Software Requirements Specification

for

Community Management System

Prepared by

Ahmed Mustafa, Haziq Naeem, Muneeb-ul-Islam

NUCES ISLAMABAD

20-3-2025

1. Table of Contents

Introduction
1.1 Purpose
1.2 Document Conventions
1.3 Intended Audience and Reading Suggestions
1.4 Product Scope
1.5 References

2. Overall Description
   2.1 Product Perspective
   2.2 Product Functions
   2.3 User Classes and Characteristics
   2.4 Operating Environment
   2.5 Design and Implementation Constraints
   2.6 User Documentation
   2.7 Assumptions and Dependencies

3. External Interface Requirements
   3.1 User Interfaces
   3.2 Hardware Interfaces
   3.3 Software Interfaces
   3.4 Communications Interfaces

4. System Features

5. Nonfunctional Requirements
   5.1 Product Requirements
   5.2 Organizational Requirements
   5.3 External Requirements

6. Use Case Diagram & User Stories
   6.1 Use Case Diagram
   6.2 User Stories with Pre/Post Conditions

7. Sequence Diagrams

8. Class Diagram

9. Product Backlog

10. Sprint Backlog

11. Meeting Minutes

12.

1. Introduction

1.1 Purpose

The **Communi3y** is Community Management System (CMS) that is a web-based application designed to streamline property management, public service requests, digital booking, facility reservations, and incident reporting within a community.

1.2 Document Conventions

This document follows the IEEE SRS template format.

1.3 Intended Audience and Reading Suggestions

- **Developers & Database Administrators** – Understand system functionalities.
- **Project Managers & Scrum Masters** – Track development milestones.
- **Residents & Admins** – Comprehend system capabilities.

1.4 Product Scope

The system provides **digital solutions** to manage properties, request public services, participate in digital voting, reserve community facilities, and report incidents. The backend will be implemented using **Next.js, Node.js, and MySQL**.

1.5 References

- IEEE 830-1998 Software Requirements Specification Standard.
- Project Diagrams: Use Case Diagram, Sequence Diagram, Class Diagram.

---

2. Overall Description

2.1 Product Perspective

This system is a standalone web-based application that digitalizes community management operations, replacing manual processes.

2.2 Product Functions

- **Property Registration**
- **Public Service Requests**
- **Digital Voting System**
- **Recreation Facility Reservation**
- **Crime & Incident Reporting**

2.3 User Classes and Characteristics

- **Residents** – Property owners or tenants accessing services.
- **Admins** – Management personnel handling operations.

2.4 Operating Environment

- Web-based system accessible via modern browsers.
- Hosted on a **local server infrastructure**.
- Uses **Html, CSS and React(if applicable at the end of sprint 3) (Frontend), Node.js (Backend), MySQL (Database)**.

2.5 Design and Implementation Constraints

- The system should be deployed on a **MySQL database**.
- Must follow **security protocols** for data encryption and access control.

2.6 User Documentation

- **User Manual** (for residents & admins)
- **API Documentation** (for developers)

2.7 Assumptions and Dependencies

- Reliable **internet connection** for cloud-based functionality.
- Compliance with **local property laws** for registration validation.

---

3. External Interface Requirements

3.1 User Interfaces

- **Web-based UI** developed using **HTML, CSS and REACT.js (applicable at the end of sprint 3)**.
- Interactive forms for **property registration, voting, and service requests**.

3.2 Hardware Interfaces

- Deployable on **standard server hardware**.
- Must support **cloud hosting** for scalability.

3.3 Software Interfaces

- Integration with **MySQL database**.

- **Node.js for backend operations**.

3.4 Communications Interfaces

- Email & SMS notifications for critical updates.
- Secure **HTTPS** for data transmissions.

---

4. System Features

- ### 4.1 User Stories & Acceptance Criteria

Here are the **User Stories with Acceptance Criteria** following the format in your attached image (`Story Card Format`):

- ### 1. Register Property

User Story:
As a resident, I want to register my property in the system so that I can be recognized as a property owner and access related services.

