# Veer Narmad South Gujarat University, Surat.

# Department of Information and Communication Technology

## M.Sc. (Information and Communication Technology) Programme

# Project Report

## 2nd Semester

## M.Sc. (Information and Communication Technology) 2 Year Course

**Year 2024 – 2025**

# Property Maintenance System


Guided By :

**Mr. Nirav Jariwala (External Guide)**
**Dr. Dhaval Joshi (Internal Guide)**

Submitted By :

**Asti Paladiya**
**(R241100180007100064)**

**Comnet Development India**
25, Canal Corridor Road, Umrao Nagar, Rupali Naher, Bhatar,
Surat-395007

# Veer Narmad South Gujarat University, Surat.

## Department of Information and Communication Technology

## M.Sc. (Information and Communication Technology) Programme

# Certificate

This is to certify that Ms. **Asti Paladiya** with Exam Seat Number: **75** and Enrollment Number: **R24110018000710064** has worked on her part time project work entitled **Property Maintenance System** at **Comnet Development India** as a partial fulfillment of the requirements for 2nd ***Semester - M.Sc. (Information and Communication Technology)***, during the academic Year 2024-2025.

Date : 16-06-2025

Place : Dept. of ICT, VNSGU, Surat.

| | | |
|---|---|---|
| **Internal Project Guide**<br>**M.Sc.(I.C.T.) 2nd Semester**<br>**Department of I.C.T.**<br>**Veer Narmad South Gujarat University, Surat** | **Course Coordinator**<br>**M.Sc. (I.C.T.) Programme**<br>**Department of I.C.T.**<br>**Veer Narmad South Gujarat University, Surat** | **Head of the Department**<br>**Department of I.C.T.**<br>**Veer Narmad South Gujarat University, Surat** |

comnet development india
digital, delivered.

Comnet Development India
25, Shree Shakti Street, Umrao Nagar,
Canal Road, Rupali Naher, Bhatar,
Surat-395007, Gujarat, India.

(L) +91 9825956676
(E) neel@comnet.com.sg
(W) www.comnet.com.sg

Date: 13/06/2025

## TO WHOMEVER IT MAY CONCERN

This is to certify that **Asti Paladiya** student of 2nd Sem of MSC(ICT) J. P. Dawer Institute of Information Science and Technology, Veer Narmad South Gujarat University, Surat has successfully completed her 5-month Internship in **Property Maintenance System** Developed with React.js as Frontend & .NET Core as Backend.

Below is the Progress in Detail.

1) Performance of the student: Very Good
2) Attendance: 99%

We wish all success to her

For,

Comnet Development India


SOCIOSPHERE

# ACKNOWLEDGEMENT

The successful completion of this project would not have been possible without the guidance, support, and encouragement of many individuals. We sincerely thank everyone who contributed their time, knowledge, and expertise to help us bring this project to fruition.

We are thankful to **Dr. Dhaval Joshi at J.P. Dawer Institute of Information Science & Technology(VNSGU)** our mentor and project coordinator, for his invaluable guidance, constant encouragement, and unwavering support throughout our journey. His ability to inspire and provide practical solutions to challenges has been instrumental in shaping this project into its current form.

We also want to thank **Comnet Development India**, the company where we had the opportunity to complete this project. Their professional environment, practical knowledge, and industry insights greatly enriched our learning experience and significantly improved the overall quality of the project.

Lastly, we want to express our appreciation to all those who have directly or indirectly supported us, be it through their technical assistance, motivation, or encouragement. Your contributions, big or small, have made a meaningful impact on the development of this system.

Thank you all for being a part of this journey and for helping us achieve this milestone.

Thanking All.
**[Asti Paladiya]**

# INDEX

# 1. Introduction

**SocioSphere** is a comprehensive web and mobile-based Property Maintenance and Society Management System designed to streamline the daily operations and communication within residential societies and apartment complexes. The name SocioSphere is a fusion of:

- **"Socio-"** – representing society, community, and collective living.
- **"Sphere"** – symbolizing completeness, structure, and an integrated system.

Together, SocioSphere represents a platform that brings social wholeness to communities by offering a centralized and structured environment for managing all aspects of residential society life.

This system simplifies complex society operations such as maintenance tracking, complaint handling, event management, visitor logging, financial record keeping, and internal communication. It ensures that all community interactions—from administration to member engagement and security coordination—are transparent, secure, and efficiently managed.

With its user-friendly interface and role-based access, SocioSphere fosters a sense of community, empowers residents, and provides administrators with the tools they need to maintain a well-functioning, harmonious living space.

## 1.1 Company Profile

| | |
|---|---|
| **Name of the Organization** | Comnet Development India |
| **Name of Guide** | Mr. Nirav Jariwala |
| **Complete Office Address** | 25, Canal Corridor Road, Umrao Nagar, Rupali<br><br>Naher, Bhatar,Surat. |
| **Telephone Number** | 9825956676 |
| **Date of Joining** | 15-01-2025 |
| **Intership Topic** | Property maintenance system |

## 1.2 Customer Profile

**SocioSphere** is designed to serve a wide range of residential communities and housing societies that require a digital solution for managing day-to-day operations, improving internal communication, and ensuring transparency in society governance. The primary customers or end-users of this system include:

- **Residential Housing Societies and Apartment Complexes** seeking to manage member records, maintenance collections, complaints, events, expenses, and visitor logs digitally.
- **Society Management Committees** or **Resident Welfare Associations (RWAs)** aiming to reduce manual workload and ensure better coordination between members, security staff, and administrators.
- **Gated Communities** that require organized visitor management, emergency contact availability, and streamlined event or complaint tracking systems.
- **Property Developers or Builders** who wish to provide a value-added service for their newly built communities by offering a structured, tech-enabled management system.
- **Residents and Flat Owners** who expect easy access to maintenance payments, event registration, community interactions, complaint tracking, and transparency in expenses.
- **Security and Support Staff**, such as watchmen or guards, who benefit from a simple mobile interface to manage visitor entries and ensure security logging.

This platform is ideal for communities that aim to digitize their ecosystem, improve collaboration, and foster a socially connected and well-maintained living environment.

---

### 1.2.1 Current System

In most residential societies and apartment complexes, the current system for managing property maintenance and community-related activities is either **manual** or based on **basic standalone software** solutions. These traditional methods often suffer from various limitations and inefficiencies, such as:

1. **Manual Record Keeping** : Member information, maintenance dues, visitor entries, complaints, and expenses are typically recorded in physical registers or

spreadsheets, increasing the chances of errors, misplacement, and lack of transparency.

2. **Inefficient Communication** : Important updates or notifications (e.g., pending maintenance, event announcements) are usually conveyed via notices on bulletin boards or word of mouth, leading to poor communication among residents.
3. **No Centralized System** : Data is scattered across multiple tools and formats, making it difficult for management to maintain a unified view or generate reports.
4. **Lack of Real-time Access** : Members and admins do not have access to real-time information such as maintenance status, complaint tracking, or visitor logs, which affects transparency and trust.
5. **No Role-based Access** : Manual systems lack user authentication or role-based access, increasing the risk of unauthorized access or data mismanagement.
6. **Security and Tracking Issues** : Visitor entries are often logged in physical notebooks by watchmen, making it hard to verify records or generate visitor reports when needed.

In summary, the existing system is largely **manual, fragmented, and inefficient**, lacking the automation, accessibility, and transparency required for modern society management.

### 1.2.2 Customer Detail

The customer for the **SocioSphere** system includes **residential housing societies, apartment complexes, and gated communities** where the society management is handled directly by an **Admin**, without the involvement of a formal committee or resident welfare association.

In the current working style of such communities:

- The **Admin** acts as the central authority responsible for managing all operational and administrative tasks. This includes adding members and watchmen, maintaining maintenance records, responding to complaints, organizing events, managing financial records, updating contact lists (like emergency agencies), and overseeing all digital interactions within the system.
- **Residents (Members)** interact with the system to fulfill essential tasks such as paying maintenance charges, submitting complaints, participating in society events, viewing important announcements, and accessing shared resources like expense reports and emergency contacts. They expect real-time updates, easy access to information, and transparency in all society-related activities.

- **Watchmen (Security Personnel)** primarily operate through a mobile interface to manage visitor entries and maintain their own profiles. They serve a crucial role in the security and verification process of all incoming guests.

This working model is ideal for small to mid-sized societies where direct management by an admin ensures quick decision-making, streamlined operations, and better accountability. **SocioSphere** is developed to perfectly align with this structure—centralizing all operations under the admin's control while providing accessible and role-specific functionality to members and support staff.

---

# 2. Proposed System

## 2.1 Scope

- The proposed system **SocioSphere** aims to automate and streamline daily operations in residential societies.
- It covers key functions such as **maintenance tracking**, **complaint management**, **event handling**, **visitor logging**, and **expense monitoring**.
- The system provides **role-based access** for Admins (web), Members (web), and Watchmen (mobile app).
- It enhances **communication**, **transparency**, and **data accessibility** across all users.
- Designed for communities **without formal committees**, allowing direct control by a central Admin.
- Helps reduce manual work, eliminate data redundancy, and ensure efficient, real-time management.

## 2.2 Objective

### User Management and Authentication

- ✓ Allow the admin to add members and watchmen with credential generation.
- ✓ Enable secure login, logout, profile update, and password recovery for all roles.
- ✓ Maintain separate access roles for Admin (web), Member (web), and Watchman (app).

### Maintenance Management

- ✓ Let members submit maintenance requests and make online payments.
- ✓ Allow the admin to review, accept/reject requests, and generate downloadable receipts.
- ✓ Maintain recent records and filter/search maintenance history by date or type.

### Complaint Management

- ✓ Enable members to raise complaints and track admin responses and status updates.
- ✓ Allow admin to manage complaint types and update the action taken.
- ✓ Provide filtering and type-based search for efficient tracking.

### Event Management

- ✓ Let members view upcoming events, register, and pay event fees online.
- ✓ Admin can manage event records, participant lists, and event-related finances.
- ✓ Both admin and members can upload and manage event gallery photos.

### Visitor and Security Management

- ✓ Watchmen can add, update, and manage visitor details via a mobile app.

- ✓ Members can view their specific visitors; Admin can view all visitor records.

**Suggestion and Feedback System**

- ✓ Members and admin can post suggestions, like/dislike others' suggestions.
- ✓ Promote interactive community involvement through voting.

**Expense and Agency Management**

- ✓ Admin can define expense types and manage all society expense records.
- ✓ Members can view expense records and official contact details of agencies like police, ambulance, etc.

**Notification System**

- ✓ Automatically send notifications for pending maintenance, event updates, or complaint responses.

**Search and Filter Functionalities**

- ✓ Provide type-wise, recent, and category-based filtering for maintenance, complaints, events, etc.
- ✓ Implement keyword-based search to simplify data access for users.

**Profile and Account Management**

- ✓ All users can update personal details and change or recover passwords when required.

## 2.3 Constraints

The SocioSphere system is developed with defined boundaries and limitations that may affect its operation under certain conditions. These constraints are categorized as follows:

### 2.3.1 Hardware Constraints

- The mobile application is supported only on **Android devices**; iOS users cannot use the app.
- Devices with **low processing power or limited memory** may experience slower performance or crashes, especially when handling large data (e.g., images or PDFs).
- **Internet connectivity is mandatory** for both web and mobile users; offline access is not supported.

- The system requires a **reliable server with sufficient CPU, RAM, and storage** to manage user accounts, event galleries, maintenance records, and PDF receipt generation.
- The application may not function properly on **outdated hardware** or **older mobile OS versions**.

### 2.3.2 Software Constraints

- The system supports only **modern web browsers** (e.g., Chrome, Firefox, Edge); older or unsupported browsers may not display features correctly.
- The mobile app is built for **Android OS only**, which limits access for iOS users.
- **Multi-society or multi-admin structure** is not supported; it is designed for single-society use with one central admin.
- PDF receipt generation, image uploads, and filtering features depend on **third-party libraries**, which may vary in behavior across platforms or devices.
- Email functionalities (e.g., OTP for password reset, complaint updates, maintenance notifications) are subject to **SMTP provider limitations**, such as daily sending limits, spam filtering, or delivery delays.
- If users enter **invalid or inactive email addresses**, critical communication (e.g., password resets or updates) will fail.
- Requires **regular software updates, server maintenance, and internet availability** to ensure consistent functionality and security.
- Some features (e.g., real-time updates or push notifications) may not work correctly in environments with **firewalls, restricted networks, or outdated software versions.**

## 2.4 Advantages

- **Centralized Management**: All society-related operations like maintenance, complaints, events, and expenses are managed from one platform.
- **Role-Based Access**: Ensures data security and personalized experience for Admins, Members, and Watchmen based on their specific responsibilities.
- **Real-Time Notifications**: Keeps users updated on maintenance dues, event schedules, complaint responses, and password resets via email alerts.

- **Digital Records & Receipt Generation**: Users can download digital receipts and track past maintenance or event records easily.
- **Time & Cost Efficient**: Reduces paperwork and manual efforts by automating tasks like visitor logging, maintenance approval, and communication.
- **Community Engagement**: Features like suggestion voting and event participation promote better member interaction and involvement.
- **Secure and Reliable**: Password recovery, OTP verification, and admin-controlled account management enhance the overall system security.
- **Search & Filter Capabilities**: Simplifies access to specific records using filters like type, date, status, and recent activities.
- **24x7 Accessibility**: Available anytime via web for admin/members and mobile app for watchmen, ensuring uninterrupted access.

## 2.5 Limitations

- **Multi-Society or Multi-Admin Support**: The system is designed for a single society with one admin. There is no provision to manage multiple societies or assign sub-admin roles.
- **iOS Mobile App Unavailability**: The mobile application is currently available only for Android users. iOS users cannot access the mobile features.
- **Real-Time Chat/Message Communication**: There is no direct chat or messaging system for communication between members, admin, or watchmen.
- **QR Code Entry for Visitors**: The visitor management module does not support digital QR-based visitor check-ins or automated gate passes.
- **Biometric or Face ID Authentication**: The system does not support biometric logins or Face ID authentication for additional security.
- **Dark Mode or Theme Customization**: Users cannot switch between different UI themes or personalize their dashboard appearance.
- **Offline Mode for Mobile App**: Watchmen must have an internet connection; there is no offline entry mode with later sync support.

# 3.Environment specification

## 3.1 Hardware and software requirement

- **Hardware requirement :**

| Component | Specification |
|---|---|
| **Processor** | Intel i5/i7 or AMD Ryzen 5+ |
| **RAM** | 8 GB (16 GB recommended) |
| **Storage** | SSD, 100 GB free space |
| **Display** | 14" Full HD (1920×1080) |
| **Network** | Broadband, minimum 10 Mbps |
| **Graphics** | Integrated or optional dedicated GPU |
| **OS** | Windows 11 (64-bit) or higher |
| **Peripherals** | Keyboard, Mouse, Webcam |

- **Software requirement**

| | |
|---|---|
| **Host Operating System (at server side)** | Windows 11 / Windows Server OS |
| **Operating System (client side)** | Any modern OS (Windows/macOS/Linux) |
| **Browser** | Chrome, Edge, Firefox |
| **Database** | SQL Server Management Studio |
| **Development** | Visual Studio 2022, Visual Studio Code |
| **Server** | IIS (for API hosting), SQL Server |
| **Mail Service** | SMTP (for OTP, complaint status, etc.) |
| **Source Control** | GitHub |

➢ **Service Requirements**

| Broadband Internet Connectivity | Required for both development and deployment |
|---|---|
| **Server Support** | SQL Server, IIS Hosting Enabled |

## 3.2Development Description

The development of **SocioSphere** is structured into three main layers: **Frontend (Web), Backend (API), and Mobile Application** to ensure a seamless, scalable, and responsive experience for all user types (Admin, Member, Watchman).

1. **Frontend Development (Website - Admin & Member)**

- ✓ Built using **ReactJS**, with UI components styled using **React Bootstrap** and **Material UI**.
- ✓ Integrated with **Axios** for consuming backend APIs securely and efficiently.
- ✓ Implements dynamic data filtering, searching, pagination, and interactive UI features like badges, modals, and real-time status indicators.
- ✓ JWT-based login system to ensure secure, role-based access.
- ✓ All forms and dashboards are responsive and optimized for desktop browsers.

2. **Backend Development (Web API)**

- ✓ Developed using **.NET Core (C#)** and exposed as a **RESTful Web API**.
- ✓ Handles all server-side logic, such as authentication, database operations, business rules, email notifications, and PDF generation (e.g., maintenance receipt).
- ✓ Connected to **SQL Server** for managing all society-related records like maintenance, events, users, complaints, suggestions, and visitors.
- ✓ Secure endpoints using **JWT (JSON Web Token)** for authorization.
- ✓ Uses **SMTP** for sending OTPs, complaint responses, and other status emails.

3. **Mobile Application (Watchman Use Only)**

- ✓ Developed in **Kotlin using Android Studio**.
- ✓ Interfaces with the backend via **Retrofit** to handle visitor management features.
- ✓ Lightweight, with a focus on adding/updating/viewing visitors and profile management.
- ✓ Includes basic authentication and password recovery (email OTP).

4. **Version Control & Deployment**

- ✓ **GitHub** is used for version control and team collaboration.
- ✓ API deployed on a Windows Server using **IIS** for hosting.
- ✓ Uses **SQL Server Management Studio (SSMS)** for database schema and data operations.

| Category | Tools/Technologies |
|---|---|
| **Frontend** | ReactJS, React Bootstrap, Material UI |
| **Backend** | ASP.NET Core Web API (C#) |
| **Mobile App** | Kotlin (Android Studio), Retrofit for API calls |
| **API Development** | Microsoft Visual Studio 2022 |
| **Database** | SQL Server (using SQL Server Management Studio) |
| **API Communication** | Axios (React), Retrofit (Android) |
| **Authentication** | JWT (JSON Web Token) |
| **Version Control** | GitHub |
| **Mail Service** | SMTP (for sending OTP, status updates, etc.) |
| **Web Server** | IIS (Internet Information Services) |
| **Browser** | Any modern browser (Chrome, Firefox, Edge, etc.) |
| **Code Editor** | Visual Studio Code |

# 4. System Planing

## 4.1 Feasibility Study

The feasibility study for the SocioSphere system evaluates various aspects to ensure the successful implementation and long-term sustainability of the project. It analyzes whether the system is practical, beneficial, and achievable in real-world scenarios based on the following key areas:

1. **Technical Feasibility**

- ✓ The technologies chosen (ReactJS, ASP.NET Core Web API, SQL Server, Android with Kotlin) are modern, reliable, and widely supported.
- ✓ The development tools such as Visual Studio, VS Code, Android Studio, and GitHub streamline the coding, testing, and deployment process.
- ✓ The availability of skilled developers and widespread community support ensures technical challenges can be addressed effectively.
- ✓ The use of APIs and modular architecture allows easy integration and future scalability.

**2. Operational Feasibility**

- ✓ The system meets the operational needs of residential societies by digitizing maintenance, complaint, event, and visitor management processes.
- ✓ Admin, Member, and Watchman roles are well-defined, with clear responsibilities and controlled access through authentication.
- ✓ Email notifications, real-time updates, and mobile accessibility (for Watchman) enhance overall user engagement and adoption.
- ✓ Easy-to-use interfaces reduce the learning curve, even for non-technical users.

**3. Economic Feasibility**

- ✓ The system uses open-source tools like ReactJS and Visual Studio Code, reducing software licensing costs.

- ✓ Development and deployment require minimal investment in hardware and server infrastructure (can even use cloud hosting or shared hosting initially).
- ✓ Reduces manual paperwork, miscommunication, and inefficiencies—resulting in long-term cost savings for housing societies.

