
            <th>Status</th>
            <th>Room Layouts</th>
        </tr>
    </thead>
    <tbody>
        <?php
        $count = 1;
        if ($result->num_rows > 0) {
            while ($row = $result->fetch_assoc()) {
                echo "<tr>";
                echo "<td>" . $count . "</td>";
                echo "<td>" . $row['id'] . "</td>";
                echo "<td>" . $row['roomname'] . "</td>";
                echo "<td>" . $row['status'] . "</td>";
                echo "<td><button id='openPdfBtn'>Open PDF</button></td>";
                echo "</tr>";
                $count++;
            }
        } else {
            echo "<tr><td colspan='5'>No records found</td></tr>";
        }
        ?>
    </tbody>
</table>
<form action="remove.php" method="post">
    <h1>Delete Rooms</h1>
    <label for="idtodelete">Enter the room ID to delete:</label>
    <input type="number" name="idtodelete" id="idtodelete">
    <button type="submit" name="submit">Delete</button>
</form>
<script>
    document.addEventListener('DOMContentLoaded', () => {
        const layoutButtons = document.querySelectorAll('.layout-btn');

        layoutButtons.forEach(button => {
            button.addEventListener('click', () => {
                const roomNumber = button.getAttribute('data-room');
                alert(`Displaying layout for Room ${roomNumber}`);
                // Implement logic to fetch and display room layout
                details.
            });
        });
    });

    document.getElementById('openPdfBtn').onclick = function() {

```

```
        window.open('uploads/pdf.pdf', '_blank');
    };
</script>
</body>
</html>
```

Explanation

- **PHP Section**
 - Establishes a database connection (`$conn`) using `mysqli_connect`.
 - Executes a query (`$sql`) to fetch all records from the `file_upload` table.
 - Handles deletion of a specific room record based on user input (`$delId`) using a prepared statement (`$stmt`).
- **HTML Section**
 - Displays room details in a table (`<table>`) with headers (`<th>`) for serial number, room ID, room name, status, and room layouts.
 - Provides a form (`<form>`) for users to input the room ID they wish to delete.
 - Includes a submit button (`<button>`) to initiate the deletion process.
 - Offers navigation links (`<a>`) to other pages, such as the home page (`roomlisting.php`) and logout (`logout.php`).
 - Utilizes JavaScript (`<script>`) to handle user interactions, such as displaying room layouts and opening PDF files.

Chapter : 3

Add Communication Details

Source Code Documentation

Enquiry Details Registration

The Rent Management System also features the facility to record the details of the consumers that come for the enquiry purposes, This is providing a form that also stores the details and produces the pdf , of those records to share and store them in the pdf format.

This will provide the feature for the Landlord to contact those consumers in future , to invite them to see the rooms available for the rental purposes , if they require it .

Folder Structure : RMS / enq_comm/

```
| -> enq_comm.php (enquiry - communication details form)  
| -> print.php
```

enq_comm.php (enquiry - communication details form)

The PHP script handles CRUD operations (Create, Read, Update, Delete) for a rent management system. The script connects to a MySQL database, performs various database operations, and displays the data in a web interface. Below is a detailed breakdown of the code:

1. Database Connection

- The script starts by establishing a connection to the MySQL database using `mysqli`. It uses `localhost` as the server, `root` as the username, and an empty password for the database `rent_management_system`.

2. Handling Form Submission

- The script checks if the form is submitted using the POST method.
- If the "save" button is clicked, it retrieves the values from the form fields (name, age, gender, address, mobile) and the hidden `id` field.
- Depending on whether an ID is provided, it either updates an existing record or inserts a new record into the `enq_comm` table.
- If the "delete" button is clicked, it deletes the record with the specified ID.
- If the "print" button is clicked, it redirects to `print.php` with the record ID as a query parameter.
- After handling the form submission, the script redirects to `enq_comm.php`.

3. Retrieving Data

- The script retrieves all records from the `enq_comm` table and stores the result in `$result`.

4. HTML Structure

- The HTML structure consists of a form for entering and editing records, and a table for displaying the records.
- The form includes fields for name, age, gender, address, and mobile number, along with a hidden field for the record ID.

- The table displays the records with options to edit, delete, and print each record.
- 5. CSS Styling**
- The CSS styles the page elements such as the body, form, form groups, table, and buttons to provide a visually appealing layout.
- 6. JavaScript Functions**
- The `editRecord` function populates the form fields with the values of the selected record for editing.
 - The `printPDF` function uses the `html2pdf` library to generate a PDF of the displayed records.

print.php (to produce the pdf reports of Enquiry Details)

The PHP script below retrieves a record from the `enq_comm` table in the `rent_management_system` database based on an ID passed via the GET method. It then displays the record in an HTML page, allowing the user to print the record to PDF. Here's a detailed breakdown of the code:

- 1. Database Connection**
 - The script establishes a connection to the MySQL database using `mysqli`. It uses `localhost` as the server, `root` as the username, and an empty password for the database `rent_management_system`.
- 2. Fetching the Record**
 - The script checks if the `id` parameter is provided in the URL via the GET method.
 - If an ID is provided, it retrieves the corresponding record from the `enq_comm` table.
 - If the record is found, it stores the data in the `$row` array.
 - If no record is found, it outputs "No record found" and terminates the script.
 - If no ID is provided, it outputs "No ID provided" and terminates the script.
- 3. HTML Structure**
 - The HTML structure displays the details of the fetched record.
 - It includes a button to print the record to PDF and a link to go back to the records page.
- 4. CSS Styling**
 - The CSS styles the page elements such as the body, the record container, headings, paragraphs, and buttons to provide a visually appealing layout.
- 5. JavaScript Functions**
 - The `printPDF` function uses the `html2pdf` library to generate a PDF of the displayed record.
 - The script also includes hover effects for the back link button.

Chapter : 4

Reset Login Credentials / Reset User Password

Source Code Documentation

Password Reset

The Rent Management System aims to be robust as well as facility rich like the other software solutions in the market. The requirement of having the feature to rest the login credentials are more than a neccessity , The Data that is very precious for the business point of view., has risk to be accessed by other party incase they are known of the password.

This application Is a Web - Browser based , and when the web-browser is online , other services / malicious users may get password. So , it is a good practice to change the password . On the other hand , creating the backup of all the data is also crucial.

The Landlord can easily change the password by the use of the entery_email that is registered in the database. Then the product key that is going to be a different code for the other computers using the software . This Software is created to showcase the learning , therefore it is 55555.

And the Landlord can be able to reset / add a new password to the System.

Folder Structure : RMS / reset.php

reset.php (Page to reset the user password)

This code is a complete PHP script for resetting a user's password. It involves server-side validation, database operations, and a user interface for input. Below is a detailed documentation of the code:

```
session_start();
include("connection.php");
include("functions.php");
```

```

Starts a new session or resumes the existing session. Includes the necessary files for database connection and additional functions.

### \*\*Handling POST Request\*\*

```
```php
if($_SERVER['REQUEST_METHOD']=="POST"){
    $user_name=$_POST['user_name'];
    $prokey=$_POST['prokey'];
    $new_password = $_POST['new_password'];
}
```

```

Checks if the request method is POST and retrieves the username, product key, and new password from the form input.

### \*\*Validation of Input Fields\*\*

```
```php
if(!empty($user_name) && !empty($prokey) && !is_numeric($user_name)){
}
```

```

Validates that the username and product key fields are not empty and that the username is not numeric.

### \*\*Database Query to Check User Existence\*\*

```
```php
$query= "SELECT * FROM users WHERE user_name='".$user_name' LIMIT 1";
$result1 = mysqli_query($con, $query);
```

```

Queries the database to find a user with the specified username.

\*\*Password Reset Logic\*\*

```
```php
if ($result1) {
    if ($result1 && mysqli_num_rows($result1) > 0) {
        $user_data = mysqli_fetch_assoc($result1);

        $update_query = "UPDATE users SET password='$new_password' WHERE user_name='$user_name' and productkey='$prokey'";
        $update_result = mysqli_query($con, $update_query);

        if ($update_result) {
            echo "<script>alert('Password reset successful'); window.location.href = 'login.php';</script>";
        } else {
            echo "<script>alert('Password reset failed'); window.location.href = 'login.php';</script>";
        }
    } else {
        echo "<script>alert('User not found'); window.location.href = 'login.php';</script>";
    }
} else {
    echo "<script>alert('Database query failed'); window.location.href = 'login.php';</script>";
}
die;
```

```
}
```

```
~~~
```

Checks if the query was successful and if the user exists. If the user is found, updates the password in the database and shows an alert indicating success or failure. Redirects to `login.php` after displaying the alert.

HTML Structure and Styling

```
~~~html
```

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Reset PWD</title>

<style>

* { margin:0; padding:0; box-sizing:border-box; }

body { min-height: 100vh; background:#eee; display:flex; font-family: sans-serif; }

.container { margin:auto; width:500px; max-width:90%; }

.container form { width:100%; height:100%; padding:20px; background:white; border-radius:4px; box-shadow: 0 8px 16px rgba(0,0,0,0.3); }

.container form h1 { text-align:center; margin-bottom: 24px; color:#222; }

.container form .form-control { width:100%; height:40px; background:white; border-radius:4px; border:1px solid silver; margin:10px 0 18px 0; padding: 0 10px; }

.container form .btn { margin-left:50%; transform:translate(-50%); width:120px; height:34px; border:none; outline:none; background:#222; cursor:pointer; font-size: 16px; text-transform:uppercase; color:white; border-radius: 4px; transition: 0.3s; }
```

```

.container form .btn:hover{opacity:0.7;}

#btn1 a{text-decoration: none; color: #eee; }

#btn1{position:relative; top:-34px; left:140px; font-size: 12px; }

#btn2 a{text-decoration: none; color: #eee; }

#btn2{position:relative; top:-68px; left:-140px; font-size: 12px; }

#logo{width:626px; position: relative; top:80px; height:480px; left:35px; }

</style>

</head>

<body>

<?php

if (isset($_GET['logout']) && $_GET['logout'] == 'success') {

    echo "<script>alert('User was logged out successfully');</script>";

} else {

    echo "<script>console.log('Logout parameter not set or incorrect');</script>";

}

?>



<div class="container">

<form action="reset.php" method="post">

    <h1>Reset Password</h1>

    <div class="formgroup">

        <label for="">Email</label>

        <input type="email" class="form-control" name="user_name" id="email" required>

    </div>

    <div class="formgroup">

```

```

<label for="">Product Key</label>

<input type="text" class="form-control" name="prokey" id="prokey" required>

</div>

<div class="formgroup">

    <label for="">New Password</label>

    <input type="password" class="form-control" name="new_password" id="password" required>

</div>

<input class="btn" type="submit" value="RESET">

<button id="btn1" class="btn"><a href="login.php">Login</a></button>

<button id="btn2" class="btn"><a href="signup.php">Register User</a></button>

</form>

</div>

</body>

</html>

```

```

Defines the HTML structure and CSS styling for the reset password form, including input fields for email, product key, and new password. Also includes buttons for login and user registration.

---

**\*\*Logout Handling Script\*\***

```

```php
<?php

if (isset($_GET['logout']) && $_GET['logout'] == 'success') {

```

```
echo "<script>alert('User was logged out successfully');</script>";  
} else {  
    echo "<script>console.log('Logout parameter not set or incorrect');</script>";  
}  
?  
```
```

Checks if the logout parameter is set and displays a success message. If not, it logs a message to the console.

## Chapter : 5

### Share Location of the Office Page

Source Code Documentation



## Share Location

The Rent Management System is a web based solution to manage the business of the room rental services. This is a web-browser operable application. It is also equipped with the feature to show the location of the office. Incase the landlord has property in a distributed manner / or he is featuring other plots of land on which rent is applicable , the location of the Google maps can be swiftly be shared. This features the location , depicted by the google maps and the share link by copy to the clipboard feature.

### Folder Structure

#### Rent Management System

```
| -> loc_office
 | -> loc_office.php
 | -> styles11.css
 | -> script11.js
```

### PHP Code Documentation

#### Overview

This php document creates a simple web page displaying an embedded Google Map with a specified location and provides two buttons for user interaction: one to copy a link to the clipboard and another to navigate back to the home page. Also it cannot be accessed by the url , 1st the login status is checked , then connection is checked then only access granted to view it.

#### HTML Elements

`<!DOCTYPE html>

Defines the document type and version of HTML being used.

`<html lang="en">

Specifies the language of the document as English.

`<head>`

Contains metadata and links to external resources:

- `<meta charset="UTF-8">`: Sets the character encoding to UTF-8.
- `<meta name="viewport" content="width=device-width, initial-scale=1.0">`: Ensures the page is responsive and scales correctly on different devices.
- `<title>Location</title>`: Sets the title of the web page.
- `<link rel="stylesheet" href="styles11.css">`: Links to an external CSS file for styling.

`<body>`

Contains the main content of the document:

- `<div class="content">`: A container for the main content.
- `<div id="mapContainer">`: A container for the embedded Google Map.
  - `<iframe>`: Embeds the Google Map with specified location, dimensions, and settings.
- `<button id="copyLink">Click to copy link</button>`: A button that triggers a JavaScript function to copy a link to the clipboard.
- `<button id="goback"><a href="../index.php">Go to home page</a></button>`: A button with a link to navigate back to the home page.

`<script src="script11.js"></script>`

Links to an external JavaScript file that contains the functionality for copying the link to the clipboard.

---

## CSS Code Documentation

### Overview

This CSS file styles the HTML document, including the body, content container, map container, and buttons.

### CSS Styles

`body`

- Sets the font to Arial, sans-serif.
- Centers the content both vertically and horizontally using flexbox.
- Sets the height to 100% of the viewport and removes default margin.
- Applies a light gray background color.

`.content`

- Centers the text within the content container.

`mapContainer`

- Adds a bottom margin, border, border-radius, and overflow properties to the map container.
- Sets a fixed width and height for the map container.
- Adds a box-shadow for a subtle 3D effect.

`button`

- Sets padding, background color, text color, border properties, border-radius, and cursor style for buttons.
- Changes the background color on hover.

`goback a`

- Sets the text color to white and removes text decoration for the link inside the "Go to home page" button.

---

JavaScript Code Documentation

Overview

This JavaScript file adds functionality to copy a predefined link to the clipboard when the "Click to copy link" button is pressed.

## JavaScript Functionality

### Event Listener for `copyLink` Button

- Selects the button with the ID `copyLink` and adds a click event listener.
- Defines the link to be copied: `https://maps.app.goo.gl/UBr1LwznqEu6pGmL7`.
- Uses the Clipboard API to write the link to the clipboard.
- Displays an alert on successful copy or logs an error if the copy fails.

## PHP Functionality

- The session is started to retain all the data of the user in the current period of online work,
- The necessary classes are imported , to carry ou their functionality.without being an extra part of the document.

## **Chapter : 6**

**Store the Registration Details of the Renter**

**Source code Documentation**



## Register Renter Details

The Rent Management System supports the preliminary purpose of registering the details of the consumers who have made their mind to have a room on rent. It is very crucial for the business to store the details. The details stored here will be then carried forward for the purpose of Agreement page. This page stores simple details .

The Agreement page will require submission of the crucial documents that include the adhar card , photo of family / photo of main member , signature.

This page will help to store to the database :rent\_management\_system

Folder Structure

Rent Management System

```
| -> reg_renter
 | -> reg_renter.php
```

Here is the documentation for the `reg\_renter.php` file:

---

File Name: `reg\_renter.php`

Purpose: This PHP file handles the registration form submission for renting a room. It collects user input from a form and inserts the data into the `reg\_renter` table in the `rent\_management\_system` database.

Connection to Database:

- Variables: `\$servername`, `\$username`, `password`, `\$database` are used to establish a connection to the MySQL database server.
- Connection Establishment: Uses `mysqli` object-oriented approach to connect to the database.
- Error Handling: Checks for connection errors and terminates execution with an error message if connection fails.

Form Handling:

- Method: Uses `POST` method to submit form data securely.
- Data Retrieval: Retrieves user input from the form fields (`name`, `age`, `gender`, `dreg`, `aadhar`, `mobile`, `address`, `roomNo`, `price`).

- Prepared Statement: Prepares an `INSERT` SQL query to insert data into the `reg\_renter` table.
- Parameter Binding: Binds form data securely to the prepared statement to prevent SQL injection attacks.
- Execution: Executes the prepared statement to insert the data into the database.
- Error Handling: Checks for errors during preparation, binding, and execution of the SQL statement. If an error occurs at any step, an appropriate error message is displayed.

#### Form Interface:

- HTML Form: Presents a structured form interface using HTML and CSS.
- Form Fields: Includes fields for `name`, `age`, `gender`, `dreg` (date of registration), `aadhar` (ID details), `mobile`, `address`, `roomNo` (selected room number), and `price` (negotiated rent price).
- Submission: On form submission, data is sent to `reg\_renter.php`.
- Styling: Uses CSS for layout and styling to enhance user experience.

#### Feedback Message:

- Alert Message: Displays a JavaScript alert message to the user upon successful submission or if an error occurs during database operation.

#### Links:

- Homepage and Logout: Provides links styled as `formlinks` for navigating to the homepage (`..../index.php`) and logging out (`..../logout.php`).

#### Error Handling and Security Measures:

- Prepared Statements: Uses prepared statements and parameter binding to prevent SQL injection attacks.
- Connection Checks: Ensures robust error handling for database connection issues.
- Feedback: Provides clear feedback to the user through alert messages for successful operations or encountered errors.

---

This documentation outlines the functionality, structure, and security measures employed in `reg\_renter.php` for handling room rental registrations.

Chapter : 7

Agreements Page

Source Code Documentation



## **Signing Agreement with the Tenant**

The Rent Management System automates the process of recording the details necessary for the verification purposes , before signing the contract. The Rental Business for rooms and accomodations has to comply with the legal regulations.

This is an electronic form that is responsible to collect and house the details , and produce a webpage that is printable to show that the agreement was signed between the consumer and the landlord.

Folder Structure

Rent Management System

```
| -> rent_agr
 | -> stores (folder that stores all the files of respective entries)
 | ->ag_resc.php
 | ->process_agreement.php
 | ->print_agr.php
```

### **1. ag\_resc.php**

**Purpose:** This PHP script fetches and displays tenant details and uploaded files associated with a registration ID from a rental management system.

**Features:**

- Retrieves tenant details (name, age, gender, contact information, etc.) from the `reg_renter` database table.
- Displays uploaded files (photo, signature, Aadhar card) stored in the `entries` database table.
- Provides a user-friendly interface to view and print tenant information and associated files.

---

### **2. process\_agreement.php**

**Purpose:** This PHP script processes and stores uploaded files (photo, signature, Aadhar card) along with registration details in a rental management system.

**Features:**

- Handles file uploads securely, storing them in designated folders on the server.

- Inserts file paths and registration details into the `entries` database table.
  - Provides feedback to the user on successful or failed file uploads via JSON responses.
- 

### **3. `print_agr.php`**

**Purpose:** This PHP script displays tenant details and uploaded files associated with a registration ID from a rental management system, formatted for printing.

#### **Features:**

- Retrieves and displays tenant details (name, age, gender, contact information, etc.) from the `reg_renter` database table.
- Displays uploaded files (photo, signature, Aadhar card) stored in the `entries` database table.
- Provides a print-friendly layout with an option to directly print the displayed information.

**Chapter : 8**

**View Renters**

**Source Code Documentation**



## Viewing Renters

The Rent Management System will show the registered renters , whom the agreement was signed with. Their Data is going to be displayed on the screen

Folder Structure

Rent Management System

| -> rentr\_list

| -> rentr\_list.php (file that retrieves data to show on the page)

Documentation ( rentr\_list.php)

Renter List Page

PHP Script

The PHP script begins by initializing a session and including necessary files for database connection and user authentication.

```
```php
<?php
session_start();
include("../connection.php"); // Includes database connection script
include("../functions.php"); // Includes user authentication functions
$user_data = check_login($con); // Checks if user is logged in and retrieves user data
```

```

Next, the script sets up parameters for establishing a MySQL database connection:

```
```php
// Database connection parameters
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "rent_management_system";
```

```

// Create MySQL connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check if connection is successful
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
```

```

After successfully connecting to the database, it fetches data from the `reg\_renter` table to display in the HTML table on the web page:

```

```php
// Fetch data from the reg_renter table
$sql = "SELECT id, name, age, gender, datereg, aadhar, mobile, address, roomNo, price
FROM reg_renter";
$result = $conn->query($sql);
?>
```

```

## HTML and CSS

The HTML section defines the structure of the web page and includes CSS for styling:

```

```html
<!DOCTYPE html>
<html>
<head>
    <title>Renter List</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: f2f2f2;
        }
    </style>
</head>
<body>
    <h1>Renter List</h1>
    <table border="1">
        <thead>
            <tr>
                <th>ID</th>
                <th>Name</th>
                <th>Age</th>
                <th>Gender</th>
                <th>Date of Registration</th>
                <th>Aadhar</th>
                <th>Mobile</th>
                <th>Address</th>
                <th>Room No.</th>
                <th>Price</th>
            </tr>
        <tbody>
            <tr>
                <td>1</td>
                <td>John Doe</td>
                <td>30</td>
                <td>Male</td>
                <td>2023-01-01</td>
                <td>1234567890</td>
                <td>9876543210</td>
                <td>123 Main St</td>
                <td>101</td>
                <td>$1000</td>
            </tr>
            <tr>
                <td>2</td>
                <td>Jane Doe</td>
                <td>28</td>
                <td>Female</td>
                <td>2023-01-01</td>
                <td>1234567890</td>
                <td>9876543210</td>
                <td>123 Main St</td>
                <td>102</td>
                <td>$1000</td>
            </tr>
        </tbody>
    </table>
</body>
</html>
```

```

```

margin: 0;
padding: 0;
}

.container {
 width: 80%;
 margin: auto;
 overflow: hidden;
}

.print-button {
 background-color: #008CBA;
 color: white;
 border: none;
 padding: 10px 20px;
 cursor: pointer;
 margin-bottom: 20px;
}

/* Additional CSS styles for table, buttons, and popup panel */
/* ... */

</style>

</head>
<body>

<!-- HTML structure for displaying renter list -->
<div class="container">
 <h1>Renter List</h1>
 <button class="print-button" onclick="printRecords()">Print All Records</button>
 <table>
 <tr>
 <th>Name</th>
 <th>Room No.</th>
 <th>Mobile</th>
 <th>Price</th>
 <th>Action</th>
 </tr>
</pre>

```

```

</tr>
<?php
// PHP code to dynamically populate table rows with fetched data
if ($result->num_rows > 0) {
 while($row = $result->fetch_assoc()) {
 echo "<tr>";
 echo "<td>" . $row["name"] . "</td>";
 echo "<td>" . $row["roomNo"] . "</td>";
 echo "<td>" . $row["mobile"] . "</td>";
 echo "<td>" . $row["price"] . "</td>";
 echo "<td><button class='details-button' onclick='showDetails(" .
json_encode($row) . ")'>View Details</button></td>";
 echo "</tr>";
 }
} else {
 echo "<tr><td colspan='5'>No records found</td></tr>";
}
?>
</table>
</div>

<!-- HTML structure for popup panel displaying details -->
<div class="popup-panel" id="popupPanel">
 <button class="close-button" onclick="closeDetails()">Close</button>
 <div id="detailsContent"></div>
</div>

<!-- JavaScript section for handling user interactions -->
<script>
 // JavaScript functions to show details and handle printing
 /* ... */
</script>

```

```
<!-- Link to return to main page -->
<h2>Go to Main Page</h2>
</body>
</html>
```

```
<?php
// Close database connection
$conn->close();
?>
```

```

JavaScript

The JavaScript section provides functionality to show details in a popup panel, close the panel, and print the page:

```
```javascript
<script>

function showDetails(details) {
 var popupPanel = document.getElementById("popupPanel");
 var detailsContent = document.getElementById("detailsContent");

 // Populate details dynamically in the popup panel
 detailsContent.innerHTML = `

 <p>Room No: ${details.roomNo}</p>
 <p>ID: ${details.id}</p>
 <p>Address: ${details.address}</p>
 `;

 // Display the popup panel
 popupPanel.style.display = "block";
}
```

```
function closeDetails() {
 var popupPanel = document.getElementById("popupPanel");
 // Close the popup panel
 popupPanel.style.display = "none";
}

function printRecords() {
 // Function to print the current page
 window.print();
}
</script>
```

## **Chapter : 9**

### **Create Bills**

**Source Code Documentation**



## Creating Bills

The Rent Management System will provide the feature to the landlord to make the bills for the Renters of the property . The simple interface provides the ability to create bill , save the details to the database and moreover making the ability to make the pdf for the bill that can be saved for future references as well as a medium of communication that conveys the Bill information to the renters.

### Rent Management System

```
| > make_bill
| | > make_bill.php (file with form to get the details)
| | > bill_logic.php (file that transfers the details to the database)
```

#### Documentation (make\_bill.php)

Certainly! Here's a comprehensive documentation explaining the purpose of each line in the provided PHP and HTML code:

#### Room Rent Bill Page

##### PHP Script

The PHP script initiates a session, includes necessary files for database connection and user authentication, and checks if the user is logged in.

```
```php  
<?php  
session_start(); // Starts a new session or resumes the existing session  
include("../connection.php"); // Includes script for database connection  
include("../functions.php"); // Includes script for user authentication functions  
$user_data = check_login($con); // Checks if user is logged in and retrieves user data  
?>  
```
```

##### HTML and CSS

The HTML section defines the structure of the web page and includes CSS for styling:

```

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF8">
    <meta name="viewport" content="width=devicewidth, initialscale=1.0">
    <title>Room Rent Bill</title>
    <style>
        /* CSS styles for body, form, containers, labels, inputs, buttons, etc. */
        /* ... */
    </style>
</head>
<body>
    <! HTML structure for displaying room rent bill form >
    <div class="container">
        <h1>Room Rent Bill</h1>
        <form method="post" action="bill_logic.php">
            <! Form fields for tenant selection, room number display, month, year, due date, rent,
            electric bill, advance paid, dues, miscellaneous charges, total amount payable >
            <?php
                // PHP code for database connection and fetching tenant data for dropdown
            ?>
        </form>
    </div>

    <! JavaScript section for handling dynamic behavior >
    <script>
        // JavaScript functions for fetching room number, calculating electric bill, adding rows
        // for miscellaneous charges, calculating total amount, and printing the bill
        /* ... */
    </script>

```

```
<! Link to return to main page >  
<h2><a href="../index.php">Go to Main Page</a></h2>  
</body>  
</html>  
```
```

### PHP Code Explanation

session\_start(): Initializes a session to persist data across multiple pages or requests.

include("../connection.php"): Includes a PHP file (`connection.php`) that establishes a connection to the database.

include("../functions.php"): Includes a PHP file (`functions.php`) that contains functions for user authentication.