Acceptance Criteria:
And I know I am done when:

- I can input property details and upload necessary documents.
- The system validates and accepts my registration request.
- I receive a confirmation notification upon successful registration.

- ### 2. Manage Resident Data

User Story:
As an admin, I want to add, update, and manage resident information so that records remain accurate and up to date.

Acceptance Criteria:
And I know I am done when:

- I can view, edit, and delete resident profiles.
- Changes made reflect immediately in the database.
- A confirmation message is shown after each update.

- ### 3. Make Payment

User Story:
As a resident, I want to pay society fees, utility bills, and other charges so that I can fulfill my financial obligations.

Acceptance Criteria:
And I know I am done when:

- I can select a bill and choose a payment method.
- Payment is processed securely and a digital receipt is generated.
- The payment status updates in my account dashboard.

- ## 4. Track & Update Payments

User Story:
As an admin, I want to monitor and update residents' payment records so that I can ensure financial accuracy.

Acceptance Criteria:
And I know I am done when:

- I can view all payment histories per resident.
- I can manually edit and update payment status.
- All changes are logged with a timestamp.

- ## 5. Manage Bills

User Story:
As an admin, I want to generate, adjust, and send utility and maintenance bills so that residents receive accurate billing.

Acceptance Criteria:
And I know I am done when:

- I can create and adjust bill details (amount, due date, etc.).
- Bills are assigned correctly to each resident.
- A notification is sent to the resident's dashboard/email.

- ## 6. Book Parking & Reserve Facility

User Story:
As a resident, I want to book parking spots and recreation facilities so that I can use shared community resources.

Acceptance Criteria:
And I know I am done when:

- I can view available parking/facilities.
- I can book a slot for a specific time/date.
- I receive confirmation and the resource is marked as reserved.

- ## 7. Submit Maintenance Request

User Story:
As a resident, I want to request maintenance services so that property issues can be resolved.

Acceptance Criteria:
And I know I am done when:

- I can describe the issue and attach supporting media (photo/video).
- The request is logged and visible in my request history.
- I receive a ticket number and status tracking option.

- ## 8. Schedule Infrastructure Maintenance

User Story:
As an admin, I want to schedule and oversee maintenance for public infrastructure so that the community remains well-maintained.

Acceptance Criteria:
And I know I am done when:

- I can create a maintenance task with schedule and location.
- The task is assigned to a maintenance team.
- Progress and completion status can be tracked.

- ## 9. Manage Event RSVPs

User Story:
As a resident, I want to RSVP for events so that I can participate in community activities.

Acceptance Criteria:
And I know I am done when:

- I can view upcoming events and RSVP.
- My attendance status updates instantly.

- I receive an event reminder or pass.

- ## 10. Send & Receive Notifications

User Story:
As an admin, I want to send important notifications so that residents stay informed.

Acceptance Criteria:
And I know I am done when:

- I can draft and send messages to all or selected residents.
- Notifications appear in residents' dashboards or inboxes.
- Sent notifications are logged with timestamp and audience.

- ## 11. Request Public Service

User Story:
As a resident, I want to request services like garbage collection so that community needs are met.

Acceptance Criteria:
And I know I am done when:

- I can submit a service request with location and description.
- A request ticket is generated.
- I receive confirmation and can track status.

- ## 12. Report Crime or Incident

User Story:
As a resident, I want to report crimes or incidents so that authorities can respond accordingly.

Acceptance Criteria:
And I know I am done when:

- I can report an incident with details and location.
- The system assigns it to the concerned department.
- I receive a response acknowledgment.

- ## 13. Submit Public Feedback

User Story:
As a resident, I want to provide feedback on community services so that improvements can be made.

Acceptance Criteria:
And I know I am done when:

- I can write and submit feedback easily.
- My feedback is acknowledged.
- The admin panel shows cumulative feedback analytics.

- ## 14. Digital Voting

User Story:
As a resident, I want to participate in digital voting so that I can have a say in community decisions.

Acceptance Criteria:
And I know I am done when:

- I can view open polls and vote.
- My vote is submitted anonymously.
- The system shows vote completion confirmation.

