PRISON MANAGEMENT SYSTEM

SOFTWARE DESIGN DOCUMENT

MEMBERS

Nihar Niranjan S

Nikith T Rajan

Niranjay Ajayan

Varun Raj R

Contents

1. Introduction

1.1 Overview

The Prison Management System (PMS) provides a complete solution to modernize the administration of prisons in a state of the art manner. The traditional manual methods of record keeping and management used in correctional facilities can often cause errors, inefficiency, and security risks. PMS does away with the need for all these processes by moving them to a secured, centralized, and user-friendly platform and presenting significant improvements in accuracy, efficiency in day-to-day operations, and safety for the inmates and the prison staff.

The management of inmate records, staff assignments, and the tracking of visitors is a subject where this system provides functionalities that cater to the above basic necessities. Therefore, real-time activities of prisoners can be better monitored, and corrective measures can be instigated by prison management. Several hurdles in paperwork can be eliminated and automated processes made available, thus increasing the accuracy of data, safety measures, and productivity of the facility

1.2 Purpose

The core objective of the existing Prison Management System (PMS) is to enhance and integrate prison functions by automating record-keeping, which is hitherto done manually. The system is able to control the flow of information about the inmates, the officers, the cells, and what the prisoners do effectively. Moreover, PMS keeps account of the visitors, the crime or the statistics, and the incidents, thus enhancing the data collection process and the operation of decision-making processes. It improves routine processes and provides analytical reports, which decreases the administrative work for the staff, fosters better operation of activities on a daily basis and

creates a safe environment both for prisoners and prison workers. The final goal of the PMC is to make the working of a corrections center more efficient, secure and economical.

1.3 Scope

The scope of the Prison Management System (PMS) extends the entire digital prison operations, which include inmate record management, staff administration, and daily task allocation. Those functionalities are also included here that prisoner intake, release, and transfers, cell and task assignments covering. The system manages visitor tracking, crime reporting, and incident logging as well. Securely, Staff can access and update data and the senior personnel at staff registration and role updates can lead. The PMS intends to eliminate the redundancies in the process and reduce manual paperwork, thus leading to a safe and secure environment, greater efficiency as well as transparency.

The PMC protection put forth is also for the inmates and guards as well. The common purpose in the PMC is to make the functioning of a corrections center more efficient, safe, and inexpensive.

1.4 Resources

The project integrates React.js for the front-end, providing an intuitive and dynamic user experience, while MySQL handles the database to store and manage data. Flask (Python) is used as the backend framework to connect the front-end with the database, ensuring efficient communication between the user interface and data layers. This combination enables a robust, scalable, and secure system for managing prison operations.

1.5 Definitions and Acronyms

1.5.1 Definitions:

- Prison Management System (PMS): A digital platform designed to streamline and manage prison operations, including inmate records, staff management, and daily activities.

- Inmate: A person held in custody within a prison.

- Staff Management: The process of handling staff information, including roles, registration, and updates within the prison.

- Cell Allocation: Assigning prisoners to available cells based on criteria like cell capacity and prisoner status.

- Task Assignment: Designating jobs or tasks to inmates as part of rehabilitation or prison routines.

- Visitor Management: The process of registering and tracking visitors, along with the details of their visits.

- Report Generation: The creation of detailed reports on prisoners, staff, crime statistics, and activities for operational and decision-making purposes.

1.5.2 Acronyms:

- PMS: Prison Management System

- UI: User Interface

- CRUD: Create, Read, Update, Delete (common database operations)

- SQL: Structured Query Language (used for managing databases like MySQL)

- API: Application Programming Interface (used for communication between different software components)

- DB: Database

- UX: User Experience

- MVC: Model-View-Controller (a software design pattern used in Flask applications)

2. System Overview

PMS is an advanced, user-friendly system aimed at automating and easing the day-to-day routine of correctional facility management. From a paper-based system, the PMS puts in place a digitized system, which guarantees a fast, accurate, accessible, and secured information system, which encompasses the whole staff management mechanism and control from the management of inmate records to cell allocation monitoring, through staff tracking and visitor attendance logging.