\$user\_data=check\_login(\$con): Calls a function `check\_login()` from `functions.php` to verify if the user is logged in and retrieves user data if authenticated.

### HTML and CSS Explanation

HTML Structure: Defines a form (`<form>`) with various input fields for capturing tenant information and bill details.

CSS Styling: Styles the elements (`body`, `container`, `h1`, `formgroup`, `label`, `input`, `printbutton`, etc.) to ensure a consistent and visually appealing layout.

### JavaScript Explanation

fetchRoomNo(): Retrieves and displays the room number based on the selected tenant.

calculateElectricBill(): Calculates the electric bill based on units used and updates the total amount.

addRow(): Adds a new row for entering miscellaneous charges dynamically.

calculateTotal(): Calculates the total amount payable considering rent, electric bill, advance paid, dues, and miscellaneous charges.

printBill(): Initiates the printing of the bill when the "Print Bill" button is clicked.

### Conclusion

This documentation provides a clear understanding of how each part of the PHP, HTML, CSS, and JavaScript code contributes to creating a room rent bill page. Adjustments can be made based on specific requirements or enhancements needed for functionality or design.

This documentation should help in understanding and maintaining the code effectively for managing room rent bills in a web application.

### Documentation (bill\_logic.php)

Certainly! Here's the detailed documentation explaining the purpose of each line in the provided PHP code:

---

#### ### Room Rent Bill Logic (bill\_logic.php)

##### #### PHP Script

The PHP script initiates a session, includes necessary files for database connection and user authentication, and checks if the user is logged in.

```
```php
<?php
session_start(); // Starts a new session or resumes the existing session
include("../connection.php"); // Includes script for database connection
include("../functions.php"); // Includes script for user authentication functions
$user_data = check_login($con); // Checks if user is logged in and retrieves user data
?>
````
```

##### #### Establish Database Connection

```
```php
// Database connection details
$servername = "localhost";
$username = "root";
```

```

$password = "";
$dbname = "rent_management_system";

// Create connection to MySQL database
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error); // Terminates script execution and
    displays error message if connection fails
}

```
Process Form Data (POST Request)
```php
// Process form data only if the request method is POST
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Retrieve form data
    $renter_id = $_POST['renter'];
    $month = $_POST['month'];
    $year = $_POST['year'];
    $due_date = $_POST['due_date'];
    $room_rent = $_POST['room_rent'];
    $units_used = $_POST['units_used'];
    $electric_bill = $_POST['electric_bill'];
    $advance_paid = $_POST['advance_paid'];
    $amount_dues = $_POST['amount_dues'];
    $miscellaneous = $_POST['miscellaneous'];
    $total_amount = $_POST['total_amount'];

    // SQL query to insert data into bill_details table
}
```

```

```

$sql = "INSERT INTO bill_details (renter_id, month, year, due_date, room_rent,
units_used, electric_bill, advance_paid, amount_dues, miscellaneous, total_amount)

VALUES ('$renter_id', '$month', '$year', '$due_date', '$room_rent', '$units_used',
'$electric_bill', '$advance_paid', '$amount_dues', "" . json_encode($miscellaneous) . "",
'$total_amount')";

// Execute SQL query and handle success or error
if ($conn->query($sql) === TRUE) {

 echo "Bill details saved successfully." // Display success message if insertion is
successful

} else {

 echo "Error: " . $sql . "
" . $conn->error; // Display error message if insertion fails
}

}
```

```

Close Database Connection

```

```php
$conn->close(); // Close the database connection
?>
```

```

Explanation

- `session_start()`: Initializes a session to persist data across multiple pages or requests.
- `include("../connection.php")`: Includes a PHP file (`connection.php`) that establishes a connection to the database.
- `include("../functions.php")`: Includes a PHP file (`functions.php`) that contains functions for user authentication.
- `$user_data = check_login($con)`: Calls a function `check_login()` from `functions.php` to verify if the user is logged in and retrieves user data if authenticated.
- `$servername, $username, $password, $dbname`: Defines variables for MySQL database connection details.
- `$conn = new mysqli($servername, $username, $password, $dbname)`: Creates a new MySQLi connection object to connect to the database.

- if (\$conn->connect_error): Checks if the database connection is successful; terminates script execution and displays an error message if connection fails.
- \$_SERVER["REQUEST_METHOD"] == "POST": Checks if the current HTTP request method is POST to process form data.
- \$_POST['...']: Retrieves form data submitted via POST method and assigns them to respective variables (`\$renter_id`, ` `\$month` , ` `\$year` , etc.).
- \$sql: Constructs an SQL INSERT statement to insert form data into the `bill_details` table.
- \$conn->query(\$sql): Executes the SQL query; displays success message if insertion is successful, otherwise displays an error message.
- \$conn->close(): Closes the database connection after processing the form data.

This documentation provides a clear understanding of how each part of the PHP script contributes to processing room rent bill data and interacting with the database. Adjustments can be made based on specific requirements or enhancements needed for functionality or error handling.

Chapter : 10

Search Bar Functionality

Source Code Documentation

Search Bar

The Rent Management System will provide the feature to the landlord to swiftly search the tenant records and details by entering either the room id or the name of the tenant.

Rent Management System

```
| -> index.php  
| -> searchAction.php
```

Sure, here's a detailed line-by-line explanation for both the HTML form and the `searchAction.php` file.

Explanation of `searchForm.html`

```
```html
```

```
<form action="searchAction.php" method="GET" id="form1"
style="width:75px;height:10px; padding-top:none;">
```

```
```
```

- `<form>`: Creates an HTML form element.
- `action="searchAction.php"`: Specifies the URL to which the form data will be sent upon submission.
- `method="GET"`: Specifies the HTTP method to use when sending form data. `GET` appends form data to the URL.
- `id="form1"`: Assigns an ID to the form for styling and JavaScript purposes.
- `style="width:75px;height:10px; padding-top:none;"`: Inline CSS to style the form.

```
```html
```

```
<input type="text" class="nav-search-input" name="query" placeholder="Search @ Rent
Management System" style="width:310px; height:16px;">
```

```
```
```

- `<input type="text">`: Creates a single-line text input field.
- `class="nav-search-input"`: Assigns a CSS class to the input field for styling.
- `name="query"`: Assigns a name to the input field, which will be used as a key in the URL parameters.

- `placeholder="Search @ Rent Management System"`: Displays placeholder text inside the input field.
- `style="width:310px; height:16px;"`: Inline CSS to style the input field.

```html

```
<button type="submit" style="position:relative; top:-37px; left:549px;">
```

```

- `<button type="submit">`: Creates a submit button that submits the form when clicked.
- `style="position:relative; top:-37px; left:549px;"`: Inline CSS to position the button.

```html

```

```

```

- ``: Embeds an image inside the button.
- `src="./assets/search_icon.png"`: Specifies the path to the image file.
- `class="nav-search-icon"`: Assigns a CSS class to the image for styling.
- `alt=""`: Provides alternative text for the image.

```html

```
</button>
```

```
</form>
```

```

- `</button>`: Closes the button element.
- `</form>`: Closes the form element.

Explanation of `searchAction.php`

```php

```
<?php
session_start();
include("connection.php");
include("functions.php");
```

```

- `<?php`: Starts a PHP script.
- `session_start();`: Starts a new session or resumes an existing session.
- `include("connection.php");`: Includes the `connection.php` file, which presumably contains code to establish a database connection.
- `include("functions.php");`: Includes the `functions.php` file, which likely contains various utility functions, including the `check_login()` function used below.

```php

```
$user_data = check_login($con);
```

```

- `check_login(\$con);`: Calls the `check_login()` function, passing the database connection `\$con` as an argument. This function likely checks if the user is logged in and returns user data if they are.

```php

```
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "rent_management_system";
```

```

- Defines variables for the database connection parameters:
- `\$servername`: The database server address.
- `\$username`: The database username.
- `\$password`: The database password.
- `\$dbname`: The name of the database.

```php

```
$conn = new mysqli($servername, $username, $password, $dbname);
```

```

- `new mysqli(\$servername, \$username, \$password, \$dbname);`: Creates a new MySQLi object and establishes a connection to the database using the specified parameters.

```
```php
if ($conn->connect_error) {
 die("Connection failed: " . $conn->connect_error);
}
```

```

- `if (\$conn->connect_error)` : Checks if there was an error connecting to the database.
- `die("Connection failed: " . \$conn->connect_error);` : If there was an error, the script terminates and displays an error message.

```
```php
$searchQuery = isset($_GET['query']) ? $_GET['query'] : '';
```

```

- `isset(\$_GET['query'])` : Checks if the `query` parameter is set in the URL.
- `\$_GET['query']` : Retrieves the value of the `query` parameter from the URL.
- `? \$_GET['query'] : ""` : If the `query` parameter is set, assigns its value to `\$searchQuery`. Otherwise, assigns an empty string to `\$searchQuery`.

```
```php
$searchQuery = $conn->real_escape_string($searchQuery);
```

```

- `{\$conn->real_escape_string(\$searchQuery);}` : Escapes special characters in the search query to prevent SQL injection.

```
```php
$sql = "SELECT * FROM reg_renter WHERE roomNo = '$searchQuery' OR name LIKE
'%$searchQuery%'";
```

```

- `{\$sql = "SELECT * FROM reg_renter WHERE roomNo = '\$searchQuery' OR name LIKE
'%\$searchQuery%'";}` : Defines an SQL query to select all columns from the `reg_renter` table where the `roomNo` matches the search query exactly or the `name` contains the search query as a substring.

```
```php
$result = $conn->query($sql);

```

```

- `'\$conn->query(\$sql);`: Executes the SQL query and stores the result set in `\$result`.

```php

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Search Results</title>

```

- `?>`: Ends the PHP script and resumes HTML.

- `<!DOCTYPE html>`: Defines the document type as HTML5.

- `<html lang="en">`: Opens the HTML document and sets the language to English.

- `<head>`: Opens the head section of the HTML document.

- `<meta charset="UTF-8">`: Sets the character encoding to UTF-8.

- `<meta name="viewport" content="width=device-width, initial-scale=1.0">`: Sets the viewport properties for responsive design.

- `<title>Search Results</title>`: Sets the title of the HTML document.

```html

<style>

body {

font-family: Arial, sans-serif;

background-color: #f2f2f2;

padding: 20px;

}

table {

width: 100%;

border-collapse: collapse;

margin-top: 20px;

```

 }
 table, th, td {
 border: 1px solid #ddd;
 }
 th, td {
 padding: 10px;
 text-align: left;
 }
 th {
 background-color: #007bff;
 color: white;
 }
</style>
</head>
```

```

- `<style>`: Opens a style block for custom CSS.
- The following CSS styles:
 - `body`: Sets the font, background color, and padding for the body element.
 - `table`: Sets the width, border, and margin for table elements.
 - `table, th, td`: Adds a border to table, th, and td elements.
 - `th, td`: Sets padding and text alignment for table header and data cells.
 - `th`: Sets the background color and text color for table header cells.
- `</head>`: Closes the head section of the HTML document.

```
```html
```

```
<body>
<h1>Search Results</h1>
```

```
```

```

- `<body>`: Opens the body section of the HTML document.
- `<h1>Search Results</h1>`: Displays a heading with the text "Search Results".

```
```php
```

```
<?php if ($result->num_rows > 0): ?>
```

```
````
```

- `<?php if (\$result->num_rows > 0): ?>`: Checks if the result set contains any rows. If so, the following HTML block will be displayed.

```
````html
```

```
 <table>
```

```
 <tr>
```

```
 <th>Serial No</th>
```

```
 <th>Date Registered</th>
```

```
 <th>Name</th>
```

```
 <th>Room No</th>
```

```
 <th>Price</th>
```

```
 </tr>
```

```
````
```

- `<table>`: Opens a table element.

- `<tr>`: Opens a table row.

- `<th>Serial No</th>`: Table header cell for the serial number.

- `<th>Date Registered</th>`: Table header cell for the date registered.

- `<th>Name</th>`: Table header cell for the name.

- `<th>Room No</th>`: Table header cell for the room number.

- `<th>Price</th>`: Table header cell for the price.

```
````php
```

```
 <?php
```

```
 $serialNo = 1;
```

```
 while($row = $result->fetch_assoc()):
```

```
 ?>
```

```
 <tr>
```

```
 <td><?php echo $serialNo++; ?></td>
```

```
 <td><?php echo $row['datereg']; ?></td>
```

```
 <td><?php echo $row['name']; ?></td>
```

```

<td><?php echo $row['roomNo']; ?></td>
<td><?php echo $row['price']; ?></td>
</tr>
<?php endwhile; ?>
</table>
<?php else: ?>
```

```

- `<?php \$serialNo = 1;`: Initializes a variable to

keep track of the serial number.

- `while(\$row = \$result->fetch_assoc()):`: Iterates through each row in the result set.
- `<tr>`: Opens a new table row for each record.
- `<td><?php echo \$serialNo++; ?></td>`: Displays the serial number and increments it.
- `<td><?php echo \$row['datereg']; ?></td>`: Displays the date registered.
- `<td><?php echo \$row['name']; ?></td>`: Displays the name.
- `<td><?php echo \$row['roomNo']; ?></td>`: Displays the room number.
- `<td><?php echo \$row['price']; ?></td>`: Displays the price.
- `</tr>`: Closes the table row.
- `<?php endwhile; ?>`: Ends the loop.
- `</table>`: Closes the table element.
- `<?php else: ?>`: If no rows are found, the following HTML block will be displayed.

```html

```

<p>No results found for "<?php echo htmlspecialchars($searchQuery); ?>"</p>
<?php endif; ?>
```

```

- `<p>No results found for "<?php echo htmlspecialchars(\$searchQuery); ?>"</p>`: Displays a message if no results are found. The `htmlspecialchars(\$searchQuery)` function ensures that any special characters in the search query are properly escaped.

- `<?php endif; ?>`: Ends the if statement.

```php

```
<?php $conn->close(); ?>
</body>
</html>
```


- `<?php $conn->close(); ?>`: Closes the database connection.
- `</body>`: Closes the body section of the HTML document.
- `</html>`: Closes the HTML document.

```

Chapter : 11

Other Functions

Source Code Documentation

Other Functions

The Rent Management System will provide some necessary features , like the details that are not used much frequent but serve to be a necessary component of the Software.

These pages are : the IT Rules followed , Developer Contacts , Landlord Details , Reset Password and the Logout page. These are the supplementary components.

The Reset Password is the Ch 4 . of the Source code and the documentation.

Rent Management System

| -> other_functions

| -> othr_func_resc (folder that is containing the image files)

| -> dev_details.php (developer details)

| -> it_rules.php (IT Rules kept in mind)

| -> ll_regist.php (Landlord's Business Registration Details)

| -> logout.php (page to log out of the system)

Documentation (dev_details.php)

Certainly! Here is the detailed documentation for each line of the provided PHP and HTML code:

Explanation of the PHP Code

```
```php
<?php
session_start();
include("../connection.php");
include("../functions.php");
$user_data = check_login($con);
?>
```

```

- `<?php`: Starts a PHP script.

- `session_start();`: Starts a new session or resumes an existing session. This is necessary for managing user login sessions.
- `include("../connection.php");`: Includes the `connection.php` file located in the parent directory. This file likely contains code to establish a database connection.
- `include("../functions.php");`: Includes the `functions.php` file located in the parent directory. This file likely contains utility functions, including the `check_login()` function used below.
- `\$user_data = check_login(\$con);`: Calls the `check_login()` function, passing the database connection `\$con` as an argument. This function checks if the user is logged in and returns user data if they are. The returned data is stored in the `\$user_data` variable.

Explanation of the HTML and CSS Code

```html

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Developer Details</title>
 <style>
```

```

- `<!DOCTYPE html>`: Defines the document type as HTML5.
- `<html lang="en">`: Opens the HTML document and sets the language to English.
- `<head>`: Opens the head section of the HTML document.
- `<meta charset="UTF-8">`: Sets the character encoding to UTF-8.
- `<meta name="viewport" content="width=device-width, initial-scale=1.0">`: Sets the viewport properties for responsive design.
- `<title>Developer Details</title>`: Sets the title of the HTML document, which appears in the browser tab.
- `<style>`: Opens a style block for custom CSS.

```css

```
body {
```

```

font-family: Arial, sans-serif;
margin: 0;
padding: 0;
background-color: #f2f2f2;
display: flex;
justify-content: center;
align-items: center;
height: 100vh;
}

```
- `body`: Styles for the body element:
  - `font-family: Arial, sans-serif;`: Sets the font family to Arial or sans-serif.
  - `margin: 0;`: Removes default margin.
  - `padding: 0;`: Removes default padding.
  - `background-color: #f2f2f2;`: Sets the background color to a light grey.
  - `display: flex;`: Uses Flexbox for layout.
  - `justify-content: center;`: Centers content horizontally.
  - `align-items: center;`: Centers content vertically.
  - `height: 100vh;`: Sets the height to 100% of the viewport height.

```

```css

```

.container {
 background-color: #fff;
 padding: 20px;
 border-radius: 10px;
 box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
 max-width: 600px;
 text-align: center;
}

```
- `.container`: Styles for the container element:
  - `background-color: #fff;`: Sets the background color to white.

```

- `padding: 20px;`: Adds padding inside the container.
- `border-radius: 10px;`: Rounds the corners of the container.
- `box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);`: Adds a subtle shadow around the container.
- `max-width: 600px;`: Sets a maximum width for the container.
- `text-align: center;`: Centers the text inside the container.

```css

```
.container img {
 border-radius: 10px;
}
```

```

- `.container img`: Styles for images inside the container:
- `border-radius: 10px;`: Rounds the corners of the images.

```css

```
.details {
 margin-top: 20px;
}
```

```

- `.details`: Styles for the details section:
- `margin-top: 20px;`: Adds a margin at the top of the details section.

```css

```
.details h1 {
 margin: 0;
 font-size: 24px;
 color: #333;
}
```

```

- `.details h1`: Styles for the heading inside the details section:
- `margin: 0;`: Removes default margin.
- `font-size: 24px;`: Sets the font size to 24 pixels.

- `color: #333;`: Sets the text color to a dark grey.

```css

```
.details p {
 margin: 5px 0;
 font-size: 18px;
 color: #555;
}
```

```

- `.`details p`: Styles for paragraphs inside the details section:

- `margin: 5px 0;`: Sets the margin to 5 pixels on top and bottom.
- `font-size: 18px;`: Sets the font size to 18 pixels.
- `color: #555;`: Sets the text color to a medium grey.

```css

```
.print-button {
 margin-top: 20px;
 padding: 10px 20px;
 font-size: 16px;
 color: #fff;
 background-color: #007bff;
 border: none;
 border-radius: 5px;
 cursor: pointer;
}
```

```

- `.`print-button`: Styles for the print button:

- `margin-top: 20px;`: Adds a margin at the top.
- `padding: 10px 20px;`: Adds padding inside the button.
- `font-size: 16px;`: Sets the font size to 16 pixels.
- `color: #fff;`: Sets the text color to white.
- `background-color: #007bff;`: Sets the background color to blue.

- `border: none;`: Removes the default border.
- `border-radius: 5px;`: Rounds the corners of the button.
- `cursor: pointer;`: Changes the cursor to a pointer when hovering over the button.

```css

```
.print-button:hover {
 background-color: #0056b3;
}
```

```

- `.`.print-button:hover` : Styles for the print button when hovered:
- `background-color: #0056b3;` : Changes the background color to a darker blue.

```css

```
#idc {
 height: 315px;
}
```

</style>

</head>

<body>

```

- `#idc` : Styles for the element with ID `idc`:
- `height: 315px;` : Sets the height to 315 pixels.
- `</style>` : Closes the style block.
- `</head>` : Closes the head section of the HTML document.
- `<body>` : Opens the body section of the HTML document.

```html

```
<div class="container">

</div>
```

- `<div class="container">` : Creates a div element with the class `container`.

- ``: Embeds an image with the specified source and alternative text.

```
```html
```

```
<div class="details">  
    <h1>Aayush Sahay</h1>  
    <p>Student of: Indian Institute of Business Management, Patna</p>  
    <p>Course: BCA</p>  
    <p>Roll No.: AKU/31</p>  
    <p>Session: 2021-2024</p>  
    <p>Enrollment No.: 21303334049</p>  
</div>
```

```
```
```

- `<div class="details">`: Creates a div element with the class `details`.
- `<h1>Aayush Sahay</h1>`: Displays the developer's name in a heading.
- `<p>Student of: Indian Institute of Business Management, Patna</p>`: Displays the developer's institution.
- `<p>Course: BCA</p>`: Displays the developer's course.
- `<p>Roll No.: AKU/31</p>`: Displays the developer's roll number.
- `<p>Session: 2021-2024</p>`: Displays the session years.
- `<p>Enrollment No.: 21303334049</p>`: Displays the developer's enrollment number.

```
```html
```

```
<br>
```

```
```
```

- ``: Embeds an image with the specified source, alternative text, and ID `idc`.
- `<br>`: Inserts a line break.

```
```html
```

```
<button class="print-button" onclick="window.print()">Print Details</button>  
</div>  
</body>
```

```

</html>
```
- `<button class="print-button" onclick="window.print()">Print Details</button>`: Creates a button with the class `print-button` that, when clicked, triggers the browser's print function.
- `</div>`:

```

## Documentation (it\_rules.php)

### ### Explanation of the PHP Code

```

```php
<?php
session_start();
include("../connection.php");
include("../functions.php");
$user_data = check_login($con);
?>
```

```

- `<?php`: Starts a PHP script.
- `session\_start();`: Starts a new session or resumes an existing session. This is necessary for managing user login sessions.
- `include("../connection.php");`: Includes the `connection.php` file located in the parent directory. This file likely contains code to establish a database connection.
- `include("../functions.php");`: Includes the `functions.php` file located in the parent directory. This file likely contains utility functions, including the `check\_login()` function used below.
- `'\$user\_data = check\_login(\$con);'`: Calls the `check\_login()` function, passing the database connection `\$con` as an argument. This function checks if the user is logged in and returns user data if they are. The returned data is stored in the `\$user\_data` variable.

### ### Explanation of the HTML and CSS Code

```

```html
<!DOCTYPE html>
<html lang="en">
<head>

```

```

<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>IT Rules</title>
<style>
```
- `<!DOCTYPE html>`: Defines the document type as HTML5.
- `<html lang="en">`: Opens the HTML document and sets the language to English.
- `<head>`: Opens the head section of the HTML document.
- `<meta charset="UTF-8">`: Sets the character encoding to UTF-8.
- `<meta name="viewport" content="width=device-width, initial-scale=1.0">`: Sets the viewport properties for responsive design.
- `<title>IT Rules</title>`: Sets the title of the HTML document, which appears in the browser tab.
- `<style>`: Opens a style block for custom CSS.

```

```

```css
body {
    font-family: Arial, sans-serif;
    line-height: 1.6;
    margin: 0;
    padding: 0;
    background-color: #f4f4f4;
    color: #333;
}
```

```

```

- `body`: Styles for the body element:
- `font-family: Arial, sans-serif;`: Sets the font family to Arial or sans-serif.
- `line-height: 1.6;`: Sets the line height to 1.6 for better readability.
- `margin: 0;`: Removes default margin.
- `padding: 0;`: Removes default padding.
- `background-color: #f4f4f4;`: Sets the background color to a light grey.
- `color: #333;`: Sets the text color to dark grey.

```

```
```css
```

```
.button {  
    height: 40px;  
    background: #50b3a2;  
    border: none;  
    padding: 0 20px;  
    color: #fff;  
    text-transform: uppercase;  
    font-size: 18px;  
    cursor: pointer;  
    border-radius: 5px;  
}
```

```
```
```

- ` `.button` : Styles for the button element:

- `height: 40px;` : Sets the height to 40 pixels.
- `background: #50b3a2;` : Sets the background color to teal.
- `border: none;` : Removes the default border.
- `padding: 0 20px;` : Adds padding of 20 pixels on the left and right.
- `color: #fff;` : Sets the text color to white.
- `text-transform: uppercase;` : Transforms text to uppercase.
- `font-size: 18px;` : Sets the font size to 18 pixels.
- `cursor: pointer;` : Changes the cursor to a pointer when hovering over the button.
- `border-radius: 5px;` : Rounds the corners of the button.

```
```css
```

```
.content {  
    padding: 20px;  
    background: #fff;  
    border-radius: 5px;  
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);  
    margin-top: 20px;
```

```
}
```

```
````
```

- ` `.content` : Styles for the content element:

- ` padding: 20px;` : Adds padding inside the content.
- ` background: #fff;` : Sets the background color to white.
- ` border-radius: 5px;` : Rounds the corners of the content.
- ` box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);` : Adds a subtle shadow around the content.
- ` margin-top: 20px;` : Adds a margin at the top.

```
````css
```

```
.content h2 {  
    border-bottom: 2px solid #e8491d;  
    padding-bottom: 10px;  
}
```

```
````
```

- ` `.content h2` : Styles for `h2` headings inside the content element:

- ` border-bottom: 2px solid #e8491d;` : Adds a bottom border with a color of orange-red and a thickness of 2 pixels.
- ` padding-bottom: 10px;` : Adds padding at the bottom.

```
````css
```

```
.content ul {  
    padding: 0;  
    list-style: none;  
}
```

```
````
```

- ` `.content ul` : Styles for unordered lists inside the content element:

- ` padding: 0;` : Removes default padding.
- ` list-style: none;` : Removes the default list bullets.

```
````css
```

```
.content li {
```

```
    margin-bottom: 10px;  
    padding-left: 20px;  
    position: relative;  
}  
~~~  
- ` `.content li` : Styles for list items inside the content element:  
- ` margin-bottom: 10px;` : Adds a margin at the bottom.  
- ` padding-left: 20px;` : Adds padding on the left.  
- ` position: relative;` : Sets the position to relative for further positioning.
```

```css

```
.content li:
```

```
before {
 content: "✓";
 position: absolute;
 left: 0;
 color: #50b3a2;
}
```

~~~

```
- ` `.content li:before` : Styles for the `before` pseudo-element of list items inside the content element:
- ` content: "✓";` : Sets the content to a checkmark symbol.
- ` position: absolute;` : Sets the position to absolute.
- ` left: 0;` : Positions the checkmark on the left.
- ` color: #50b3a2;` : Sets the checkmark color to teal.
```

### Explanation of the JavaScript Code

```
```html  
</style>  
</head>
```

```

<body>
  <header>
    <div class="container">
      <div id="branding">
        <h1>IT Rules & Best Practices</h1>
      </div>
    </div>
  </header>
  <div class="container">
    <div class="content">
      <h2>Security Measures</h2>
      <ul>
        <li>Data Validation and Sanitization</li>
        <li>Authentication and Authorization</li>
        <li>Secure Password Storage</li>
        <li>Session Management</li>
        <li>Encryption</li>
        <li>Prevent Cross-Site Scripting (XSS)</li>
      </ul>
    </div>
  </div>

```

Documentation (ll_regist.php)

Uses the Simple CSS and HTML to display the Registration details of the owner of the business

Documentation (logout.php)

Explanation of the PHP Code

```

```php
<?php
session_start();
if(isset($_SESSION['user_id'])){

```

```

unset($_SESSION['user_id']);

}

header("Location: login.php?logout=success");

die;

?>

```

```

- `<?php`: Starts a PHP script.
- `session_start();`: Starts a new session or resumes an existing session. This is necessary to manipulate session variables.
- `if(isset(\$_SESSION['user_id'])){ }`: Checks if the `\$_SESSION['user_id']` variable is set.
- `unset(\$_SESSION['user_id']);`: If `\$_SESSION['user_id']` is set, it unsets (deletes) the `user_id` session variable.
- `header("Location: login.php?logout=success");`: Sends an HTTP header to redirect the user to `login.php` with a query parameter `logout=success`.
- `die;`: Terminates execution of the script immediately after the header redirect.