- ## 15. Manage Institutes & Healthcare

User Story:
As an admin, I want to oversee educational and healthcare facilities so that community members receive quality services.

Acceptance Criteria:
And I know I am done when:

- I can add/edit institute profiles.
- I can assign and manage healthcare service schedules.
- Updates are reflected in the community portal.

Would you like these filled into a **template** like your uploaded image (in a Word or PDF document)?

- .

---

5. Nonfunctional Requirements

5.1 Product Requirements

- The system should support **1000+ concurrent users**.
- Should process **real-time transactions**.

5.2 Organizational Requirements

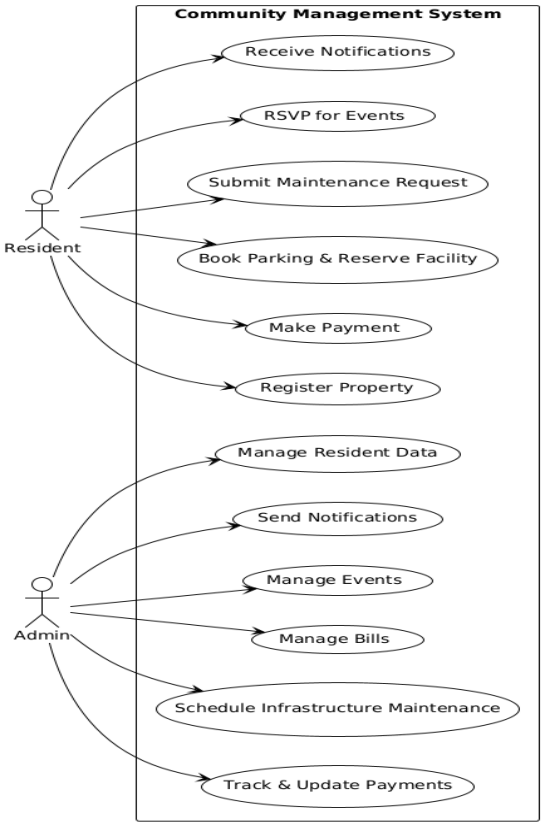
- Uses **Agile methodology** for iterative development.

5.3 External Requirements

- Should comply with **GDPR & local property laws**.