With this, staff management will allow authorized staff to log in securely, add staff, update roles, and remove employees. Hence, all Employee information will always be up-to-date and easily manageable.

For prisoner management, all prisoner records can be added or retrieved easily with PMS. The generated reports may detail for the activities of prisoners. It provides cell assignments and work task checks so that the administrators can monitor prisoners more closely and track progress.

With the Visitor Control module, all details of visitors are recorded in PMS, and visit history is maintained. Facilities for fast search for records by inmate or visitor are provided. In addition, crime and incident tracking can assist in analyzing crime data and even in developing statistics with the aim of improving security and decision-making.

PMS, as a whole, is a smooth, secure, efficient operation for all concerned.

3. System Architecture

3.1 Architectural Design

The prison management system is a modular, multi-functional system divided into subsystems that manage various aspects of the prison, such as staff, prisoners, cells, tasks, visitors, and crime tracking. Each subsystem handles a specific set of responsibilities, allowing for seamless coordination between them to achieve the overall functionality.

Subsystem overview is as follows:

3.1.1 Staff Management Subsystem:

This subsystem is responsible for managing all staff-related operations, including logging in, viewing, registering, updating, and removing staff members. It ensures that staff information is properly maintained and updated. It works closely with the Task Management Subsystem to assign tasks to staff and manage their duties within the prison system.

3.1.2 Prisoner Management Subsystem:

This subsystem handles the management of prisoners, from adding new inmates to releasing them, updating their information, and generating reports. It collaborates with the Cell Management Subsystem to allocate cells based on prisoner details and with the Task Management Subsystem to assign work to prisoners. These collaborations ensure smooth operations related to housing and prisoner labor assignments.

3.1.3 Cell Management Subsystem:

Responsible for managing the allocation and reassignment of cells to prisoners, this subsystem tracks the availability of cells and ensures that prisoners are appropriately housed. It works in collaboration with the Prisoner Management Subsystem for cell assignments and with the Task Management Subsystem if tasks involve relocation or prisoner movement between cells.

3.1.4 Task Management Subsystem:

This subsystem oversees task assignments for both prisoners and staff, tracks work hours, and records the completion of jobs. It coordinates with the Prisoner Management Subsystem to ensure that prisoners are assigned tasks based on their status and with the Staff Management Subsystem to delegate work to staff. Additionally, it interacts with the Cell Management Subsystem if tasks require movement between cells.

3.1.5 Visitor Management Subsystem:

The Visitor Management Subsystem manages the registration of visitors and keeps a record of visitor history, including details such as who they are visiting and the time of the visit. It works with the Security and Authentication Subsystem to ensure that visitor access is authorized and securely tracked within the prison system.