Explanation of the Code Flow

1. **Session Start**: Starts or resumes a session.
2. **Check Session Variable**: Checks if the `user_id` session variable is set.
 - If set, it deletes (`unset`) the `user_id` session variable using `unset(\$_SESSION['user_id']);`.
3. **Redirect**: Sends a redirect header to `login.php` with `logout=success` appended as a query parameter.
4. **Termination**: Terminates further execution of the script with `die;`.

Purpose of the Code

This PHP script is typically used to log out a user from a session-based login system. Here's how it works in steps:

- **Session Check**: It checks if the `user_id` session variable exists (`isset(\$_SESSION['user_id'])`).

- **Session Unset**: If the `user_id` session variable exists, it is unset (deleted) using `unset(\$_SESSION['user_id'])`. This effectively logs out the user by removing their session data.
- **Redirect**: After unsetting the session variable, the script redirects the user to `login.php` with `logout=success` in the URL query string. This can be used by the `login.php` page to display a logout success message or perform any other relevant action.
- **Script Termination**: The `die;` statement ensures that no further PHP code is executed after the redirect header, preventing unintended output or behavior.

This logout mechanism ensures that the user's session is properly terminated, enhancing security by preventing unauthorized access to protected areas of the application after logout.

Chapter : 12

Payment System

Source Code Documentation

Online Payments Interface

The Rent Management System will provide online payments interface

Rent Management System

```
| -> pay_online  
|   | -> assets_ptm (folder that is containing the image files for the app)  
|   | -> ptm.php (Main payments page)  
|   | -> ptmAction.php (handler of the transactions )  
|   | -> pay_records.php (Landlord's interface to manage cash transactions )  
|   | -> delete_record.php (delete details of specific records (for correction))  
| -> see_records  
|   | -> see_records.php (page to see the transaction details of all kind)
```

The above files are responsible for carrying out the online / offline transactions of the Business. The source code of the respective files are as follows :

Documentation (ptm.php)

This PHP script sets up a web page for a Paytm Payment Gateway clone. Here's a summary of its key components:

1. **PHP Setup**:

- Starts a session and includes necessary PHP files (`connection.php` and `functions.php`).
- Retrieves user data using the `check_login` function.

2. **HTML Structure**:

- Defines a standard HTML5 structure with `<head>` and `<body>` sections.
- Includes meta tags for character set and viewport settings.

3. **Styling**:

- Uses embedded CSS to style the entire page with a clean, modern look.

- Defines styles for headers, navigation, buttons, hero section, forms, features section, and footer.

4. **Header and Navigation**:

- Displays a header with a logo and navigation links.
 - Navigation links include typical sections like Home, Products, Solutions, Pricing, Resources, About Us, Contact Us, Sign Up, and Login.

5. **Main Content**:

- **Hero Section**: Promotes Paytm Payment Gateway with a title, subtitle, and a QR code panel for payment scanning.
- **Payment Form**: Allows users to input payment details (Name, Mobile Number, Renter ID, Payment Amount) and submit them to `ptmAction.php`.

6. **Features Section**:

- Lists reasons to choose Paytm Payment Gateway, highlighting security, ease of integration, and 24/7 support with descriptive icons and text.

7. **Footer**:

- Displays a copyright notice.

8. **JavaScript**:

- Includes a script that listens for key events (`Ctrl + /` and `Ctrl + ;`) and alerts success or failure messages for transactions.

This script essentially creates a web page resembling a Paytm Payment Gateway landing page, complete with styling, navigation, payment form, and feature highlights.

Source code (ptmAction.php)

This PHP script sets up a web page for handling payment transactions and displaying transaction details. Here's a summary of its key components:

1. **PHP Setup**:

- Starts a session and includes necessary PHP files (`connection.php` and `functions.php`).
- Defines a function `generateUPIID()` to generate a random 6-character UPI ID.

2. **HTML Structure**:

- Defines a standard HTML5 structure with `<head>` and `<body>` sections.
- Includes meta tags for character set and viewport settings.

3. **Styling**:

- Uses embedded CSS to style the entire page with a clean, responsive design.
- Defines styles for the body, containers, sections, tables, buttons, etc.

4. **JavaScript**:

- Includes a script to enable printing the page using `window.print()` when the "Produce the Copies" button is clicked.

5. **Main Content**:

- **Container Setup**: Uses a `<div class="container">` to center content and manage width.
- **Payment Transaction Section**:
 - Includes a `<div class="section">` with a title "Payment Transaction Details".
 - Handles form submission (^POST^ method) to capture `name`, `mobile`, `renter_id`, and `amount` from the form.
 - Generates a random `upi_id` using `generateUPIID()`.
 - Retrieves current date and time for `date_pay`.
 - Connects to the database (^rent_management_system^) using `mysqli`.
 - Executes an SQL `INSERT` query to store transaction details into the `payment_status` table.
 - Displays an alert message indicating whether the transaction was successful or failed based on the query execution.
 - Displays transaction details in a formatted table (^<table>^) including Name, Mobile, Renter ID, Amount, UPI ID, and Date & Time.

6. **Form Handling**:

- Extracts form data, generates UPI ID, executes SQL query, and displays transaction details dynamically within the same page upon form submission.

7. **Button**:

- Includes a button styled as `print-button` that triggers the `printPage()` function to print the transaction details section.

This script essentially creates a dynamic web page where users can submit payment transaction details, store them in a database, and view/print transaction details on the same page based on form submission.

Source code (pay_records.php)

Here's a brief documentation of the provided PHP and HTML code:

Overview

This script is designed to manage payment records for a rent management system. It allows users to add new payment records and view existing ones, with functionality to delete records.

PHP Code (Server-side)

1. **Session and Includes**: Starts a session and includes necessary PHP files (`connection.php` and `functions.php`).

2. **Database Connection**: Connects to the MySQL database (`rent_management_system`).

3. **Adding Payment Record**:

- Handles form submission ('POST' method) to add a new payment record to the `payment_status` table.
- Retrieves form data (`date_pay`, `renter_id`, `month`, `amount`, `pay_status`, `description`) and inserts them into the database.
- Uses `mysql` for database operations and checks for successful insertion.

4. **Fetching and Displaying Records**:

- Retrieves all records from the `payment_status` table and displays them in a formatted table (`<table>`).
- Provides an option to delete records using a confirmation prompt that requires a PIN (`55555`).

5. **Error Handling**: Displays error messages if database connection fails or if there are errors during SQL operations.
6. **Automatic Page Refresh**: Utilizes `<meta http-equiv='refresh' content='0'>` to refresh the page after a successful record insertion, ensuring updated display of records.
7. **Security Considerations**: Includes a PIN-based confirmation for record deletion to prevent unauthorized deletions.

HTML and CSS (Client-side)

1. **HTML Structure**:
 - Defines a standard HTML5 structure with `<head>` and `<body>` sections.
 - Includes meta tags for character set and viewport settings.
2. **Styling**:
 - Uses embedded CSS to style the entire page with a clean, responsive design.
 - Defines styles for body, containers, sections, tables, buttons, forms, and input fields.
3. **JavaScript**:
 - Provides client-side functionality for confirming record deletion (`confirmDelete()` function).
 - Uses `window.location.href` to redirect to `delete_record.php` for record deletion.

Functionality

- **Adding Records**: Users can input details such as date of payment, renter ID, month, amount, status, and description to add a new payment record.
- **Viewing Records**: Displays existing payment records in a table format with columns for renter ID, date of payment, month, amount, status, description, and a delete button.
- **Deleting Records**: Provides a delete button for each record, which prompts for a PIN before allowing deletion.

Enhancements

- **Validation**: Implement form validation to ensure all required fields are filled out before submission.

- **Pagination**: Add pagination for better management of large datasets.
- **Sorting and Filtering**: Allow sorting and filtering of records based on different criteria (e.g., date, amount).
- **Security**: Implement more robust security measures, such as parameterized queries, to prevent SQL injection attacks.

This script provides a basic but functional interface for managing payment records within a rent management system, with room for additional features and security improvements.

Source code (delete_record.php)

Here is the brief documentation of the entire PHP script:

Overview

This PHP script facilitates the deletion of payment records from the `payment_status` table in a MySQL database. It checks for the presence of an `id` parameter in the URL, establishes a database connection, executes a deletion query based on the provided `id`, and manages redirection and error handling.

PHP Code (Server-side)

1. **Session Start and Includes**:

- Initiates a PHP session to manage user authentication and state.
- Includes necessary files (`./connection.php` and `./functions.php`) for database connectivity and user authentication (`check_login` function).

2. **ID Check and Database Connection**:

- Verifies if the `id` parameter is present in the URL using the `GET` method.
- Creates a MySQL database connection using `mysqli` for database operations.

3. **Deletion Process**:

- Constructs an SQL query (`DELETE FROM payment_status WHERE id = \$id`) to remove a specific record from the `payment_status` table based on the provided `id`.
- Executes the SQL query using `\$conn->query(\$sql)` to delete the record.
- If successful, redirects the user to `pay_records.php` using `header("Location: pay_records.php")`.

- If the deletion query fails, an error message displaying the specific error (`\$conn->error`) is echoed.

4. **Error Handling and Cleanup**:

- Closes the database connection (`\$conn->close()`) after executing the deletion query.
- If no `id` parameter is found in the URL, the script redirects back to `pay_records.php` to prevent unauthorized or accidental access.

Purpose

This script ensures the secure and controlled deletion of payment records from the database, adhering to best practices in database management. It protects against unauthorized deletion attempts by verifying the presence of the `id` parameter. Proper error handling ensures that users are informed of any issues encountered during the deletion process, maintaining data integrity and user experience.

Source code (see_records.php)

Here's the brief documentation of the provided PHP script:

Overview

This PHP script generates an HTML page (`see_records.php`) that displays bill details and payment records from the `rent_management_system` database. It fetches data from the `bill_details` and `payment_status` tables, presents them in tabular format, and includes a button to print the page content.

PHP Code (Server-side)

1. **Session Start and Includes**:

- Initiates a PHP session for managing user authentication and state.
- Includes necessary files (`./connection.php` and `./functions.php`) for database connection and user authentication (`check_login` function).

2. **HTML Structure**:

- Defines the structure of an HTML document with metadata and styling using CSS.
- Uses `

` containers styled as sections (`class="section"`) to organize content.

3. **Bill Details Section**:

- Displays a table (`<table>`) containing columns (`<th>`) for various bill details fetched from the `bill_details` table.
- Executes a MySQL query to retrieve records (`SELECT * FROM bill_details`) and iterates over the result set to output each row in HTML format.
- Handles cases where no records are found by displaying a corresponding message (`<tr><td colspan='14'>No records found</td></tr>`).

4. **Payment Records Section**:

- Presents another table (`<table>`) for displaying payment records fetched from the `payment_status` table.
- Similar to the bill details section, retrieves data (`SELECT * FROM payment_status`), iterates over the result set, and formats each row in HTML.
- Handles the scenario of no records found with an appropriate message (`<tr><td colspan='7'>No records found</td></tr>`).

5. **Database Connectivity and Error Handling**:

- Establishes a MySQL database connection using `mysqli`.
- Checks the connection status and terminates with an error message (`die("Connection failed: " . \$conn->connect_error`) if the connection fails.
- Closes the database connection after executing each query to free resources (`\$conn->close()`).

6. **Print Button**:

- Includes a button (`<button class="print-button" onclick="window.print()">Produce Copies</button>`) styled with CSS to allow users to print the displayed records directly from the browser.

Purpose

This script serves to display and manage bill details and payment records within a web application. It ensures a user-friendly interface for viewing these records, integrates database operations for data retrieval, and provides functionality to print the displayed content for offline use or documentation purposes.

Chapter : 13

Data - Backup System

Source Code Documentation

Data - Backup System

The Rent Management System will provide the feature to make the folder to make the data backup , that can be used to utilised in future occurrences , when we face a data failure.

Data Failure can only occur in the below consequences :

- (i) The internal files can get deleted by any data cleanup software.
- (ii) the data can be mistakenly be deleted.

To overcome above consequences to prevent and overcome the problems , we have added the Data-backup system

Rent Management System

```
| -> backup_proc  
| -> backup.php
```

The responsibility of the above file is to produce all the documents submitted to make the agreement , and all the data stored inside the database tables.

This will produce the folder that will contain all the files and the data in the form of the html , that can be viewed and printed to suitable page format

Source code (backup.php)

Sure, let's go through the provided PHP script and explain each part in detail:

```
```php  
<?php
// Function to generate random 6-character UPI ID
function generateUPIID() {
 $characters = 'ABCDEFGHIJKLMNPQRSTUVWXYZ0123456789';
 $upi_id = 'UPI';
 for ($i = 0; $i < 6; $i++) {
 $upi_id .= $characters[rand(0, strlen($characters) - 1)];
 }
 return $upi_id;
}
```

```

1. **Function `generateUPIID()`**:

- This function generates a random 6-character UPI (Unified Payment Interface) ID.
- It initializes a string `\$upi_id` with 'UPI'.
- It iterates 6 times, appending a random character from the string `\$characters` (which contains letters A-Z and digits 0-9) to `\$upi_id`.
- Finally, it returns the generated `\$upi_id`.

```php

```
// Handling form submission
if ($_SERVER["REQUEST_METHOD"] == "POST") {
 // Verify the PIN
 $pin = $_POST['pin'];
 if ($pin != '55555') {
 echo "<script>alert('Incorrect PIN');</script>";
 exit();
 }

 // Set up the backup directory on the Desktop
 $backup_dir = getenv("HOMEDRIVE") . getenv("HOMEPATH") . '\Desktop\Backup_' .
 date('Ymd');
 if (!file_exists($backup_dir)) {
 mkdir($backup_dir, 0777, true);
 }
}
```

## 2. \*\*Form Submission Handling\*\*:

- Checks if the HTTP request method is POST (`\$\_SERVER["REQUEST\_METHOD"] == "POST"`).
- Retrieves the PIN from the POST data (`\$\_POST['pin']`).
- Checks if the PIN matches '55555'. If not, it displays an alert and exits the script.

- Constructs a backup directory path on the Desktop using environment variables (`HOMEDRIVE` and `HOME PATH`) and appends the current date (`date('Ymd')`) to create a unique directory for each backup.
- Checks if the backup directory doesn't exist (`!file\_exists(\$backup\_dir)`), then creates it with full permissions (`mkdir(\$backup\_dir, 0777, true)`).

```
```php
// Function to copy directory
function copyDirectory($src, $dst) {
    $dir = opendir($src);
    @mkdir($dst);
    while (false !== ($file = readdir($dir))) {
        if (($file != '.') && ($file != '..')) {
            if (is_dir($src . '/' . $file)) {
                copyDirectory($src . '/' . $file, $dst . '/' . $file);
            } else {
                copy($src . '/' . $file, $dst . '/' . $file);
            }
        }
    }
    closedir($dir);
}
```

```

### 3. \*\*Function `copyDirectory(\$src, \$dst)`\*\*:

- This function recursively copies all files and directories from `'\$src` to `'\$dst`.
- Opens the source directory (`'\$src` and creates the destination directory (`'\$dst` if it doesn't exist.
- Iterates through each file in the source directory, copying files or recursively calling itself for subdirectories until all files are copied.
- Closes the directory handle after copying.

```
```php
```

```

// Copy the stores directory to the backup folder
$source_dir = 'C:/xampp/htdocs/rmstest11/rent_agr/stores';
$destination_dir = $backup_dir . '/stores';
copyDirectory($source_dir, $destination_dir);

```

```

#### 4. \*\*Copying Directory\*\*:

- Defines `'\$source\_dir` as the absolute path to the 'stores' directory in the local XAMPP server.
- Sets `'\$destination\_dir` as the backup directory path appended with '/stores'.
- Calls `copyDirectory(\$source\_dir, \$destination\_dir)` to copy the entire 'stores' directory to the backup folder created earlier.

```

```php
// Database connection
$conn = new mysqli('localhost', 'root', "", 'rent_management_system');

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

```

```

#### 5. \*\*Database Connection\*\*:

- Establishes a connection to the MySQL database named 'rent\_management\_system' hosted on localhost with the username 'root' and no password.
- Checks if the connection fails (`\$conn->connect\_error`), then terminates the script and displays an error message.

```

```php
// Function to fetch data from a table and return as HTML
function fetchTableData($conn, $tableName, $fields) {
    $sql = "SELECT " . implode(", ", $fields) . " FROM " . $tableName;
}

```

```

$result = $conn->query($sql);

if ($result->num_rows > 0) {
    $html = "<h2>$tableName</h2>";
    $html .= "<table border='1'><tr>";
    foreach ($fields as $field) {
        $html .= "<th>$field</th>";
    }
    $html .= "</tr>";

    while ($row = $result->fetch_assoc()) {
        $html .= "<tr>";
        foreach ($fields as $field) {
            $html .= "<td>" . $row[$field] . "</td>";
        }
        $html .= "</tr>";
    }
    $html .= "</table>";
    return $html;
} else {
    return "<h2>$tableName</h2><p>No records found</p>";
}
}

```

```

## 6. \*\*Function `fetchTableData(\$conn, \$tableName, \$fields)`\*\*:

- This function fetches data from a specified table (`\$tableName`) using specified fields (`\$fields`) and returns it formatted as HTML.
- Constructs an SQL query (`\$sql`) to select columns specified in `'\$fields` from `'\$tableName`.
- Executes the query (`\$conn->query(\$sql)`), fetches the result, and constructs an HTML table (`\$html`) if there are rows returned.

- Returns the HTML table with column headers and row data formatted accordingly. If no rows are returned, it returns a message indicating no records found.

```
```php
// Fetch data from each table

$users_fields = ['id', 'user_id', 'user_name', 'password', 'productkey', 'date'];

$reg_renter_fields = ['id', 'name', 'age', 'gender', 'datereg', 'aadhar', 'mobile', 'address', 'roomNo', 'price'];

$entries_fields = ['id', 'room_id', 'photo_path', 'sign_path', 'aadhar_path', 'created_at'];

$enq_comm_fields = ['id', 'name', 'age', 'gender', 'address', 'mobile', 'time_of_saving'];

$bill_details_fields = ['id', 'renter_id', 'month', 'year', 'due_date', 'room_rent', 'units_used', 'electric_bill', 'advance_paid', 'amount_dues', 'miscellaneous', 'total_amount', 'created_at'];

$payment_status_fields = ['id', 'renter_id', 'date_pay', 'month', 'amount', 'pay_status', 'description'];

$data_html = "";

$data_html .= fetchTableData($conn, 'users', $users_fields);

$data_html .= fetchTableData($conn, 'reg_renter', $reg_renter_fields);

$data_html .= fetchTableData($conn, 'entries', $entries_fields);

$data_html .= fetchTableData($conn, 'enq_comm', $enq_comm_fields);

$data_html .= fetchTableData($conn, 'bill_details', $bill_details_fields);

$data_html .= fetchTableData($conn, 'payment_status', $payment_status_fields);

```

```

## 7. \*\*Fetching Data from Tables\*\*:

- Defines arrays (`\$users\_fields`, `'\$reg\_renter\_fields`), etc.) containing field names for each table to be fetched.
- Initializes an empty string (`\$data\_html`) to accumulate HTML table data.
- Calls `fetchTableData()` for each table with respective fields and appends the returned HTML to `'\$data\_html`.

```
```php
// Close connection
```

```

$conn->close();

// Save the data HTML to a file
$data_file = $backup_dir . '/data.html';
file_put_contents($data_file, $data_html);

echo "<script>alert('Backup created successfully');</script>";
echo "<script>window.location.href='backup.php';</script>";
}

?>
```

```

#### 8. \*\*Final Steps\*\*:

- Closes the database connection (`\$conn->close()`).
- Saves the accumulated HTML data (`\$data\_html`) to a file (`\$data\_file`) in the backup directory as 'data.html' using `file\_put\_contents()`.
- Displays a success alert indicating the backup creation (`echo "<script>alert('Backup created successfully');</script>"`).
- Redirects the user to 'backup.php' (`echo "<script>window.location.href='backup.php';</script>"`).

```

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Backup Module</title>
    <style>
        /* CSS styles for the HTML content */
    </style>
</head>
<body>

```

```
<!-- HTML structure for displaying backup module interface -->
</body>
</html>

>
```

```

## 9. \*\*HTML Interface\*\*:

- Defines a standard HTML5 document structure with `<html>`, `<head>`, and `<body>` tags.
- Sets the document charset and viewport.
- Includes a `<title>` tag for the page title.
- Defines CSS styles in the `<style>` section to format the interface elements.

## 10. \*\*CSS Styling\*\*:

- CSS styles define the appearance of elements (`body`, `.container`, `.section`, `table`, `.backup-button`, etc.).
- Styles control margins, padding, backgrounds, borders, and text formatting to create a visually appealing backup module interface.

```
```html
<div class="container">
  <div class="section">
    <h2>Data Backup</h2>
    <button class="backup-button" onclick="createBackup()">Create Backup</button>
    <div id="data-container">
      <!-- Data will be loaded here by JavaScript -->
    </div>
  </div>
</div>
```

```

## 11. \*\*HTML Structure within Body\*\*:

- Defines a `<div>` with class `container` to center and limit width of content.
- Inside, a `<div>` with class `section` contains:
  - `<h2>` for heading 'Data Backup'.
  - A `<button>` with class `backup-button` triggers the `createBackup()` function on click.
  - A `<div>` with id `data-container` to display data loaded dynamically by JavaScript.

```javascript

```
<script>
  function createBackup() {
    var pin = prompt("Enter Pin:");
    if (pin !== "55555") {
      alert("Incorrect PIN");
      return;
    }
    document.forms["backupForm"].submit();
  }
</script>
```

```
<form id="backupForm" method="POST" style="display: none;">
  <input type="hidden" name="pin" value="55555">
</form>
```

```

## 12. \*\*JavaScript Functions\*\*:

- Defines a JavaScript function `createBackup()`:
  - Prompts the user to enter a PIN.
  - Checks if the entered PIN matches '55555'. If not, alerts 'Incorrect PIN' and exits.
  - Submits the hidden form `backupForm` to initiate the backup process.

## 13. \*\*Hidden Form for PIN Submission\*\*:

- Defines a hidden `<form>` with id `backupForm` and method POST.
- Contains a hidden input `<input>` named 'pin' with value '55555'.

- This form is submitted by JavaScript when the correct PIN is entered, triggering the PHP script for backup creation.

This script integrates PHP for server-side processing (backup creation, database interaction) and JavaScript for client-side interaction (PIN validation, form submission), providing a robust backup module for managing data integrity.

**Chapter : 14**

**Homepage : Rent Management System**

**Source Code Documentation**



## Homepage

The Rent Management System , features are accessed only through the homepage.

The login page will bring us to the homepage of the Rent Management System

### Rent Management System

- | - > index.php (homepage of the system)
- | - > style.css (for styling the homepage)
- | - >script1.js (providing interactivity to the system)

### Documentation (index.php)

Certainly! Here's a brief but expressive documentation for the provided PHP and HTML code:

#### PHP Code Explanation

##### 1. Session Start:

- `session\_start();`: Initiates a session to persist data across multiple pages for the user.

##### 2. Including Files:

- `include("connection.php");`: Includes a PHP file (`connection.php`) that likely establishes a database connection.

- `include("functions.php");`: Includes a PHP file (`functions.php`) containing reusable functions.

##### 3. User Authentication:

- `'\$user\_data = check\_login(\$con);` : Calls the `check\_login()` function from `functions.php` with the database connection (`\$con`) to authenticate and retrieve user data. Stores the result in `'\$user\_data` .

#### HTML Code Explanation

##### 1. Document Structure:

- Standard HTML5 structure (`<!DOCTYPE html>`).
- `<head>` section includes metadata (`charset`, `viewport`) and references to external resources (`style.css` for styling).

## 2. Navigation (`<nav>`):

- Contains various links and elements for navigation:
- Logo linking to the homepage.
- Navigation items like 'Location', 'Search', 'Create Bills', 'Save Contacts', 'Registration', etc.
- Icons and dropdown indicators for additional functionality.

## 3. Header and Navigation Links:

- Links to various pages and functionalities related to the Rent Management System.
- Structured into sections (`IT-Rules Followed`, `Developer Contacts`, `Landlord Details`, `Reset Password`, `Logout`) for easy access.

## 4. Header Slider (`<div class="header-slider">`):

- Displays a slideshow of images (`header1.jpg` to `header6.jpg`) for visual appeal.

## 5. Content Boxes (`<div class="box-row header-box">`):

- Divided into columns (`<div class="box-col">`) showcasing different functionalities:
  - Online Payments (`Pay Online`, `Feed Details Manually`).
  - Records & Reports (`Click to Open Early Records`, `Export Reports (in pdf)`).
  - List of Renters (`Click To Proceed`).
  - Rooms Available (`View List of Available Rooms`, `+ Add Rooms`, `+ Remove Rooms`).

## 6. User Data Display:

- `<?php echo "{\$user\_data['user\_name']} &password :{\$user\_data['password']}"; ?>`:  
Displays user-specific data retrieved during login, likely for debugging or testing purposes.

## 7. Script Inclusion:

- `<script src="script1.js"></script>`: Includes a JavaScript file (`script1.js`) for client-side scripting functionality.

## Summary

This code integrates PHP for server-side logic such as session management, user authentication, and database operations (`connection.php`, `functions.php`). HTML provides the structure and interface elements for navigating various functionalities of the Rent Management System, enhancing user interaction and usability with visual elements like sliders and navigation links.

### Documentation (script1.js)

Certainly! Here's a brief and expressive documentation for the provided JavaScript code:

#### ### JavaScript Code Explanation

##### 1. \*\*Query Selection\*\*:

- `const imgs = document.querySelectorAll('.header-slider ul img');`: Selects all `` elements within `
` under elements with class `.header-slider`. These represent the images in the slideshow.

- `const prev\_btn = document.querySelector('.control\_prev');`: Selects the element with class `.control\_prev`, which serves as the previous slide control button.