---

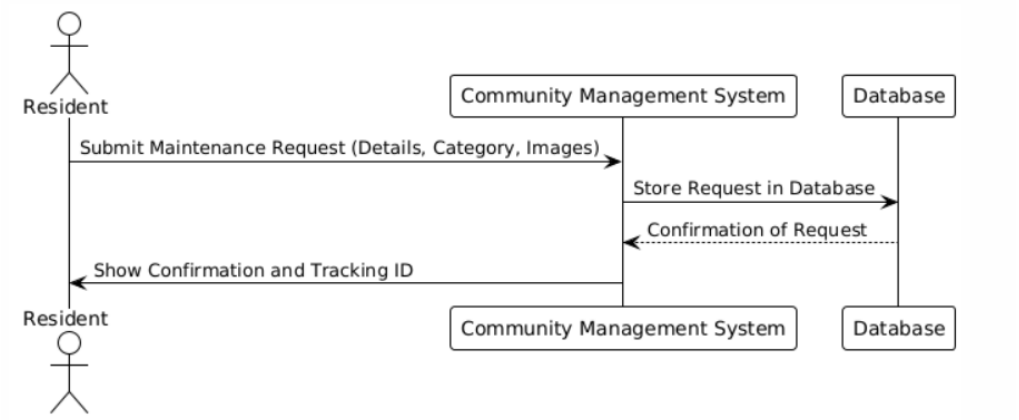
6. Use Case Diagram & User Stories

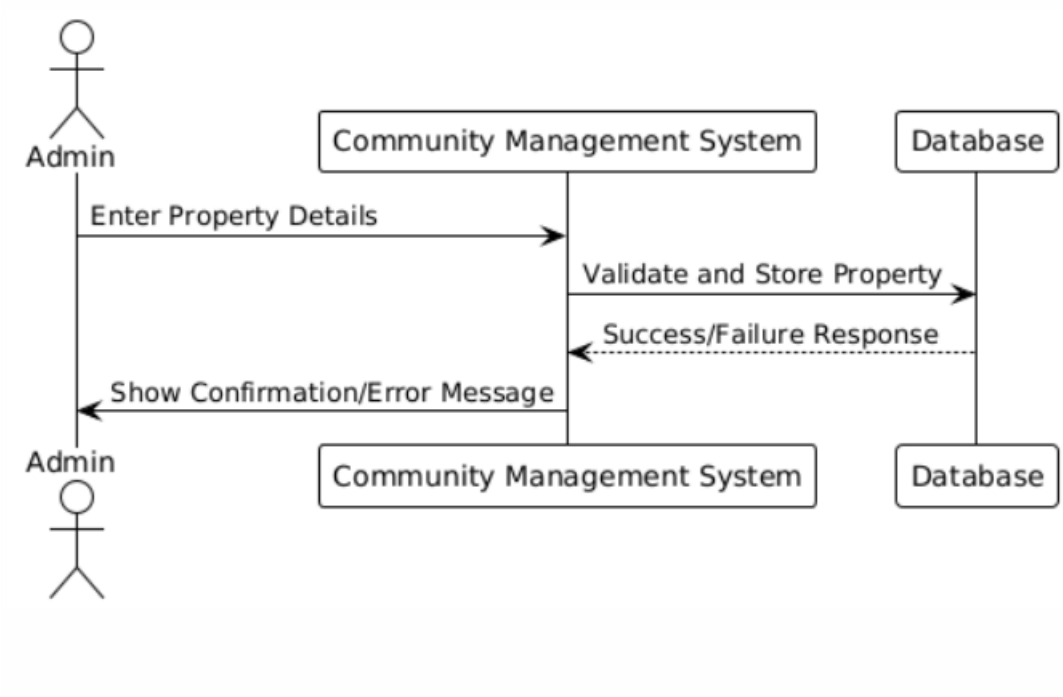


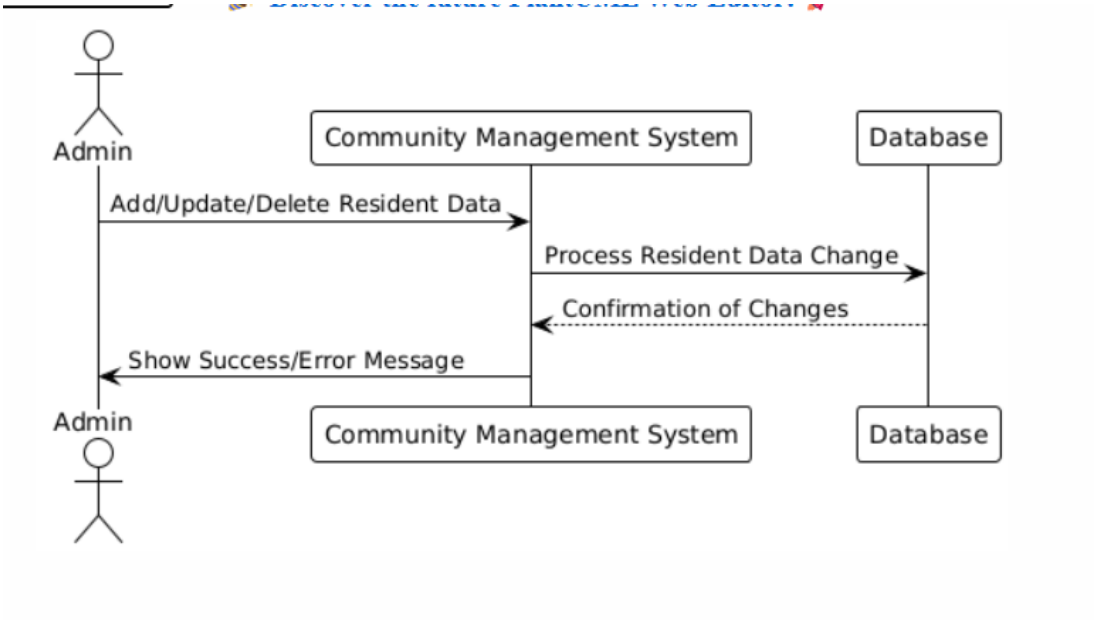
---

7. Sequence Diagrams

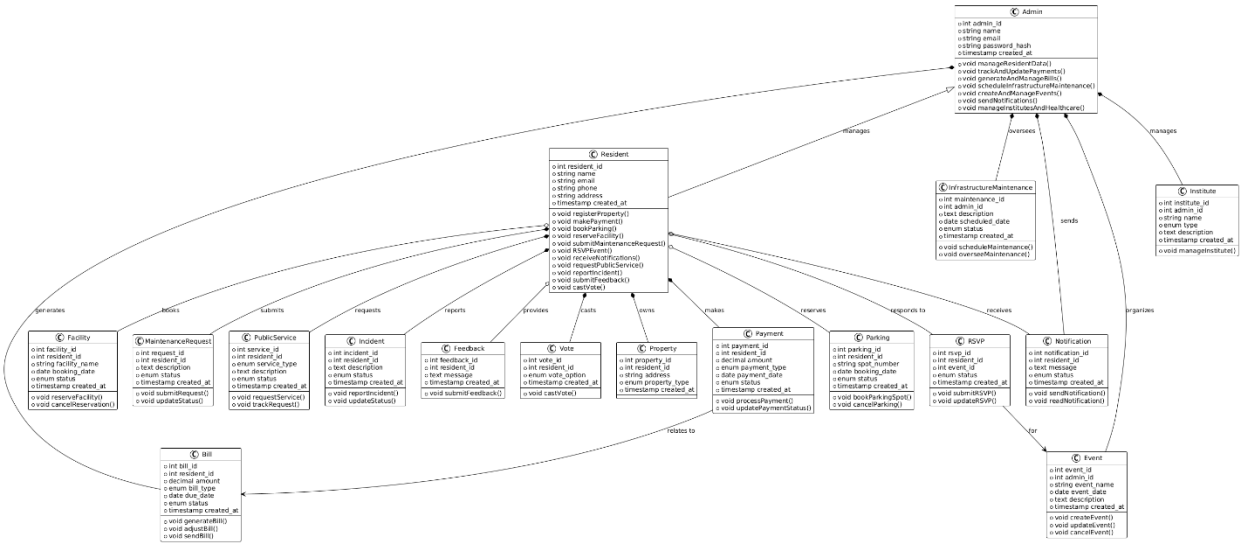
- **Three key activities visualized.**


Resident
Community Management System
Database
Submit Maintenance Request (Details, Category, Images)
Store Request in Database
Confirmation of Request
Show Confirmation and Tracking ID
Resident
Community Management System
Database

- 
Admin
Community Management System
Database
Enter Property Details
Validate and Store Property
Success/Failure Response
Show Confirmation/Error Message
Admin
Community Management System
Database



8. Class Diagram



9. Product Backlog

The product backlog includes all user stories prioritized based on their importance.

User Story	Priority
Register Property	**High**
Manage Resident Data	**High**
Make Payment	**High**
Track & Update Payments	**High**
Manage Bills	**High**
Book Parking & Reserve Facility	**Medium**
Submit Maintenance Request	**Medium**
Schedule Infrastructure Maintenance	**Medium**
RSVP & Manage Events	**Low**
Send & Receive Notifications	**Low**

10. Sprint Backlog

Sprint 1 Backlog

User Story	Assigned To	Tasks	Milestones
Register Property	Haziq	Database schema, API, UI	March 16: Backend, March 18: UI, March 20: Testing
Manage Resident Data	Muneeb	Schema, API, UI	March 15: API, March 17: UI, March 19: Testing
Make Payment	Ahmed	Payment API, UI, Security	March 16: Backend, March 18: UI March 20: Security
Track & Update Payments	Haziq	Database, Admin panel, Notifications	March 17: Backend March 19: Testing
Manage Bills	Muneeb	Billing system, API, UI	March 15: Backend March 18: UI March 20: Testing

Sprint 2 Backlog (Subset of User Stories)