3.1.6 Crime and Incident Tracking Subsystem:

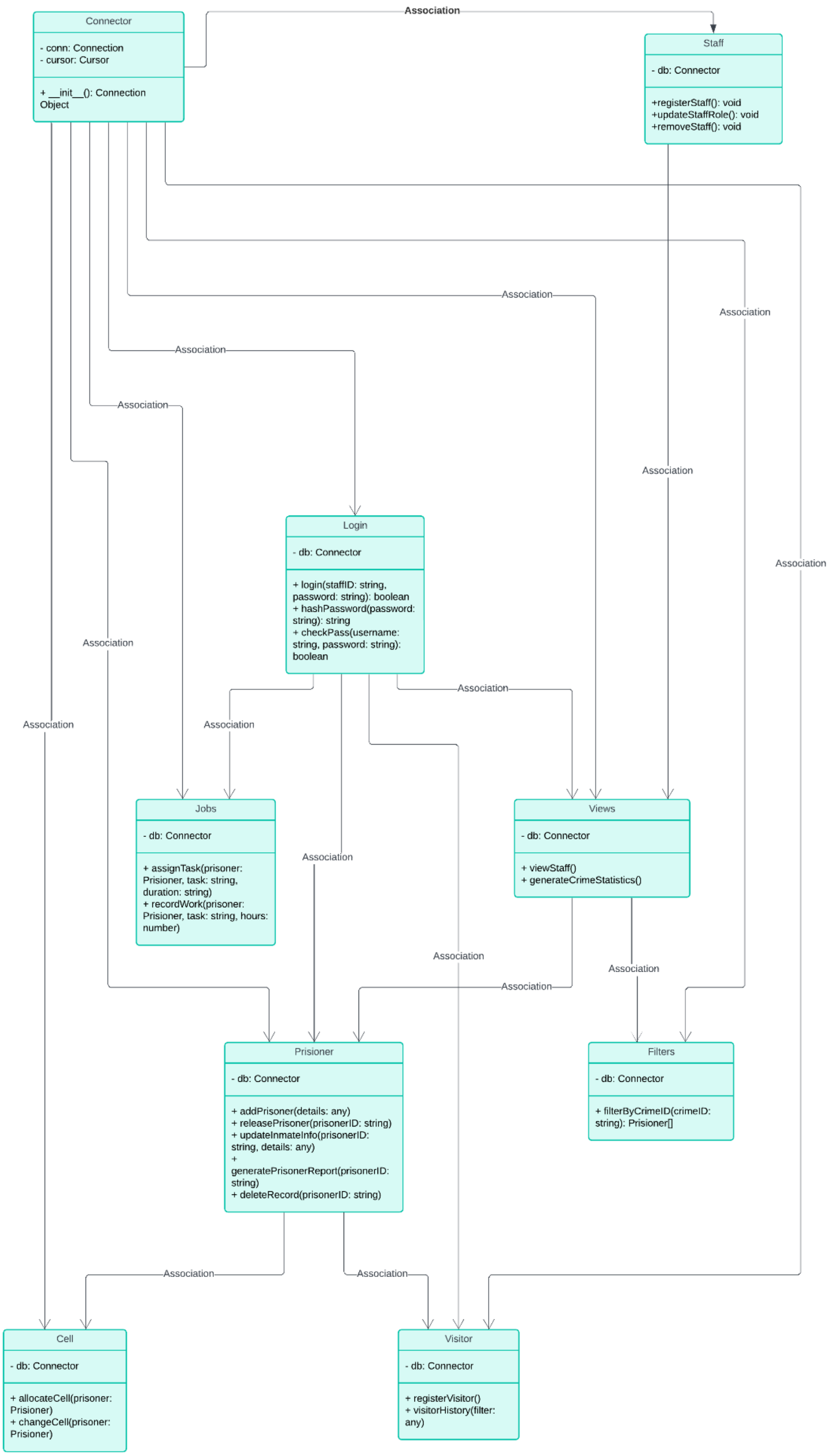
This subsystem tracks crimes and incidents associated with prisoners, provides filters by crime ID, and generates statistical reports. It interacts with the Prisoner Management Subsystem to keep prisoner profiles up-to-date with crime details and offers comprehensive reports for better insight into prison crime data.