- `const next\_btn = document.querySelector('.control\_next');`: Selects the element with class `.control\_next`, which serves as the next slide control button.

##### 2. \*\*Initial State and Function Definition\*\*:

- `let n = 0;`: Initializes a variable `n` to keep track of the current image index in the slideshow.

- `function changeSlide() { ... }`: Defines a function `changeSlide()` to hide all images except the one at index `n`, making it visible (`display: block`).

##### 3. \*\*Initial Slide Display\*\*:

- `changeSlide();`: Calls `changeSlide()` initially to display the first image in the slideshow.

#### 4. \*\*Event Listeners\*\*:

- `prev\_btn.addEventListener('click', (e) => { ... });`: Listens for a click event on the previous button. When clicked:
  - Checks if `n` is greater than 0. If true, decrements `n` to show the previous image. If `n` is 0, sets `n` to the last image index (`imgs.length - 1`).
  - Calls `changeSlide()` to update the displayed image.
- `next\_btn.addEventListener('click', (e) => { ... });`: Listens for a click event on the next button. When clicked:
  - Checks if `n` is less than `imgs.length - 1`. If true, increments `n` to show the next image. If `n` is already at the last image, sets `n` back to 0 to loop back to the first image.
  - Calls `changeSlide()` to update the displayed image.

#### ### Summary

This JavaScript code manages a slideshow functionality:

- It selects slideshow images and control buttons.
- Defines a function to display the current image and hide others.
- Initializes a variable to keep track of the current image index (`n`).
- Listens for click events on previous and next buttons to navigate through the slideshow, updating the displayed image accordingly.

Overall, this code enhances user experience by enabling interactive navigation through slideshow images using simple JavaScript event handling and DOM manipulation techniques.

# **CHAPTER - 6**

## **SOFTWARE TESTING**

### **RENT MANAGEMENT SYSTEM**

| <b>SYSTEM DESIGN AND IMPLEMENTATION</b> |                                             |
|-----------------------------------------|---------------------------------------------|
| 6.1                                     | Introduction to Software Testing            |
| 6.2                                     | Strategic Approaches for Software Testing   |
| 6.3                                     | Unit Testing For the Rent Management System |



## 6.1 Introduction To Software Testing

. Software testing is a crucial process in the software development lifecycle that involves evaluating and verifying that a software application or system meets specified requirements and functions as intended. It is a methodical approach to identify defects, ensure quality, and validate the software's performance, security, and usability before it is deployed to end-users. By systematically executing test cases, testers can uncover issues related to functionality, reliability, and user experience, enabling developers to address these problems early on. This proactive process not only enhances the overall quality of the software but also minimizes the risk of failures and reduces maintenance costs post-deployment. Software testing encompasses various levels and types, including unit testing, integration testing, system testing, and acceptance testing, each serving a specific purpose in ensuring that the software is robust, secure, and fit for use. Ultimately, software testing is an integral part of delivering high-quality software products that meet user expectations and business requirements.

## 6.2 Strategic Approaches for Software Testing

### Strategic Approaches for Software Testing

In the dynamic world of software development, ensuring the quality and reliability of software products is paramount. Software testing plays a critical role in achieving this goal, and adopting strategic approaches to software testing can significantly enhance the effectiveness and efficiency of the testing process. A well-defined testing strategy not only helps in identifying and fixing defects early in the development cycle but also ensures that the software meets the desired quality standards and user expectations. This article explores several strategic approaches for software testing, highlighting their importance and implementation in the software development lifecycle.

#### 1. Risk-Based Testing

Risk-based testing is a strategic approach that prioritizes testing efforts based on the risk of failure and the impact of defects. By identifying and focusing on the most critical areas of the application, testers can allocate resources effectively to mitigate the highest risks. This approach involves assessing the probability of defects and their potential consequences on the system. Test cases are then designed to target these high-risk areas, ensuring that the most significant risks are addressed first. This method not only optimizes testing efforts but also enhances the overall reliability of the software.

#### 2. Test Automation

Incorporating test automation into the testing strategy is essential for improving test coverage, increasing efficiency, and reducing human error. Automated tests can be executed repeatedly, providing quick feedback on the stability of the application with each code change. Test automation is particularly beneficial for regression testing, performance testing,

and continuous integration/continuous deployment (CI/CD) pipelines. By automating repetitive and time-consuming tasks, testers can focus on more complex and exploratory testing activities. However, it is crucial to identify the right test cases for automation to maximize the return on investment.

### 3. Behavior-Driven Development (BDD)

Behavior-Driven Development (BDD) is a collaborative testing approach that involves stakeholders, developers, and testers in the process of defining the expected behavior of the application. BDD uses natural language constructs to create test scenarios that describe the desired behavior from the user's perspective. These scenarios are then translated into automated tests. This approach fosters better communication among team members, ensures a clear understanding of requirements, and aligns testing efforts with business objectives. BDD not only improves the quality of the tests but also enhances the overall development process.

### 4. Shift-Left Testing

Shift-left testing is a proactive approach that emphasizes early testing in the software development lifecycle. By integrating testing activities from the beginning of the development process, teams can identify and resolve defects sooner, reducing the cost and effort associated with fixing issues later. This strategy involves practices such as test-driven development (TDD), where tests are written before the code, and continuous testing, which ensures that testing is an ongoing activity throughout the development cycle. Shift-left testing leads to faster feedback loops, higher code quality, and shorter development cycles.

### 5. Exploratory Testing

Exploratory testing is an unscripted and adaptive testing approach where testers actively explore the application to uncover defects that may not be identified through traditional testing methods. This approach relies on the tester's intuition, experience, and creativity to discover issues in real-time. Exploratory testing is particularly useful for identifying usability issues, unexpected behaviors, and edge cases. By combining exploratory testing with other structured testing methods, teams can achieve a more comprehensive understanding of the application's quality and user experience.

### 6. Performance Testing

Performance testing is a strategic approach focused on evaluating the responsiveness, stability, and scalability of the software under various conditions. It involves simulating real-world scenarios to assess how the application performs under different loads and stress levels. Performance testing helps identify bottlenecks, resource constraints, and performance degradation issues that could impact the user experience. By conducting thorough

performance testing, teams can ensure that the application meets performance requirements and can handle expected user traffic and data volumes.

## 7. Continuous Testing

Continuous testing is a modern approach that integrates testing into every stage of the software delivery pipeline. By leveraging automation, continuous integration, and continuous delivery practices, continuous testing ensures that code changes are validated continuously throughout the development process. This approach provides immediate feedback on the impact of changes, enabling teams to detect and fix defects quickly. Continuous testing supports the delivery of high-quality software at a faster pace, making it an essential component of agile and DevOps methodologies.

## Conclusion

Strategic approaches to software testing are essential for delivering high-quality software products that meet user expectations and business goals. By adopting methods such as risk-based testing, test automation, BDD, shift-left testing, exploratory testing, performance testing, and continuous testing, teams can enhance the effectiveness and efficiency of their testing efforts. These strategies not only help in identifying and fixing defects early but also ensure that the software is reliable, performant, and user-friendly. As the software development landscape continues to evolve, adopting and adapting these strategic approaches will be crucial for achieving success in software testing.

## 6.3 Unit Testing : Rent Management System

### Definition of Unit Testing

Unit testing is a software testing technique where individual units or components of a software application are tested in isolation. The primary goal of unit testing is to validate that each unit of the software performs as expected. Units are the smallest testable parts of an application, such as functions, methods, or procedures. Unit tests are typically automated and written by developers during the development phase to ensure that each unit of the software behaves correctly under various conditions. By identifying and fixing bugs early in the development process, unit testing helps improve the overall quality of the software and facilitates easier maintenance and refactoring.

### Suitability of Unit Testing

Unit testing is a suitable practice for several reasons:

1. Early Bug Detection: Unit testing helps in identifying defects at an early stage of development, reducing the cost and effort required to fix them later.

2. Code Quality: By testing individual components, unit testing ensures that each part of the application functions correctly, leading to higher overall code quality.
3. Simplifies Integration: When units are tested in isolation and verified to work correctly, it simplifies the process of integrating them into a larger system.
4. Facilitates Refactoring: Unit tests provide a safety net that allows developers to refactor code with confidence, knowing that any changes can be verified against existing tests.
5. Documentation: Unit tests serve as documentation for the code, providing examples of how each unit is expected to behave.
6. Automated Testing: Unit tests can be automated and run frequently, ensuring that the codebase remains stable over time.

### Unit Testing Results

| Serial No. | Module                                    | Unit Testing Result |
|------------|-------------------------------------------|---------------------|
| 1          | Login / Logout / Signup / Reset Password  | Success             |
| 2          | Store Enquiry & Display Enquiry Details   | Success             |
| 3          | Tenant Registration & Agreement           | Success             |
| 4          | Add Rooms, Remove Rooms, Display Rooms    | Success             |
| 5          | Online Payments Interface                 | Success             |
| 6          | Manual Transaction Entry                  | Success             |
| 7          | Display all Records of Bills & Receivings | Success             |
| 8          | Databackup Module                         | Success             |
| 9          | PDF Making Functionality                  | Success             |

**CHAPTER - 7**

**FUTURE ENHANCEMENTS**

**RENT MANAGEMENT SYSTEM**



## 7.1 Future Enhancements of the RENT MANAGEMENT SYSTEM

### Future Enhancements of the Rent Management System

The Rent Management System, already encompassing a robust suite of modules including Login/Logout/Signup/Reset Password, Store Enquiry & Display Enquiry Details, Tenant Registration & Agreement, Add/Remove/Display Rooms, Online Payments Interface, Manual Transaction Entry, Display all Records of Bills & Receivings, Databackup Module, and PDF Making Functionality, has significant potential for future enhancements. To begin with, the Login/Logout/Signup/Reset Password module can be augmented with multi-factor authentication and biometric login options to enhance security. Additionally, implementing machine learning algorithms can analyze login patterns to detect and prevent unauthorized access.

The Store Enquiry & Display Enquiry Details module can benefit from a more sophisticated data analytics feature that can predict future rental trends based on historical enquiry data. Integrating a chatbot with natural language processing capabilities can provide instant responses to enquiries, improving user engagement. For the Tenant Registration & Agreement module, incorporating blockchain technology can ensure the immutability and security of rental agreements, providing tenants and landlords with an added layer of trust and transparency.

Enhancing the Add/Remove/Display Rooms module with virtual tours and augmented reality can offer prospective tenants a more immersive experience, allowing them to visualize the space before making a decision. The Online Payments Interface can be expanded to include more payment options such as cryptocurrency and offer automatic rent payment reminders and split payments for roommates. For Manual Transaction Entry, introducing voice-to-text capabilities can streamline the data entry process, making it quicker and more efficient.

The module for Displaying all Records of Bills & Receivings can be enhanced with advanced reporting and visualization tools, providing users with customizable dashboards that display real-time financial metrics and trends. The Databackup Module can be improved by adopting decentralized storage solutions and automated backup schedules to ensure data integrity and availability. Finally, the PDF Making Functionality can be upgraded to support dynamic PDF templates and digital signatures, facilitating a more seamless and professional document management process.

In summary, the Rent Management System holds great promise for future advancements that can further elevate its functionality, security, and user experience. By leveraging emerging technologies and focusing on user-centric enhancements, the system can continue to evolve and meet the dynamic needs of the rental market.

**CHAPTER - 8**

**CONCLUSION**

**RENT MANAGEMENT SYSTEM**



## 8.1 Conclusion : RENT MANAGEMENT SYSTEM

The Rent Management System stands as a testament to the integration of technology and property management, offering a centralized platform to manage various aspects of rental operations seamlessly. By providing modules for critical functions such as login/logout, tenant registration, enquiry management, room management, payment processing, and data backup, the system ensures that property managers can handle their tasks with greater efficiency and accuracy. Each module is designed with the user in mind, offering intuitive interfaces and automated features that reduce the administrative burden and allow for more focus on strategic management and tenant relations.

Moreover, the system's current capabilities lay a strong foundation for future enhancements. As the rental market evolves and new technologies emerge, the Rent Management System can incorporate these advancements to stay ahead of industry trends. For instance, the integration of machine learning for predictive analytics, blockchain for secure transactions, and augmented reality for virtual tours can significantly elevate the system's functionality. These enhancements not only improve the operational aspects but also enhance the user experience, making the system more valuable to both landlords and tenants.

In conclusion, the Rent Management System is not just a tool but a comprehensive solution that addresses the multifaceted needs of property management. Its design ensures that it can grow and adapt, providing long-term value and supporting the dynamic nature of rental property management. By leveraging technology to automate processes, secure transactions, and offer advanced features, the Rent Management System paves the way for more efficient, transparent, and user-friendly property management. This adaptability and forward-thinking approach position the system as an indispensable asset in the modern property management landscape, ensuring it remains relevant and effective in meeting the needs of property managers and tenants alike.



## **CHAPTER - 9**

### **Bibliography & References**

### **RENT MANAGEMENT SYSTEM**



## **9.1 Bibliography & References : RENT MANAGEMENT SYSTEM**

### **Bibliography and References for PHP Rent Management System**

#### **Books**

1. **PHP and MySQL Web Development**" by Luke Welling and Laura Thomson

This book provides comprehensive coverage of PHP and MySQL, including techniques for building dynamic and interactive web applications.

Citation: Welling, L., & Thomson, L. (2016). PHP and MySQL Web Development. Addison Wesley Professional.

2. **Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5**" by Robin Nixon

An excellent resource for learning how to build dynamic web applications using PHP, MySQL, JavaScript, and related technologies.

Citation: Nixon, R. (2018). Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5. O'Reilly Media.

3. **PHP Objects, Patterns, and Practice**" by M. Zandstra

This book explores advanced PHP concepts, including object oriented programming, design patterns, and best practices.

Citation: Zandstra, M. (2017). PHP Objects, Patterns, and Practice. Apress.

#### **Online Resources**

4. **PHP Official Documentation**

The official PHP documentation provides detailed information on PHP functions, classes, and best practices.

URL : <https://www.php.net/manual/en/>

5. **MySQL Official Documentation**

Comprehensive guide to using MySQL, including installation, configuration, and SQL commands.

URL : <https://dev.mysql.com/doc/>

6. **W3Schools PHP Tutorial**

A beginner friendly tutorial covering the basics of PHP, including syntax, functions, and database integration.

URL : <https://www.w3schools.com/php/>

## 7. Stack Overflow

A community driven Q&A site where developers can ask and answer questions related to PHP and other programming languages.

URL : <https://stackoverflow.com/>

## Articles and Tutorials

### 8. Building a Simple PHP Login System" by Tania Rascia

A step by step guide to building a basic PHP login system, including user registration and authentication.

Citation : Rascia, T. (2018). Building a Simple PHP Login System. Retrieved from [https://www.taniarascia.com/create a simple database app php/](https://www.taniarascia.com/create-a-simple-database-app-php/)

### 9. Secure PHP Development" by John Conde

An article focused on security best practices in PHP development, covering common vulnerabilities and mitigation techniques.

Citation : Conde, J. (2019). Secure PHP Development. Retrieved from [https://johnconde.net/blog/secure php development/](https://johnconde.net/blog/secure-php-development/)

## References

1. Welling, L., & Thomson, L. (2016). PHP and MySQL Web Development. Addison Wesley Professional.
2. Nixon, R. (2018). Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5. O'Reilly Media.
3. Zandstra, M. (2017). PHP Objects, Patterns, and Practice. Apress.
4. PHP Official Documentation. Retrieved from <https://www.php.net/manual/en/>
5. MySQL Official Documentation. Retrieved from <https://dev.mysql.com/doc/>
6. W3Schools PHP Tutorial. Retrieved from <https://www.w3schools.com/php/>

This bibliography and references page provides a comprehensive list of resources that can support the development and enhancement of a PHP based Rent Management System.

## **CHAPTER - 10**

### **SYNOPSIS**

### **RENT MANAGEMENT SYSTEM**





आर्यभट्ट क्षान विश्वविद्यालय

ARYABHATTA KNOWLEDGE UNIVERSITY

## BCA SYNOPSIS

**SESSION :** 2021-2024

**NAME OF THE LEARNER :** AAYUSH SAHAY

**ENROLLMENT NUMBER :** 21303334049

**PROGRAMME :** BCA

**COURSE CODE :** 303

**MOBILE NUMBER :** 9234817383

**EMAIL :** aayushsahay947@gmail.com

**Under Guidance of :** Prof. Shahbaz Shakeb

**COLLEGE CODE :** 334

**NAME OF THE INSTITUTION :** INDIAN INSTITUTE OF BUSINESS  
MANAGEMENT , PATNA

# **PERFORMA OF BCA PROJECT PROPOSAL (BCA-303)**

## **PROJECT TITLE AND GUIDE DETAILS**

**Session : 2021-2024**

**Enrollement No. : \_\_\_\_\_ Course Code : \_\_\_\_\_ College Code : \_\_\_\_\_**

**1. Details of the Student :**

**Name and Address of the Student :**

---

---

---

**Mobile No. : \_\_\_\_\_ E-Mail : \_\_\_\_\_**

**2. Title of the Project :**

---

**3. Details of the Guide :**

**Name and Address of the Guide :**

---

---

---

**4. Qualification of the Guide :**

---

**5. Industrial / Teaching Experience of the Guide :**

---

**6. Software Used to make the Project :**

---

**Signature of the Student**

**Signature of the Guide**

**Date : \_\_\_\_\_**

**Date : \_\_\_\_\_**

**For Office Use Only**

**Approved**

**Not Approved**

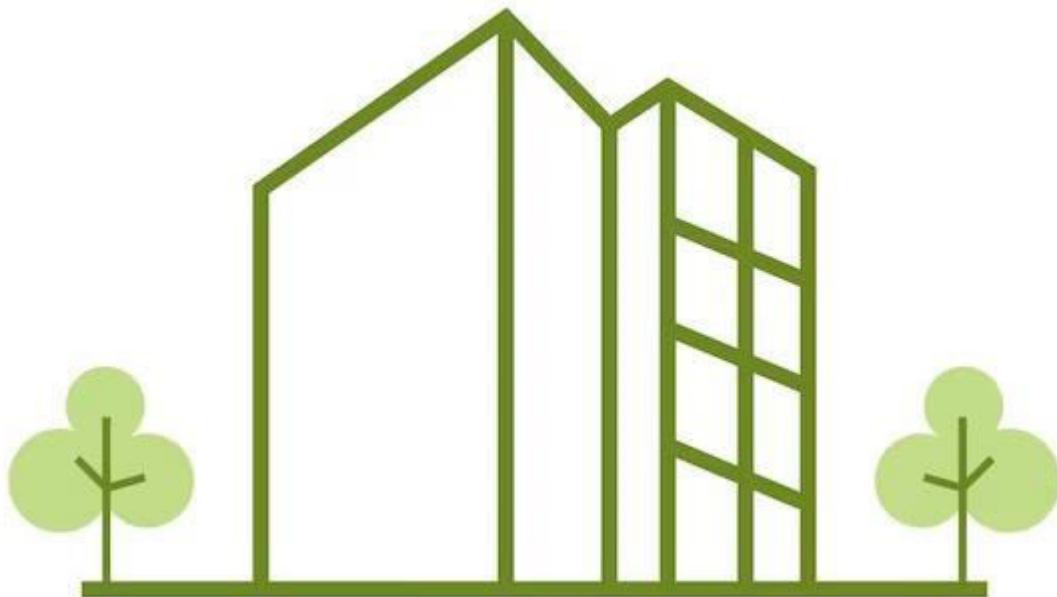
**Signature , Designation , Stamp  
of the Project Prosposal Evaluator**

**Suggestions for Re -Formulating the Project**

## **Biodata of the Guide**

## **Resume of the Guide**

## TOPIC OF THE PROJECT



# RENT MANAGEMENT SYSTEM

**Developed By :** AAYUSH SAHAY

**Enrollement No :** 21303334049

**Session :** 2021-2024

**Institution :** INDIAN INSTITUTE OF BUSINESS MANAGEMENT , PATNA

**College Code :** 334

**Under Guidance Of:** Prof. Shahbaz Shakeb

**Submitted To ARYABHATTA KNOWLEDGE UNIVERSITY , PATNA in Partial  
fulfillement of the Requirements for the Award of the Degree**

**Bachelor in Computer Applications (BCA)**

## Contents

| Sno. | Topics                                                  | Page No. |
|------|---------------------------------------------------------|----------|
| 1    | Introduction : Rent Management System                   | 315-316  |
| 2    | Objectives of the System                                | 317-318  |
| 3    | Features of the Rent Management System                  | 319-320  |
| 4    | Category of the Project                                 | 321-321  |
| 5    | Scope of the Project                                    | 322-323  |
| 6    | Hardware and the Software Requirements                  | 324-325  |
| 7    | Problem Definition and Requirements gathering           | 326-332  |
| 8    | Project Planning and Scheduling                         | 333-338  |
| 9    | System Analysis Rent Management System                  | 339-340  |
| 10   | Use Case Diagrams                                       | 341-341  |
| 11   | DFD digarams : 0th Level DFD ,Level 1 DFD , Level 2 DFD | 342-351  |
| 12   | Er Diagrams AND Sql Tables                              | 352-362  |
| 13   | Future Scope of the System                              | 363-364  |
| 14   | Limitations of the System                               | 365-367  |
| 15   | Bibliography and References                             | 368-369  |



Chapter : 1

## Introduction

# **RENT MANAGEMENT SYSTEM**



## Introduction To The Project : Rent Management System

Streamlining Landlord-Tenant Relations: Introducing the Rent Management System

In the world of real estate, where landlords oversee properties and tenants seek convenient rooms / housing , efficient management is required by the landlord side. The solution : the Rent Management System , a comprehensive software designed to facilitate seamless interactions between landlords and renters while ensuring legal compliance and financial transparency.

At its core, the Rent Management System acts as a Management Software where landlords can effortlessly register tenants, maintain detailed records, and track the financial aspects of their rental properties .

Let's Consider a landlord, let's call him Mr. AMIT , He owns multiple rental units or Rooms within a residential complex . As tenants come and go, Mr. AMIT needs a reliable Software system to manage the influx of renters, streamline rent collection , and maintain meticulous records for verification and legal purposes.

The journey begins with Mr. Amit , the diligent landlord, who logs into the Rent Management System. Here, he can effortlessly register new tenants, inputting vital details such as their personal information and lease agreements , and rental payment terms . The Software will Allow Mr.Amit to ensure that each tenant is officially documented within the RENT MANAGEMENT SYSTEM, laying the foundation for a transparent landlord - tenant relationship.

As tenants settle into the Rooms They Desire, the Rent Management System becomes a central platform for rent collection and Managing . Mr. AMIT can easily monitor rental payments, schedule recurring payments, and generate detailed reports showcasing the financial performance of his rental properties. Whether it's tracking overdue payments, sending reminders, or reconciling accounts, the system empowers Mr.AMIT to Maintain and Monitor his Room rental Business effortlessly.

Furthermore , the Rent Management System offers Needed features to enhance legal compliance and verification processes. By maintaining the records of lease agreements, rental payments, and communication logs, landlords like Mr.AMIT can navigate legal proceedings with confidence. Whether it's providing proof of tenancy, resolving disputes, or conducting audits, the system serves as a reliable repository of crucial information, safeguarding the interests of both landlords and tenants.

In essence, the Rent Management System revolutionizes the way landlords manage their rental properties , offering a seamless blend of efficiency , transparency , and compliance . From streamlining tenant onboarding to optimizing rent collection and record - keeping , the system empowers landlords to navigate the complexities of property management with ease. With the Rent Management System , the Business Resolves Risk of Ambiguity Thus , by replacing the Pen and Paper Records by the Digital Software Storing and Logic , Thus Making it a Mangement and Monitoring System For Enhancement of The Room Rental Business.

The Software : Rent Management System Provides the Interface Landlords To Register the Renters and also track the details and the income from this business. This is going to Provide the Landlords the Convinience in Handling the Details Legally and More Reliably.

## Chapter : 2

### Objectives of the Project

### **RENT MANAGEMENT SYSTEM**



## Objectives of : RENT MANAGEMENT SYSTEM

### (i) To Register the Details of the Tenant :

The need of Data for Verification Purposes and contact purposes are must to be recorded prior agreeing to provide the property to rent . The System Provides us the form of Registration in by help of which the necessary details of the Customer could be effectively be recorded.

### (ii) To Make the Tenants Aware of the Rules that They must Comply..

The System will provide the Interface to Provide the Tenant the Legal Guideline or the Tenant Laws that they must comply to in-order to Ensure the legal Compliance , The Eg. of Such Laws is That a Tenant is not supposed to keep another person as his /her Tenant . This Displays the Contract that the user understands and in his accomodation , he will keep that things in mind, There shall be the Form that shows the Guidelines and the Sign of the Tenant Entrusting / Agreeing to them . A copy will be given to the tenant , and the Landlord will keep that safe in his data store

### (iii) To Monitor and Manage the Electricity Bills : In the Type of City,

The Electricity bills are needed to be filled by the Landlord and Ensure the timely Payment by the Tenant for their bills is a must to avoid the abrupt stoppage of Electricity . This Provides a Module Such that the Accomodations , Units Used and Rate Per Unit is utilised to produce the Electricity bill the Tenant has to pay .

### (iv) Extensibility : The Term Extensibility Means that the LandLord Can add more rooms to be subjected for rental services , if the Landlord wishes , can add or remove the rooms from the list of available rooms The System Provides the Interior as Well as the dimensions of the room proposed for rental service. The landlord will be able to display the

design of the rooms , incase there are customers who are for inquiry for rooms / accomodations for rent.

- (v) To Ensure Timely Rent Collection by Generating the Reminders.  
The Reminder System provides the Convience to the landlord as the message can be sent to the tenant requesting payment for their accomodation.
- (vi) To Track every Transaction of Each of the Tenant : that may be rent receivings , amount in dues or advance payments , electric bills and their payments. This System allows the Tracking of the Date of Joining and the Date of Leaving .
- (vii) To Ensure the Legal Compliance : The Data of Each Tenant will be collected in a digital form . The copy of that is necessary to be submitted to the nearest police authority for the verification purposes , the System provides a swift way to store the necessary id and communication details that are needed prior to Agreeing to provide the accomodation for rent.

- (Viii) To Make the Comprhensive Report on the Income Per Tenant.  
The Income relies on the amount received , and the data needed To be formulated in such a way that shows the true picture of the Rent the Landlord has been receiving . This plays a crucial role when the landlord has to clear his income taxes.