---

**□ 4. Legal Feasibility**

- ✓ The application ensures user data is handled securely using JWT and HTTPS protocols, reducing the risk of data breaches.
- ✓ Complies with general data privacy practices such as secure login, password recovery via OTP, and email verification.
- ✓ All records (visitor logs, complaints, payments) are maintained for accountability and transparency.

---

**□ 5. Schedule Feasibility**

- ✓ The project is divided into multiple modules (maintenance, events, visitor, complaint, user management) for systematic and timely development.
- ✓ With clear milestones and modular development using GitHub versioning, the project can be completed within a reasonable timeframe.

Here is a detailed explanation of the **Software Engineering Model** for your **SocioSphere – Property Maintenance System**, structured and explained according to a specific model approach:

### 4.2 Software Engineering Model

To ensure efficient development, testing, and delivery of the **SocioSphere** system, we adopted the **Incremental Software Development Model**. This model is highly suitable for web-based applications with multiple user roles, complex functionalities, and evolving requirements like SocioSphere.

## ➢ Chosen Model: Incremental Model

The **Incremental Model** is a process where the overall system is built and delivered in small functional segments (increments). Each increment includes planning, design,

coding, and testing activities, and it gradually adds to the complete functionality of the system.

✓ **Why Incremental Model for SocioSphere?**

| Aspect | How it applies to SocioSphere |
|---|---|
| **Multiple Modules** | The system includes Maintenance, Complaints, Events, Visitors, Suggestions, etc., which are developed incrementally. |
| **User Roles** | Supports different roles like Admin, Member, and Watchman, whose functionalities can be built step-by-step. |
| **Progressive Feedback** | Feedback can be taken from users after each module is released, allowing real-time improvement. |
| **Risk Management** | Risk is reduced by breaking down the system into smaller, manageable parts. |
| **Time-Constrained** | Due to the five-month project window, prioritizing and delivering core features early helped with project tracking. |

✓ **How SocioSphere Followed the Incremental Model**

Each of the following phases was applied repeatedly to different modules like Maintenance, Complaints, Events, etc.

1. **Requirement Gathering and Analysis**

✓ Requirements were collected based on real-world needs of housing societies.
✓ Separate use cases were defined for Admin, Member, and Watchman roles.

2. **System Design**

✓ Database schema was designed using SQL Server.
✓ API architecture was planned using RESTful standards.
✓ UI/UX design was created using ReactJS and Material UI.

3. **Implementation (Development)**

✓ Developed modules one by one, starting with user authentication and profile management.

- ✓ Maintenance module, Complaint module, Event module, etc., followed incrementally.

4. **Testing**

- ✓ Each module was unit tested after development.
- ✓ Integration testing ensured the system worked properly when modules interacted.
- ✓ Mobile app testing was also done using Android emulators and real devices.

5. **Deployment**

- ✓ Modules were deployed on a live server using IIS.
- ✓ Mobile app was deployed separately using Android Studio.

6. **Feedback and Enhancement**

- ✓ Based on feedback, additional features like:
  - Email notifications (complaint status, password recovery)
  - PDF receipt generation
  - Dynamic search and filtering.
  - Event and maintenance reminders via notifications

  were added in later increments.

➢ **Benefits Realized with This Model**

- ✓ Early working system to demonstrate core functionality.
- ✓ Easy maintenance and bug tracking due to modular build.
- ✓ Enhanced collaboration using GitHub for incremental version control.
- ✓ Better end-user satisfaction through phased feedback and refinement.

---

## 4.3 Risk Analysis

Risk analysis plays a critical role in identifying potential issues that may affect the successful development and deployment of the SocioSphere system. Addressing these risks early helps in reducing project delays, financial loss, and performance issues.

➢ **Types of Risks and Their Impact**

| Risk Category | Description | Impact Level | Mitigation Strategy |
|---|---|---|---|
| **Technical Risk** | Complexity in integrating multiple technologies like ReactJS, .NET Core, Android (Kotlin), and SQL Server. | High | Proper API documentation, modular coding, and frequent testing in different environments. |
| **Security Risk** | Data leakage or unauthorized access due to weak authentication. | High | JWT-based authentication, HTTPS communication, encrypted password storage, role-based access. |
| **Resource Risk** | Limited development team and short timeline (5 months). | Medium | Prioritized core modules first; followed incremental model for faster deliveries. |
| **Hardware/Software Failure** | Unexpected failure of server, database, or development machines. | Medium | Regular backups, version control (GitHub), deployment on reliable server using IIS. |
| **User Acceptance Risk** | Members or admins may find the system difficult to use or resist adoption. | Medium | Developed intuitive UI/UX using React + Material UI; involved sample users in testing. |
| **Communication Risk** | Misunderstanding between team or stakeholders leading to incorrect implementation. | Low | Clear documentation, team meetings, and use of GitHub for centralized tracking. |
| **Email Delivery Issues** | Emails for complaint status, OTP, or maintenance response may | Medium | Verified SMTP setup, fallback for retries, and user alerts for mail failure. |

| Risk Category | Description | Impact Level | Mitigation Strategy |
|---|---|---|---|
| | fail or be delayed due to SMTP issues. | | |
| **Mobile Integration Risk** | API might behave inconsistently when accessed via Android application (Retrofit). | Medium | Performed mobile API testing and error handling in Kotlin app. |

## 4.4 Project Schedule

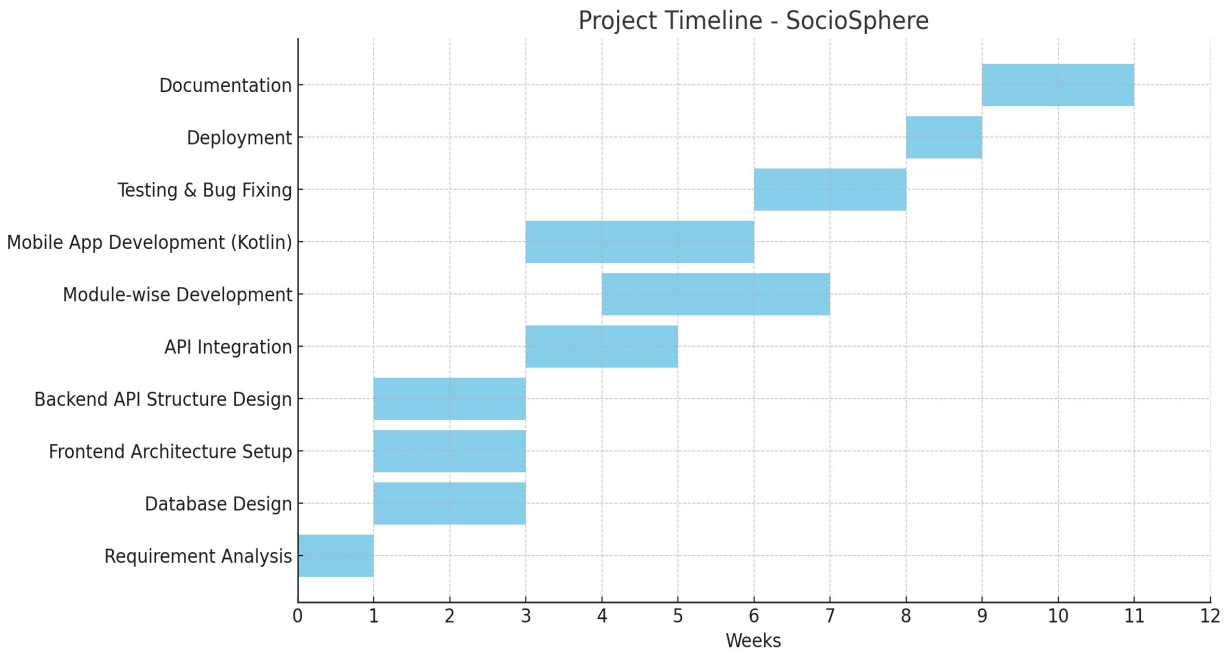
A well-structured schedule is critical to ensuring the timely and successful delivery of the SocioSphere system. The following breakdown outlines task dependencies, estimated timelines, and resource allocation.

### 4.4.1 Task Dependency

Below is the logical order in which project tasks depend on one another:

| Task | Depends On |
|---|---|
| Requirement Analysis | - |
| Database Design | Requirement Analysis |
| Frontend Architecture Setup | Requirement Analysis |
| Backend API Structure Design | Requirement Analysis |
| API Integration | Backend API Design, Frontend Setup |
| Module-wise Development | API Integration, Database Design |
| Mobile App Development (Kotlin) | Backend API Design |
| Testing & Bug Fixing | Module-wise Development |
| Deployment | Testing & Bug Fixing |
| Documentation | Entire Development Process |

### 4.4.2 Timeline Chart




### 4.4.3 Project Table

Here is the **Project Table** in a clear table format without specific start or end dates — ideal for documentation when dates aren't finalized:

| Sr. No. | Task | Duration | Dependency |
|---|---|---|---|
| 1 | Requirement Analysis | 1 week | - |
| 2 | Database Design | 1 week | Task 1 |
| 3 | Frontend & Backend Setup | 1 week | Task 1 |
| 4 | API + Authentication | 2 weeks | Task 3 |
| 5 | Maintenance Module | 2 weeks | Task 2, Task 4 |
| 6 | Complaint Module | 1 week | Task 4 |
| 7 | Suggestion, Events, Expenses | 2 weeks | Task 5, Task 6 |

| Sr. No. | Task | Duration | Dependency |
|---|---|---|---|
| 8 | Mobile App (Visitor Only) | 1 week | Task 4 |
| 9 | Testing + Bug Fixing | 1 week | Task 5, Task 6, Task 7, 8 |
| 10 | Deployment + Hosting | 1 week | Task 9 |
| 11 | Documentation | 1 week | All Tasks Completed |

# 5. System Analysis

## 5.1 Detailed SRS :

1. User Management Module :

| Section | Description |
|---|---|
| **Input** | User registration details: name, email, mobile, password, role |
| **Events** | Registration, login, forgot password, OTP verification, profile update |
| **Output** | Registration success, login token, OTP mail, updated profile confirmation |
| **Validation** | Unique email/mobile, strong password, valid OTP format |
| **Constraints** | Cannot register duplicate user, OTP expiry within 5 mins |

2. Maintenance Module:

| Section | Description |
|---|---|
| **Input** | Maintenance amount, due date, payment details |
| **Events** | Admin adds record, Member pays, Admin accepts/rejects, Receipt generates |
| **Output** | Receipt (PDF), Payment status update, Mail confirmation |
| **Validation** | Valid amount, date format, correct user ID |
| **Constraints** | Cannot pay after due date without late fee, only one payment per cycle |

3. Complaint Module

| Section | Description |
|---|---|
| **Input** | Complaint type, description, optional image |
| **Events** | Complaint submission, admin action update, status change, email notification |
| **Output** | Status update, mail to member with action details |
| **Validation** | Non-empty subject & description |

| Section | Description |
|---|---|
| **Constraints** | Complaint cannot be updated after "Completed" stage |

4. Suggestion Module

| Section | Description |
|---|---|
| **Input** | Suggestion text, category |
| **Events** | Suggestion add, like/dislike, view others' suggestions |
| **Output** | Updated suggestion list with votes |
| **Validation** | No duplicate suggestion by same user |
| **Constraints** | Cannot vote more than once on same suggestion |

5. Event Management Module

| Section | Description |
|---|---|
| **Input** | Event title, date, fee, gallery photo, description |
| **Events** | Add event, Member registers/pays, Admin manages gallery |
| **Output** | Event list, Payment confirmation, Gallery images |
| **Validation** | Valid date, title, fee > 0 |
| **Constraints** | Payment cannot be canceled once made |

6. Visitor Management (Watchman App)

| Section | Description |
|---|---|
| **Input** | Visitor name, contact, person to meet, in/out time |
| **Events** | Add/update visitor, view daily visitor log |
| **Output** | Updated list with in/out status |
| **Validation** | Mobile number format, required fields |

| Section | Description |
|---|---|
| **Constraints** | Only Watchman can access this module via mobile app |

7. Notification & Mail Module

| Section | Description |
|---|---|
| **Input** | System events (complaint update, maintenance due, password reset, etc.) |
| **Events** | Trigger email or system notification |
| **Output** | Mail sent confirmation, frontend alert popup |
| **Validation** | Valid mail ID, template mapping |
| **Constraints** | Mail send failure fallback mechanism must exist |

8. Reports & Filtering Module

| Section | Description |
|---|---|
| **Input** | Filter criteria (by date, type, status) |
| **Events** | Admin/member filter/search maintenance, complaint, expenses |
| **Output** | Table/List of filtered results, downloadable report (PDF) |
| **Validation** | Valid search inputs, pagination limits |
| **Constraints** | Large result sets may slow performance without pagination |

9.Expenses Module

| Section | Description |
|---|---|
| **Input** | Expense type, amount, date, description, related maintenance cycle |
| **Events** | Admin adds expense record, edits or deletes expense |
| **Output** | Expense list, updated balance reports |
| **Validation** | Valid amount (positive), valid date, non-empty description |
| **Constraints** | Expenses must be linked to a valid expense type; cannot delete expenses |

| Section | Description |
|---|---|
| | linked to finalized reports |

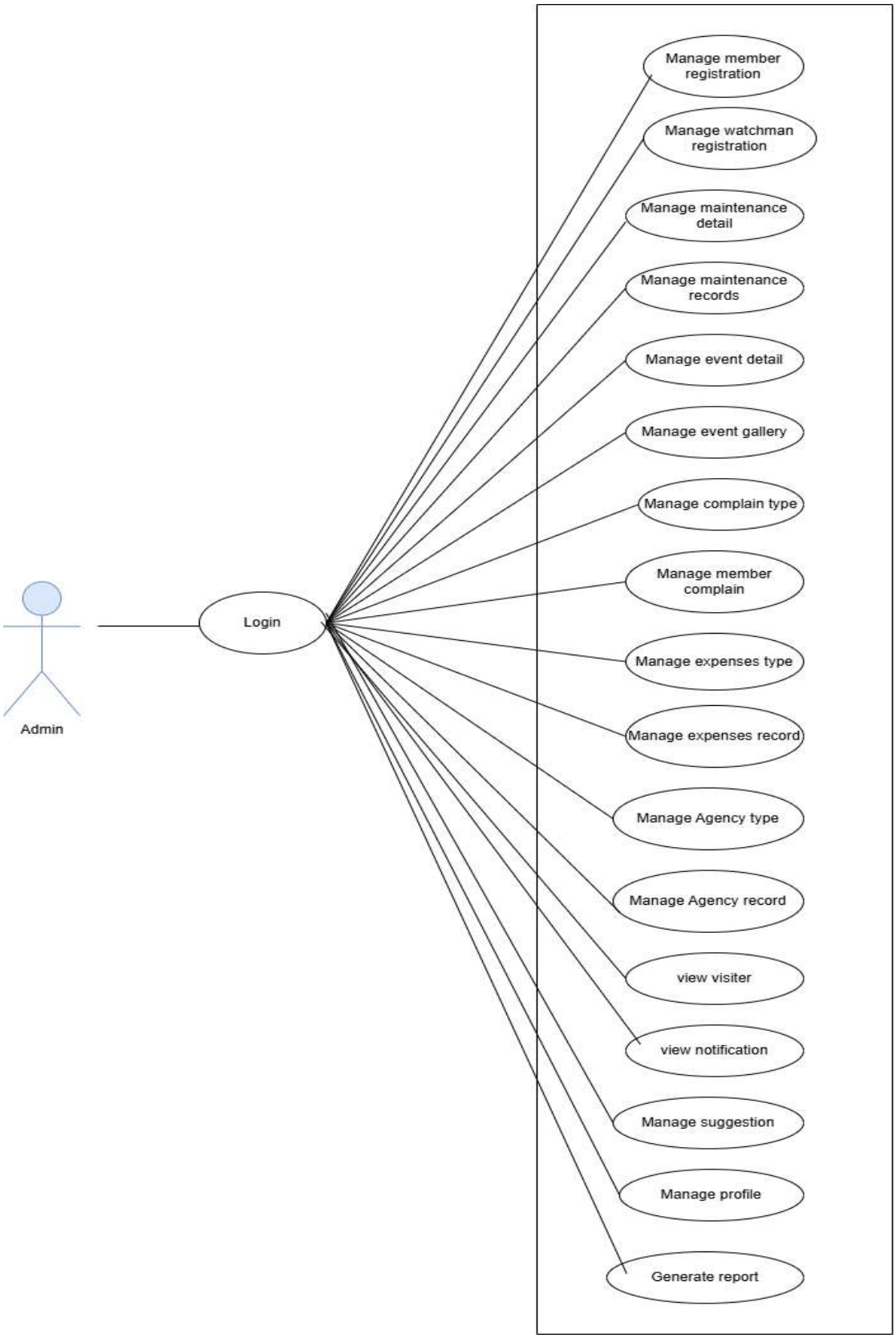
10.Agency Module

| Section | Description |
|---|---|
| **Input** | Agency name, contact details (phone, email), agency type, address |
| **Events** | Admin adds/updates/deletes agency records |
| **Output** | Agency contact list, detailed view for each agency |
| **Validation** | Unique agency name, valid contact formats |
| **Constraints** | Agency deletions only allowed if no pending contracts or expense links |