3.1.7 Login Subsystem:

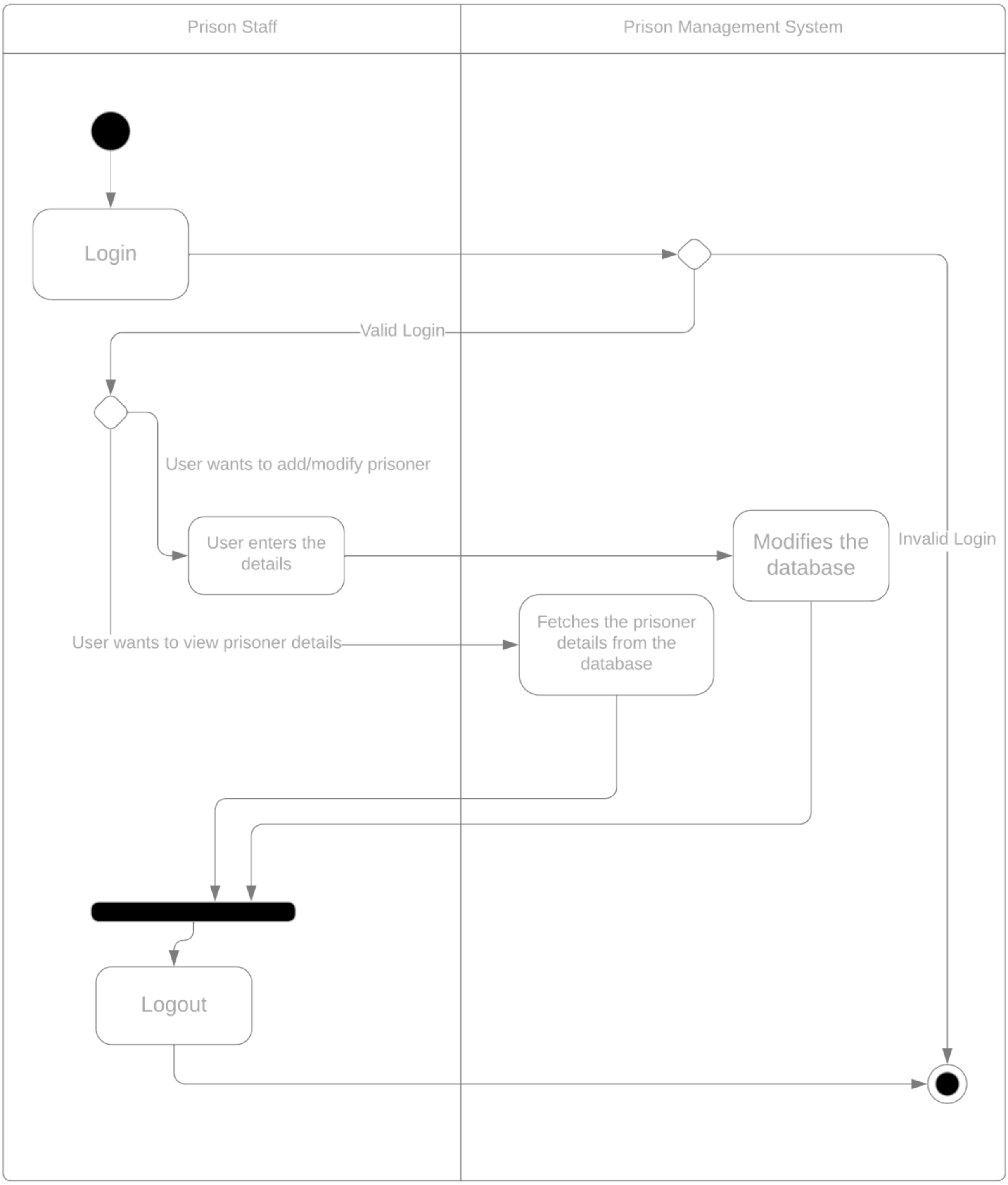
This subsystem ensures secure access to the system by managing logins for staff, enforcing password policies, and providing role-based access control. It collaborates with all other subsystems to grant access and manage permissions, ensuring that only authorized staff can interact with specific subsystems based on their role.