## Chapter : 3

### Features of the Project

### **RENT MANAGEMENT SYSTEM**



## Features of the Rent Management System

The Rent Management System is Tailored to Meet the Requirements of the Landlord , or for the managers of the Room Rental Services. We tend to visualise the processes involved in a sequence is automated by the help of Software Engineering as :

- (i) To Display or to showcase the rooms that are available for the Landlord / Manager to provide the customers for the purpose of accommodations.
- (ii) To Keep the Details of the Interested Customers , that can be reminded of incase they require to rent a room / flat.
- (iii) To Register the Details of the Customer willing to Rent a Room. Registration will involve the recording of the necessary Communication Details and Identity Credentials.
- (iv) Signing the Agreement With the Landlord by the help of an Electronic Form that involves to enter the Picture and the Signature with a voice recording : By reading a statement aloud that he is aware of the rules and regulations and will abide by them.
- (v) There will be Record keeping of the Transactions that took place between the customer and the Landlord.
- (vi) The Records will contain the Monthly Rent , Monthly Rent Dues , Monthly Rent Advance , Room Electricity Meter , Monthly Electricity Bill , Electricity Bill Dues , Electricity Bill Advance , Miscellaneous Charges and Reasons , Deductions Made and the Reason of the deductions. The PDF of the bills will be made and provided and stored.
- (vii) Then to Generate the Reports for the purpose of future References and Audit , the Software will be capable of making the receipts in the form of PDF documents that will be the portable format of the Documents Like the Registration and Agreement that can be stored as a necessary proof.
- (viii) When there arises the need for the customer/renter to leave the room , then he /she can request to the manager or the Landlord , to which the Certificate of Ending Room Renting Services will be Made that will be provided , the copy of that is going to be provided to the renter,

The Above Process Help us to Visualise the Flow of the Processes that occur in the Business. The Rent Management System will help us to manage the Entire Business in a Systematic Form by providing us with the below features :

- (i) Ease of Storing the Communication Details of the Customers who came to inquire. They can be reminded incase they are in the need of finding the accommodations.
- (ii) Ease of Registering and Signing the Agreement with the Customer.
- (iii) Ease of Maintaining the Records for the Landlord or the Manager of the Business to carry out the process of Accounting that is the most important task of any Business.
- (iv) Ease of Producing the Receipts in the form of PDF that can be delivered to the Renter and that is the most portable way to produce the document.
- (v) To Store the details and the necessary proof submitted at the time of Registration and Agreement in the Landlords Database.

Thus the RENT MANAGEMENT SYSTEM works as a Web Based Application Software that will carry out the general processes stated exactly above with providing much ease as well as Automating every part of the Business.

## Key Features of Rent Management Software

### 1. Tenant Management

Tenant Database: Centralized storage of tenant information including contact details, lease agreements, and payment history.

Communication Tools: Integrated messaging systems for sending notifications and reminders to tenants.

### 2. Rent Collection

Online Payments: Secure online payment gateways for tenants to pay rent via credit/debit cards, bank transfers, or digital wallets.

Automated Billing: Generation and distribution of invoices, along with automated reminders for upcoming or overdue payments.

### 3. Property Management

Property Listings: Management of property details, availability, and rental terms.

Maintenance Requests: System for tenants to submit maintenance requests and track their status.

### 4. Financial Reporting

Expense Tracking: Logging of expenses related to property maintenance, utilities, and other operational costs.

Income Reports: Detailed reports on rental income, payment history, and financial performance of properties.

### 5. Lease Management

Lease Creation and Renewal: Tools to create, manage, and renew lease agreements.

Document Storage: Secure storage of lease agreements and other important documents.

### 6. Compliance and Security

Data Security: Implementation of security measures to protect sensitive tenant and financial data.

Regulatory Compliance: Features to ensure compliance with local rental laws and regulations.

**Chapter : 4**

**Project Category**

**RENT MANAGEMENT SYSTEM**



## Introduction

Rent management software is essential for landlords, property managers, and real estate companies to efficiently manage their rental properties. These tools streamline various processes such as tenant management, rent collection, maintenance tracking, and financial reporting. PHP (Hypertext Preprocessor) is a popular server-side scripting language that is widely used in web development, including the creation of rent management software. This guide explores the key features, benefits, and popular solutions within the category of rent management software built using PHP.

### Benefits of Using PHP for Rent Management Software

#### 1. Cost-Effective Development

PHP is an open-source language, which reduces licensing costs and makes it a budget-friendly option for developing rent management software.

#### 2. Scalability

PHP-based applications can be easily scaled to handle increasing numbers of properties and tenants as your business grows.

#### 3. Flexibility

PHP offers flexibility in terms of integration with other systems and customization of features to meet specific business needs.

#### 4. Community Support

A large and active community of PHP developers provides extensive resources, libraries, and frameworks to support development and troubleshooting.

#### 5. Performance

PHP is known for its fast execution and ability to handle high traffic loads, making it suitable for managing large volumes of rental data and transactions.

The Project : RENT MANAGEMENT SYSTEM encompasses the use of WebTechnology

Like PHP for the purpose of creating Dynamic Project that support the Room Rental Business

By handling the tasks of Record keeping , The Business is greatly dependent on the data and the records that are generated by the actions , will be used to be stored in the MySQL Database Technology to meet the requirements of storage of data in the best way.

## Chapter : 5

### Scope of the Project

### RENT MANAGEMENT SYSTEM



# Scope of the Project: Rent Management System

## 1. Project Overview

The Rent Management System (RMS) project aims to develop a comprehensive software solution to streamline the management of rental properties. The system will cater to the needs of landlords, property managers, and real estate companies, providing tools for tenant management, rent collection, property maintenance, and financial reporting. This project will be developed using PHP, ensuring a robust, scalable, and cost effective solution.

## 2. Objectives

Enhance Efficiency : Automate and streamline property management tasks to reduce manual work and errors. Improve Communication : Facilitate better communication between property managers and tenants. Simplify Rent Collection : Provide multiple payment options and automate billing processes.

Ensure Compliance : Help property managers comply with local rental laws and regulations. Provide Insights : Offer detailed financial reports and analytics to help property managers make informed decisions.

## Features

### (i) Tenant Management

- Centralized database for tenant information.
- Tools for tenant communication and notification.

### (ii) Rent Collection

- Integration with online payment gateways.
- Automated billing and invoicing.
- Payment tracking and reminders.

### (iii) Property Management

- Property listing and availability management.
- Maintenance request submission and tracking.
- Document storage for leases and important documents.

### (iv) Financial Reporting

- Expense tracking and management.
- Income and payment history reports.
- Comprehensive financial performance analytics.

### (v) Lease Management

- Tools for creating, managing, and renewing leases.
- Secure storage of lease agreements.

(vi) Compliance and Security

Data encryption and secure access controls.

Features to ensure compliance with rental laws and regulations.

## Technical Requirements

Programming Language : PHP

Database : MySQLi

Web Server : Xammp Server

Frontend Technologies : HTML, CSS, JavaScript

Payment Integration : Stripe, PayPal, or other payment gateways

Hosting : Cloud based solutions (AWS, Google Cloud, or similar)

## Stakeholders

Property Managers : Primary users who will manage properties, tenants, and financials.

Landlords : Owners of the properties who will access reports and financials.

Tenants : End users who will use the tenant portal for payments and communication.

Developers : Technical team responsible for building and maintaining the system.

## Expected Outcomes

A fully functional rent management system that meets the needs of property managers and landlords. Streamlined property management processes resulting in time and cost savings.

Enhanced tenant satisfaction due to improved communication and convenient payment

options. Comprehensive financial reporting providing valuable insights into property performance.

## Conclusion

The Rent Management System project aims to deliver a robust and efficient tool for managing rental properties. By leveraging PHP and modern web technologies, the system will provide essential features to streamline operations, enhance communication, and ensure compliance, ultimately contributing to more effective and profitable property management.

## **Chapter : 6**

**Hardware and Software Requirements For the Project**

**RENT MANAGEMENT SYSTEM**



# Software and Hardware Requirements for the Rent Management System Using PHP

## Software Requirements

### 1. Operating System

Development: Windows 10

Deployment: Windows 10 or Windows 11

### 2. Web Server

Apache HTTP Server 2.4 or higher

Nginx 1.18 or higher (optional)

### 3. PHP

PHP 7.4 or higher

### 4. Database

MySQL Lite by Xammp Server

### 5. Frontend Technologies

HTML5

CSS3

JavaScript

### 6. Development Tools

Code Editor: Visual Studio Code

## Hardware Requirements

### 1. Development Machine

Processor: Intel Core i5 or equivalent

RAM: 8 GB (16 GB recommended for smooth operation)

Storage: 256 GB SSD (512 GB recommended)

Display: 1920x1080 resolution or higher

## 2. Server Machine (for hosting the application)

Processor: Intel Xeon E3 or equivalent

RAM: 8 GB (16 GB recommended for handling multiple users)

Storage: 256 GB SSD (preferably in a RAID configuration for redundancy)

Network: High speed internet connection with static IP

## Deployment Environment

### The System to Be setup on the Client's Browser Application

#### 1. Web Server Setup

Install Xammp Server for the Apache Container to run the PHP support.

#### 2. Database Setup

Install MySQL or PostgreSQL.

Create a database and user with necessary permissions.

Secure the database server by configuring firewalls and access controls.

## Summary

The Rent Management System built using PHP on a Windows 10 computer will be easily deployable on both Windows 10 and Windows 11 environments. By following these software and hardware requirements, you can ensure a robust, scalable, and secure setup for developing and running the rent management system efficiently. The outlined tools and configurations will help in achieving optimal performance and seamless user experience for property managers and tenants alike.

## Chapter : 7

### Problem Definition & Requirements Gathering

#### **RENT MANAGEMENT SYSTEM**



## Problem Definition And Requirements Gathering : Rent Management System

### Problem Definition

Actually the study phase is considered as the requirement analysis , which is the most vital part of the software development life cycle (SDLC). Without proper analysis something will remain hidden from the developer may generate certain fatality in near future even in after the implementation, too. Analysis is a software engineering task that bridges the gap between system level requirements engineering and system design. This phase consists of...

- (i) System Engineering
- (ii) Software Requirement Analysis
- (iii) Software Design

### Problem Recognition

In this phase the analyst should clarify

- (i) Who is behind the request for this work?
- (ii) Who will use the solution?
- (iii) What will be the economic benefit of the successful solution?
- (iv) Is there any other source for the solution that the organization needs?

Getting answers all those questions by himself/herself, the analyst starts working and he issues some questionnaires to the future users for gathering information and then after that concentrates on the feasibility studies.

## Preliminary Investigations

The purpose of the preliminary investigation is to collect information for developing broad solutions for the purpose of feasibility study. Material i.e. information and facts to be collected in the preliminary investigation not only act as a basis for forming the several broad solutions of proposed system but it also provides the much needed feedback for selection of the final candidates system among the solutions suggested in the course of feasibility study. Actually it means to find out the way that how the proposed site will be developed containing which facts and figures.

Following the four broad methodologies, which are described as follows:-

- (i) Interview.
- (ii) Questionnaire.
- (iii) Fact finding studies, etc.

After the verbal interview session is over, it was very clear that the team requires to Requirements Specifications.

## Requirements Specifications:-

In the traditional system files were used to maintain the database which was done manually. This existing system consumes a lot of time. This time consuming evaluation coupled by the huge maintenance problem and may also lead to erroneous results. The various operations performed on these files like sorting, adding, modifying and deletion of the records are very tedious. Moreover these manually maintained files have the possibility of getting worn out. Thus, less durability is achieved. Thus the demerits of the existing system can be summarized as follows:

- (i) There is no consistency as the data may not be updated on time.
- (ii) Feasibility is reduced
- (iii) Less reliability
- (iv) Security is not provided and any one can access
- (v) Prioritization of records is difficult.

- (vi) More erroneous
- (vii) Difficult to maintain
- (viii) As everything is done manually its slow process
- (ix) No timely acknowledgement service

Taking the demerits into consideration, an alternative system which uses Oracle as both front end and back end was used. In front end, retrieval of the data from the database is done through SQL queries i.e. using D2K forms. This is not a web application and the data is not distributed as only a single system is used. As it is confined only to a particular system, scope is limited and there is a hindrance to the reliability if the system fails.

The demerits of this alternate system are

- (i) Only single system used.
- (ii) If the system crashes then the data is lost
- (iii) too overburdened
- (iv) not reliable
- (v) slow processing
- (vi) less flexible
- (vii) not so user friendly

### Proposed system

The proposed system is developed based on the client server architecture, a request-response paradigm and is implemented with the help of advanced java using the tomcat web container. The employees can maintain and do the transactions online. The application starts by asking for user name and password which provides authentication. This system provides high security where the unauthorized users cannot access the data. Later we have different options for the Landlord like :

- (i) Room Management
- (ii) Tenant Details
- (iii) Rent Collection
- (iv) Billing and Accounting
- (v) Reports
- (vi) Electricity Bill Details
- (vii) Add Rooms / Remove Rooms

The objectives of the proposed system are as follows:

- (i) Easy to use, effective and efficient
- (ii) Accurate results.
- (iii) Easy maintenance.
- (iv) Fast access
- (v) More feasibility
- (vi) More secure.
- (vii) Provides high consistency.
- (viii) More reliable

## Feasibility Study

Feasibilities are studied from Economical, technical, operational and legal point of view and hence found no obstacles to continue with our proposed project to be developed. So, the feasibility studies are undergone as follows:

- (i) **Economic Feasibility:** More commonly known as Cost/Benefit Analysis.

The procedure is to determine the benefits and savings that are expected from a candidate system and compare them with costs. If the benefits outweigh costs, then decision is made to design and implement the system. Considering the facts it is becoming evident that the system will be economically feasible

both for developer as well as for client's respect.

(ii) **Technical Feasibility:** Technical feasibility centers on the existing computer system (hardware, software, etc.) and to what extent it can support the proposed addition. If the budget is a serious constraint, then the project is judged not feasible. In our case this does not become an obstacle.

(iii) **Behavioral Feasibility:** People are inherently resistant to change, and computers have been known to facilitate change. An estimate should be made of how strong a reaction the user staff is likely to have toward the development of a computerized system. This was not such a problem in our mentioned project.

(iv) **Legal Feasibility:** A determination of any infringement, violation or liability that could result from the development of the system. But the system to be developed will be 100% legal and complying strictly to the regulations under the Consumer Acts , Landlord Acts and Information and Identity Acts in India.

(v) **Operational Feasibility:** The management & operators desire to be well acquainted with the requisite skill needed. Here most of the members in development team having technical expertise.

(vi) **Time Feasibility:** The management & operators here concern about whether the project will completed timely or not. But considering the facts and figures collected by us regarding our project it can be easily assumed that the project will be completed within the specified time frame

## Cost and Benefit Analysis

In developing the cost estimates for this system several cost elements were taken into consideration. Among them was hardware, software, facility, operating and supply costs. Hardware costs related to the actual purchase of the computer and peripherals (for example, printer, disk drive, tape unit). Determining the actual cost of hardware is generally more difficult when various users than for dedicated stand-alone system share the system. Software costs relate to the buying of the software required to develop the project as well as the software required to run the application in the organization.

## Functional Requirements

In software engineering, a functional requirement defines a function of a software

system or its component. A function is described as a set of inputs, the behavior,

and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that show how a use case is to be fulfilled. They are supported by non-functional requirements, which impose constraints on the design or implementation (such as performance requirements, security, or reliability). As defined in requirements engineering, functional requirements specify particular behaviors of a system. This should be contrasted with non-functional requirements which specify overall characteristics such as cost and reliability.

## Technical Requirements:-

- (i) Performance Requirements
- (ii) Safety Requirements
- (iii) Security Requirements
- (iv) Hardware Constraints
- (v) Software Constraints
- (vi) Design Constraints

## Nonfunctional Requirements:-

In systems engineering and requirements engineering, non-functional requirements are requirements which specify criteria that can be used to judge the operation of a system, rather than specific behaviors. This should be contrasted with functional requirements that specify specific behavior or functions. Non-functional requirements are often called qualities of a system. Other terms for non-functional requirements are "constraints", "quality attributes", "quality goals" and "quality of service requirements". Qualities, of Non-functional requirements can be divided into two main categories. Execution qualities, such as security and usability, are observable at run time. Evolution qualities, such as extensibility and scalability, embody in the static structure of the software system.

## Software Quality Attributes

The main goals can be accomplished by standing true to the below main pillars :

(i) **Reliability:** Reliability will mostly depend on the client's connection status.

The Systems Reliability solely relies upon its key files and folders.

(ii) **Availability:** The server on which this system will be running is expected

to be available at all hours of the day to provide worldwide accessibility.

A graphical user interface (GUI) is provided to have online interaction with the user.

(iii) **Maintainability:** The business logic here is that there can be accounting

feature in the future development of the product.. All development will be provided with good documentation.

(iv) **Portability:** As the system is designed using Java, it can work on any platform

Or architecture

Chapter : 8

Project Planning and Scheduling

**RENT MANAGEMENT SYSTEM**



## **Project Planning & Scheduling:-**

### **PERT Chart/ Task Network Chart:-**

PERT stands for the Program Evaluation Review Technique , is a project management tool used in project planning to estimate time. A single project can be divided into various sub- tasks , to complete all of them in time , we estimates each subtask completion time and then group them or sequence them that is the most favourable for the process of development It was made by the U.S. Navy in 1950's to manage the Polaris Submarine Missile Programme.

#### **Working Rule of PERT Chart :**

This chart is a graphical representation of subtasks in a network diagram . All the subtasks are Linked by the labelled vectors . Directions Show us the sequence of the tasks . We arrange the tasks in such a way that all activities are finished in managed way in least possible time.

#### **Steps in making PERT Chart :**

Step 1 : We will list and define all the activities involved in making the System.

Step 2 : We will portray any dependency that is involved in any of the tasks.

Step 3: We will draw the tasks as nodes .

Step 4: We will make the Connection between all the nodes ,  
this will indicate the various path combinations .

Step 5 : Identify the process and its estimate of the completion time .

Step 6 : We select that Route in Which all subtasks are covered in the least time

## Advantages of PERT Chart :

1. We can improve the sequence of work we have to finish work in minimum time.
2. Improves planning and decision making.
3. Uncertainty can be identified , managed and resolved.
4. We can minimise the delay in the development process.

The tasks identified in the proposed System : Rent Management System

- (i) Login Module
- (ii) Room Management Module
- (iii) Consumer / Tenant Management Module
- (iv) Agreement Module
- (v) Rent Collection Module
- (vi). Produce Reports and Data Backup Module
- (vii) Change Password Module

**We will make the Plan keeping the above modules as the Subtasks :**

| <u>Activity</u> | <u>Activity Name</u>                             |
|-----------------|--------------------------------------------------|
| A               | Login by Landlord                                |
| B               | Room Management                                  |
| C               | Consumer / Tenant Management Module              |
| D               | Agreement Module                                 |
| E               | Rent Collection module With Payments Integration |
| F               | Reports and Data Backup Module                   |
| G               | Change Password                                  |

**We Make the Activities in a tabular form and mention the time estimates in weeks**

| <b>Activity</b> | <b>Predecessor</b> | <b>Time Estimated Weeks</b> |
|-----------------|--------------------|-----------------------------|
|                 |                    |                             |

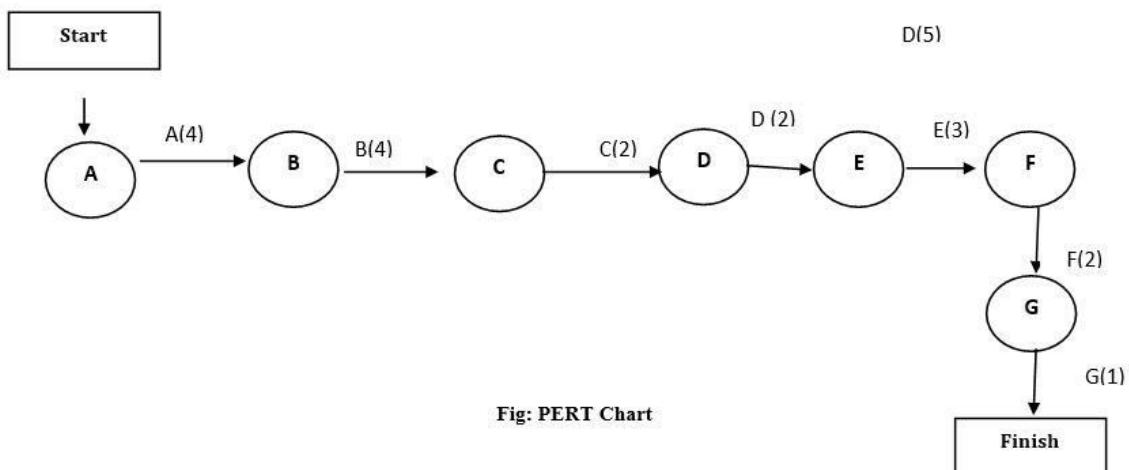
|                              | <b>Activity</b> | <b>(Individual)</b> |
|------------------------------|-----------------|---------------------|
| A                            | .....           | 4                   |
| B                            | A               | 4                   |
| C                            | B               | 2                   |
| D                            | C               | 2                   |
| E                            | D               | 3                   |
| F                            | E               | 2                   |
| G                            | F               | 1                   |
| Total Time Estimated (Weeks) |                 | 18                  |

### Critical Path Method (CPM) :

CPM provides a structured framework for monitoring the progress of a project. By comparing actual progress against the planned schedule. This allows project managers to calculate the earliest and latest start and finish times for each activity, helping them determine the overall duration of the project. This helps in setting realistic project timelines and managing stakeholder expectations.

### Critical Path Method (CPM): -

#### Critical Path Method (CPM): -



### Time Line Path: -

| <b>Path</b>                | <b>Length Of Time</b>  |
|----------------------------|------------------------|
| Start-A-B-C-D-E-F-G-Finish | 4+4+2+2+3+2+1= 18WEEKS |

## GANTT Chart:-

A Gantt chart is a popular type of bar chart that illustrates a project schedule. Gantt charts illustrate the start and finish dates of the terminal elements and summary elements of a project. Terminal elements and summary elements comprise the work breakdown structure of the project.

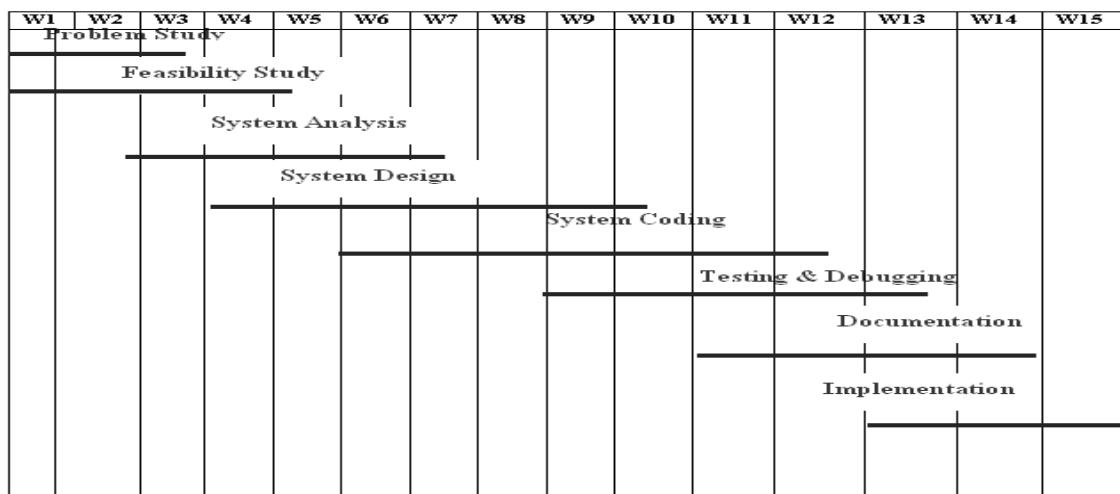


Fig: GANTT Chart

| Task | Name of the Task               | Time of the Task (weeks) |
|------|--------------------------------|--------------------------|
| A    | Login Module                   | 4                        |
| B    | Room Management Module         | 4                        |
| C    | Tenant Management Module       | 2                        |
| D    | Agreement Module               | 2                        |
| E    | Rent Collection Module         | 3                        |
| F    | Reports and Data Backup Module | 2                        |
| G    | Change Password Module         | 1                        |

We will Refer to the Table of Tasks and Plan the Subtasks with Accordance to the Weeks Assigned To them

| Task | Main Task and Below Subtasks    | 4 Weeks | 4 Weeks | 2 Weeks | 2 Weeks | 3 Weeks | 2 Weeks | 1 Weeks |
|------|---------------------------------|---------|---------|---------|---------|---------|---------|---------|
| A    | 1 Login Module                  |         |         |         |         |         |         |         |
|      | 1.1 Designing                   |         |         |         |         |         |         |         |
|      | 1.2 Coding And Testing          |         |         |         |         |         |         |         |
|      | 1.3 Implementing                |         |         |         |         |         |         |         |
|      | 1.4 Documenting                 |         |         |         |         |         |         |         |
| B    | 2 Room Management Module        |         |         |         |         |         |         |         |
|      | 2.1 Designing                   |         |         |         |         |         |         |         |
|      | 2.2 Coding and Testing          |         |         |         |         |         |         |         |
|      | 2.3 Implementing                |         |         |         |         |         |         |         |
|      | 2.4 Documenting                 |         |         |         |         |         |         |         |
| C    | 3 Tenant Management Module      |         |         |         |         |         |         |         |
|      | 3.1 Designing                   |         |         |         |         |         |         |         |
|      | 3.2 Coding and Testing          |         |         |         |         |         |         |         |
|      | 3.3 Implementing                |         |         |         |         |         |         |         |
|      | 3.4 Documenting                 |         |         |         |         |         |         |         |
| D    | 4 Agreement Module              |         |         |         |         |         |         |         |
|      | 4.1 Designing                   |         |         |         |         |         |         |         |
|      | 4.2 Coding and Testing          |         |         |         |         |         |         |         |
|      | 4.3 Implementing                |         |         |         |         |         |         |         |
|      | 4.4 Documenting                 |         |         |         |         |         |         |         |
| E    | 5 Rent Collection Module        |         |         |         |         |         |         |         |
|      | 5.1 Designing                   |         |         |         |         |         |         |         |
|      | 5.2 Coding and Testing          |         |         |         |         |         |         |         |
|      | 5.3 Implementing                |         |         |         |         |         |         |         |
|      | 5.4 Documenting                 |         |         |         |         |         |         |         |
| F    | 6 Reports and DataBackup Module |         |         |         |         |         |         |         |
|      | 6.1 Designing                   |         |         |         |         |         |         |         |
|      | 6.2 Coding and Testing          |         |         |         |         |         |         |         |
|      | 6.3 Implementing                |         |         |         |         |         |         |         |
|      | 6.4 Documenting                 |         |         |         |         |         |         |         |
| G    | 7 Change Password Module        |         |         |         |         |         |         |         |

|  |                        |  |  |  |  |  |  |  |
|--|------------------------|--|--|--|--|--|--|--|
|  | 7.1 Designing          |  |  |  |  |  |  |  |
|  | 7.2 Coding and Testing |  |  |  |  |  |  |  |
|  | 7.3 Implementing       |  |  |  |  |  |  |  |
|  | 7.4 Documenting        |  |  |  |  |  |  |  |

Chapter : 9

## System Analysis

# RENT MANAGEMENT SYSTEM



## Analysis : Rent Management System

The whole approach of analysis of problem should however be based around critical factors like the availability of information for making the decision, the time available for processing the data i.e. the realism. System Requirement Specification or SRS had been prepared after proper discussion with the persons attached with the mentioned “OSCM”. Software project management begins with a set of activities collectively called PROJECT PLANNING. Software project planning actually encompasses all of the activities. Planning involves estimation- to determine how much money, how much effort, how many resources, and how much time it will take to build a specific software-based system or product.