User Story	Assigned To	Tasks	Milestones
Book Parking & Reserve Facility	Ahmed	Reservation System, UI, API	March 19: Backend March 200: UI March 21: Testing
Submit Maintenance Request	Haziq	Request submission, File uploads, Status tracking	March 21: Backend, March 21: UI, March 21: Testing
Schedule Infrastructure Maintenance	Muneeb	Admin scheduling UI, Notifications	March 22: Backend, March 22: UI March 22: Testing

11. Meeting Minutes

Project: Communi3y – Community Management System (CMS)
Meeting Title: Functional and Technical Requirements Review
Date: 15 MARCH 2025
Time: 8:30 pm
Location: Virtual
Chairperson: Muneeb ul Islam, Product Owner
Recorder: Ahmed Mustafa, Scrum Master
Attendees:

1) Ahmed (Developer, Scrum Master)
2) Muneeb (Developer, Product Owner)
3) Haziq (Developer, Scrum Team)

---

a. Agenda

1. Review of SRS Document
2. Clarification of Roles & Responsibilities
3. Discussion on Sprint 1 and Sprint 2 Backlog
4. Confirmation of Design Artifacts (Use Case, Sequence, Class Diagrams)
5. Roadmap Review and Action Items

---

b. Meeting Notes:

- Reviewed the document and project purpose: A web-based CMS designed to facilitate digital community management (property registration, service requests, incident reporting, etc.).

- Reiterated the use of IEEE SRS format and audience scope (Developers, Admins, Residents).
- Confirmed product scope: Includes digital voting, payment, reservations, and feedback systems.
- Technologies finalized:
    - **Frontend**: HTML, CSS, React.js (planned post-Sprint 3)
    - **Backend**: Node.js
    - **Database**: MySQL
- Two main user classes: **Residents** and **Admins**
- System will be hosted locally but must support cloud scalability.

C. Sprint Planning

- **Sprint 1 Stories Reviewed:**
    - *Register Property*, *Manage Resident Data*, *Make Payment*, *Track Payments*, *Manage Bills*
    - Tasks assigned with clear milestones. Developers confirmed task deadlines.
- **Sprint 2 Stories Reviewed:**
    - *Book Parking*, *Submit Maintenance Request*, *Schedule Infrastructure Maintenance*
    - Emphasis placed on testing and UI readiness by March 22.

d. Non-Functional & Compliance Requirements

- Agreed on:
    - System must support **1000+ concurrent users**
    - Real-time transaction capability

---

e. Decisions Made

- React.js integration will be postponed until end of Sprint 3, meanwhile use HTML and CSS for testing the frontend purposes.
- Documentation (User Manual & API Docs) to be created alongside system features.

- Working on Backend using Node.js and MySQL for database.
- Digital voting and feedback systems and other user stories marked for Sprint 3 due to lower priority.