3.2 Decomposition Description

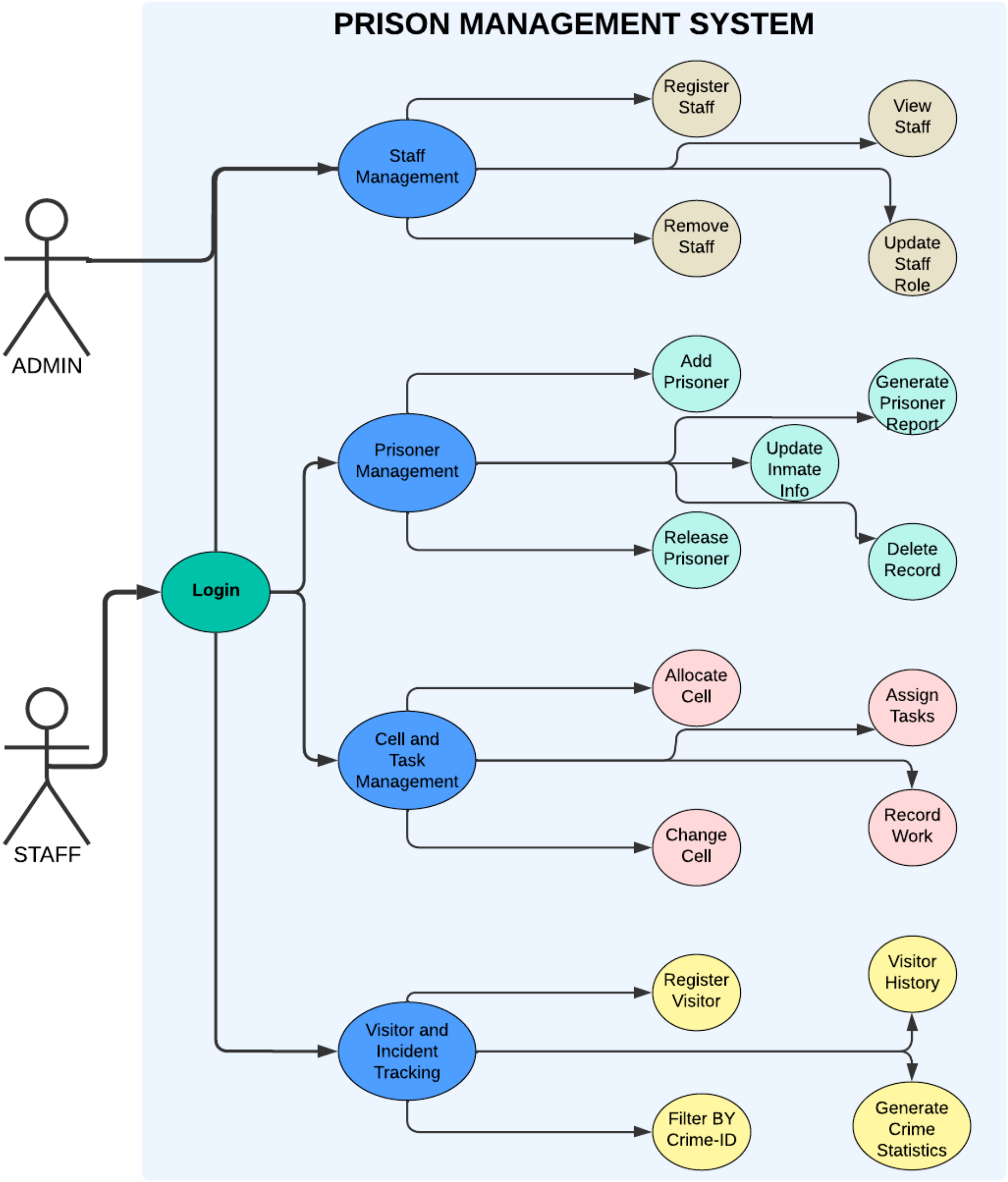
3.2.1 Class Diagram


Connector
- conn: Connection
- cursor: Cursor
+ __init__(): Connection Object
Association
Staff
- db: Connector
+registerStaff(): void
+updateStaffRole(): void
+removeStaff(): void
Association
Association
Association
Association
Association
Association
Association
Association
Login
- db: Connector
+ login(staffID: string, password: string): boolean
+ hashPassword(password: string): string
+ checkPass(username: string, password: string): boolean
Association
Association
Association
Jobs
- db: Connector
+ assignTask(prisoner: Prisioner, task: string, duration: string)
+ recordWork(prisoner: Prisioner, task: string, hours: number)
Association
Views
- db: Connector
+ viewStaff()
+ generateCrimeStatistics()
Association
Association
Association
Prisioner
- db: Connector
+ addPrisoner(details: any)
+ releasePrisoner(prisonerID: string)
+ updateInmateInfo(prisonerID: string, details: any)
+ generatePrisonerReport(prisonerID: string)
+ deleteRecord(prisonerID: string)
Filters
- db: Connector
+ filterByCrimeID(crimeID: string): Prisioner[]
Association
Association
Cell
- db: Connector
+ allocateCell(prisoner: Prisioner)
+ changeCell(prisoner: Prisioner)
Visitor
- db: Connector
+ registerVisitor()
+ visitorHistory(filter: any)