### Phases Covered:-

- (i) Pre–Analysis Studies
- (ii) System Analysis
- (iii) System Design
- (iv) Project Coding
- (v) Project Testing
- (vi) Implementation & Documentation

These Phases are described as follows:

- (i) **Pre–Analysis Phase:** In this phase problems with existing system are to be determined and do the investigation to make the solutions.
- (ii) **System Analysis Phase:** In this phase system analysis is done by preparation of Software Requirement Specification.
- (iii) **System Design Phase:** The purpose of the design phase is to plan a solution for the problem specified in the requirements documents.

- (iv) **Project Coding Phase:** The goal of the coding phase is to translate the design of the system into a program code by help of a programming language like Visual Studio, Sublime Text ,Atom etc.
- (v) **Project Testing Phase:** Testing concerned with the elimination of errors introduced during coding phase.
- (vi) **Implementation & Documentation Phase:** - This phase includes all the activities performed to keep the system operational after the installation of the software.

Chapter : 10

Use Case Diagrams

**RENT MANAGEMENT SYSTEM**



## Rent Management System : Use Case / Behaviour Diagram

We use the Tool Use Case Model for analysis of behaviour of the System. It is a part of Unified Modelling Language. The Functional Requirements of the System and its interaction woth external agents (actors) are used. A Use Case Diagram gives us high level view of System without going in the details of the implementation.

The figure below illustrates the Use Case Model of Proposed System :

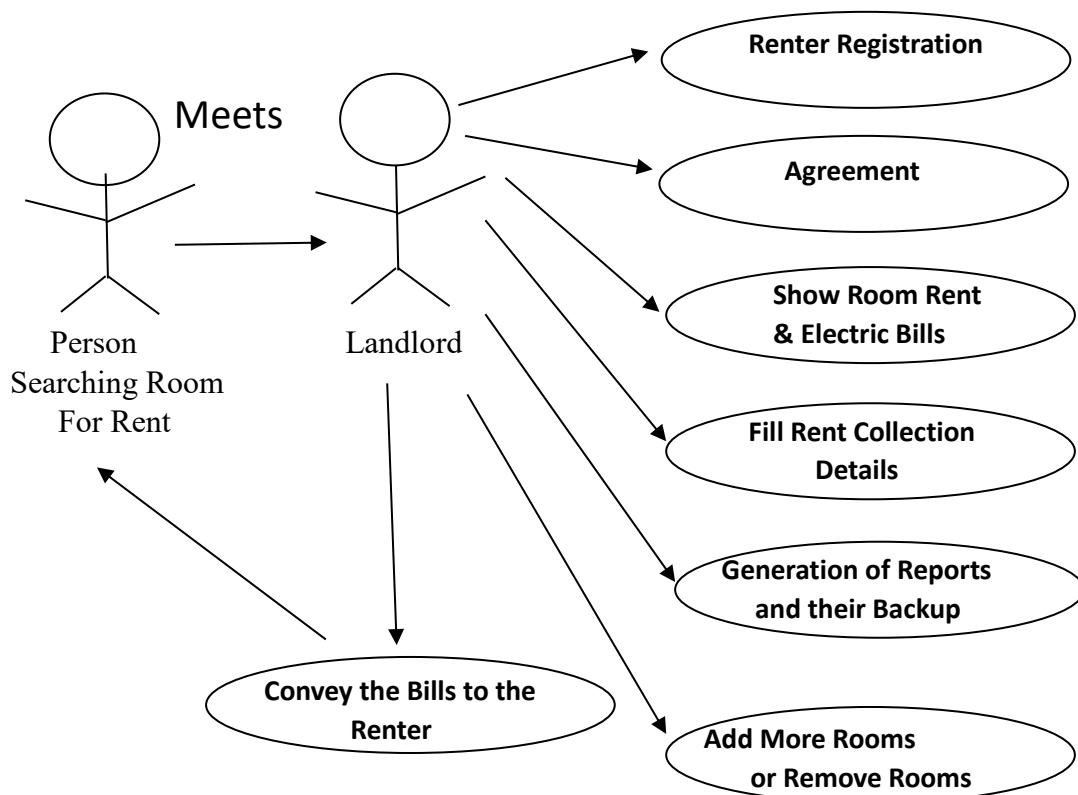


Fig : Use Case Diagram : RENT MANAGEMENT SYSTEM

**Chapter : 11**

**DFD 0<sup>th</sup> Level , Level-1 & Level-2**

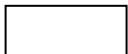
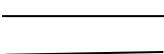
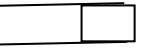
**RENT MANAGEMENT SYSTEM**



## Data Flow Diagram (DFDs):-

DFD stands for Data Flow Diagrams . This is also known as Bubble Charts through which we can represent the flow of data graphically in any information management system. We can easily understand the overall functionality of system because diagram represents the incoming dataflow and the outgoing dataflow and Stored data in the Graphical Form. It describes the flow of data in terms of input and output .

A DFD uses a number of notations or symbols that represent the flow of data in any System. These notations are illustrated as follows:

| Serial No. | Name of Symbol         | Figure/Symbol                                                                                                                                                                | Purpose                          |
|------------|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------|
| 1          | Entity                 |                                                                                            | A person / Entity                |
| 2          | Direction of Data Flow |                                                                                          | Flow of Data from Entity/Process |
| 3          | Process or Bubble      |                                                                                          | Signify the process of a Module  |
| 4          | Data Store             |  OR  | Place of Storing Data            |

The Rules We Follow while preparing the DFD's are as follows :

1. Each Process Should have atleast one input and one output.
2. Each Data Store should have atleast one data flow in and one data flow out.
3. All Process in DFD either go to another process or to another Data Store.

## 0 th Level DFD :

This is the highest level DFD which provides overview of entire System. It shows the major processes , data flow and the data stores in the System. We also call it the Context Diagram. Here we can see the entire System working as a Complete Blackbox to which user provides input , processing is carried out by the system and the result is produced . Thus, Providing the Business the Desireable Advantages .

The 0th Level DFD illustrates the Rent Management System as :

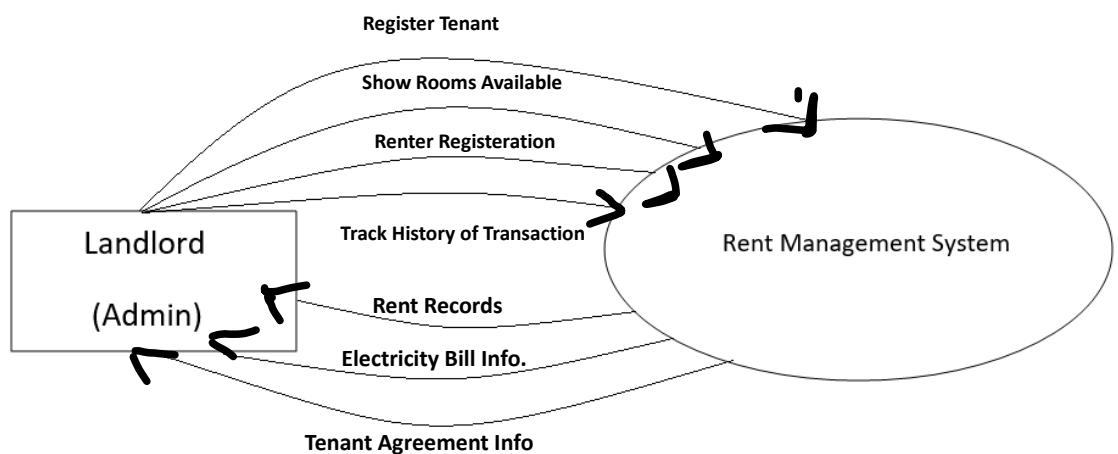


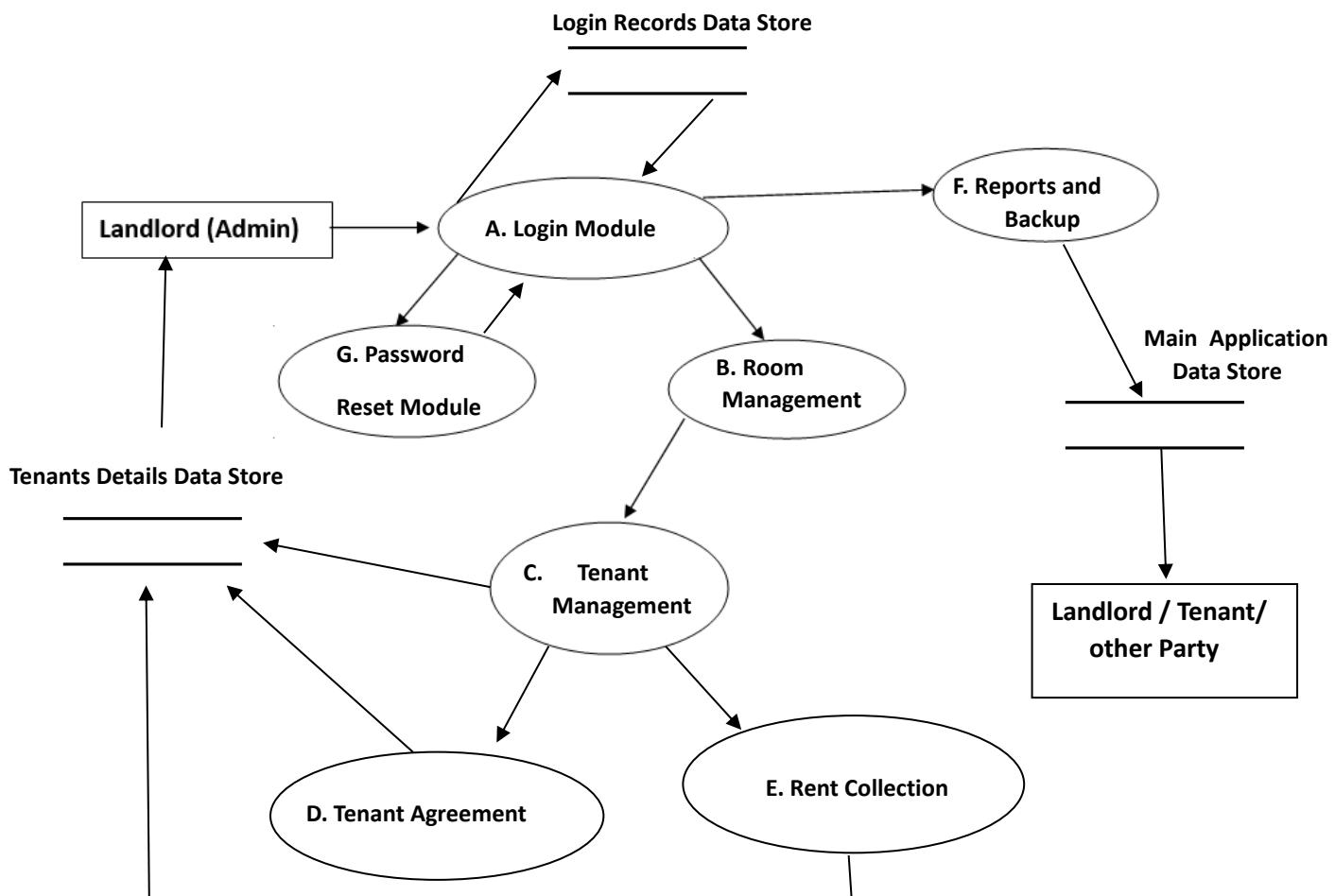
Fig : 0<sup>th</sup> Level DFD : Rent Management System

## 1 th Level DFD :

In 1 th Level DFD we decompose the Entire System or the Context Diagram That breaks into multiple branches of modules or sub processes . In this Level DFD all the Processes are rendered by the Modules that are present in the table of modules that is as follows :

| Task | Name of the Task               |
|------|--------------------------------|
| A    | Login Module                   |
| B    | Room Management Module         |
| C    | Tenant Management Module       |
| D    | Agreement Module               |
| E    | Rent Collection Module         |
| F    | Reports and Data Backup Module |
| G    | Change Password Module         |

The 1 th Level DFD illustrates the Rent Management System as :

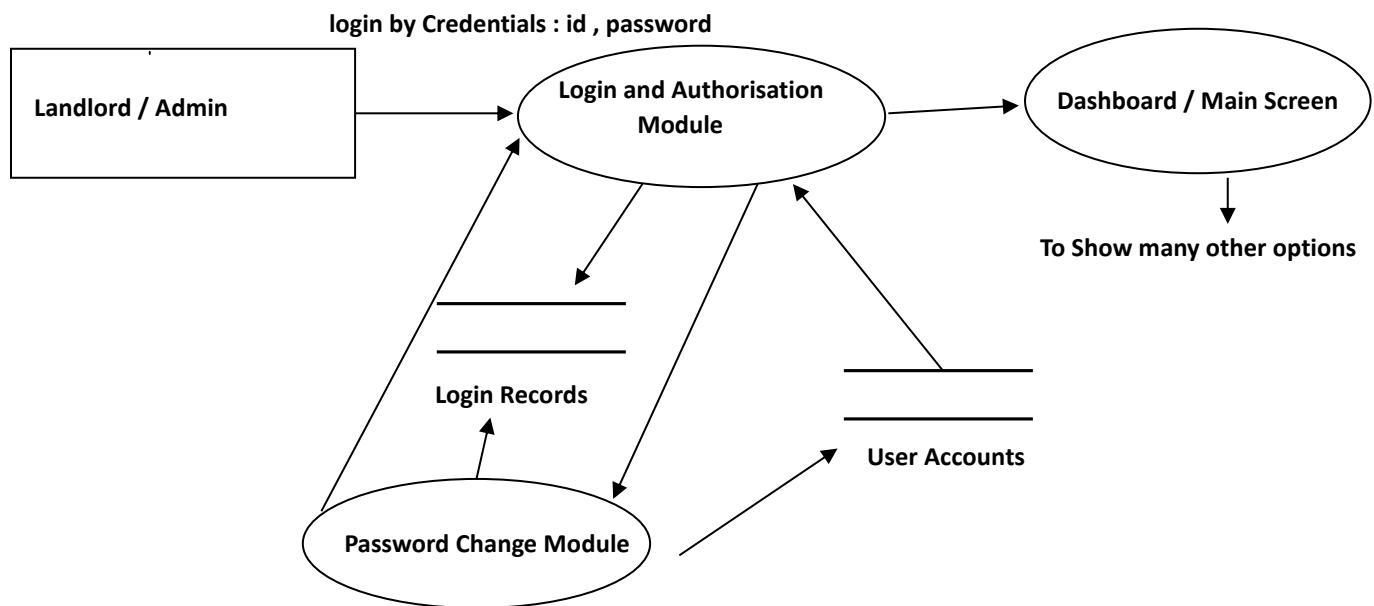


## 2 th Level DFD :

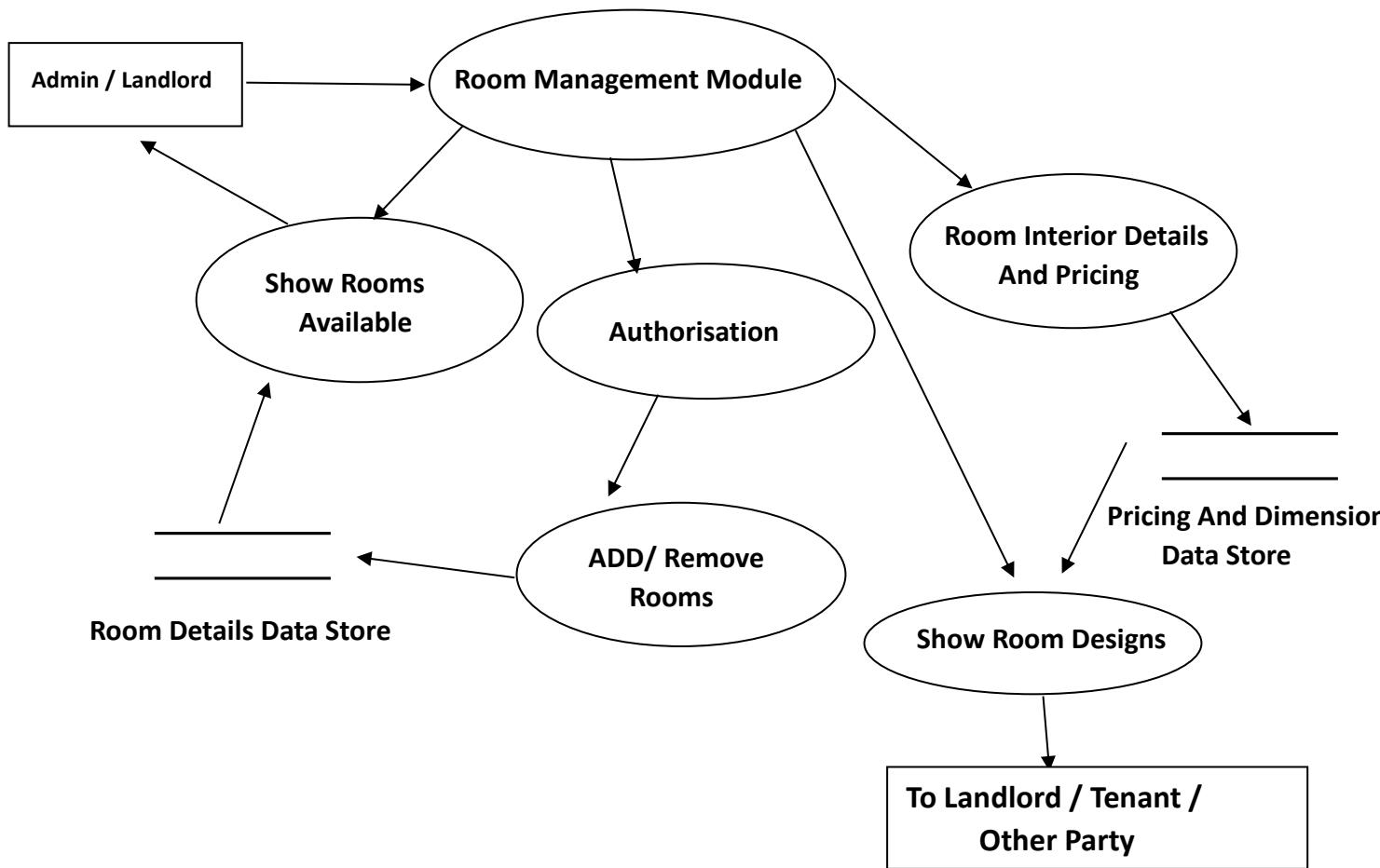
In 2th Level DFD , We go one step deep into 1th Level DFD . Thus , illustrating the Sub Processes and their dependencies in each Module. This is necessary as it is used to plan or record specific details about system functioning . Since , the Rent Management System Comprises of 7 Modules, Their individual DFD helps us to know their Internal Data Flow better.

The 2th Level DFD of Each Sub System / Module of Rent Management System are illustrated as follows:

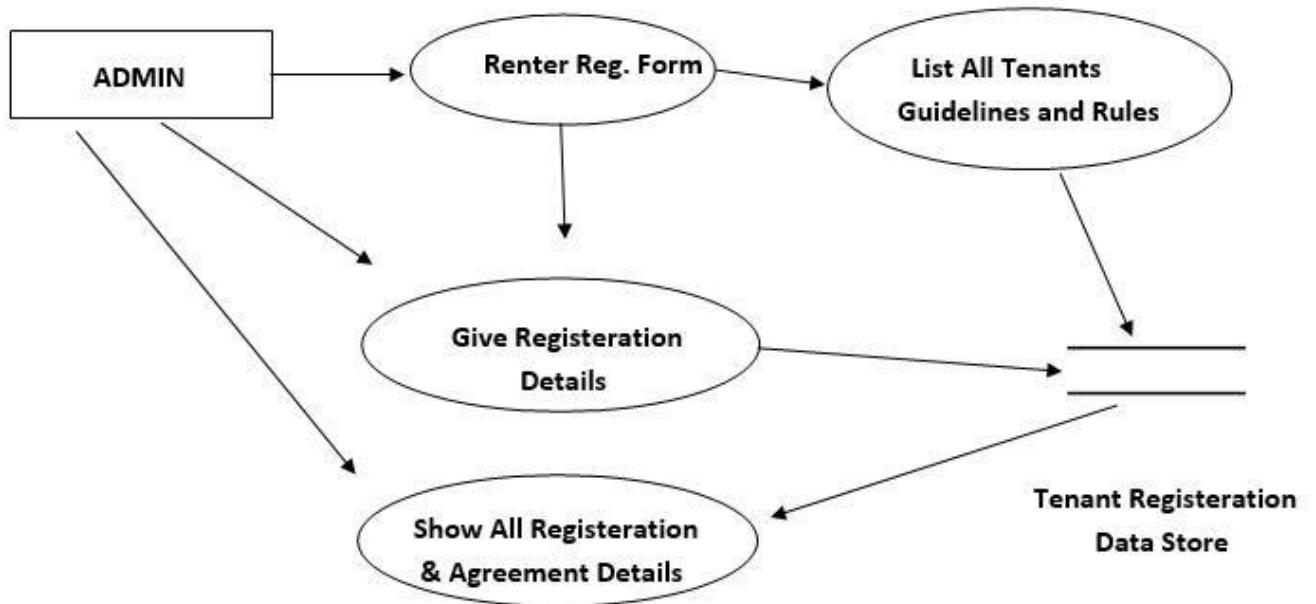
| Task | Name of the Task               |
|------|--------------------------------|
| A    | Login Module                   |
| B    | Room Management Module         |
| C    | Tenant Management Module       |
| D    | Agreement Module               |
| E    | Rent Collection Module         |
| F    | Reports and Data Backup Module |
| G    | Change Password Module         |



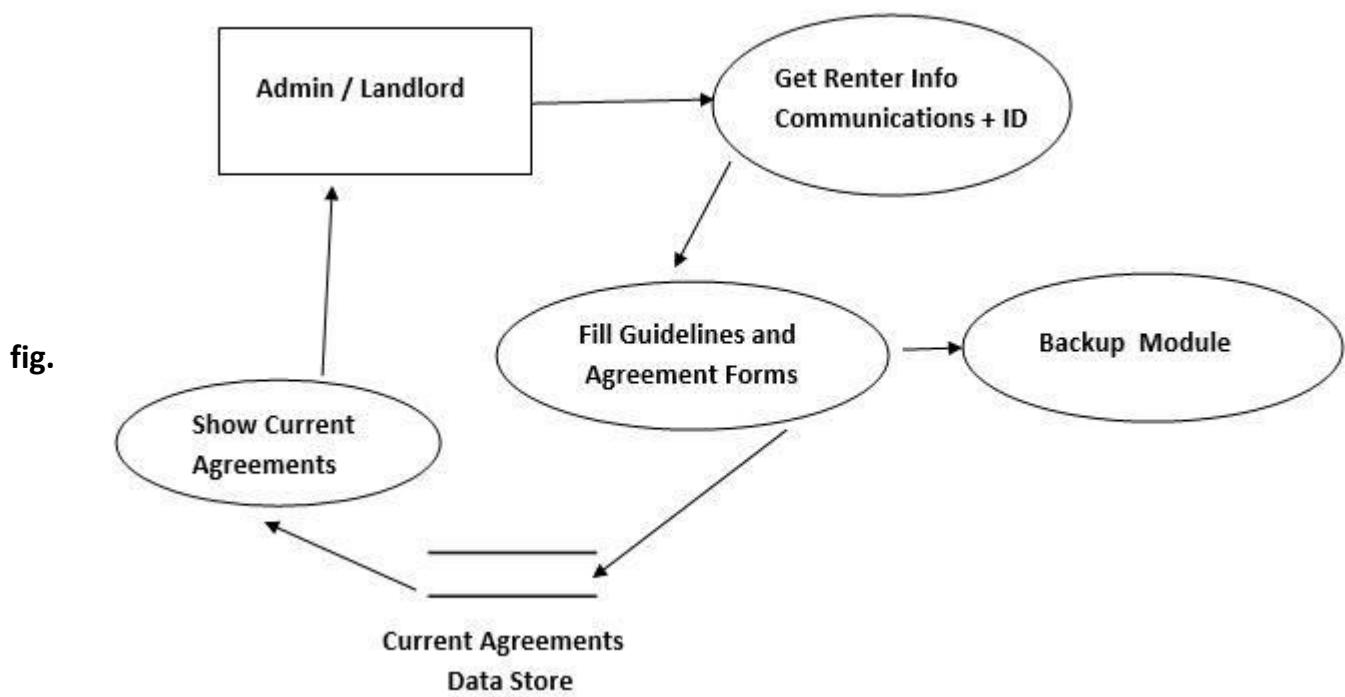
**fig. Module A : illustration of the Login Module**