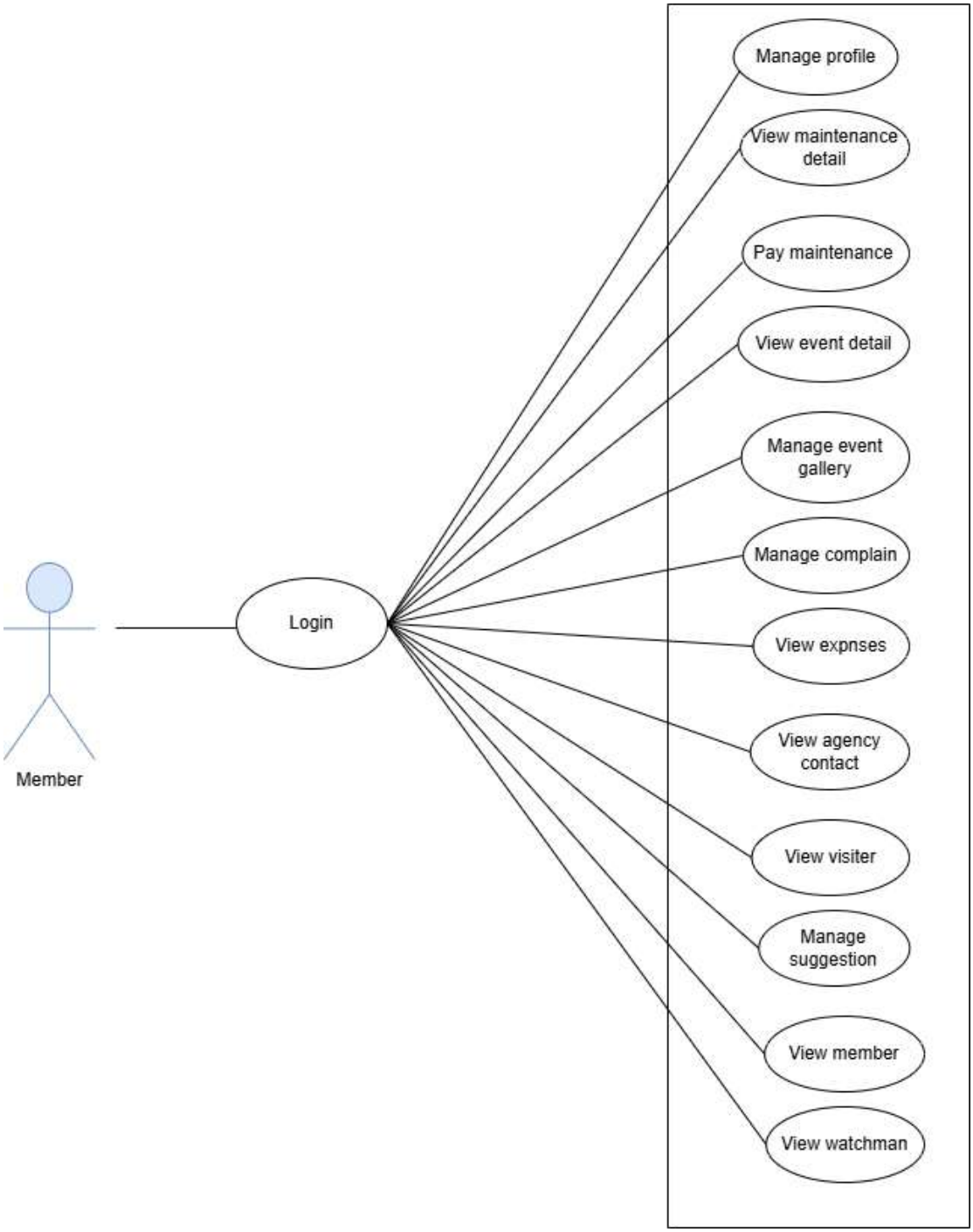
## 5.2 UML Diagram

### 5.2.1 Use Case Diagram

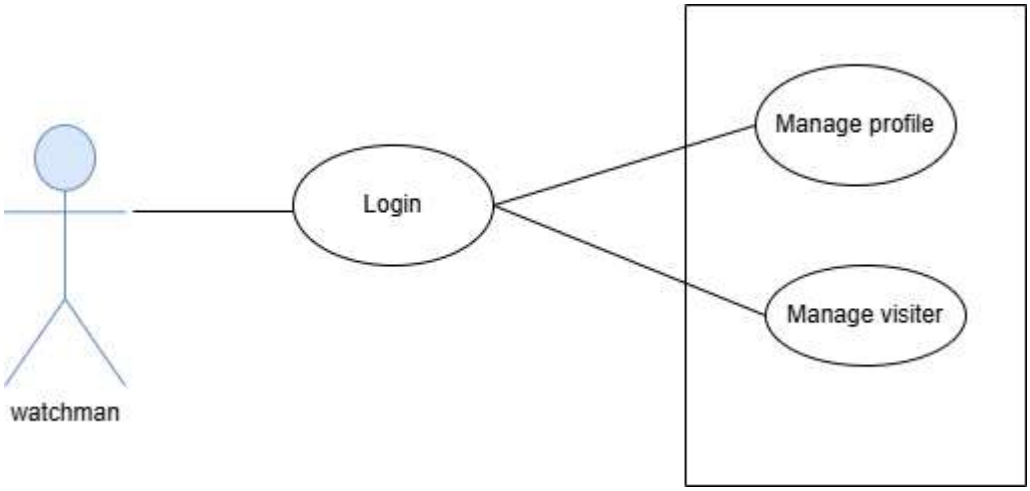
**1. Admin :**



## 2. Member


Manage profile
View maintenance detail
Pay maintenance
View event detail
Manage event gallery
Manage complain
View expnses
View agency contact
View visiter
Manage suggestion
View member
View watchman
Login
Member

**3. Watchmen**




## 5.2.2 CRC – Class Responsibility Collaborator Table

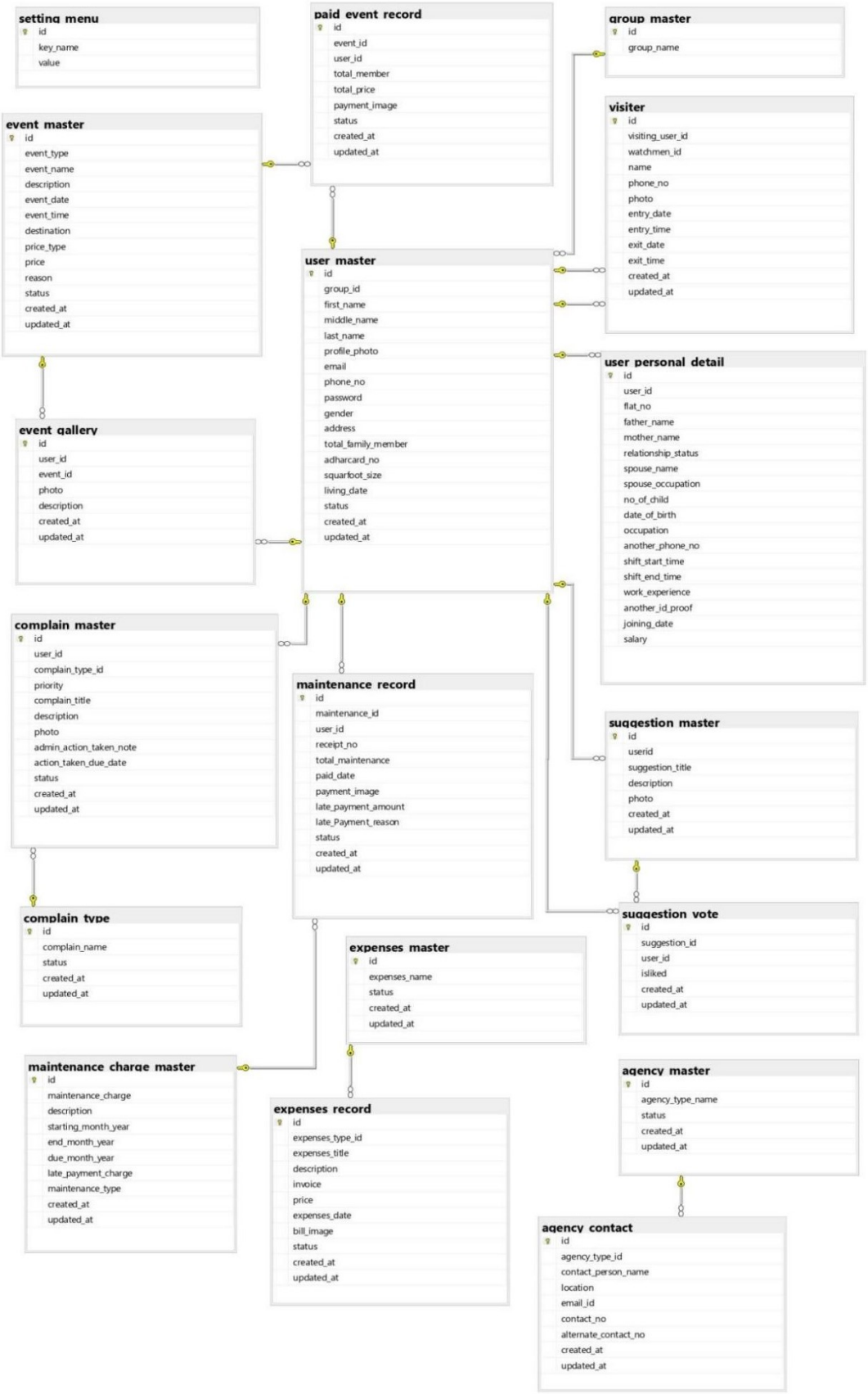
The following table presents the CRC Cards for major classes involved in the **SocioSphere** platform. Each class is defined with its **primary responsibilities** and the **collaborating entities** it interacts with.

- CRC Table

| Class Name | Responsibilities | Collaborators |
|---|---|---|
| **User** | Register/Login, update profile, manage roles and access modules | Admin, NotificationService, Member |
| **Admin** | Manage users, maintenance, complaints, events, expenses, suggestions, agencies | User, Maintenance, Complaint, Event, Expense, Agency, NotificationService |
| **Member** | Pay maintenance, file complaints, give suggestions, register for events | Maintenance, Complaint, Suggestion, Event, NotificationService |
| **Watchman** | Add/view visitor entries, manage in/out time | Visitor, NotificationService |
| **Maintenance** | Store maintenance cycles, process payments, generate receipts | Member, Admin, Expense, NotificationService |

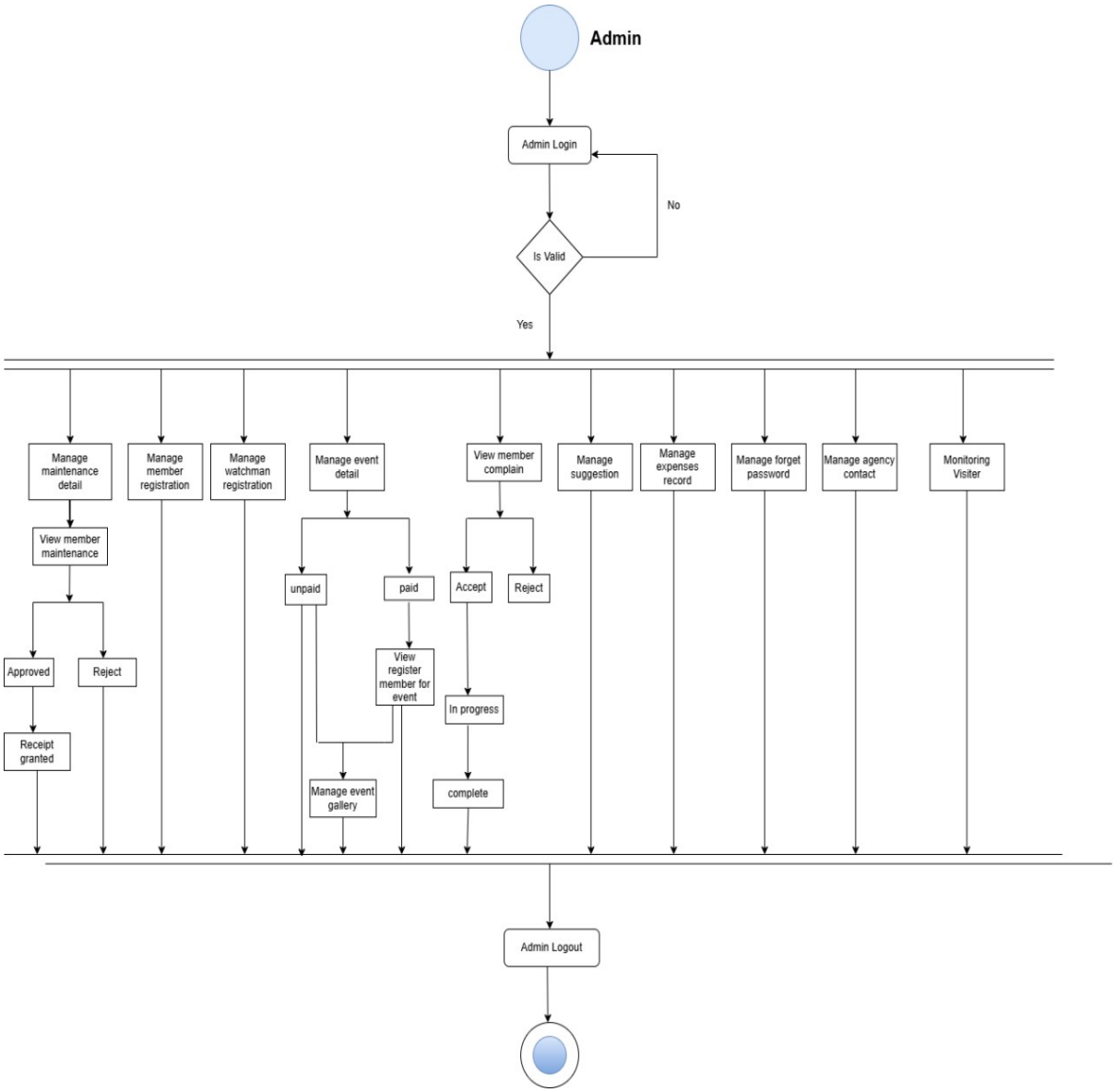
| Class Name | Responsibilities | Collaborators |
|---|---|---|
| **Complaint** | Store complaint info, update status, notify admin and user | Member, Admin, NotificationService |
| **Suggestion** | Add/view suggestions, allow likes/dislikes | Member, Admin |
| **Event** | Create/update events, manage gallery, handle registration/payments | Member, Admin, Gallery |
| **Visitor** | Store visitor info, manage check-in/out timestamps | Watchman, Admin |
| **Expense** | Track expenses, link to maintenance and agencies | Admin, Maintenance, Agency |
| **Agency** | Store agency info (type, name, contact), linked with expense | Admin, Expense |
| **NotificationService** | Send system alerts, emails for password reset, maintenance updates, etc. | User, Admin, Member, Watchman |