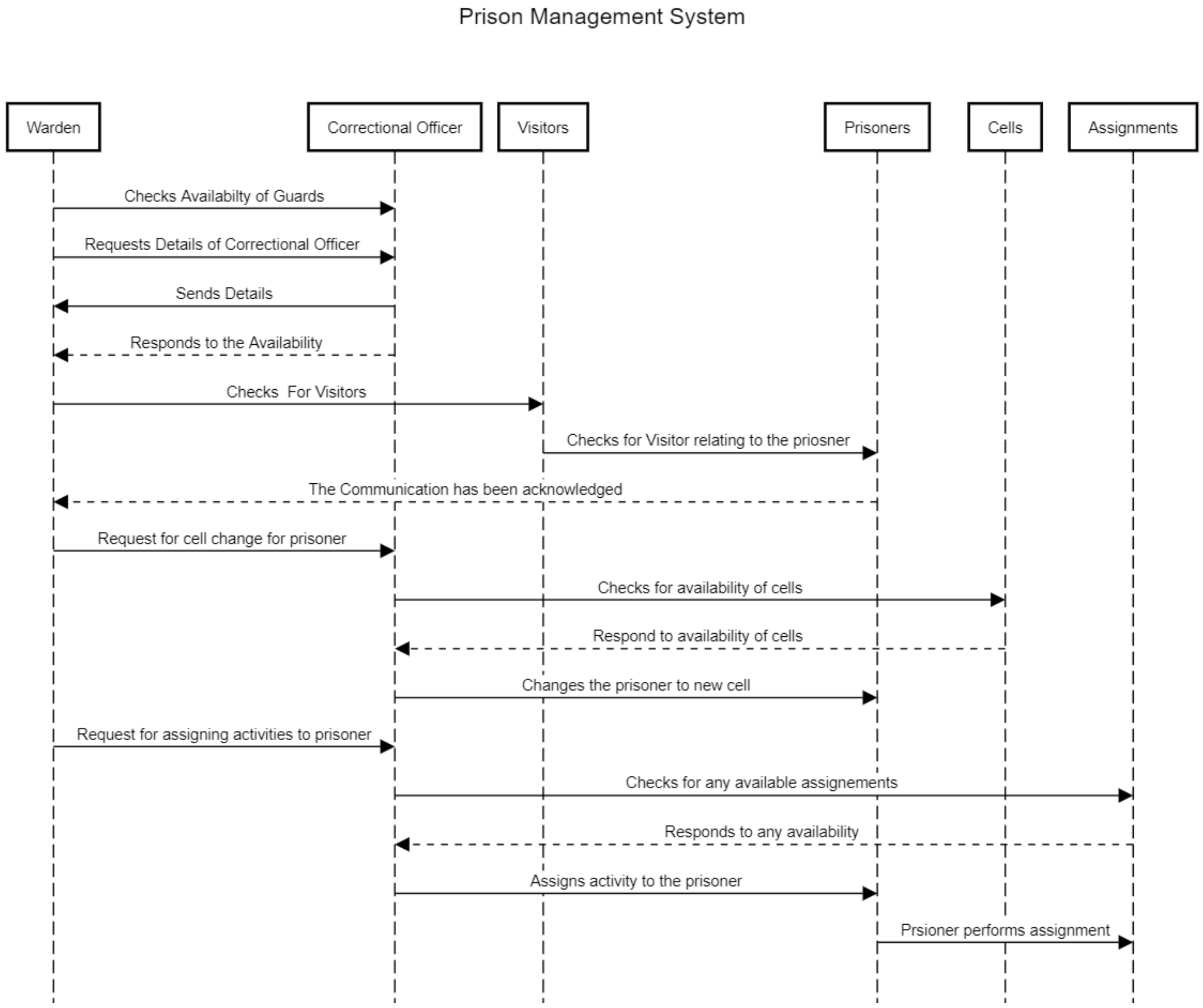
3.2.2 Activity Diagram



3.2.3 Use Case Diagram



3.2.4 Sequence Diagram


Prison Management System
Warden
Correctional Officer
Visitors
Prisoners
Cells
Assignments
Checks Availabilty of Guards
Requests Details of Correctional Officer
Sends Details
Responds to the Availability
Checks For Visitors
Checks for Visitor relating to the priosner
The Communication has been acknowledged
Request for cell change for prisoner
Checks for availability of cells
Respond to availability of cells
Changes the prisoner to new cell
Request for assigning activities to prisoner
Checks for any available assignements
Responds to any availability
Assigns activity to the prisoner
Prsioner performs assignment

3.3 Design Rationale

The prison management system is designed with modularity, security, and scalability at its core. Each subsystem, such as Staff Management, Prisoner Management, and Crime Tracking, operates independently, ensuring flexibility and easier maintenance. The object-oriented approach maps real-world entities like prisoners, staff, and cells to corresponding classes, promoting clarity and abstraction. Security is prioritized with a Login class for officers and administrators, while dedicated classes like Filters and Connector maintain data integrity. The user-centered Views class offers a straightforward interface for efficient system interaction. Subsystems communicate seamlessly to keep data synchronized, supporting real-time operations. Additionally, the system's scalable architecture allows for future enhancements, like AI integration, without disrupting current functionality.

4. Data Design