---

4) Action Items

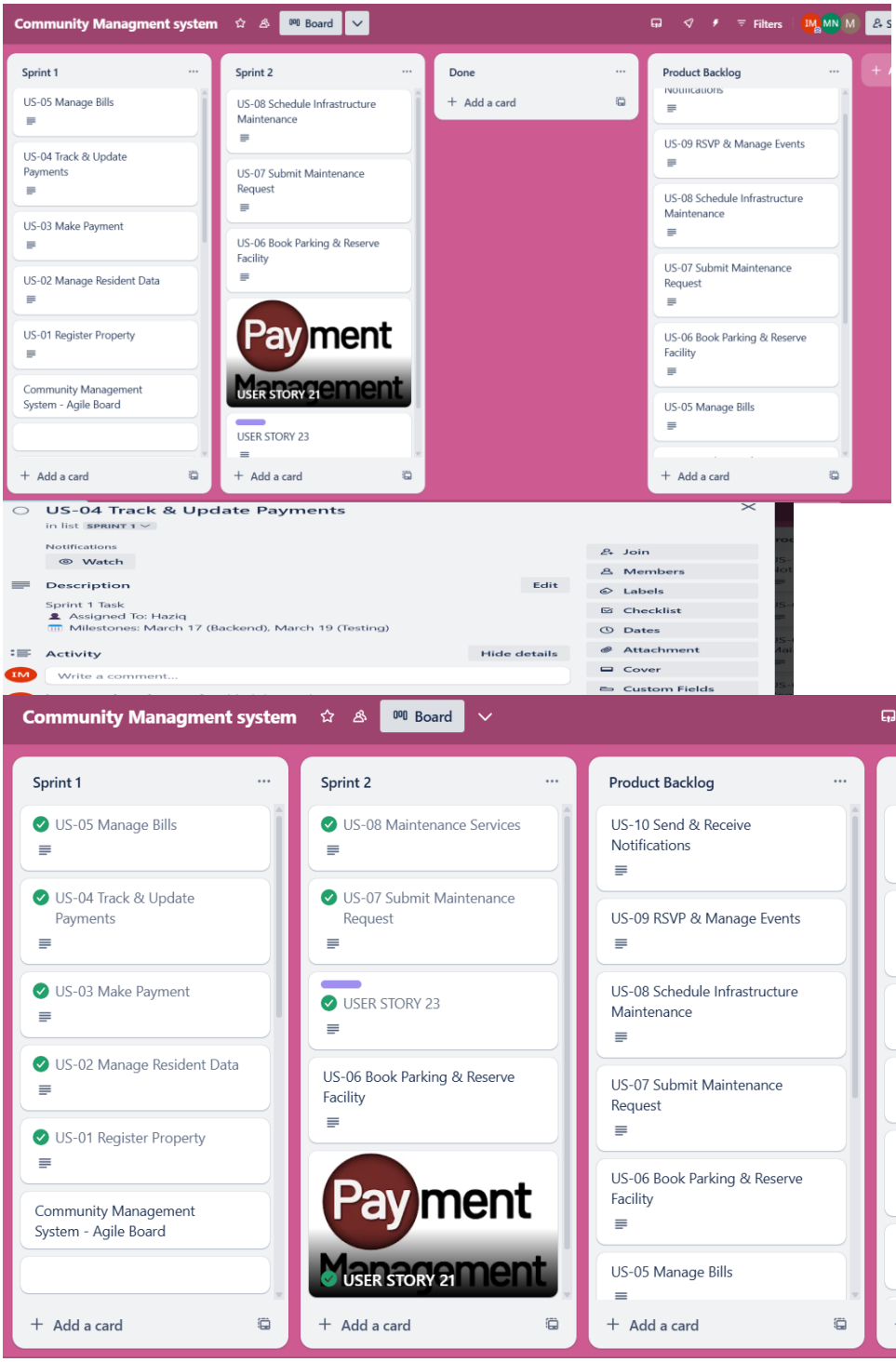
Action Item	Responsible	Deadline
Update Class and Sequence Diagrams	Haziq, Ahmed	March 15
Begin Sprint 2 Development	All Devs	March 19
Finalize UI/UX designs	Ahmed	March 22
Backend and DB setup	Muneeb. Haziq	March 16

Next Meeting: 15 April 2025
Objective: Review Sprint 3 progress and initial testing feedback.

---

12. Version Control & Contribution Evidence

- https://github.com/Mustafaahmed10/SE_Project
- https://trello.com/b/CmcUFss2/community-managment-system
- ScreenSHots:

- 
Community Managment system
Board
Filters
Sprint 1
US-05 Manage Bills
US-04 Track & Update Payments
US-03 Make Payment
US-02 Manage Resident Data
US-01 Register Property
Community Management System - Agile Board
Add a card
Sprint 2
US-08 Schedule Infrastructure Maintenance
US-07 Submit Maintenance Request
US-06 Book Parking & Reserve Facility
USER STORY 21
USER STORY 23
Done
Product Backlog
US-09 RSVP & Manage Events
US-10 Send & Receive Notifications
US-04 Track & Update Payments
in list SPRINT 1
Notifications
Watch
Description
Edit
Sprint 1 Task
Assigned To: Haziq
Milestones: March 17 (Backend), March 19 (Testing)
Activity
Hide details
Write a comment...
Join
Members
Labels
Checklist
Dates
Attachment
Cover
Custom Fields