## 5.2.3 Class Diagram

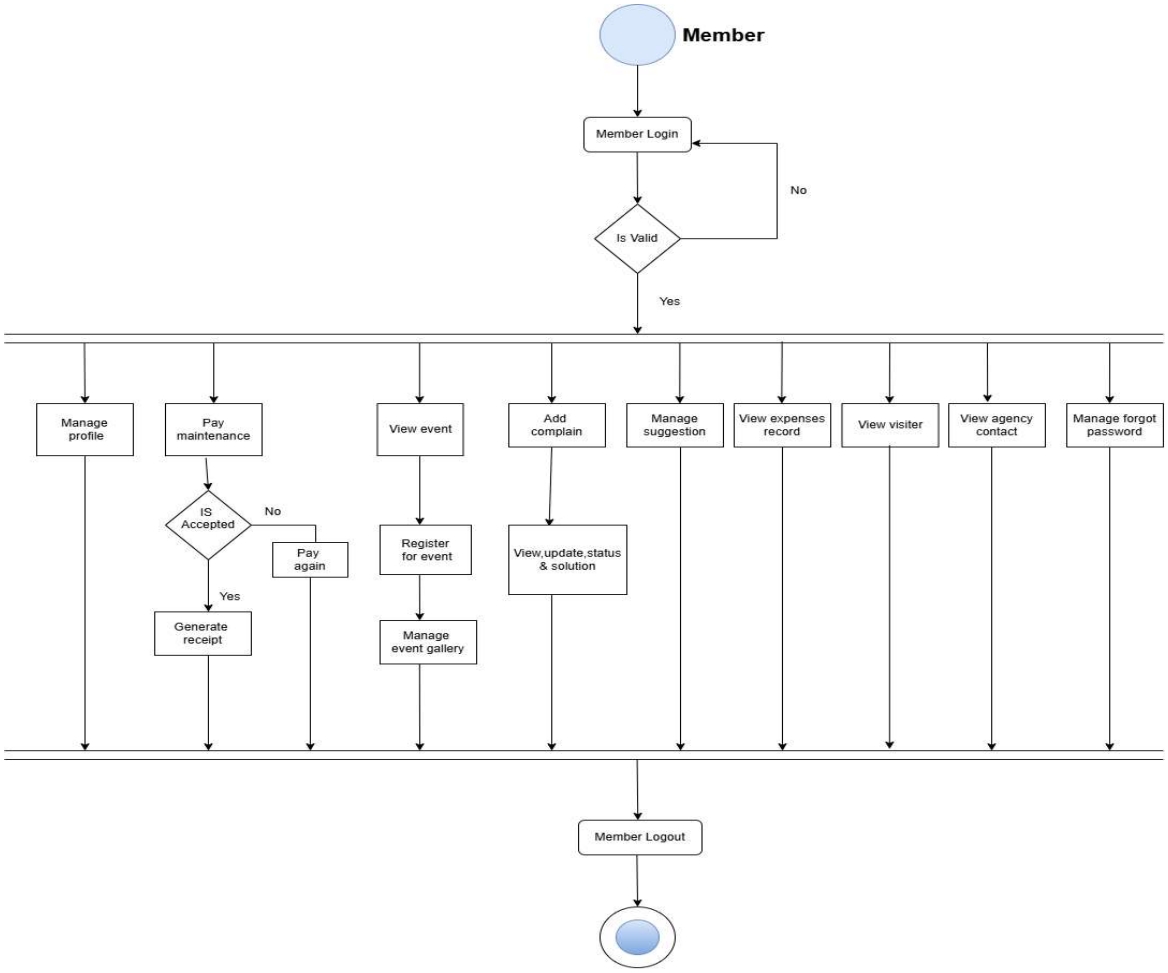


## 5.2.4 Activity Diagram

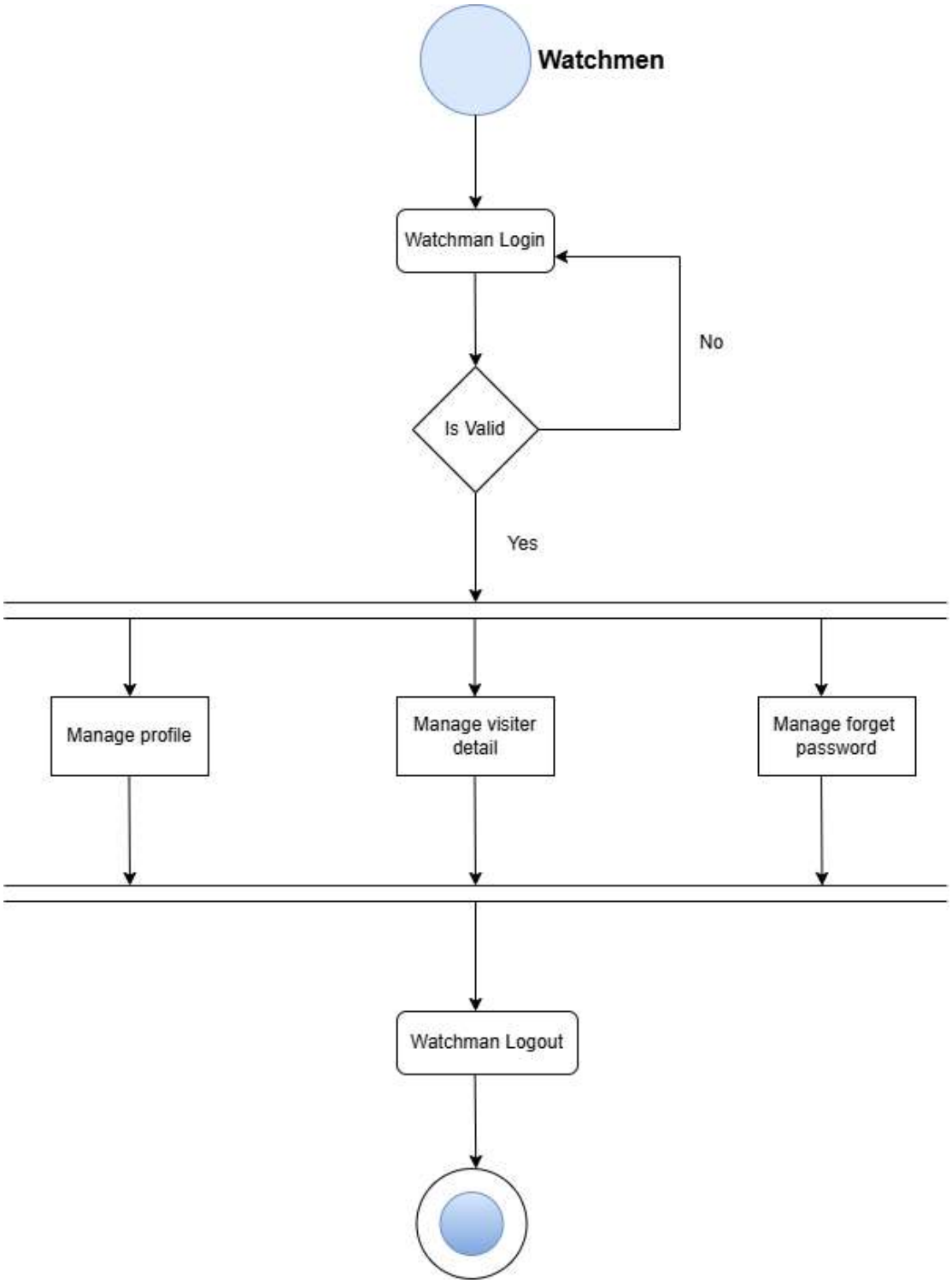
### 1. Admin



## 2.Member

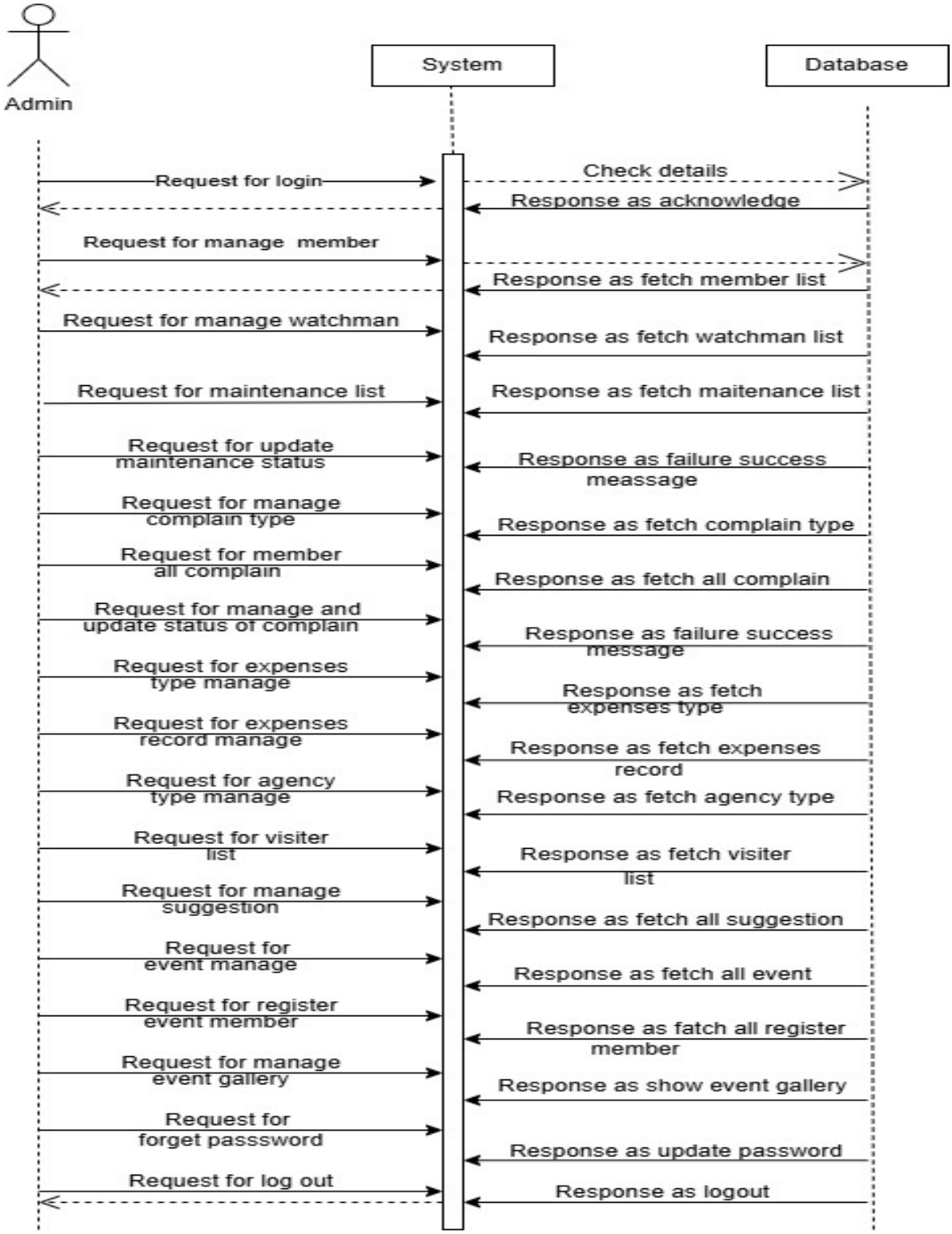

Member
Member Login
No
Is Valid
Yes
Manage profile
Pay maintenance
View event
Add complain
Manage suggestion
View expenses record
View visiter
View agency contact
Manage forgot password
IS Accepted
No
Pay again
Yes
Generate receipt
Register for event
Manage event gallery
View,update,status & solution
Member Logout

**3.Watchmen**

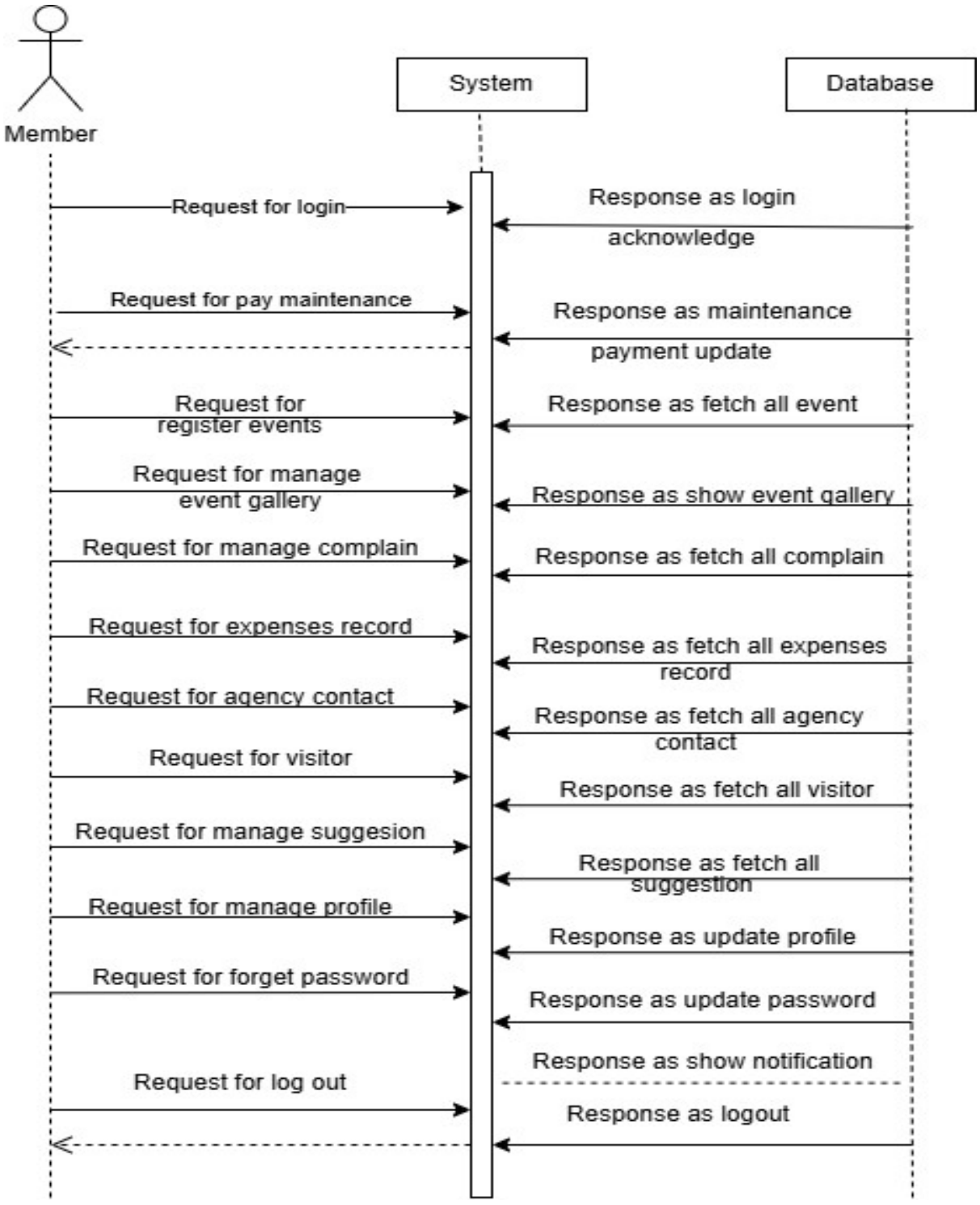

Watchmen
Watchman Login
No
Is Valid
Yes
Manage profile
Manage visiter detail
Manage forget password
Watchman Logout

## 5.2.5 Sequence Diagram

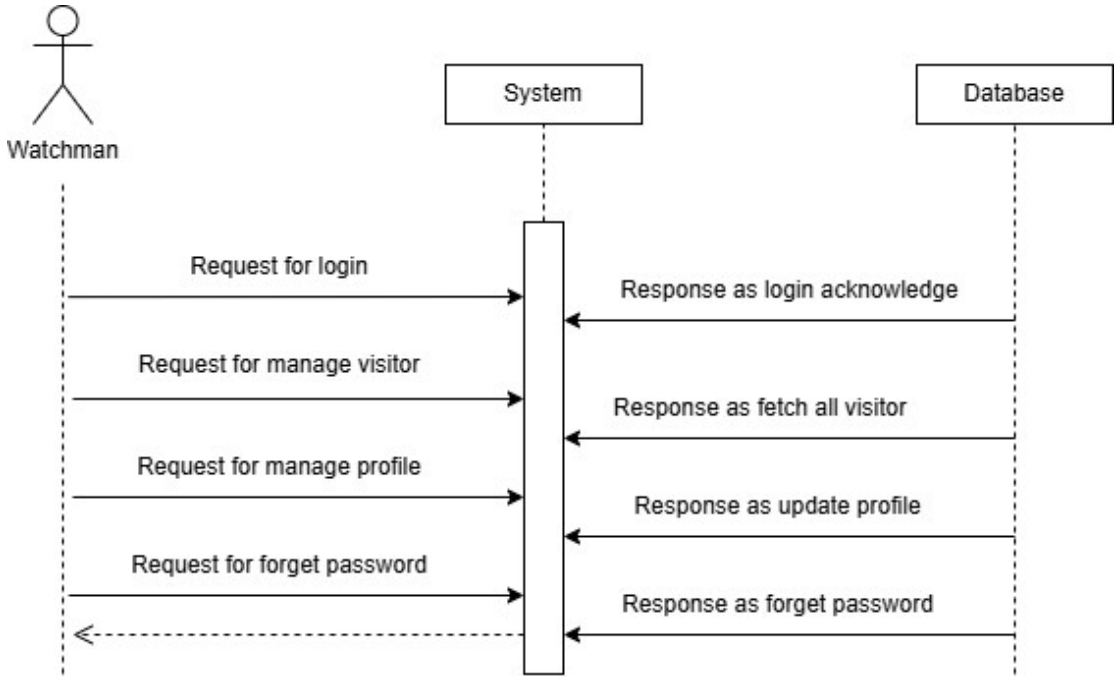
### 1.Admin



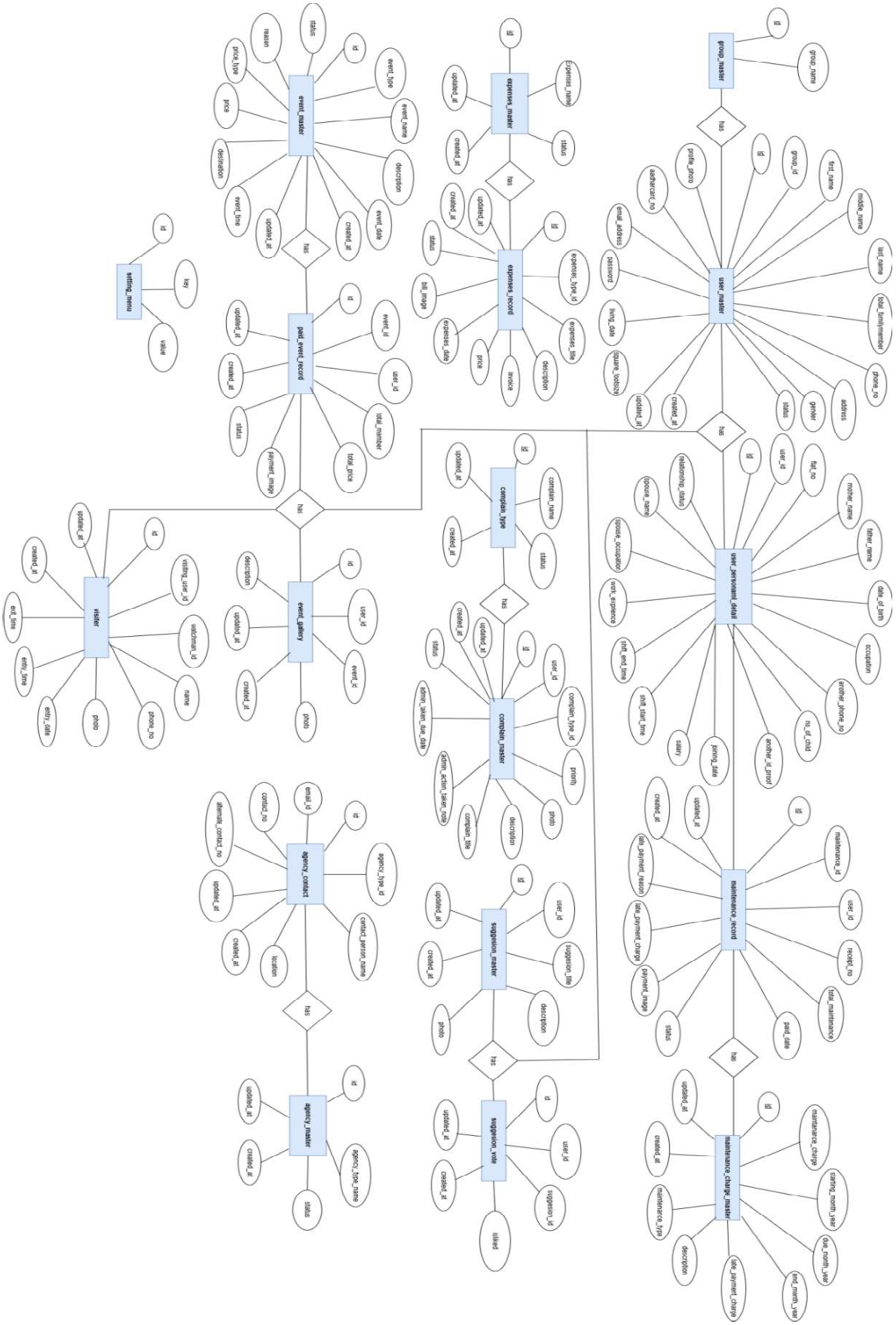
**2.Member**


Member
System
Database
Request for login
Response as login acknowledge
Request for pay maintenance
Response as maintenance payment update
Request for register events
Response as fetch all event
Request for manage event gallery
Response as show event gallery
Request for manage complain
Response as fetch all complain
Request for expenses record
Response as fetch all expenses record
Request for agency contact
Response as fetch all agency contact
Request for visitor
Response as fetch all visitor
Request for manage suggesion
Response as fetch all suggestion
Request for manage profile
Response as update profile
Request for forget password
Response as update password
Response as show notification
Request for log out
Response as logout

## 3.Watchmen


Watchman
System
Database
Request for login
Response as login acknowledge
Request for manage visitor
Response as fetch all visitor
Request for manage profile
Response as update profile
Request for forget password
Response as forget password

## 5.3 E-R Diagram



# 6.Software Design

## 6.1 Database Design

### 1. Group_master

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | No | Primary Key |
| **group_name** | Varchar(255) | Yes | |

### 2. User_master

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **group_id** | Int(11) | YES | Foreign key |
| **commite_member_id** | Int(11) | YES | |
| **first_name** | Varchar(255) | YES | |
| **middle_name** | Varchar(255) | YES | |
| **last_name** | Varchar(255) | YES | |
| **profile_photo** | Varchar(400) | YES | |
| **email_address** | Varchar(300) | YES | |
| **phone_no** | Bigint(20) | YES | |
| **password** | Text | YES | |
| **gender** | Varchar(255) | YES | |
| **address** | Text | YES | |
| **total_familymember** | Int(11) | YES | |
| **adharcard_no** | Int(11) | YES | |
| **square_footsize** | Float | YES | |
| **living_date** | Date | YES | |
| **status** | Varchar(300) | YES | |
| **created_at** | Datetime | YES | |

| | | |
|---|---|---|
| **updated_at** | Datetime | YES |

- group_id(foreign_key)->Group_master(id)

## 3. User_personal_detail

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **user_id** | Int(11) | YES | Foreign key |
| **flat_no** | Int(11) | YES | |
| **father_name** | Varchar(255) | YES | |
| **mother_name** | Varchar(255) | YES | |
| **relationship_status** | Varchar(200) | YES | |
| **spouse_name** | Varchar(200) | YES | |
| **spouse_occupation** | Varchar(255) | YES | |
| **no_of_child** | Int(11) | YES | |
| **date_of_birth** | Date | YES | |
| **occupation** | Varchar(300) | YES | |
| **another_phone_no** | Bigint(20) | YES | |
| **shift_start_time** | Time | YES | |
| **shift_end_time** | Time | YES | |
| **work_experience** | Varchar(400) | YES | |
| **another_id_proof** | Varchar(255) | YES | |
| **joining_date** | Date | YES | |
| **salary** | Float | YES | |

- user_id(foreign_key)->User_master(id)

## 4. Maintenance_charge_master

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **maintenace_charge** | Double | YES | |

| | | | |
|---|---|---|---|
| **description** | Text | YES | |
| **starting_month_year** | Date | YES | |
| **due_month_year** | Date | YES | |
| **end_month_year** | Date | YES | |
| **maintennace_charge** | float | YES | |
| **maintenance_type** | Varchar(255) | YES | |
| **created_at** | Datetime | YES | |
| **updated_at** | Datetime | YES | |

## 5. Maintenance_record

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **maintenance_id** | Int(11) | YES | Foreign key |
| **user_id** | Int(11) | YES | |
| **receipt_no** | Int(11) | YES | |
| **total_maintenance** | Double | YES | |
| **paid_date** | Date | YES | |
| **from_month** | Date | YES | |
| **to_month** | Date | YES | |
| **payment_image** | Varchar(400) | YES | |
| **late_payment_charge** | Float | YES | |
| **late_payment_reason** | Varchar(255) | YES | |
| **IsLatePayment** | Boolean | YES | |
| **status** | Varchar(300) | YES | |
| **created_at** | Datetime | YES | |
| **updated_at** | Datetime | YES | |

- User_id(foreign_key)->User_master(id)
- Maintenance_id(foreign_key)->Maintenance_charge_master(id)

## 6. **Expenses_master**

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **expenses_name** | Varchar(255) | YES | |
| **status** | Varchar(300) | YES | |
| **created_at** | Datetime | YES | |
| **updated_at** | Datetime | YES | |

## 7. Expenses_record

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **expenses_type_id** | Int(11) | YES | Foreign key |
| **expenses_title** | Varchar(255) | YES | |
| **description** | Text | YES | |
| **invoice** | Varchar(255) | YES | |
| **price** | Double | YES | |
| **expenses_Date** | Date | YES | |
| **bill_image** | Varchar(400) | YES | |
| **status** | Varchar(300) | YES | |
| **created_at** | Datetime | YES | |
| **updated_at** | Datetime | YES | |

- Expenses_type_id(foreign_key)->Expenses_master(id)