4.1 Data Description

The information is stored in the form of a relational database using MySQL. The database contains tables for staff, cells, crime, prisoner, prisoner details, login details and visitor details. The data is transferred between the frontend and the backend in the form of JSON objects.

4.2 Data Dictionary

The following data dictionary defines the tables and fields for the Prison Management System database schema. It describes each field, its data type, and any constraints or relationships with other tables.

4.2.1 CELLS table

Field Name	Data Type	Description	Constraints
CELL_NUMBER	INT	Unique identifier for each cell	Primary Key
VACANT	BOOLEAN	Indicates whether the cell is vacant or occupied	NOT NULL
PRISONER_ID	INT	ID of the prisoner occupying the cell (if any)	Foreign Key, References `PRISONER_ID` in `PRISONER` table, NULL allowed

4.2.2 CRIME table

Field Name	Data Type	Description	Constraints
CRIMEID	INT	Unique identifier for each crime	Primary Key, Auto Increment
DESCRIPTION	TEXT	Detailed description of the crime	NOT NULL

4.2.3 JOBS Table

Field Name	Data Type	Description	Constraints
JOBID	INT	Unique identifier for each job assignment	Primary Key, Auto Increment
JOB_DESC	VARCHAR(255)	Brief description of the job	NOT NULL
WORK_START	TIME	Start time of the prisoner's work shift	NOT NULL
WORK_END	TIME	End time of the prisoner's work shift	NOT NULL

4.2.4 LOGIN_DETAILS Table

Field Name	Data Type	Description	Constraints
STAFF_ID	INT	Unique identifier for each staff member	Primary Key, Foreign Key, References `STAFF_ID` in `STAFF` table
PASSWORD	VARCHAR(100)	Encrypted password for staff login	NOT NULL