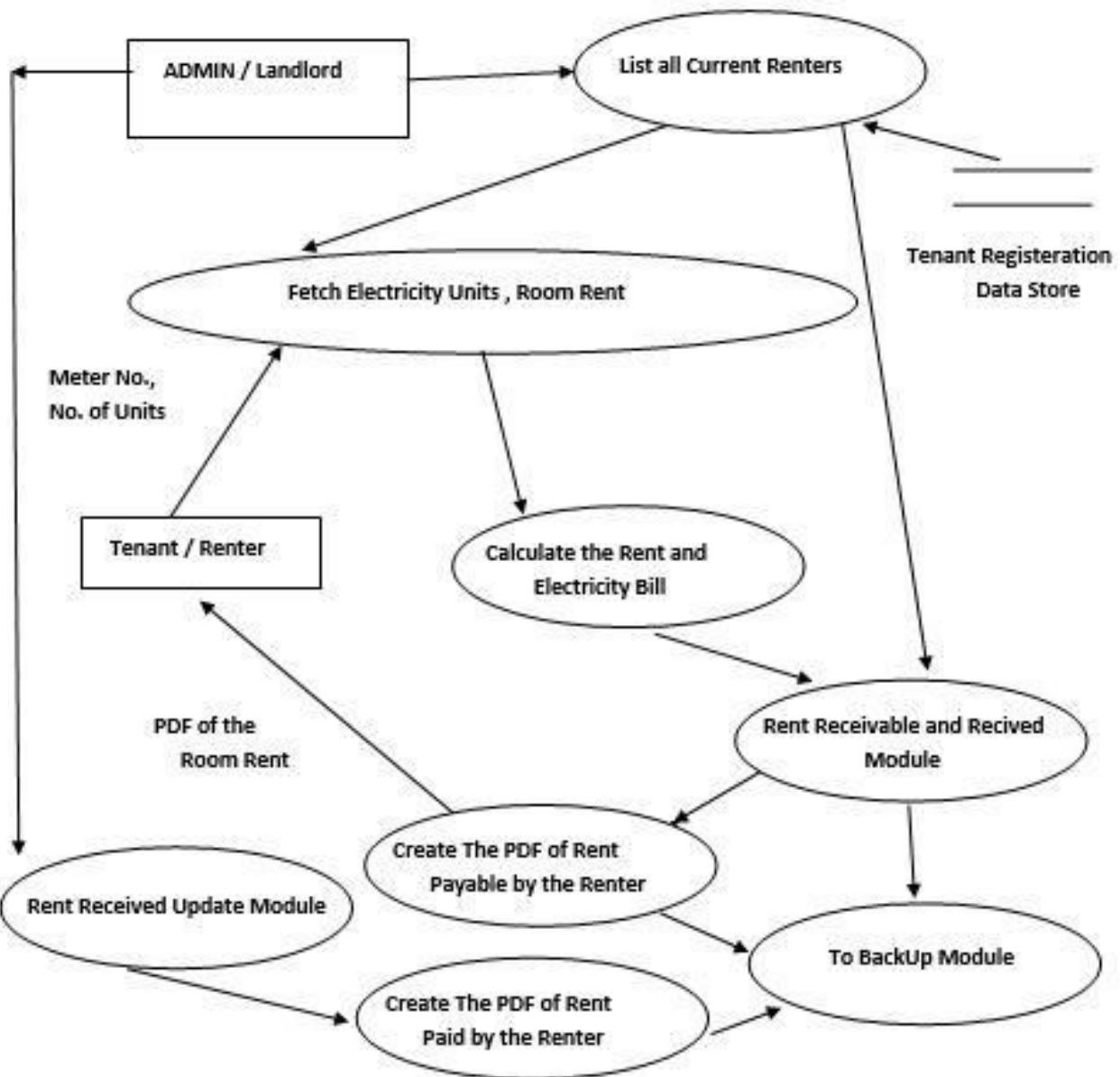
**fig. Module B : illustration of the Room Management Module**



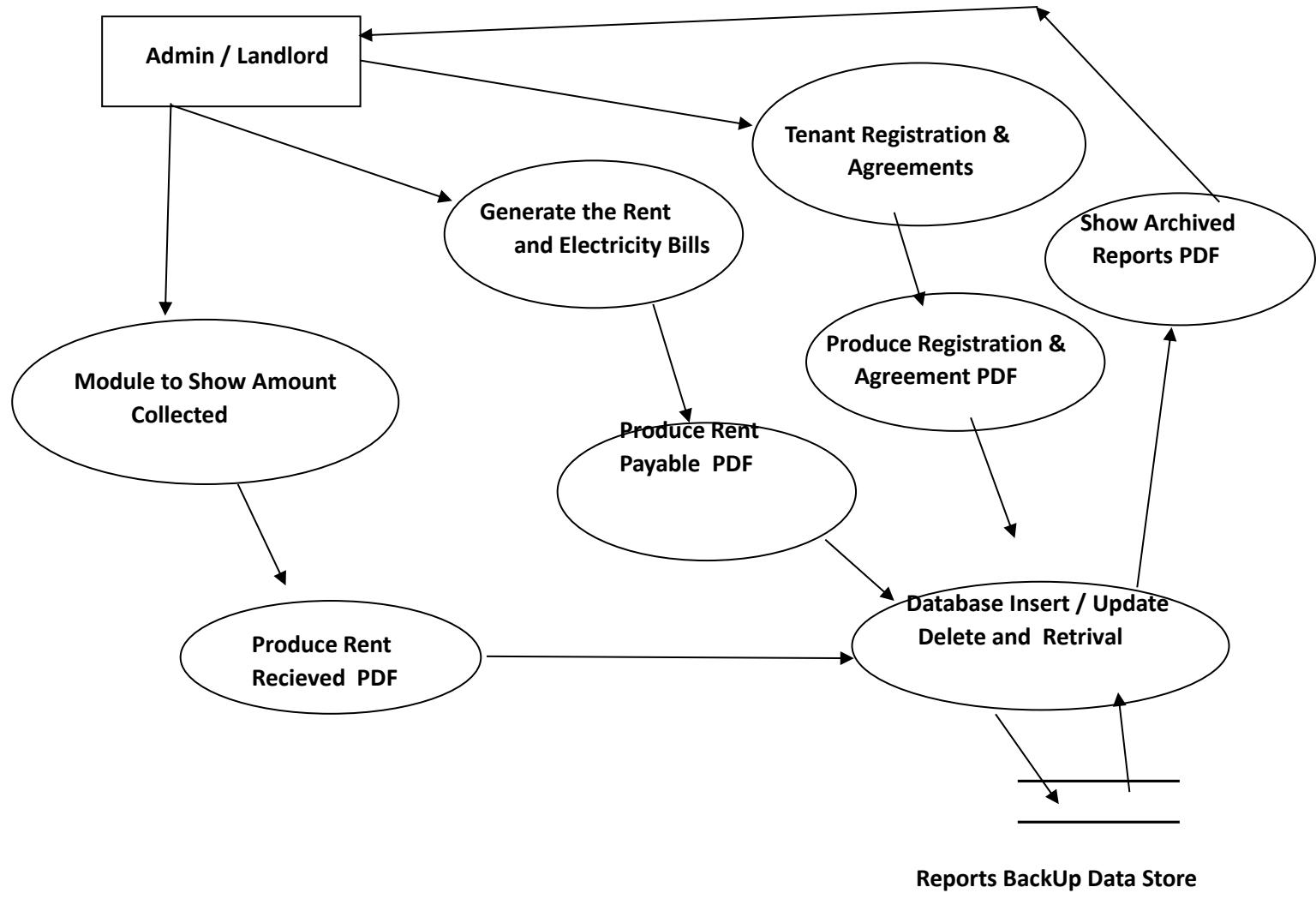
**fig. Module C : illustration of the Tenant Management Module**



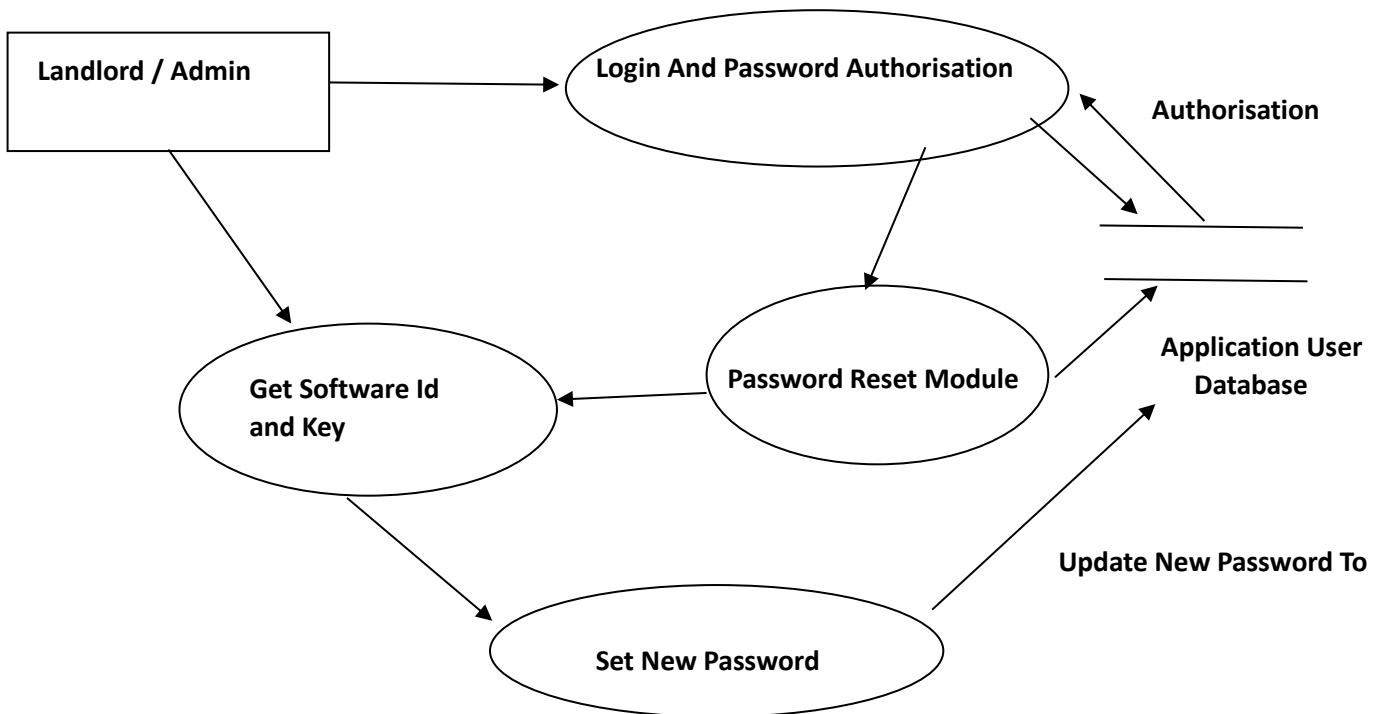
**Module D : illustration of the Tenant Agreement Module**



**fig. Module E : illustration of the Rent Collection Module**



**fig. Module F : illustration of the Rent Collection Module**



**fig. Module G : illustration of the Rent Collection Module**

Chapter : 12

Entity Relationship Diagrams & SQL Tables

**RENT MANAGEMENT SYSTEM**



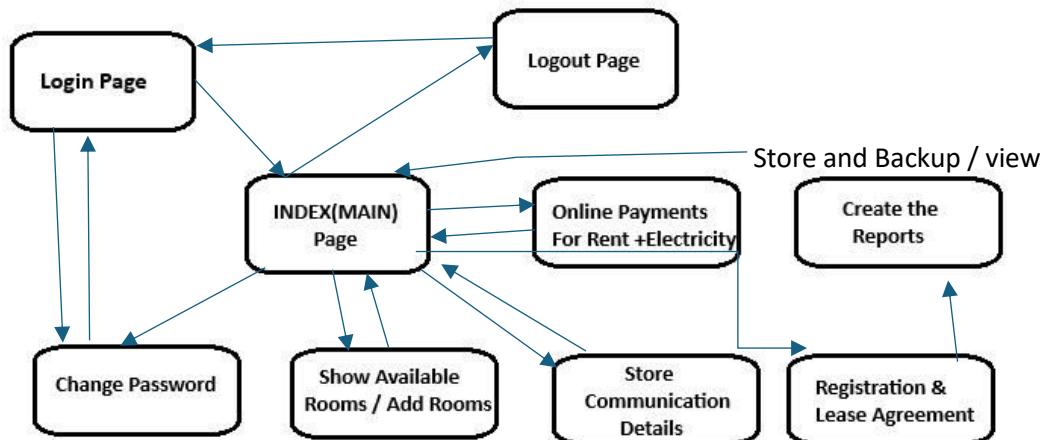
## Introduction of Entity Relationship Diagrams of the Rent Management System

An Entity-Relationship (ER) diagram is a type of structural diagram for use in database designing for the Rent Management System. It provides graphical representation of the entities involved in a system and relations between them on the basis of the attributes they possess and need to be stored inside of the database.

The Database involved in the Rent Management System is the MySQL lite that comes along with the Xammp Server that provides us ease in clearly making the designs for the database that will be utilised to store the data in format with the special attributes that are involved within the entities of the System that can be the landlord , Tenant , Rooms , Bills etc.

For the preparation of the Required ER – Diagrams we follow the below Graphical Scheme inorder to produce the precise Designs for the Database for the Rent Management System. This Diagram serves as the blueprint ensuring that all necessary data relationships are efficiently Accounted.

Beginning to Design the ER model , we must consider the flow of the process inside the Rent Management System with the help of the Site map that is as follows.



The Site Map Depicts the Rent Management System's key pages that involve to deal with the main entities of the Rent Management System.

The Below Graphical Representations Identify the Main Entities involved in the System and also the Relations between them.

### Entity

An Entity is defined as the object , being the part of the System has some attributes that are likeable to be stored in the database inorder to carry out the necessary tasks.

An Entity that is the part of the system is depicted as the square / rectangle shape



Fig. : Entity in ER Diagram

### Attribute

Attributes are defined of an entity that are crucial points over which necessary actions are required to be carried out by the system. This is the clear feature or the set of values that can help the entity be identified and differentiated with the other entities involved inside the system.

An attribute in the system is depicted as the oval shaped .

There are also some sub categories inside the attributes : that are :

Derrived attributes : These attributes depend on the other attributes to make a decision.

Eg. : The Age is a Derrived Attribute type that can be derrived from the date of Birth.

Multivalued Attributes : These category can have more than one types and that are legally accepted by te System .

Eg. : The Online Payment can be Done by various Online Payment Interfaces that will create the transactions with the System.

The Attributes and their sub types are depicted as follows:



Single valued / Simple



Multi Valued



Derrived Attribute

### Links between Two Entities

The Links are depicted by a single straight line , they define the relation between the two / more entities , they also bind the entity with their attributes and with the relationships that pair them respectively.

Fig . The Link between Two/ More Entity in the ER Diagram

## Relation

The Relation is depicted by the shape of the rhombus that contains a keyword of English in the center of the Place.



Fig.: The Relation Specifying Symbol in the ER Diagram

The Relations in the Real World Scenarios are of many types:

- (i) One to One Relation
- (ii) One to Many Relation
- (iii) Many to One Relation
- (iv) Many to Many Relation

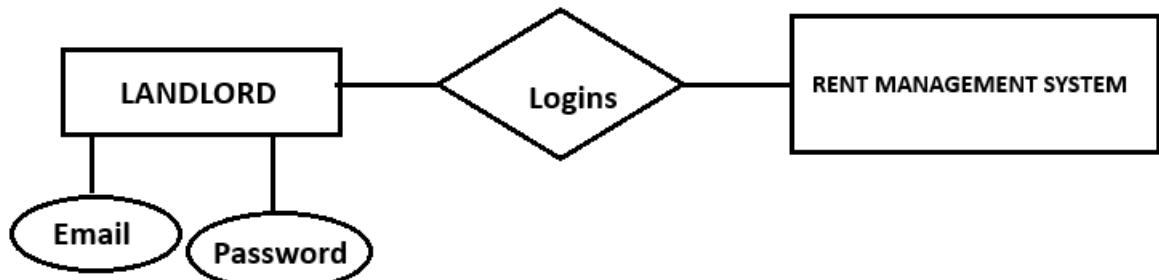
These are exemplified as follows:

- (i) One to One Relation : One Renter has One Identity Card for Registration.
- (ii) One To Many Relation : One landlord is Capable to manage many renters by the help of the Rent Management System.
- (iii) Many to One Relation : Many Customers use the Same Payments Gateway Interface for their online transactions.
- (iv) Many to Many Relation : A student can register for many classes, and a class can include many students.

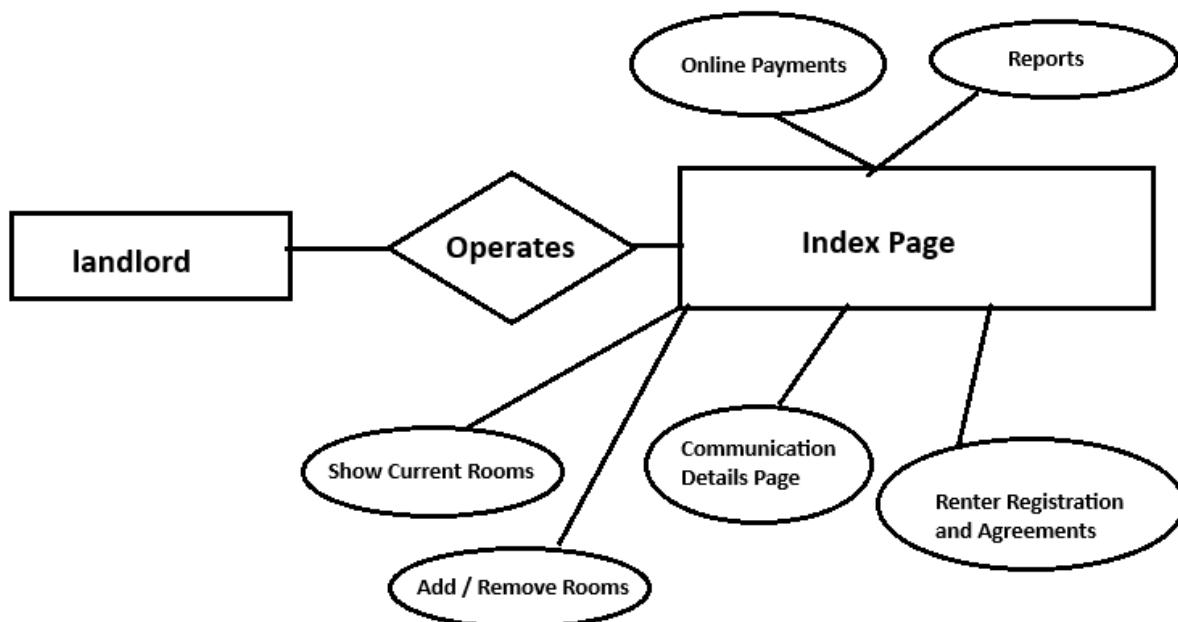
## Login Page

The Landlord Logins using the Email and the Password into the system.

One LandLord has One Email and one Password.

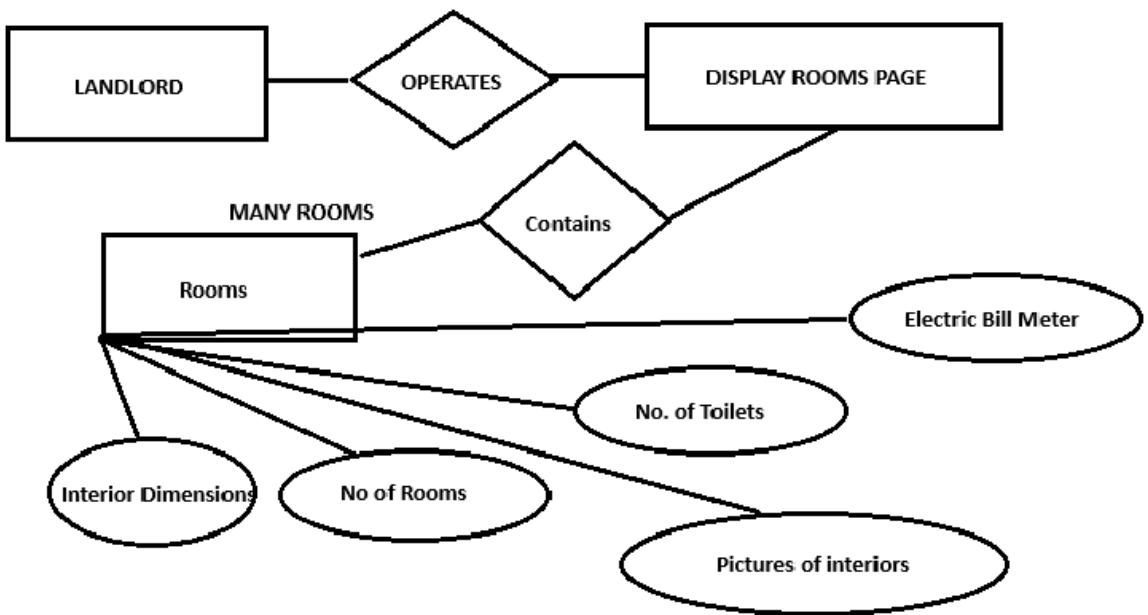


Index Page is the Main Page for the Rent Management System that Comprises of the various Pages that carry out the various tasks.



## Display Rooms Page

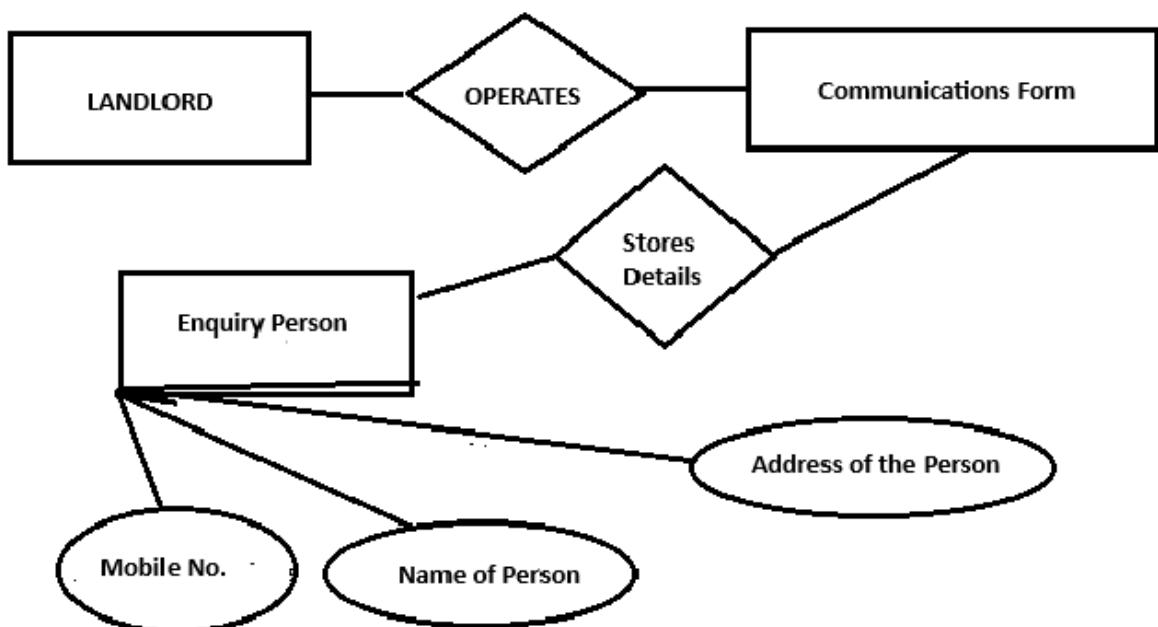
The Display Rooms page helps the Landlord to showcase the Rooms that are in his property that are available for the renting purpose , This will be beneficial for the landlord to display to the customers for enquiry and also share over the internet for the purpose of online advertisement.



#### Store Details Page:

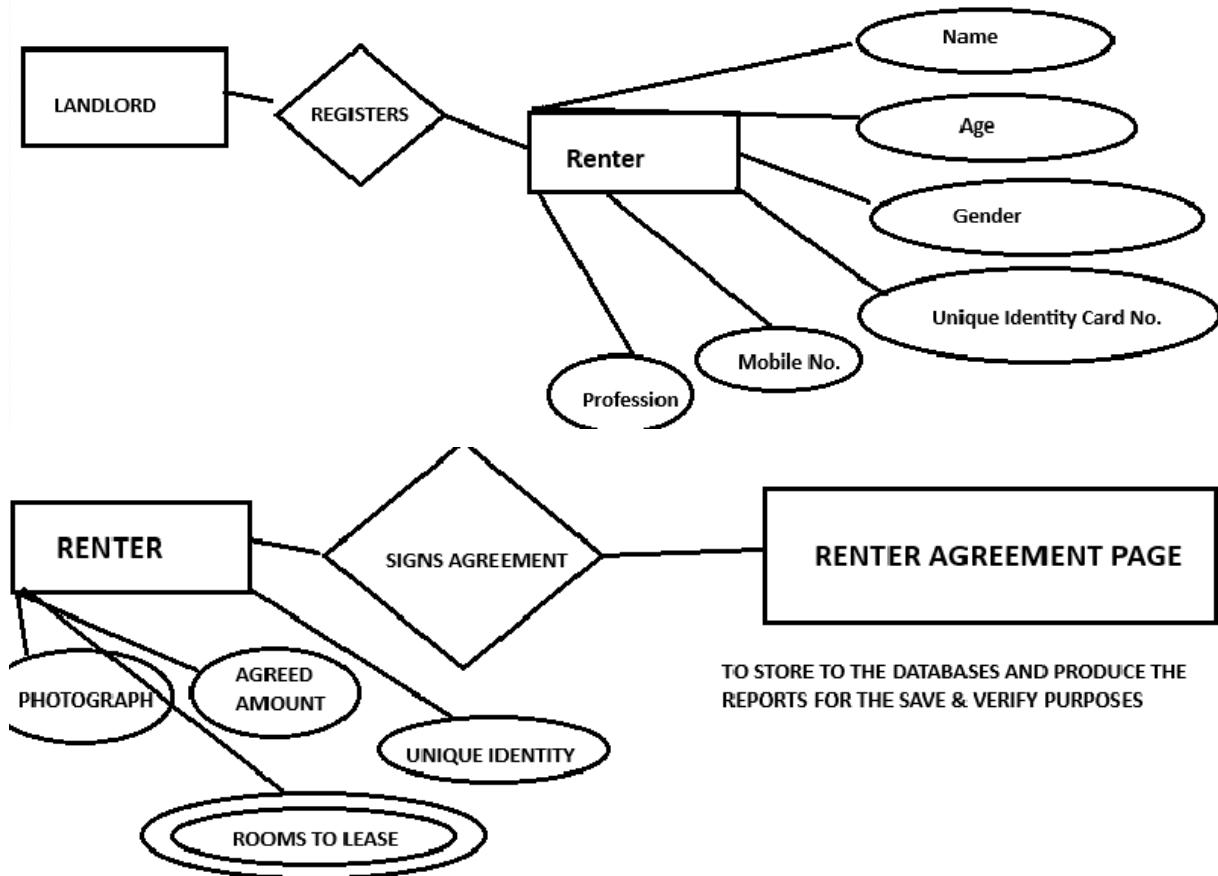
Sometimes , it is crucial to store the details of the consumers inorder to contact them in future , incase they are in need to find accomodations. There is always a chance that One who comes for enquiry , not be a customer , in case we may remind them that we have some rooms for renting, they may come to negotiate incase they are interested , or in dire need of.

This page is a form that will store the necessary Details like the Name and the Mobile no. , by which future communication may be established .