## 8. Complain_type

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|

| | | | |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **complain_name** | Varchar(255) | YES | |
| **status** | Varchar(300) | YES | |
| **created_at** | Datetime | YES | |
| **updated_at** | Datetime | YES | |

## 9. Complain_master

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **user_id** | Int(11) | YES | Foreign key |
| **complain_type_id** | Int(11) | YES | Foreign key |
| **priority** | Varchar(255) | YES | |
| **complain_title** | Varchar(255) | YES | |
| **description** | Text | YES | |
| **photo** | Varchar(400) | YES | |
| **admin_action_taken_note** | Varchar(255) | YES | |
| **action_taken_due_date** | Date | YES | |
| **status** | Varchar(300) | YES | |
| **created_at** | Datetime | YES | |
| **updated_at** | Datetime | YES | |

- User_id(foreign_key)->User_master(id)
- Complain_type_id(foreign_key)->Complain_type(id)

## 10. Suggesion_master

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |

| | | | |
|---|---|---|---|
| **userid** | Int(11) | YES | Foreign key |
| **suggestion_title** | Varchar(255) | YES | |
| **description** | Text | YES | |
| **photo** | Varchar(400) | YES | |
| **created_at** | Datetime | YES | |
| **updated_at** | Datetime | YES | |

- User_id(foreign_key)->User_master(id)

## 11. Suggesion_vote

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **suggestion_id** | Int(11) | YES | Foreign key |
| **user_id** | Int(11) | YES | |
| **isliked** | Bit | YES | |
| **created_at** | Datetime | YES | |
| **updated_at** | Datetime | YES | |

- User_id(foreign_key)->User_master(id)
- Suggestion_id(foreign_key)->Suggestion_master(id)

## 12. Event_master

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **event_type** | Varchar(255) | YES | |
| **event_name** | Varchar(255) | YES | |
| **description** | Text | YES | |
| **event_date** | Date | YES | |
| **event_time** | Time | YES | |
| **destination** | Varchar(400) | YES | |

| | | | |
|---|---|---|---|
| **price_type** | Varchar(255) | YES | |
| **price** | Float | YES | |
| **reason** | Varchar(500) | YES | |
| **status** | Varchar(300) | YES | |
| **created_at** | datetime | YES | |
| **updated_at** | datetime | YES | |

## 13. Paid_event_record

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **event_id** | Int(11) | YES | Foreign key |
| **user_id** | Int(11) | YES | Foreign key |
| **total_member** | Int(11) | YES | |
| **total_price** | Float | YES | |
| **payment_image** | Varchar(400) | YES | |
| **status** | Varchar(300) | YES | |
| **created_at** | Datetime | YES | |
| **updated_at** | Datetime | YES | |

- User_id(foreign_key)->User_master(id)
- Event_id(foreign_key)->Event_master(id)

## 14. Event_gallery

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **user_id** | Int(11) | YES | Foreign key |

| | | | |
|---|---|---|---|
| **event_id** | Int(11) | YES | Foreign key |
| **photo** | Varchar(300) | YES | |
| **description** | Text | YES | |
| **created_at** | Datetime | YES | |
| **updated_at** | Datetime | YES | |

## 15. Agency_master

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **agency_type_name** | Varchar(255) | YES | |
| **status** | Varchar(300) | YES | |
| **created_at** | Timestamp | YES | |
| **updated_at** | Timestamp | YES | |

## 16. Agency_contact

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | **Int(11)** | **NO** | **Primary Key** |
| **agency_type_id** | **Int(11)** | **YES** | **Foreign key** |
| **contact_person_name** | **Varchar(255)** | **YES** | |
| **location** | **Varchar(500)** | **YES** | |
| **email_id** | **Varchar(300)** | **YES** | |
| **contact_no** | **Bigint(20)** | **YES** | |
| **alternate_contact_no** | **Bigint(20)** | **YES** | |
| **created_at** | **Datetime** | **YES** | |
| **updated_at** | **Datetime** | **YES** | |

- Agency_type_id(foreign_key)->Agency_master(id)

## 17. Visiter

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | NO | Primary Key |
| **visiting_user_id** | Int(11) | YES | Foreign key |
| **watchmen_id** | Int(11) | YES | Foreign key |
| **name** | Varcha(255) | YES | |
| **phone_no** | BigInt(20) | YES | |
| **photo** | Varchar(300) | YES | |
| **entry_date** | Date | YES | |
| **entry_time** | Time | YES | |
| **exit_date** | Date | YES | |
| **exit_time** | Time | YES | |
| **created_at** | Datetime | YES | |
| **updated_at** | Datetime | YES | |

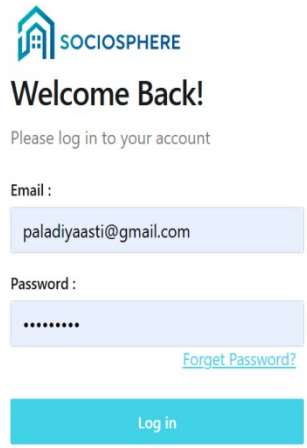
- Visiting_user_id(foreign_key)->User_master(id)
- Watchmen_id(Foreign_key)->User_master(id)

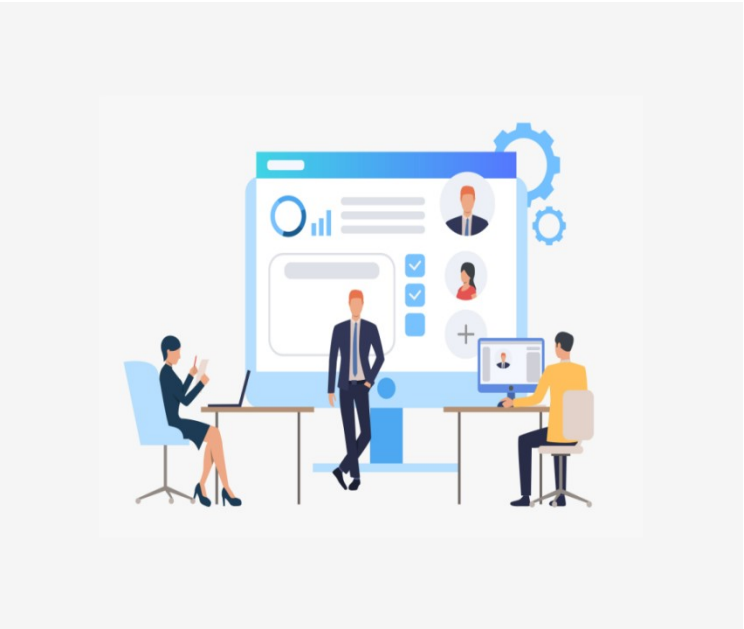
## 18. Setting_menu

| Column Name | Data Type | Null | Constraints |
|---|---|---|---|
| **id** | Int(11) | YES | Primary Key |
| **key** | Varchar(255) | YES | |
| **value** | Varchar(255) | YES | |

## 6.2 Interface Design sitemap followed with page snapshots

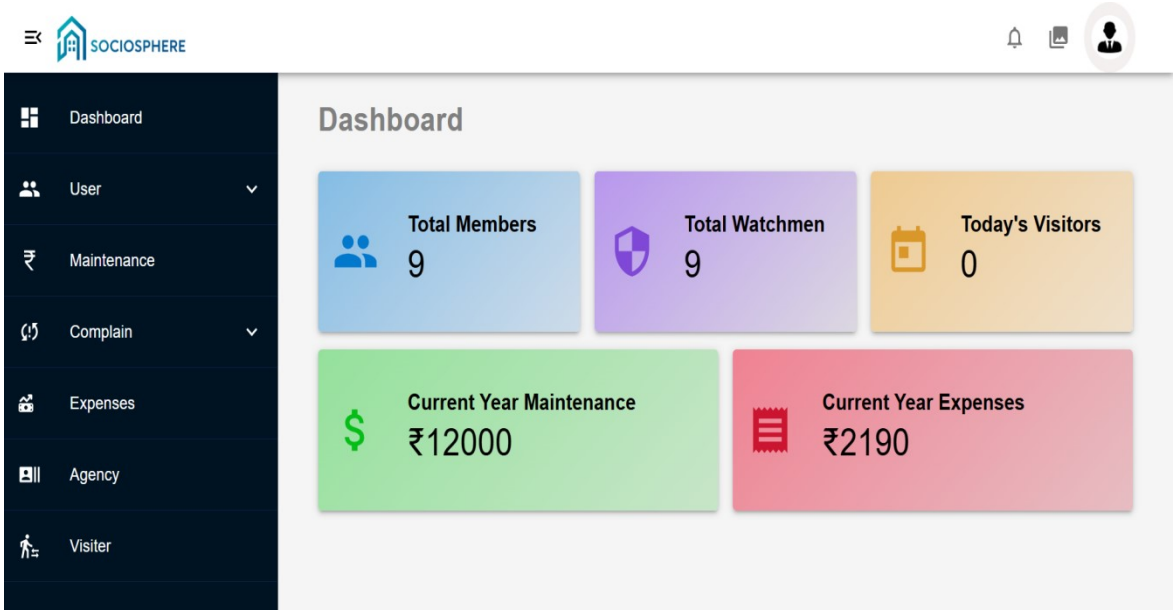
### 1. Admin and user login:-



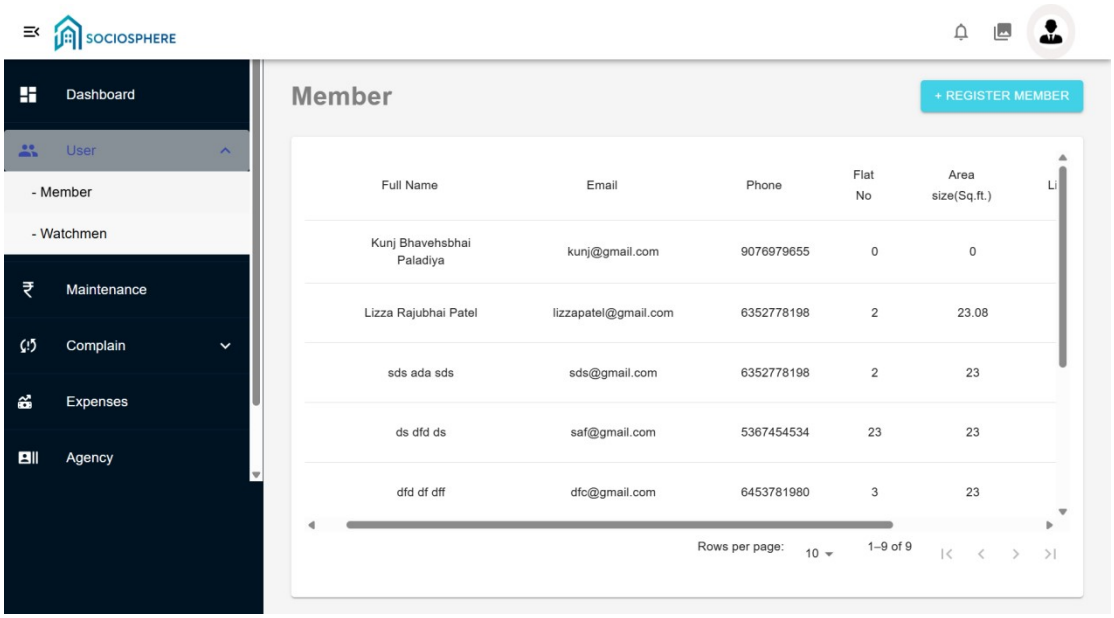



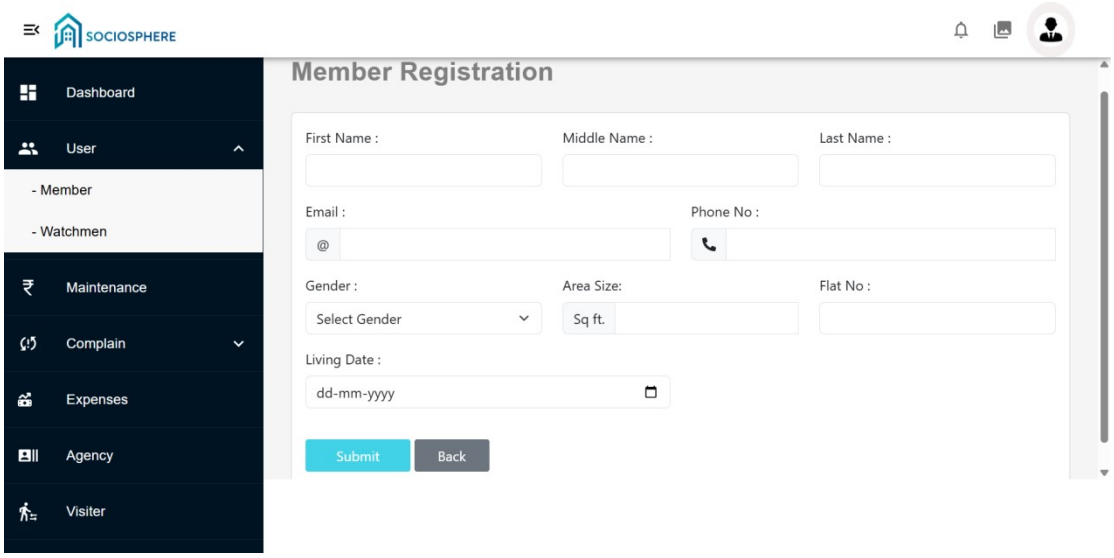
### 1. Admin :

- **Dashboard**

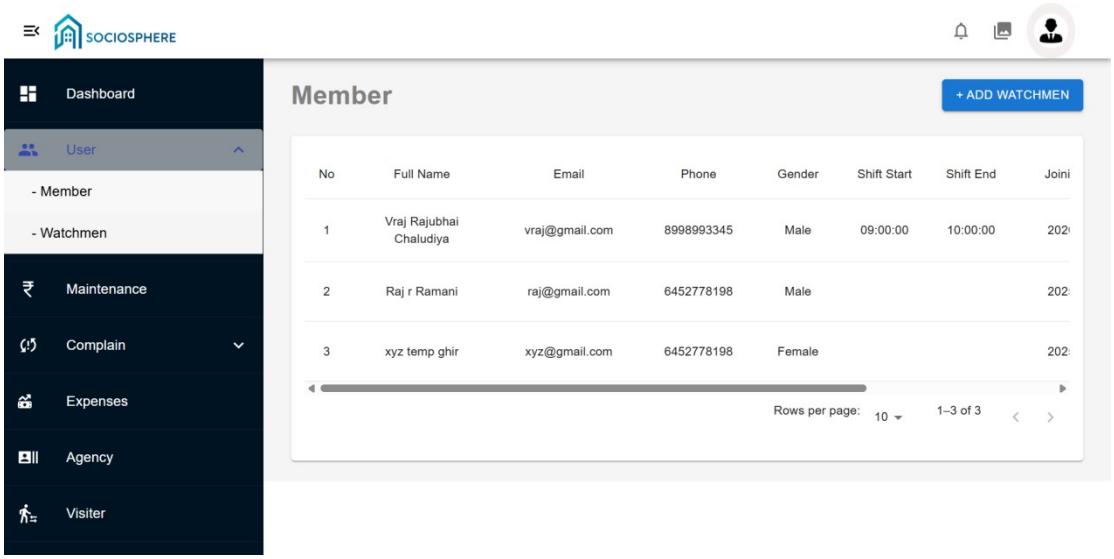


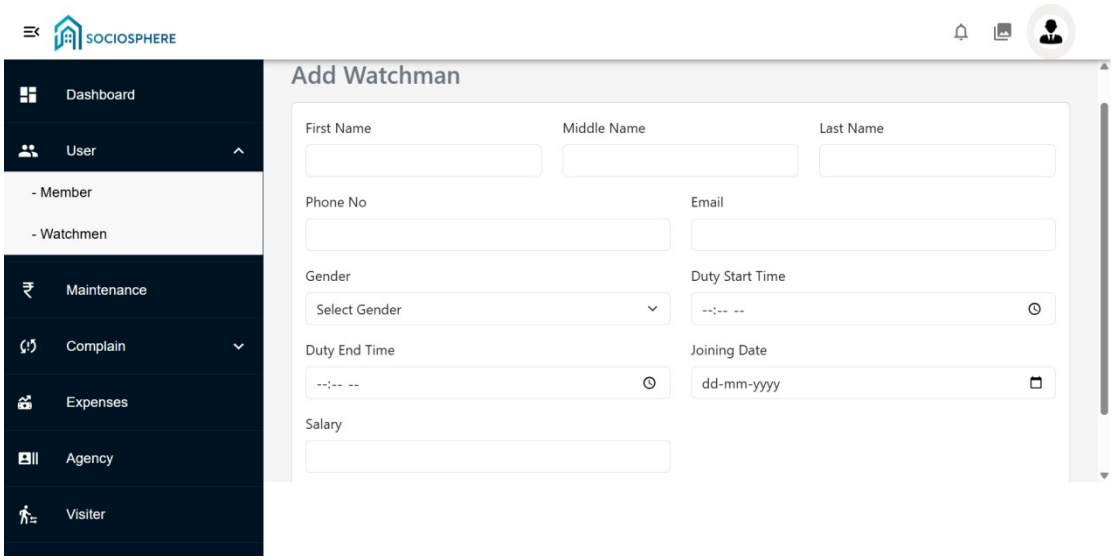

### Member Registration


SOCIOSPHERE
Dashboard
User
- Member
- Watchmen
Maintenance
Complain
Expenses
Agency
Member
+ REGISTER MEMBER
Full Name
Email
Phone
Flat No
Area size(Sq.ft.)
Kunj Bhavehsbhai Paladiya
kunj@gmail.com
9076979655
0
0
Lizza Rajubhai Patel
lizzapatel@gmail.com
6352778198
2
23.08
sds ada sds
sds@gmail.com
6352778198
2
23
ds dfd ds
saf@gmail.com
5367454534
23
23
dfd df dff
dfc@gmail.com
6453781980
3
23
Rows per page:
10
1–9 of 9

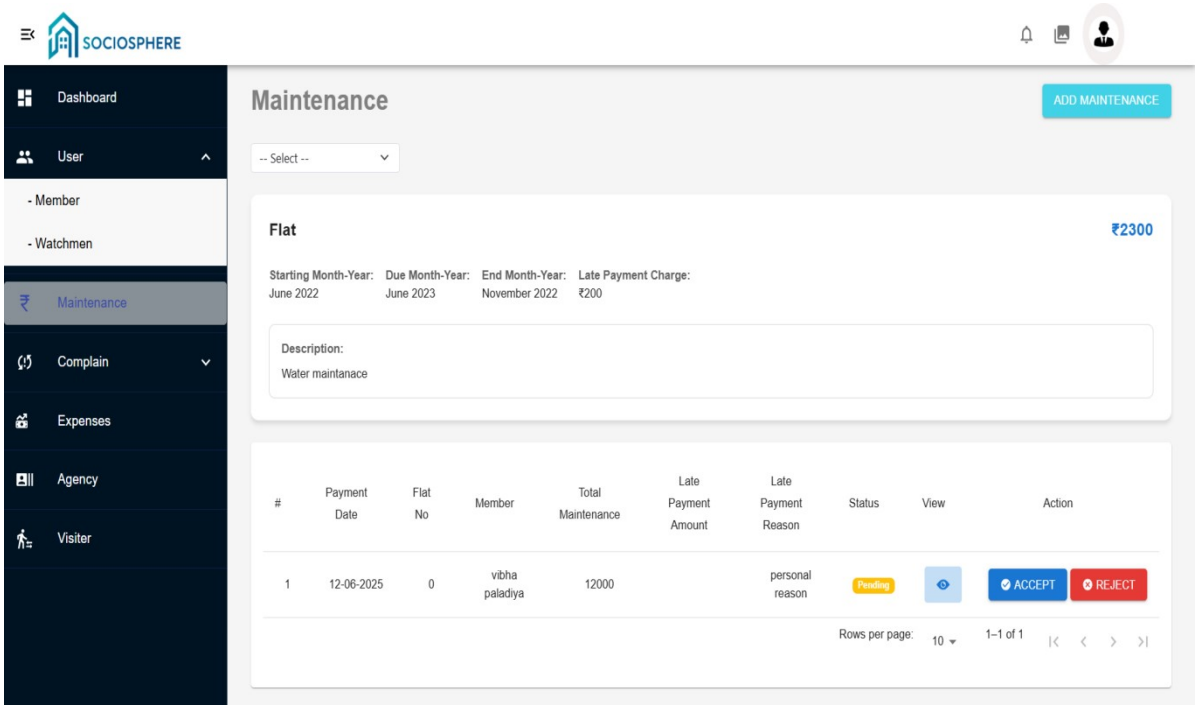

SOCIOSPHERE
Dashboard
User
- Member
- Watchmen
Maintenance
Complain
Expenses
Agency
Visiter
Member Registration
First Name :
Middle Name :
Last Name :
Email :
@
Phone No :
Gender :
Select Gender
Area Size:
Sq ft.
Flat No :
Living Date :
dd-mm-yyyy
Submit
Back

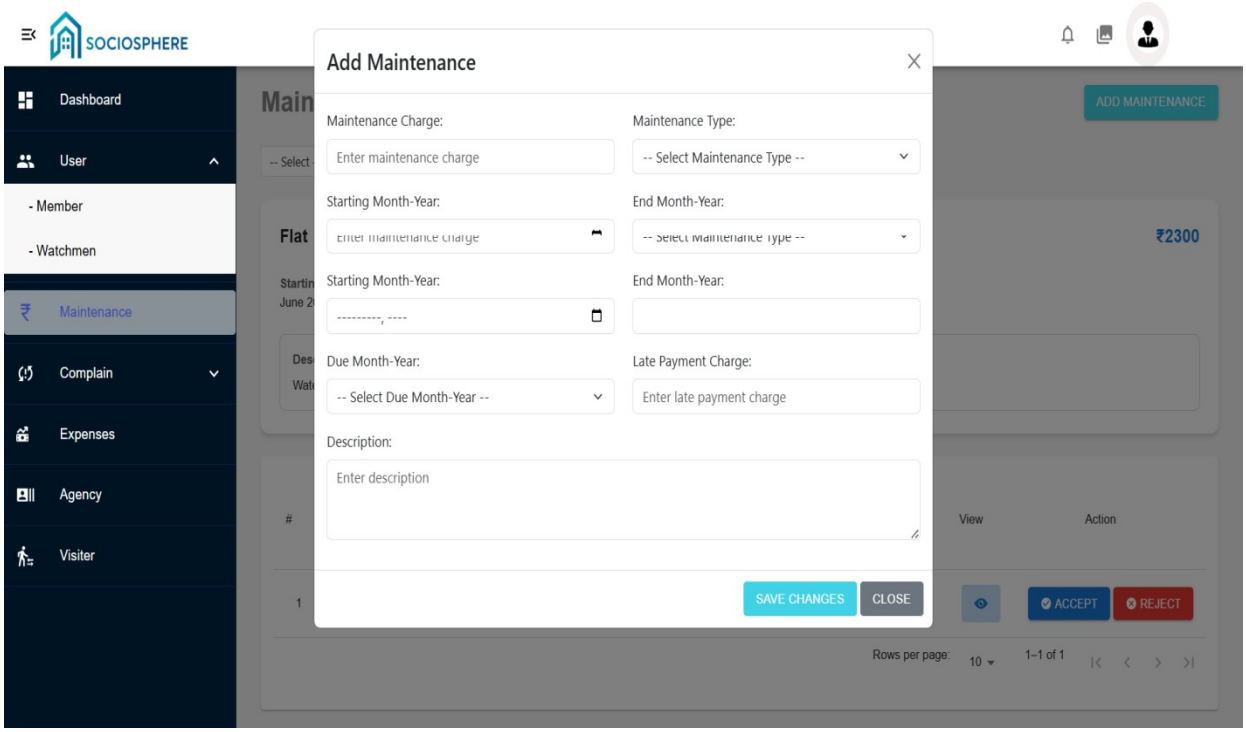
# Watchmen registration



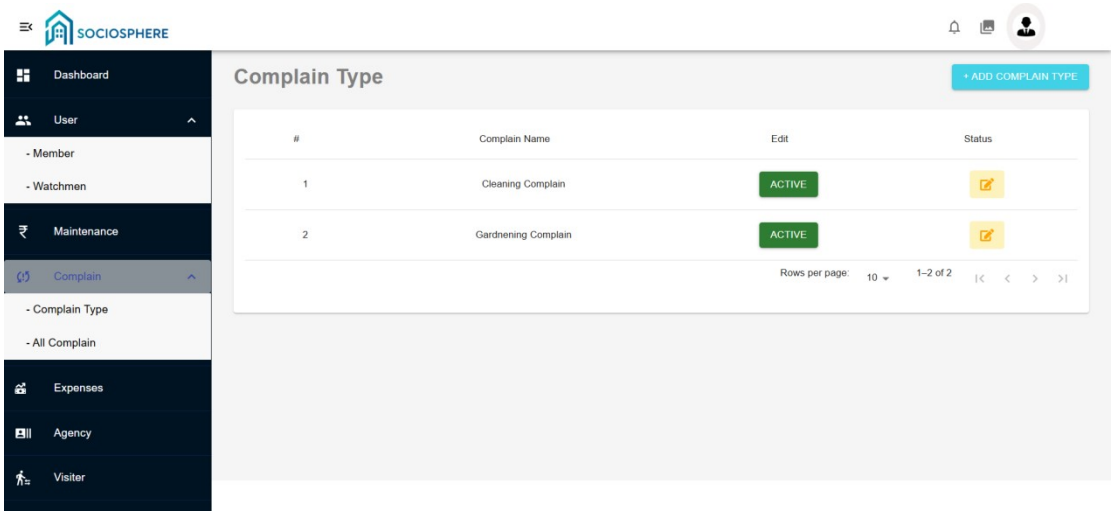



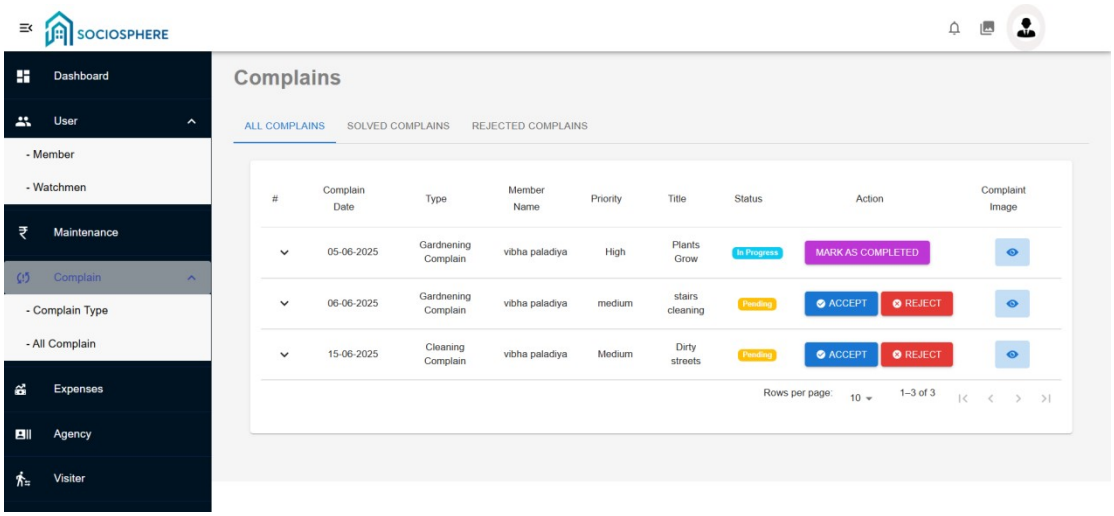
# Maintenance



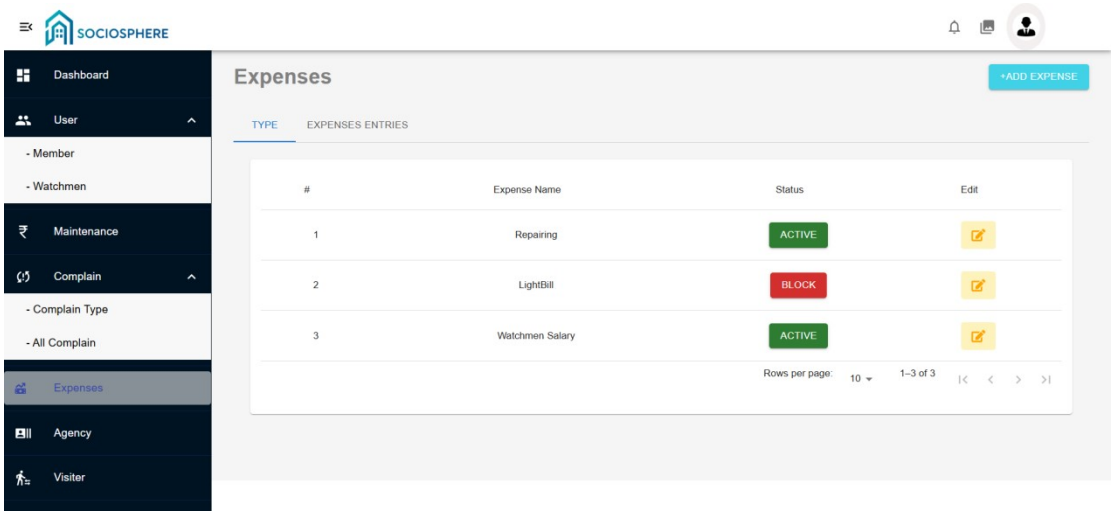



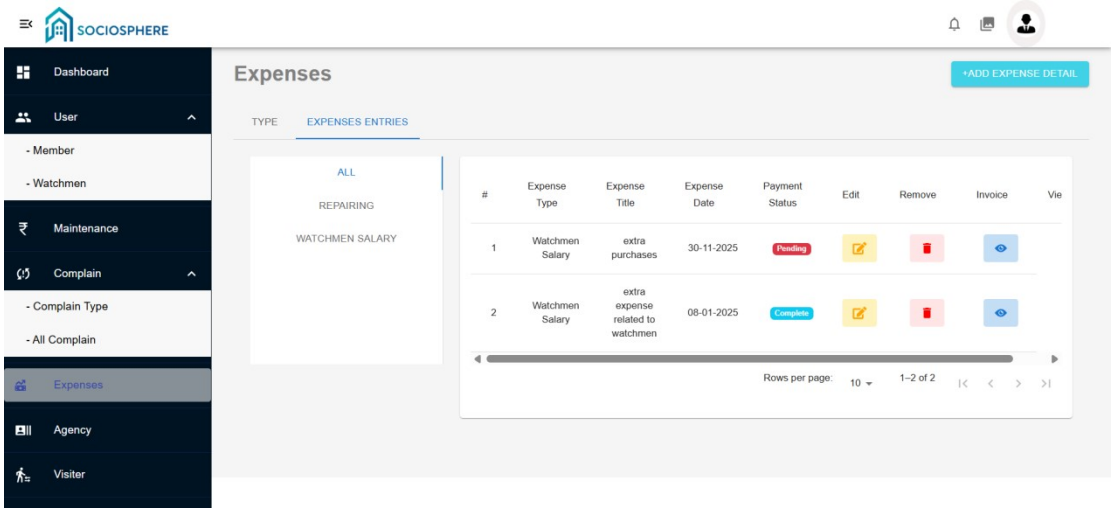
# Complain

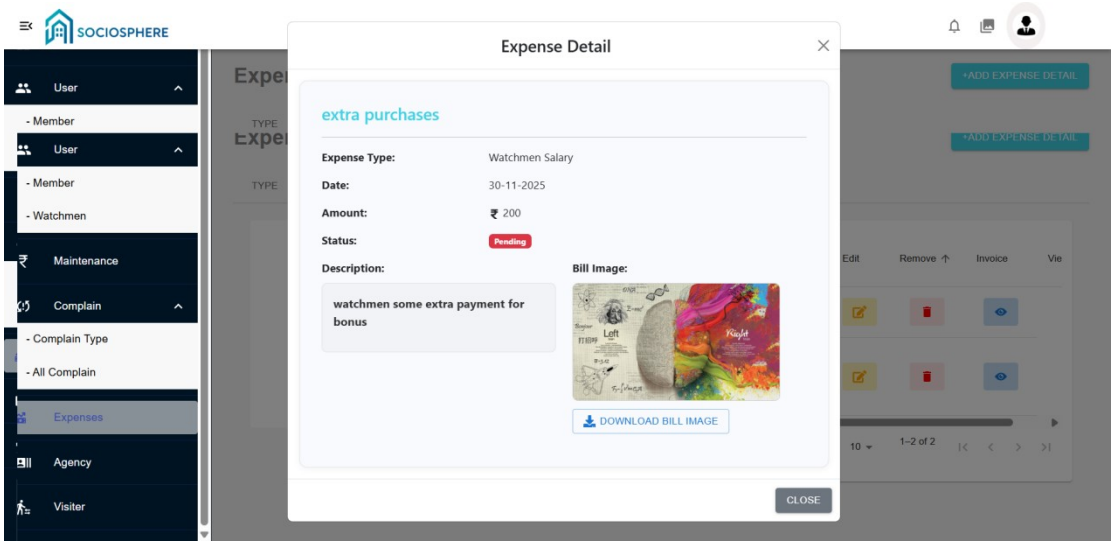

SOCIOSPHERE
Dashboard
User
- Member
- Watchmen
Maintenance
Complain
- Complain Type
- All Complain
Expenses
Agency
Visiter
Complain Type
+ ADD COMPLAIN TYPE
# Complain Name Edit Status
1 Cleaning Complain ACTIVE
2 Gardnening Complain ACTIVE
Rows per page: 10 1–2 of 2


SOCIOSPHERE
Dashboard
User
- Member
- Watchmen
Maintenance
Complain
- Complain Type
- All Complain
Expenses
Agency
Visiter
Complains
ALL COMPLAINS
SOLVED COMPLAINS
REJECTED COMPLAINS
# Complain Date Type Member Name Priority Title Status Action Complaint Image
05-06-2025 Gardnening Complain vibha paladiya High Plants Grow In Progress MARK AS COMPLETED
06-06-2025 Gardnening Complain vibha paladiya medium stairs cleaning Pending ACCEPT REJECT
15-06-2025 Cleaning Complain vibha paladiya Medium Dirty streets Pending ACCEPT REJECT
Rows per page: 10 1–3 of 3

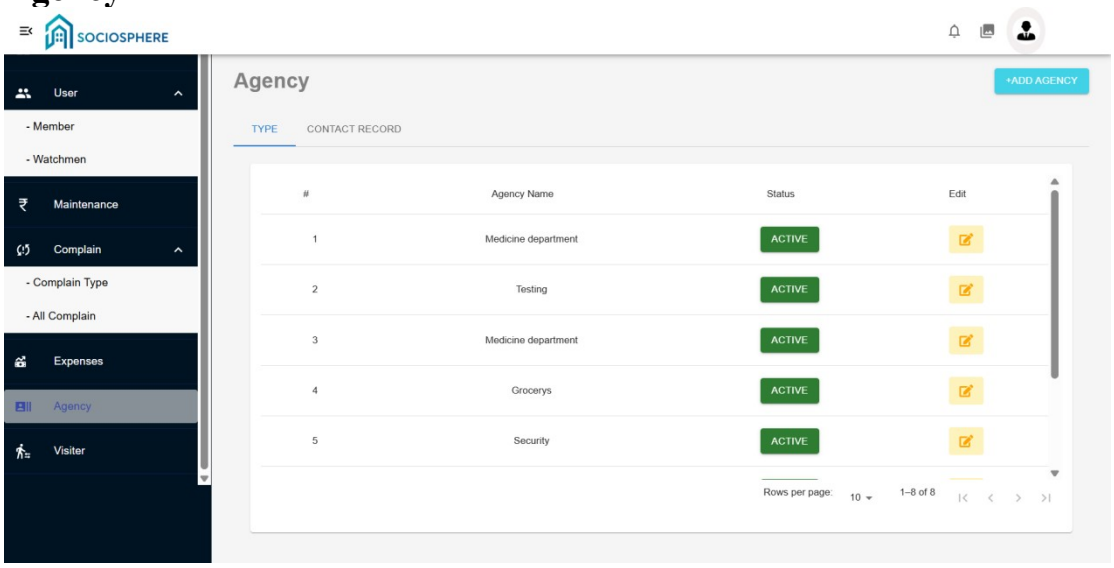
# Expenses

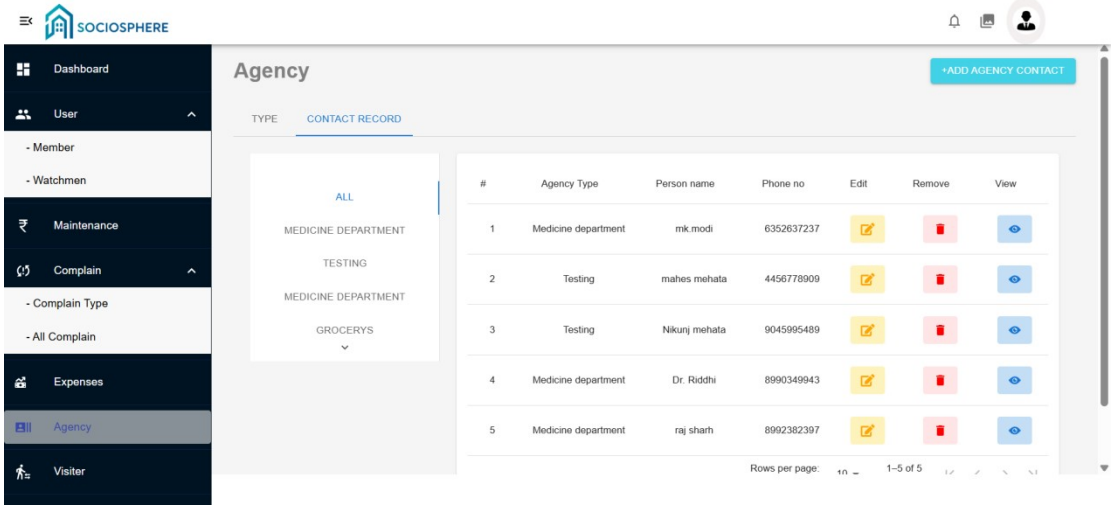

SOCIOSPHERE
Dashboard
User
- Member
- Watchmen
Maintenance
Complain
- Complain Type
- All Complain
Expenses
Agency
Visiter
Expenses
+ADD EXPENSE
TYPE
EXPENSES ENTRIES
# Expense Name Status Edit
1 Repairing ACTIVE
2 LightBill BLOCK
3 Watchmen Salary ACTIVE
Rows per page: 10 1–3 of 3