## Renter Registration and Agreement Page

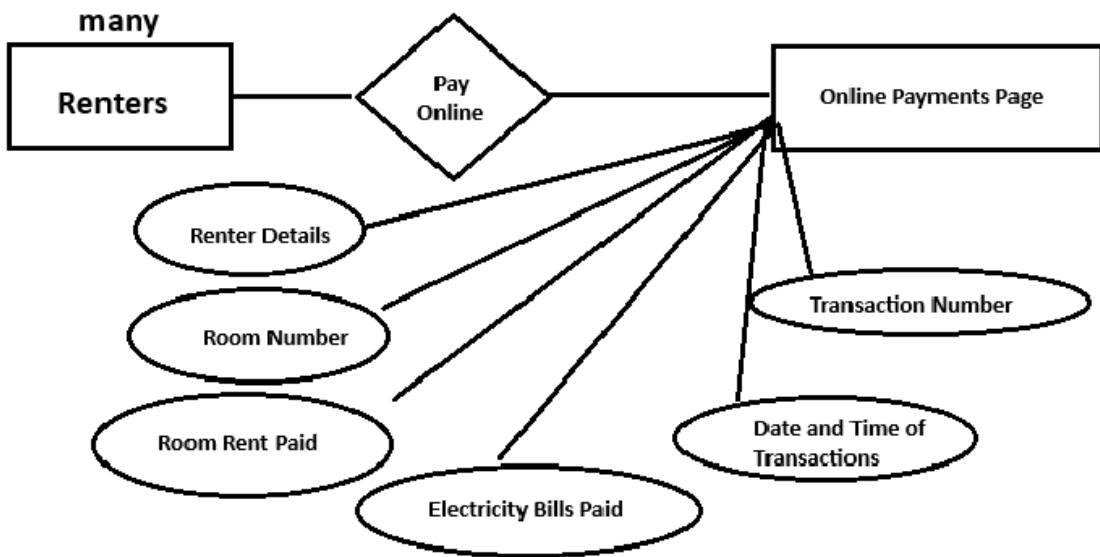
This is the most necessary and the crucial page for the purpose of legally storing the details of the consumer (Renter / Tenant) for the Rental Services that are provided by the Landlord(Service Provider). The Landlord Tends to Store the Necessary Details that must be Genuine for the purpose of occupying the room for the rent and for the safe future of the business.



## Online Payments Page

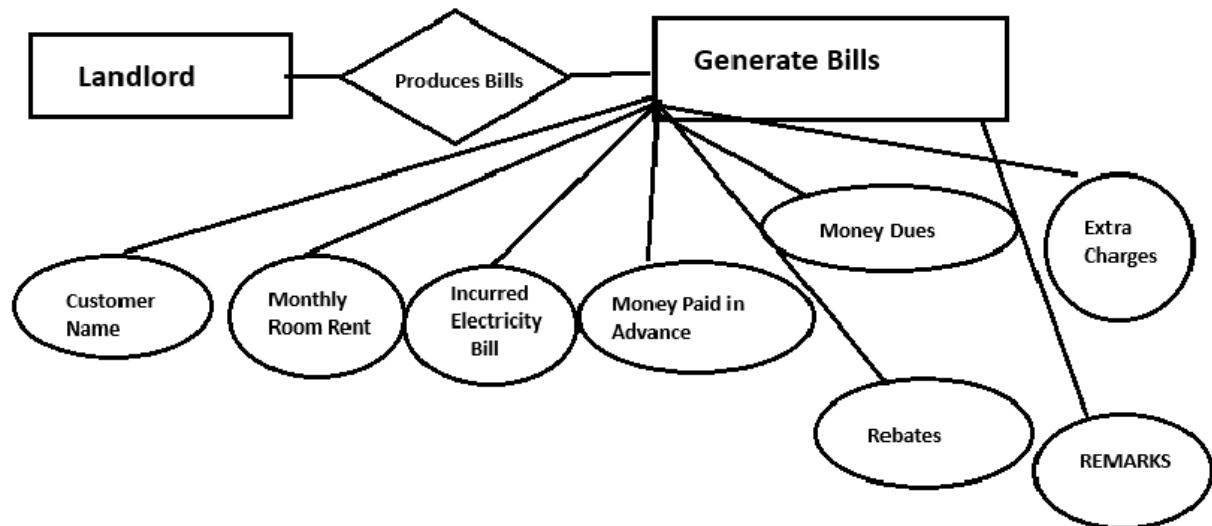
There will be many numbers of the renters to be managed by the System.

This page is responsible for the task for receiving the payments if made in online mode by the Renters . This will contain a Page containing the QR-code of the Landlord . In the best cases , this can be linked to secure portal for the purpose of Online Transactions that is also known as the Payments Gateway Integration with the Software.



### Produce Reports Page

This Page is important for the task to make the bills for the room renters and to send them , such that they get prepared about paying the monthly and the electricity bill keeping in mind the process of producing the bills.



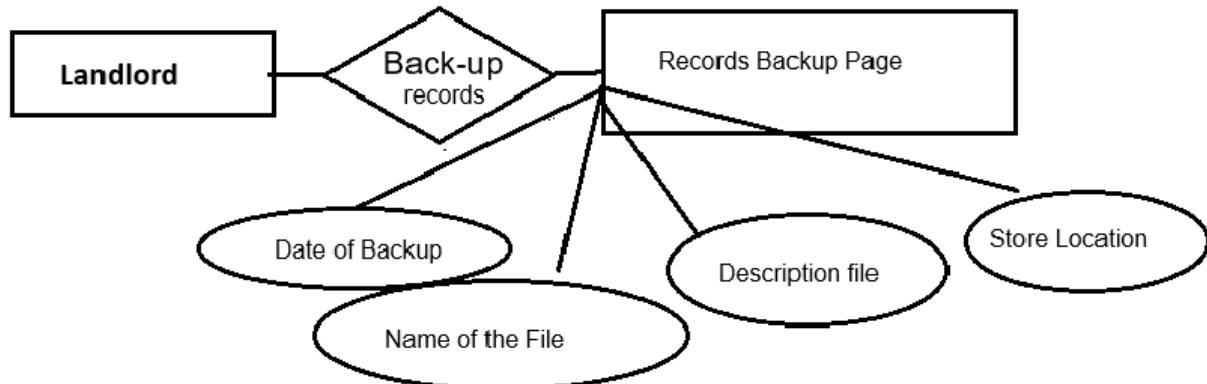
### The Reports Backup Page :

This Page is the crucial component in the Rent Management System. The Reports are made stored in the computer , send a copy to the renter , and also , the main file is stored inside the main computer. There is always an uncertain risk that can lead to loss of those sensitive files.

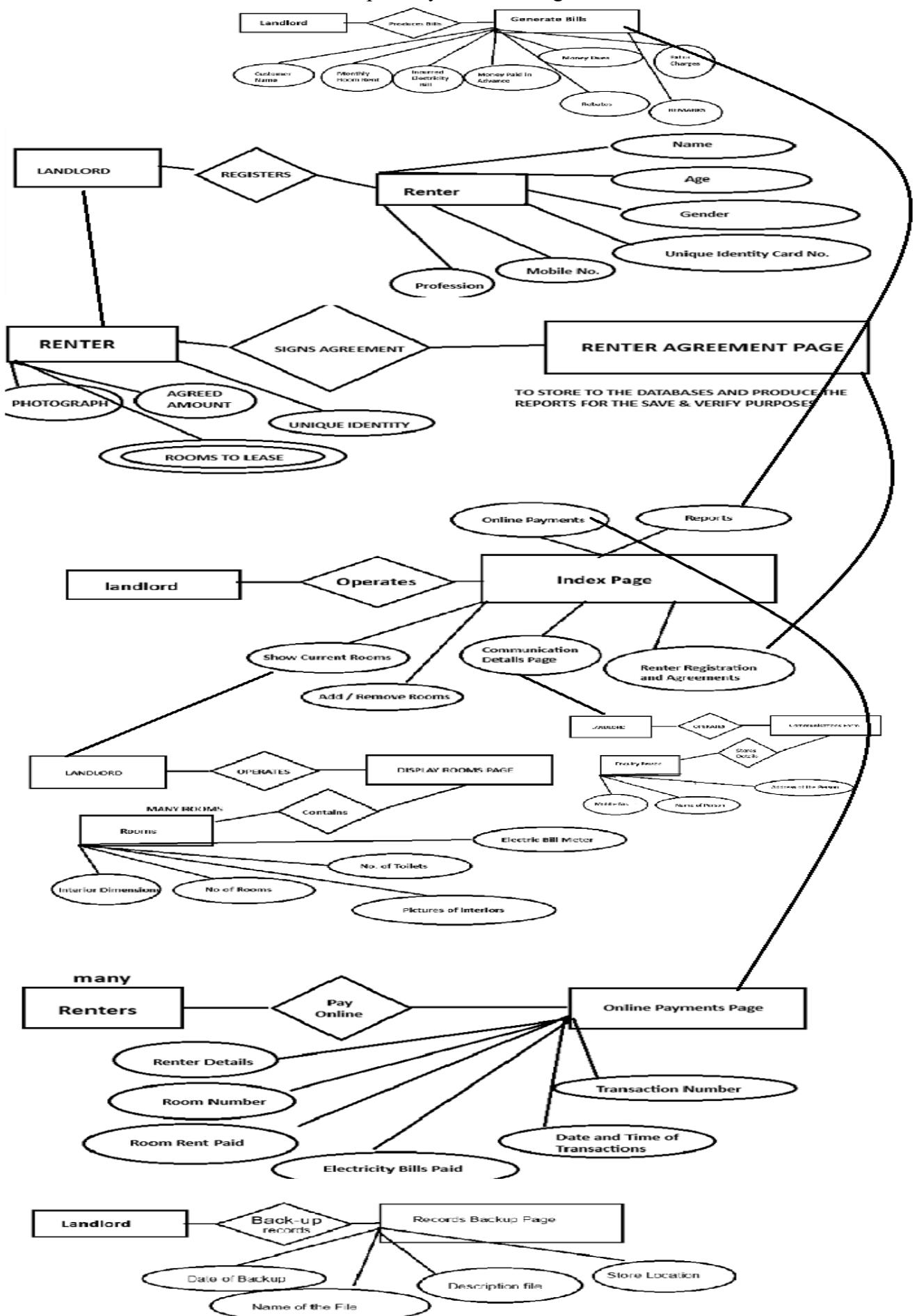
The term ‘backup’ means to reliably make a copy . The Agreements contain the crucial data , incase a critical error may cause its loss , and in future the problems may be faced.

This will utilise the File Management System to Store the Report with a Details file . Since all the files are going to be made in .pdf format , it will store it to a good location , where the landlord may copy all of them in a disk / or in the cloud storage .

This is also going to maintain the database records , keeping the log of the events that took place , what file was made the copy of , what were the details it was stored with and which location its copy was made. This provides the convinience to the Landlord to produce separate copies inorder to store to the place he may feel convinient.



## Complete System ER Diagram



## SQL Tables : For the Rent Management System

*Table name: Rooms Listing*

| <i>Columns</i>       | Data Types   |
|----------------------|--------------|
| <i>room_no</i>       | varchar(255) |
| <i>no_of_rooms</i>   | int          |
| <i>no_of_toilets</i> | int          |
| <i>Meter_no</i>      | varchar(255) |
| <i>Dimensions</i>    | varchar(255) |

*Table name: CommDetails*

| <i>Columns</i>    | Data Types      |
|-------------------|-----------------|
| <i>Mobile No.</i> | BigInt (unique) |
| <i>Name</i>       | Varchar(50)     |
| <i>Address</i>    | Varchar(50)     |

*Table name: RenterRegistration*

| <i>Columns</i>       | Data Types   |
|----------------------|--------------|
| <i>Name</i>          | Varchar(100) |
| <i>Age</i>           | Int          |
| <i>Gender</i>        | Varchar(20)  |
| <i>Unique Id</i>     | Varchar(25)  |
| <i>Profession</i>    | Varchar(255) |
| <i>Address</i>       | Varchar(255) |
| <i>Date of Birth</i> | Date         |

*Table name: RenterAgreement*

| <i>Columns</i>    | Data Types   |
|-------------------|--------------|
| <i>Name</i>       | Varchar(100) |
| <i>Age</i>        | Int          |
| <i>Gender</i>     | Varchar(20)  |
| <i>Unique Id</i>  | Varchar(25)  |
| <i>Profession</i> | Varchar(255) |
| <i>Address</i>    | Varchar(255) |
| <i>PhotoFace</i>  | Varchar(255) |
| <i>PhotoSign</i>  | Varchar(255) |
| <i>PhotoUID</i>   | Varchar(255) |

*T&Ccheck* | Varchar(10)

*Table name: OnlinePayments*

| <i>Columns</i>        | Data Types   |
|-----------------------|--------------|
| <i>Name</i>           | Varchar(100) |
| <i>Room No.</i>       | Int          |
| <i>Transaction id</i> | Varchar(20)  |
| <i>Room Rent</i>      | Varchar(25)  |
| <i>No.of.units</i>    | int          |
| <i>Cost Per Unit</i>  | Varchar(255) |
| <i>DatePayment</i>    | Date         |
| <i>Remarks</i>        | Varchar(255) |

*Table name: BillsCreated*

| <i>Columns</i>      | Data Types   |
|---------------------|--------------|
| <i>Name</i>         | Varchar(100) |
| <i>Room No.</i>     | Int          |
| <i>MonthlyRent</i>  | Decimal      |
| <i>ElectricBill</i> | Decimal      |
| <i>MoneyDues</i>    | Decimal      |
| <i>MoneyAdvance</i> | Decimal      |
| <i>Rebates</i>      | Decimal      |
| <i>ExtraCharges</i> | Decimal      |
| <i>Remarks</i>      | Varchar(255) |

*Table name: RecordsBackup*

| <i>Columns</i>       | Data Types   |
|----------------------|--------------|
| <i>Date</i>          | DateTime     |
| <i>File Name</i>     | Varchar(255) |
| <i>FileDesc</i>      | Varchar(255) |
| <i>LocationExist</i> | varchar(255) |
| <i>cpylocation</i>   | Varchar(255) |
| <i>Remarks</i>       | Varchar(255) |

Chapter : 13

Future Scope of The System

**RENT MANAGEMENT SYSTEM**



## Future Scope of the Rent Management System

As the Rent Management System evolves, there are several key areas where future enhancements and expansions can be made to further improve its functionality, user experience, and market reach. These future scopes can help the system adapt to changing market needs, integrate new technologies, and provide additional value to property managers, landlords, and tenants.

### Advanced Analytics and Reporting

**Predictive Analytics :** Implement machine learning algorithms to predict trends such as tenant turnover rates, rent price optimization, and maintenance needs.

**Customizable Reports :** Allow users to create and customize their reports based on specific criteria and data points.

**Dashboard Enhancements :** Develop more interactive and visually appealing dashboards to display key metrics and insights in real time.

### Enhanced Security Features

**Two Factor Authentication (2FA) :** Introduce two factor authentication to enhance the security of user accounts.

**Data Encryption :** Implement advanced encryption techniques for data at rest and in transit to ensure the highest level of data security.

**Security Audits :** Regular security audits and compliance checks to adhere to the latest security standards and regulations.

### Enhanced Payment Options

**Cryptocurrency Payments :** Introduce support for cryptocurrency payments to provide more flexibility to tenants.

**Automated Recurring Payments :** Allow tenants to set up automated recurring payments for rent and other bills

### Integration with Third Party Services

**CRM Systems :** Integrate with Customer Relationship Management (CRM) systems to enhance tenant relationship management.

**Accounting Software :** Seamless integration with popular accounting software like QuickBooks and Xero for streamlined financial management.

**Real Estate Platforms :** Connect with real estate listing platforms for automated property listings and vacancy management.

## Conclusion

The future scope of the Rent Management System is vast and promising. By leveraging advancements in technology and addressing the evolving needs of property managers, landlords, and tenants, the system can continue to provide comprehensive, efficient, and secure solutions for rental property management. Continuous improvement and innovation will be key to maintaining a competitive edge and delivering enhanced value to all stakeholders involved.

**Chapter : 14**

## **Limitations of the Project**

# **RENT MANAGEMENT SYSTEM**



## Limitations of the PHP Rent Management System

While a PHP based Rent Management System can offer a robust and scalable solution for managing rental properties, there are certain limitations that need to be considered:

### 1. Performance Issues

Scalability : As the number of users and data increases, PHP applications may face performance issues, especially if not optimized correctly.

Concurrency : Handling multiple simultaneous requests can be challenging without proper configuration and optimization of the web server and database.

### 2. Security Concerns

Vulnerabilities : PHP is known to have certain vulnerabilities if not properly secured. Common issues include SQL injection, cross site scripting (XSS), and cross site request forgery (CSRF).

Session Management : Proper session management is crucial to prevent session hijacking and other session related attacks.

### 3. Maintenance and Upgrades

Code Management : Managing and maintaining large PHP codebases can become cumbersome without proper coding standards and practices.

Dependencies : Keeping up with PHP and library updates is necessary to ensure security and compatibility, which can be a time consuming task.

### 4. Customization and Flexibility

Customization : While PHP is flexible, extensive customization may require significant effort and expertise, particularly when integrating with other systems or adding new features.

Legacy Code : Older PHP code may not follow modern best practices, making it difficult to extend or modify.

### 5. Compatibility

Browser Compatibility : Ensuring the application works seamlessly across different browsers and devices can be challenging.

Integration with Modern Technologies : Integrating PHP applications with modern technologies like microservices, serverless architectures, and real time data processing can be complex.

## Use of Filters to Prevent Malicious Scripts

To mitigate the risk of malicious users running scripts via forms, PHP provides several mechanisms to filter and validate input data. Here's how filters can help:

### 1. Input Validation

**Sanitizing Data :** Sanitizing input data ensures that potentially harmful characters are removed or encoded, reducing the risk of injection attacks.

**Validation Functions :** PHP offers built in functions to validate data types, formats, and values.

### 2. Using PHP Filters

**filter\_var Function :** The filter\_var function is used to validate and sanitize data.

```
php
```

```
// Example of validating an email address
```

```
$email = filter_var($_POST['email'], FILTER_VALIDATE_EMAIL);
```

```
// Example of sanitizing a string
```

```
$name = filter_var($_POST['name'], FILTER_SANITIZE_STRING);
```

**Filter Constants :** PHP provides a variety of filter constants to handle different types of input.

```
php
```

```
// Example of validating an integer
```

```
$age = filter_var($_POST['age'], FILTER_VALIDATE_INT);
```

```
// Example of sanitizing a URL
```

```
$url = filter_var($_POST['url'], FILTER_SANITIZE_URL);
```

### 3. Preventing SQL Injection

**Prepared Statements :** Using prepared statements with parameterized queries helps prevent SQL injection attacks.

```
php
```

```
$stmt = $pdo->prepare("SELECT FROM users WHERE email = :email");
```

```
$stmt->bindParam(':email', $email);
```

```
$stmt->execute();
```

#### 4. Cross Site Scripting (XSS) Protection

HTML Encoding : Encode user input before rendering it on web pages to prevent XSS attacks.

```
php
echo htmlspecialchars($user_input, ENT_QUOTES, 'UTF 8');
```

#### 5. Cross Site Request Forgery (CSRF) Protection

CSRF Tokens : Use CSRF tokens to ensure that requests are coming from authenticated users.

```
php
// Generating a CSRF token
$_SESSION['csrf_token'] = bin2hex(random_bytes(32));

// Including the CSRF token in forms
<input type="hidden" name="csrf_token" value="= $_SESSION['csrf_token']; ?>></pre
```

## Conclusion

While a PHP based Rent Management System has certain limitations, such as performance issues, security concerns, and maintenance challenges, these can be mitigated with proper coding practices, optimization, and regular updates. The use of filters and other security mechanisms is crucial to prevent malicious users from exploiting the system via forms, ensuring a more secure and reliable application. By addressing these limitations and implementing robust security measures, a PHP Rent Management System can effectively manage rental properties and provide a valuable tool for property managers and landlords.

Chapter : 15

## Bibliography & References

### **RENT MANAGEMENT SYSTEM**



## Bibliography and References for PHP Rent Management System

### Books

#### 1. PHP and MySQL Web Development" by Luke Welling and Laura Thomson

This book provides comprehensive coverage of PHP and MySQL, including techniques for building dynamic and interactive web applications.

Citation: Welling, L., & Thomson, L. (2016). PHP and MySQL Web Development. Addison Wesley Professional.

#### 2. Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5" by Robin Nixon

An excellent resource for learning how to build dynamic web applications using PHP, MySQL, JavaScript, and related technologies.

Citation: Nixon, R. (2018). Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5. O'Reilly Media.

#### 3. PHP Objects, Patterns, and Practice" by M. Zandstra

This book explores advanced PHP concepts, including object oriented programming, design patterns, and best practices.

Citation: Zandstra, M. (2017). PHP Objects, Patterns, and Practice. Apress.

### Online Resources

#### 4. PHP Official Documentation

The official PHP documentation provides detailed information on PHP functions, classes, and best practices.

URL : <https://www.php.net/manual/en/>

#### 5. MySQL Official Documentation

Comprehensive guide to using MySQL, including installation, configuration, and SQL commands.

URL : <https://dev.mysql.com/doc/>

#### 6. W3Schools PHP Tutorial

A beginner friendly tutorial covering the basics of PHP, including syntax, functions, and database integration.

URL : <https://www.w3schools.com/php/>

## 7. Stack Overflow

A community driven Q&A site where developers can ask and answer questions related to PHP and other programming languages.

URL : <https://stackoverflow.com/>

## Articles and Tutorials

### 8. Building a Simple PHP Login System" by Tania Rascia

A step by step guide to building a basic PHP login system, including user registration and authentication.

Citation : Rascia, T. (2018). Building a Simple PHP Login System. Retrieved from [https://www.taniarascia.com/create a simple database app php/](https://www.taniarascia.com/create-a-simple-database-app-php/)

### 9. Secure PHP Development" by John Conde

An article focused on security best practices in PHP development, covering common vulnerabilities and mitigation techniques.

Citation : Conde, J. (2019). Secure PHP Development. Retrieved from [https://johnconde.net/blog/secure php development/](https://johnconde.net/blog/secure-php-development/)

## References

1. Welling, L., & Thomson, L. (2016). PHP and MySQL Web Development. Addison Wesley Professional.
2. Nixon, R. (2018). Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5. O'Reilly Media.
3. Zandstra, M. (2017). PHP Objects, Patterns, and Practice. Apress.
4. PHP Official Documentation. Retrieved from <https://www.php.net/manual/en/>
5. MySQL Official Documentation. Retrieved from <https://dev.mysql.com/doc/>
6. W3Schools PHP Tutorial. Retrieved from <https://www.w3schools.com/php/>

This bibliography and references page provides a comprehensive list of resources that can support the development and enhancement of a PHP based Rent Management System.



## **CHAPTER - 11**

### **Installation Guide**

### **RENT MANAGEMENT SYSTEM**

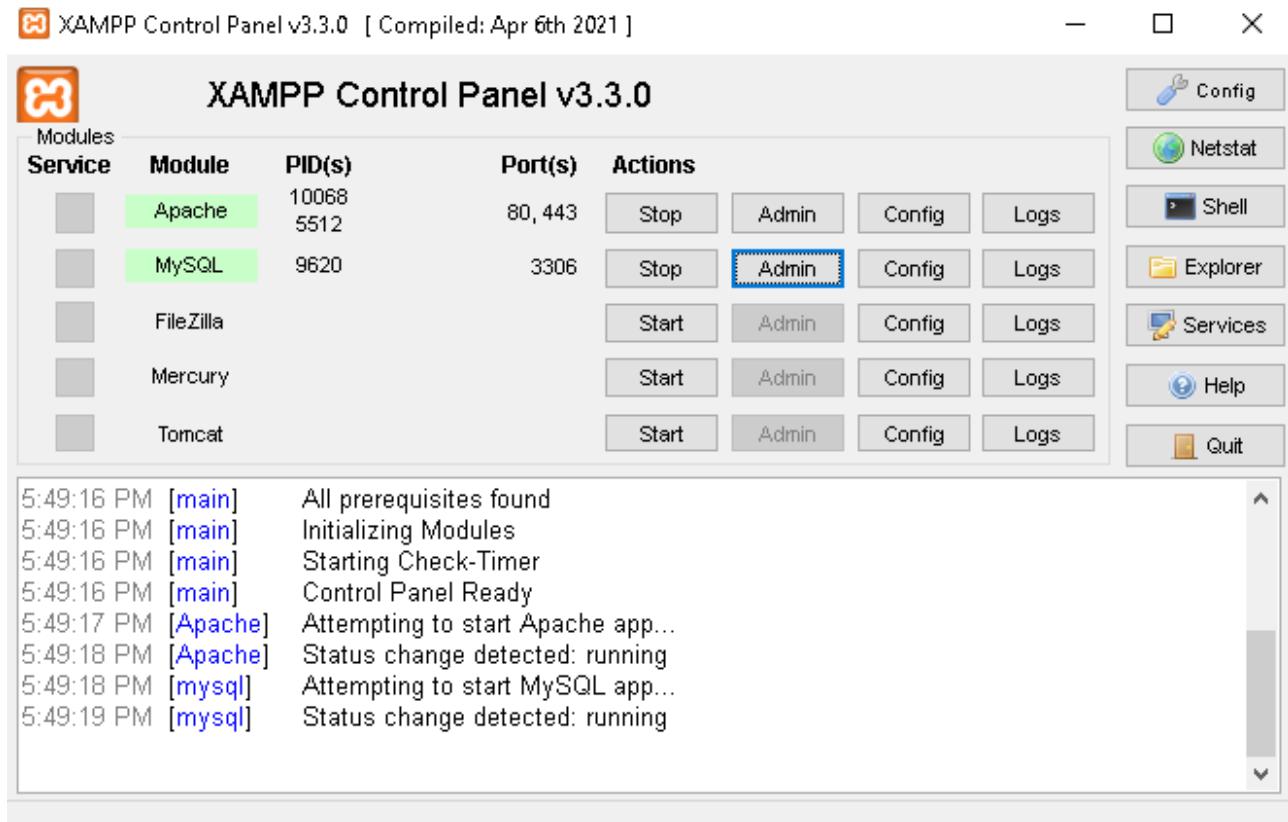


## 10.1 Installation Guide : RENT MANAGEMENT SYSTEM

We will go to the cd Drive , and Copy the folder : rms to the location:

we will past it in the location : C:\xampp\htdocs\

Then we will open the xammp Control panel



We will click on the Start Services button .

We will click the Admin button of the MySQL : This will open the Main page for the mysql db (Maria Db) .There we will create a new Database with the

name : rent\_management\_system .Then we will select it and open the next page

Then we will click on the Import Button. We will import the sql file : rent\_management\_system.sql  
The queries will run perfectly.

Then we will open the link : localhost/rms in the chrome / microsoft edge and hit enter.

We can login , create the new user and operate the Software . We can test the software by entering the test@gmail.com and password :test

We can also download the file from the below link in the form of qr code.

We will download the entire package as zip file , we will place the rms folder to the htdocs folder in the Xammp .

We will open the Admin page of mysql , make the new database with the name : rent\_management\_system and then select it.

Then finally we will import the given sql file in the Database .

Incase , the CD files are corrupt / missing , Kindly Download the Software files for installation