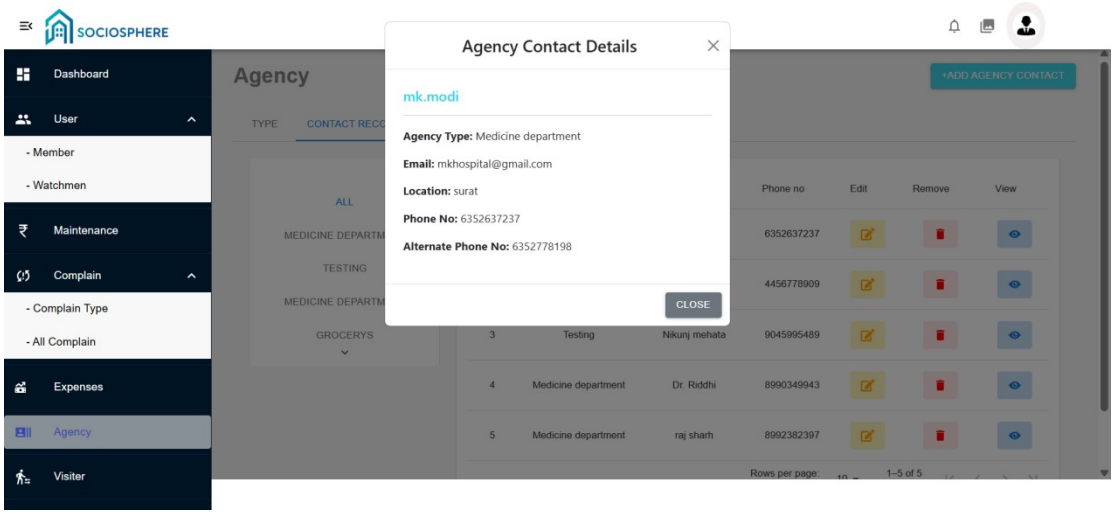

SOCIOSPHERE
Dashboard
User
- Member
- Watchmen
Maintenance
Complain
- Complain Type
- All Complain
Expenses
Agency
Visiter
Expenses
+ADD EXPENSE DETAIL
TYPE
EXPENSES ENTRIES
ALL
REPAIRING
WATCHMEN SALARY
# Expense Type Expense Title Expense Date Payment Status Edit Remove Invoice Vie
1 Watchmen Salary extra purchases 30-11-2025 Pending
2 Watchmen Salary extra expense related to watchmen 08-01-2025 Complete
Rows per page: 10 1–2 of 2


SOCIOSPHERE
Expense Detail
extra purchases
Expense Type: Watchmen Salary
Date: 30-11-2025
Amount: ₹ 200
Status: Pending
Description:
watchmen some extra payment for bonus
Bill Image:
DOWNLOAD BILL IMAGE
CLOSE

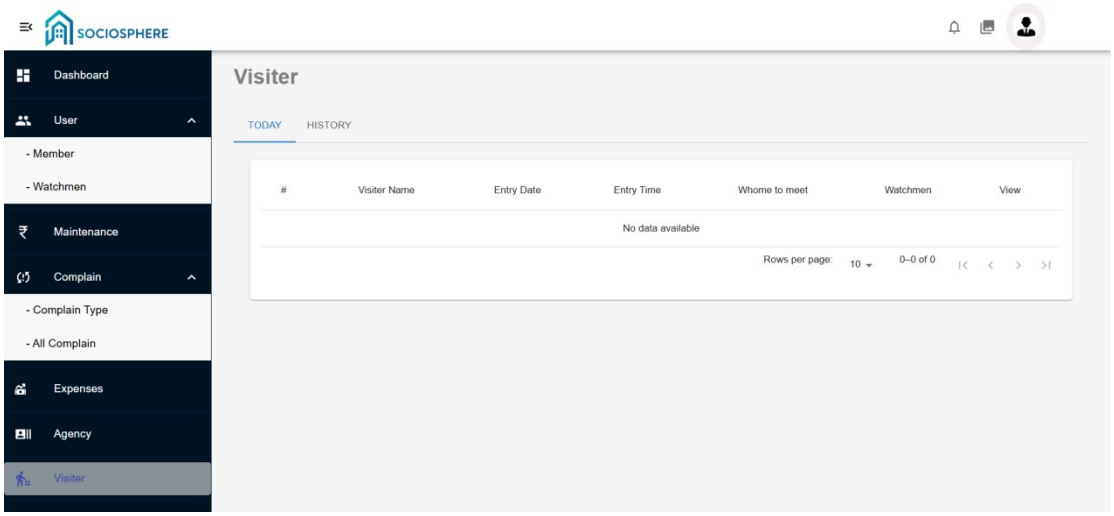
# Agency

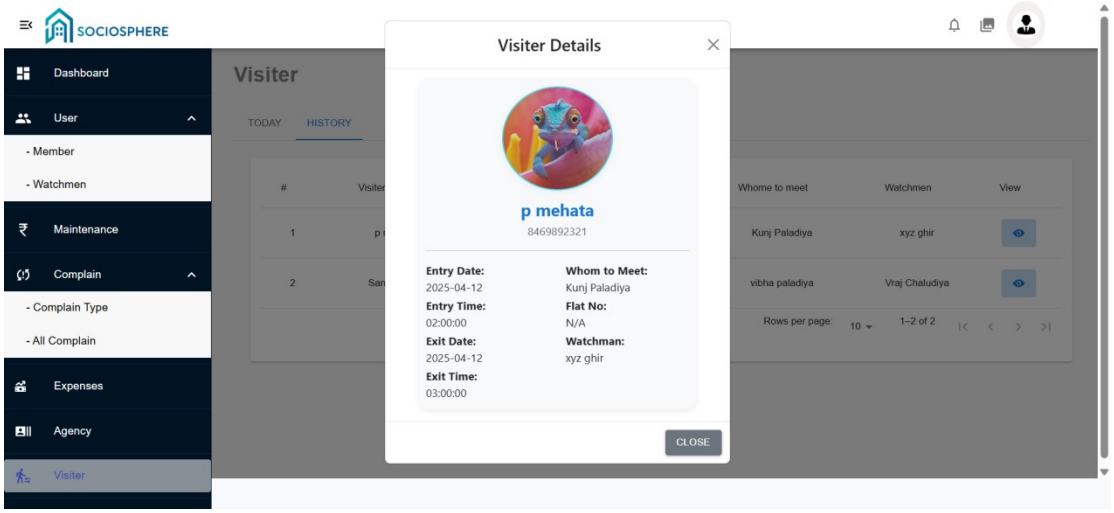

SOCIOSPHERE
User
- Member
- Watchmen
Maintenance
Complain
- Complain Type
- All Complain
Expenses
Agency
Visiter
Agency
+ADD AGENCY
TYPE
CONTACT RECORD
# Agency Name Status Edit
1 Medicine department ACTIVE
2 Testing ACTIVE
3 Medicine department ACTIVE
4 Grocerys ACTIVE
5 Security ACTIVE
Rows per page: 10 1–8 of 8

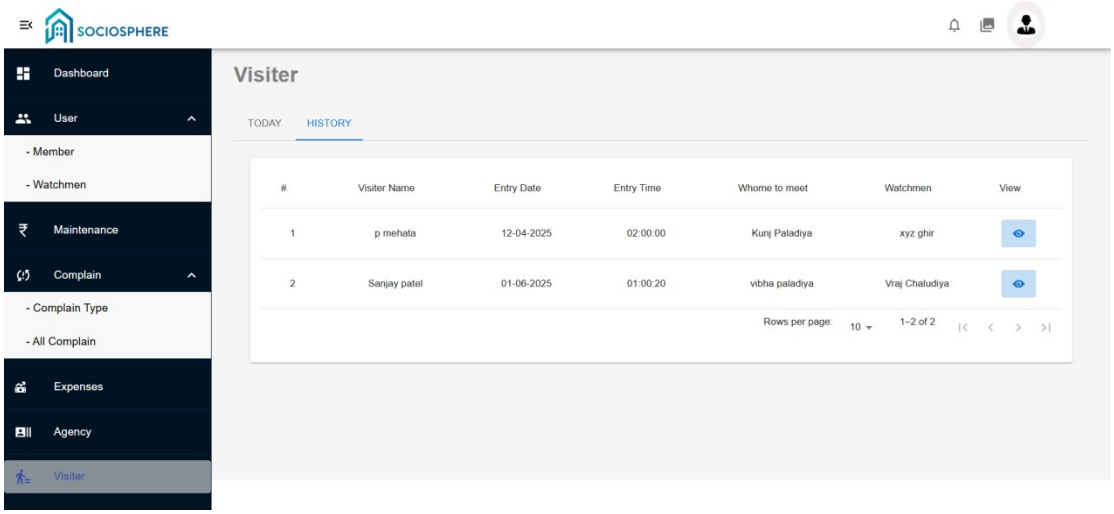

SOCIOSPHERE
Dashboard
User
- Member
- Watchmen
Maintenance
Complain
- Complain Type
- All Complain
Expenses
Agency
Visiter
Agency
+ADD AGENCY CONTACT
TYPE
CONTACT RECORD
ALL
MEDICINE DEPARTMENT
TESTING
MEDICINE DEPARTMENT
GROCERYS
# Agency Type Person name Phone no Edit Remove View
1 Medicine department mk.modi 6352637237
2 Testing mahes mehata 4456778909
3 Testing Nikunj mehata 9045995489
4 Medicine department Dr. Riddhi 8990349943
5 Medicine department raj sharh 8992382397
Rows per page: 1–5 of 5


SOCIOSPHERE
Agency Contact Details
mk.modi
Agency Type: Medicine department
Email: mkhospital@gmail.com
Location: surat
Phone No: 6352637237
Alternate Phone No: 6352778198
CLOSE

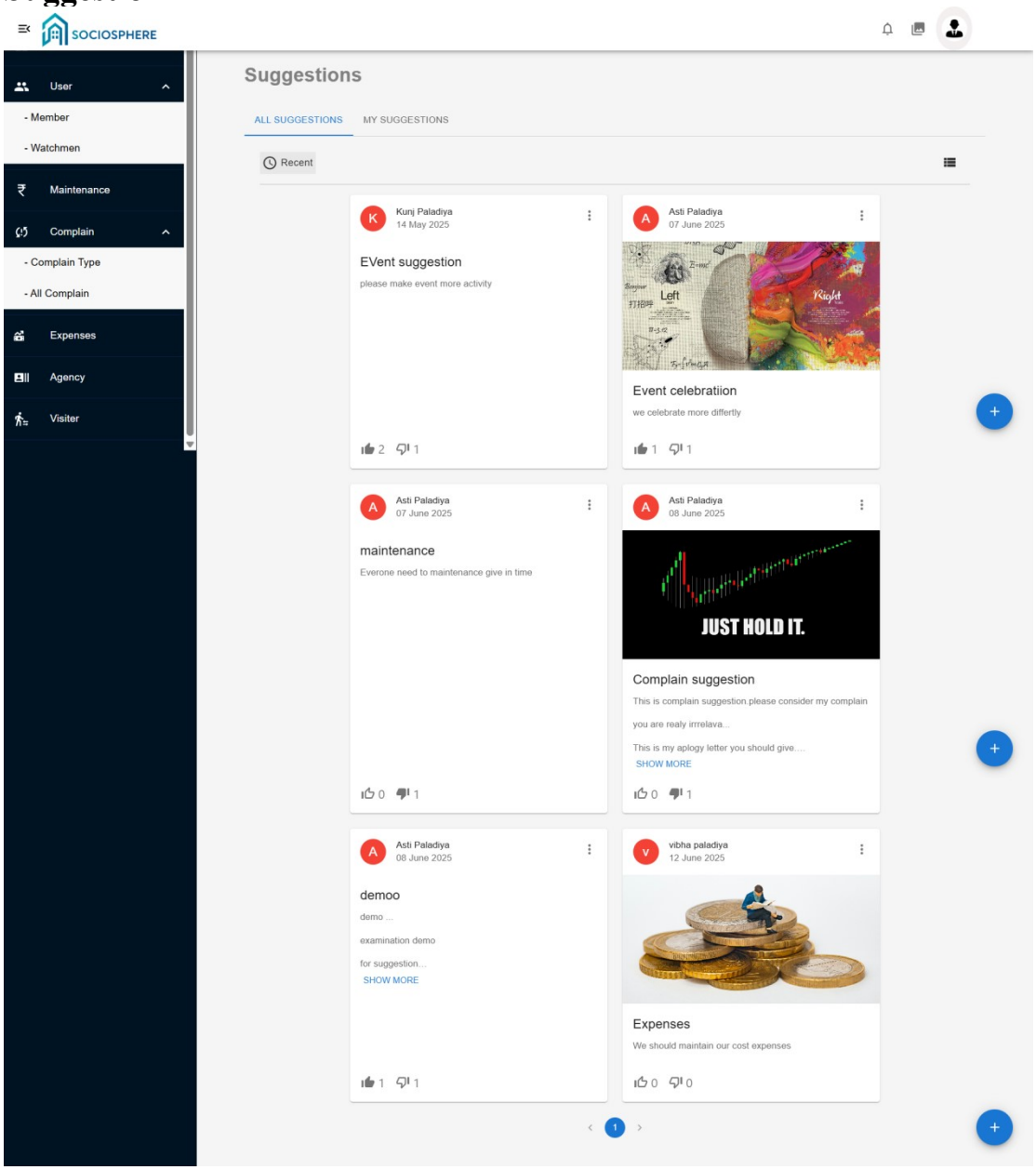
# Visiter

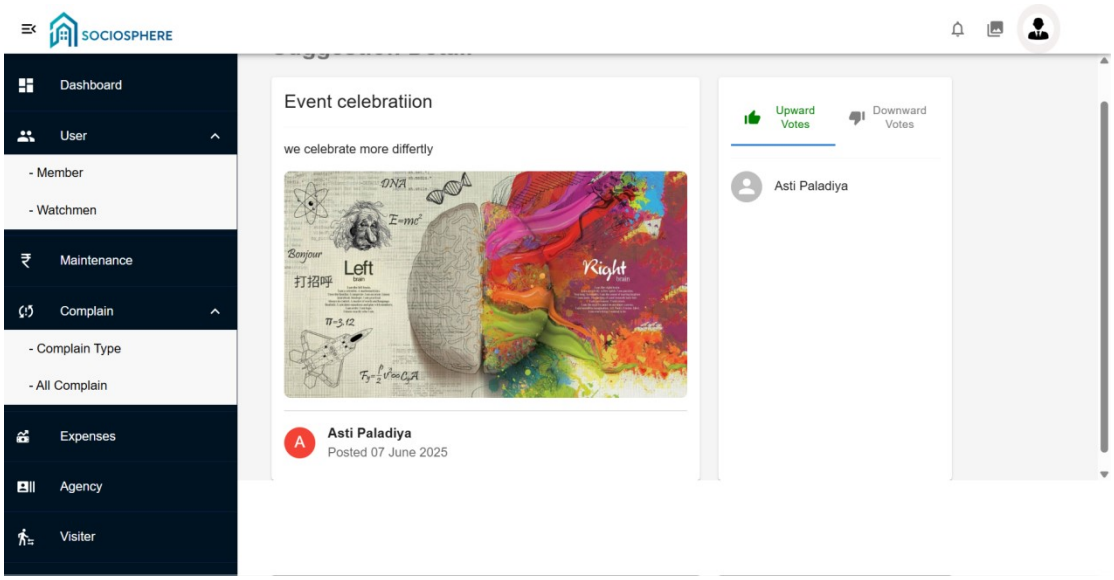

SOCIOSPHERE
Dashboard
User
- Member
- Watchmen
Maintenance
Complain
- Complain Type
- All Complain
Expenses
Agency
Visiter
Visiter
TODAY
HISTORY
# Visiter Name Entry Date Entry Time Whome to meet Watchmen View
No data available
Rows per page: 10 0–0 of 0


SOCIOSPHERE
Visiter
TODAY
HISTORY
Visiter Details
p mehata
8469892321
Entry Date: 2025-04-12
Entry Time: 02:00:00
Exit Date: 2025-04-12
Exit Time: 03:00:00
Whom to Meet: Kunj Paladiya
Flat No: N/A
Watchman: xyz ghir
CLOSE
Kunj Paladiya
xyz ghir
vibha paladiya
Vraj Chaludiya
Rows per page: 10 1–2 of 2


SOCIOSPHERE
Visiter
TODAY
HISTORY
# Visiter Name Entry Date Entry Time Whome to meet Watchmen View
1 p mehata 12-04-2025 02:00:00 Kunj Paladiya xyz ghir
2 Sanjay patel 01-06-2025 01:00:20 vibha paladiya Vraj Chaludiya
Rows per page: 10 1–2 of 2

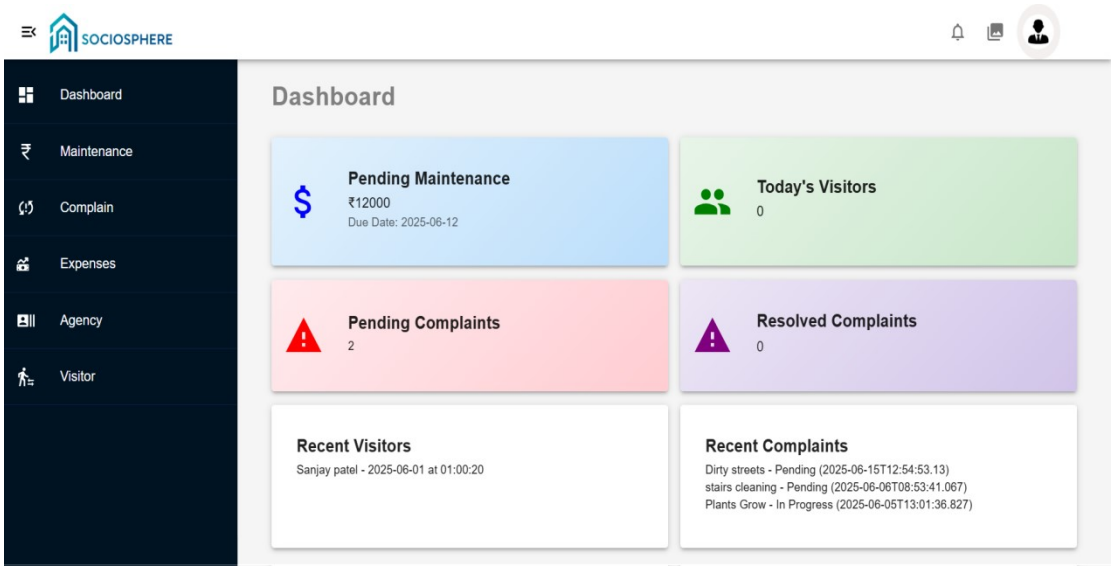
# Suggestion


SOCIOSPHERE
User
- Member
- Watchmen
Maintenance
Complain
- Complain Type
- All Complain
Expenses
Agency
Visiter
Suggestions
ALL SUGGESTIONS
MY SUGGESTIONS
Recent
Kunj Paladiya
14 May 2025
EVent suggestion
please make event more activity
2
1
Asti Paladiya
07 June 2025
Left
Right
Event celebratiion
we celebrate more differtly
1
1
Asti Paladiya
07 June 2025
maintenance
Everone need to maintenance give in time
0
1
Asti Paladiya
08 June 2025
JUST HOLD IT.
Complain suggestion
This is complain suggestion.please consider my complain
you are realy irrelava...
This is my aplogy letter you should give....
SHOW MORE
0
1
Asti Paladiya
08 June 2025
demoo
demo ...
examination demo
for suggestion...
SHOW MORE
1
1
vibha paladiya
12 June 2025
Expenses
We should maintain our cost expenses
0
0
1

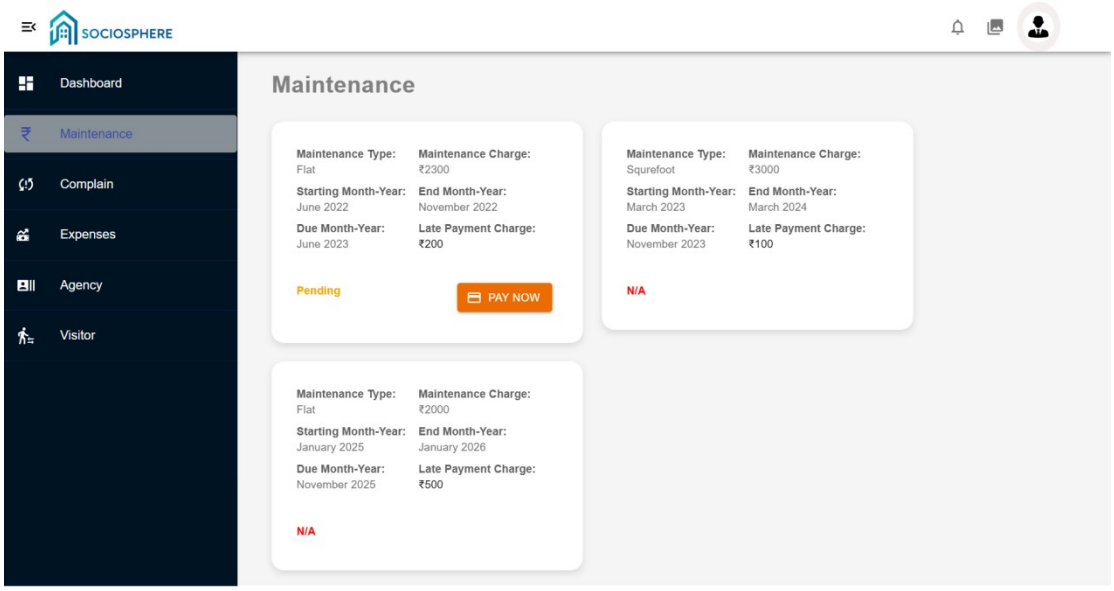



## 2. Member:-

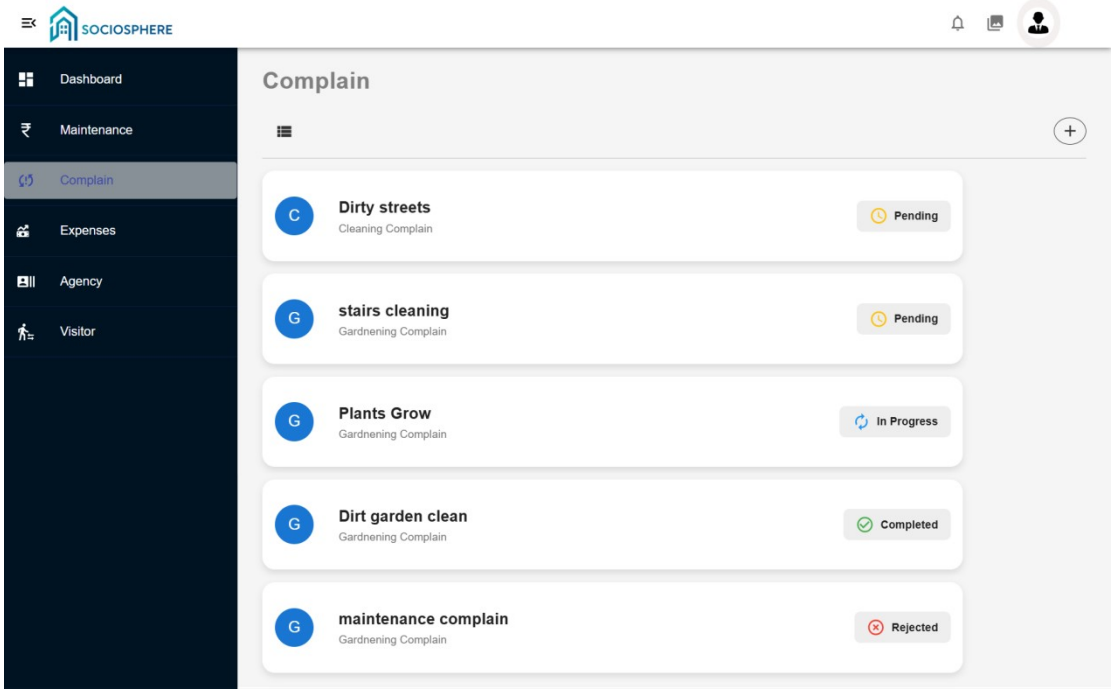
## Dashboard

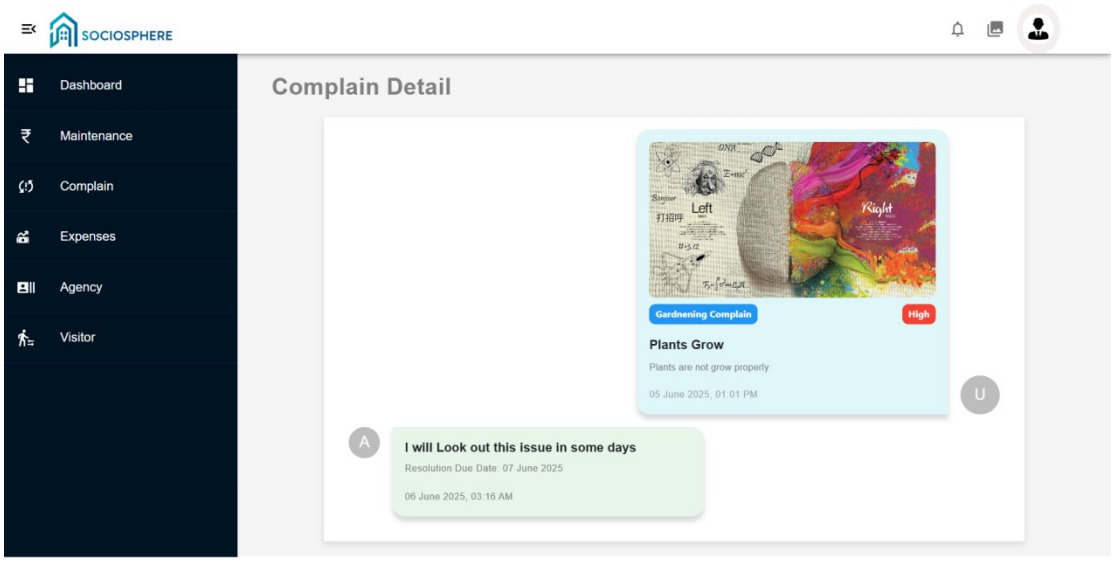


## Maintenance

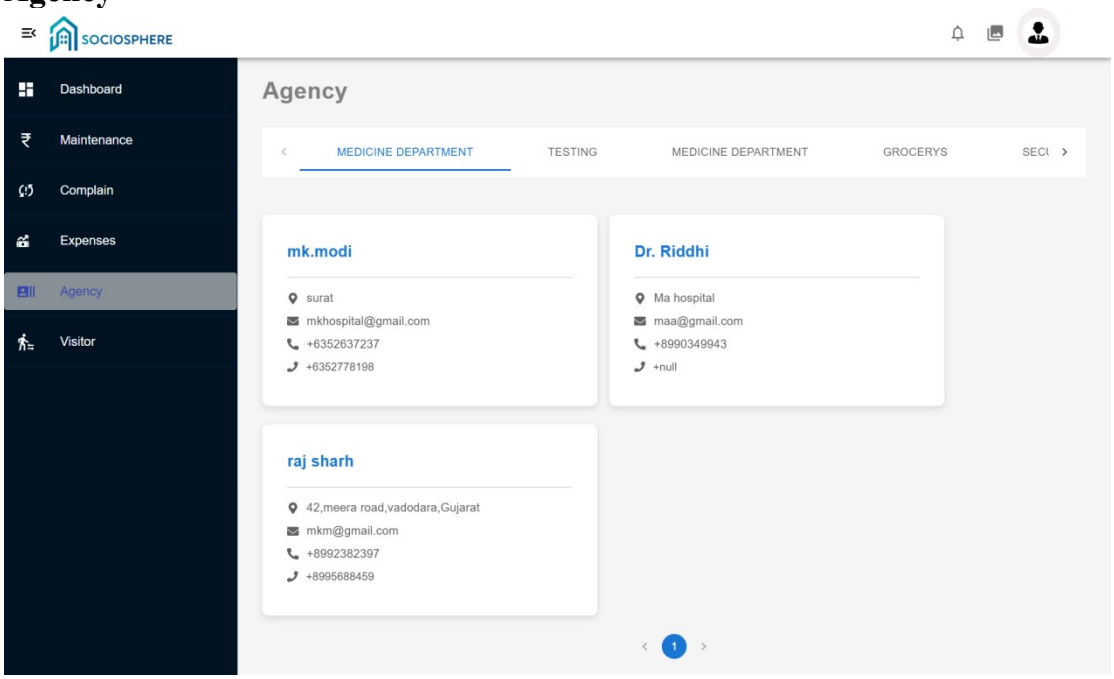



## Complain

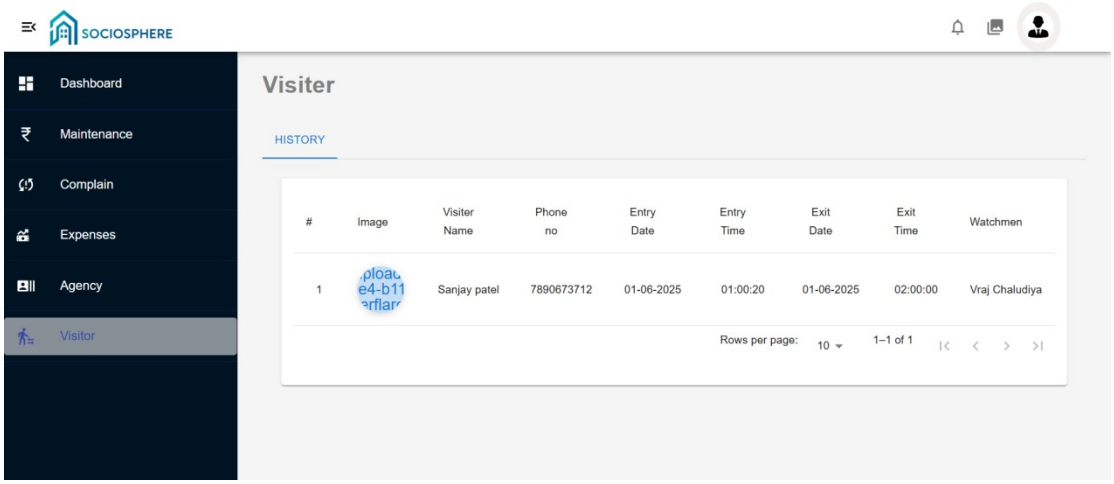



SOCIOSPHERE
Dashboard
Maintenance
Complain
Expenses
Agency
Visitor
Complain Detail
Gardnening Complain
High
Plants Grow
Plants are not grow properly
05 June 2025, 01:01 PM
U
A
I will Look out this issue in some days
Resolution Due Date: 07 June 2025
06 June 2025, 03:16 AM

## Agency


SOCIOSPHERE
Dashboard
Maintenance
Complain
Expenses
Agency
Visitor
Agency
MEDICINE DEPARTMENT
TESTING
MEDICINE DEPARTMENT
GROCERYS
mk.modi
surat
mkhospital@gmail.com
+6352637237
+6352778198
Dr. Riddhi
Ma hospital
maa@gmail.com
+8990349943
+null
raj sharh
42,meera road,vadodara,Gujarat
mkm@gmail.com
+8992382397
+8995688459
1

# Visitor


SOCIOSPHERE
Dashboard
Maintenance
Complain
Expenses
Agency
Visitor
Visiter
HISTORY
# Image Visiter Name Phone no Entry Date Entry Time Exit Date Exit Time Watchmen
1 Sanjay patel 7890673712 01-06-2025 01:00:20 01-06-2025 02:00:00 Vraj Chaludiya
Rows per page: 10
1–1 of 1

## 6.3 Architecture Design

The **SocioSphere** application is developed using a **Three-Tier Architecture** that separates the system into three distinct layers to ensure modularity, scalability, and maintainability.

**1. Presentation Layer (Frontend)**

- ✓ Developed using **ReactJS**, with styling from **React Bootstrap** and **Material UI**.
- ✓ Responsible for handling user interactions and displaying data.
- ✓ Uses **Axios** to communicate with the backend API.
- ✓ Supports responsive design for Admin and Member portals.
- ✓ Includes form validations, dynamic filters, pagination, and JWT-based authentication.

**2. Business Logic Layer (Backend API)**

- ✓ Built using **.NET Core Web API (C#)**.
- ✓ Handles all business operations such as login, registration, maintenance processing, event handling, and complaint responses.
- ✓ Ensures data security and access control using **JWT**.
- ✓ Sends email notifications using **SMTP** (e.g., OTPs, status updates, password recovery).
- ✓ Manages role-based logic (Admin, Member, Watchman).

**3. Data Access Layer (Database)**

- ✓ Uses **SQL Server** to store and manage all data including users, maintenance records, complaints, events, visitors, and suggestions.
- ✓ CRUD operations are performed through Entity Framework in the API.
- ✓ Ensures data consistency and integrity through validations and relationships.

This architecture ensures that the system is **easily maintainable**, **secure**, and **adaptable to future enhancements** such as analytics and multi-society support.

# 7. Testing

## 7.1Unit Testing

| Sr. No. | Module | Test Case Description | Expected Result | Status |
|---|---|---|---|---|
| 1 | Login (JWT Auth) | Check valid login credentials | Token generated and dashboard loads | Pass |
| 2 | Forget Password | OTP should be sent to registered email | OTP mail is received | Pass |
| 3 | Maintenance Payment | Member makes payment and receipt is generated | Status updated in DB and PDF receipt is downloaded | Pass |
| 4 | Add Complaint | Member submits new complaint | Complaint stored and visible in admin panel | Pass |
| 5 | Event Registration | Member registers and pays for event | Entry added in DB and confirmation mail sent | Pass |
| 6 | Visitor Entry (Mobile) | Watchman adds visitor entry | Visitor is added and linked to the respective member | Pass |
| 7 | Suggestion Submission | Member submits a new suggestion | Suggestion saved and visible to others | Pass |
| 8 | Suggestion Voting | Member likes/dislikes a suggestion | Vote count updates correctly | Pass |
| 9 | Add Expenses | Admin adds a new expense record | Expense saved and visible to members | Pass |
| 10 | Agency Contact Record | Admin adds or updates agency contact | Contact saved and displayed in contact list | Pass |

## 7.2 Intefration Testing

| Sr. No. | Integration Flow | Test Case Description | Expected Result | Status |
|---|---|---|---|---|
| 1 | Login (Frontend → API → DB) | Authenticate and load role-based dashboard | User redirected based on role with valid token | Pass |
| 2 | Complaint Workflow (Member → Admin → Member) | Member adds complaint → Admin updates status → Member sees response | Status updated and email sent to member | Pass |
| 3 | Maintenance Payment Flow | Member pays → API updates DB → Receipt PDF generated → Email sent | All steps successful, receipt downloaded | Pass |
| 4 | Event Registration Flow | Member registers → API saves data → Email confirmation sent | Confirmation email and event record inserted | Pass |
| 5 | Visitor Module (Mobile → API → DB) | Watchman adds visitor → DB stores data → Member can view | Visitor correctly mapped and visible in member dashboard | Pass |
| 6 | Suggestion Module | Member submits suggestion → All users can view and vote | Suggestion saved and votes reflect in real-time | Pass |
| 7 | Expenses Visibility | Admin adds expense → DB updates → Members can view it | Expense record reflects on member dashboard | Pass |
| 8 | Agency Contact Management | Admin adds contact → DB saves → Member views in contact list | Contact visible with correct details | Pass |

## 8. Future Enhancement

- ✓ **Role-Based Chat System**

  Secure internal messaging between Admin, Members, and Watchmen for smoother communication.

- ✓ **Advanced Reports & Analytics**

  Graphical dashboards for maintenance dues, complaint resolution stats, event participation trends, etc.

- ✓ **Mobile App Extension (Admin & Member)**

  Extend the existing Kotlin app to allow Admins and Members to access system functionalities on mobile.

- ✓ **Online Payment Integration**

  Integrate payment gateways (Razorpay, Stripe, etc.) for secure maintenance and event fee payments.

- ✓ **Biometric Authentication (App)**

  Add Face ID / Fingerprint login for better mobile app security.

- ✓ **Multi-Language Support**

  Allow users to use the application in their preferred local language.

- ✓ **AI Chatbot or Voice Assistant**

  Help users navigate and use the platform via chat or voice.

- ✓ **Digital Society Notice Board**

  Admins can post announcements, and members can receive real-time updates.

- ✓ **Polling & Voting System**

  Allow members to vote on society decisions like new rules, budgets, or events.

- ✓ **Society Meeting Management**
  - Schedule society meetings (general or emergency).
  - Share agendas before the meeting.
  - Record attendance and meeting minutes.
  - Upload decisions and downloadable MOM (Minutes of Meeting) post-meeting.

- ✓ **Society Committee Management**
  - Create multiple committees (e.g., Finance, Event, Security).
  - Assign roles (President, Treasurer, etc.) and manage their tenure.
  - Allow members to propose or vote for committee formation.
  - Maintain transparency of committee activities for all residents.

- ✓ **IoT Integration for Smart Societies**

  Integrate gate access control, water tank sensors, lift monitoring, etc., using IoT systems.

- ✓ **Cloud Sync and Backup**
  Automate backups to the cloud to prevent data loss and improve availability.

# 9.Glossary

| Term | Description |
|---|---|
| **Admin** | The system controller who manages all modules, users, data records, and system configuration. |
| **Member** | A resident of the society/flat who can raise complaints, pay maintenance, view events, and interact with other members. |
| **Watchman** | A security personnel who manages visitor details and performs limited functions via the mobile app. |
| **Maintenance** | Monthly/quarterly fees collected from members for society upkeep and services. |
| **Complaint** | An issue or concern raised by a member related to society infrastructure or services. |
| **Suggestion** | A feedback or proposal submitted by members for society improvement. |
| **Event** | Social, cultural, or community gatherings scheduled by the society. |
| **Visitor** | Any non-member individual entering the society premises. Logged by the watchman. |
| **Agency Contact** | Emergency or essential contacts such as Police, Ambulance, Fire Brigade, etc. |
| **Maintenance Receipt** | PDF or document proof generated after a member makes a successful maintenance payment. |
| **OTP** | One-Time Password used for secure authentication during password recovery. |
| **JWT (JSON Web Token)** | A secure token used for authenticating API requests and user sessions. |
| **Retrofit** | A REST API client library used in Android for making server requests. |
| **Axios** | A promise-based HTTP client used in React to call backend APIs. |

| Term | Description |
|---|---|
| **SMTP** | Simple Mail Transfer Protocol used for sending system-generated emails. |
| **Dashboard** | A visual summary of key statistics and actions for Admin and Members. |
| **Society Committee** | A designated group of residents responsible for decision-making and governance. |
| **Meeting Agenda** | A list of items or topics to be discussed during a scheduled society meeting. |
| **Event Gallery** | A media section where event photos or videos are uploaded and managed. |

# 10.Refrence


1. https://www.c-sharpcorner.com/
2. https://stackoverflow.com/
3. https://learn.microsoft.com/en-us/dotnet/csharp/
4. https://react.dev
5. https://mui.com/material-ui/getting-started/
6. https://react-bootstrap.netlify.app/
7. https://kotlinlang.org/docs/home.html
8. https://firebase.google.com/docs/cloud-messaging