4.2.5 PRISONER Table

Field Name	Data Type	Description	Constraints
PRISONER_ID	INT	Unique identifier for each prisoner	Primary Key, Auto Increment
AADHAR_NUMBER	VARCHAR(12)	Aadhar number of the prisoner	NOT NULL, Foreign Key, References `AADHAR_NUMBER` in `PRISONER_DETAILS` table
CRIME_ID	INT	ID of the crime the	Foreign Key, References `CRIMEID` in `CRIME` table

Field Name	Data Type	Description	Constraints
		prisoner is convicted for	
ENTER_DATE	DATE	Date when the prisoner entered the prison	NOT NULL
RELEASE_DATE	DATE	Expected release date of the prisoner	NULL allowed

4.2.6 PRISONER_DETAILS Table

Field Name	Data Type	Description	Constraints
AADHAR_NUMBER	VARCHAR(12)	Unique Aadhar number of the prisoner	Primary Key
NAME	VARCHAR(100)	Full name of the prisoner	NOT NULL
AGE	INT	Age of the prisoner	NOT NULL
NUMBER_OF_CONVICTIONS	INT	Number of convictions the prisoner has	Default: 0, NOT NULL

4.2.7 STAFF Table

Field Name	Data Type	Description	Constraints
STAFF_ID	INT	Unique identifier for each staff member	Primary Key, Auto Increment
NAME	VARCHAR(100)	Full name of the staff member	NOT NULL
AGE	INT	Age of the staff member	NOT NULL

Field Name	Data Type	Description	Constraints
PHONE_NUMBER	VARCHAR(15)	Staff member's contact phone number	NOT NULL, Unique
ROLE	VARCHAR(255)	Role of the staff member in the prison	NOT NULL

4.2.8 VISITOR_DETAILS Table

Field Name	Data Type	Description	Constraints
VISITOR_NAME	VARCHAR(100)	Full name of the visitor	NOT NULL
PHONE_NUMBER	VARCHAR(15)	Contact phone number of the visitor	NOT NULL
PRISONER_ID	INT	ID of the prisoner being visited	Foreign Key, References `PRISONER_ID` in `PRISONER` table
DATE	DATE	Date of the visit	NOT NULL
TIME	TIME	Time of the visit	NOT NULL

This data dictionary provides a clear and structured description of the database schema, ensuring that both developers and stakeholders have a common understanding of the data elements used in the Prison Management System.

5. Component Design

5.1 User Interface Component

This componenet is responsible for all interactions between the user and the system. Different users can interact with the system in different ways. Admins have access to all parts of the system while other staff may only access a subset of the interfaces. The user interface is implemented using HTML/CSS/JS and React as a framework.

5.2 Prisoner Management Component

This component is responsible for managing all prisoner related activities. It contains the following functions.
`addPrisoner(prisonerID,aadharNo,crimeId,enterDate,releaseDate)`: Adds a prisoner
`addPrisonerDetails(aadharNo,name,age,noOfConvictions)`:Adds prisoner details

5.3 Staff Management Component

This component is responsible for managing all staff related activities. Only the admin can access this component. It contains the following functions.
`addStaff(name,age,phoneNo,role)`: Adds a staff member
`removeStaff(staffID)`:Removes a staff member

5.4 Visitor Management Component

This component is responsible for managing all visitor related activities. It contains the following functions.
`addVisitor(visitorName,phoneNo,prisonerID,date,time)`

5.5 Cell Management Component

This component is responsible for managing all cell assignment related activities. It contains the following functions.
`assignCell(cellNo,prisonerID)`: Assigns a cell to the prisoner `markCellAsUnoccupied(cellNo)`:Marks a cell as unoccupied

5.6 Account Management Component

This component is responsible for managing all user account related activities. It is only accessible to the admin. It contains the following functions.
`addAccount(staffID,password)`: Creates a new account for a staff member
`changePassword(staffID,newPassword)`: Changes the password of a pre-existing account
`removeAccount(staffID)`: Removes an account

5.7 Login Component

This component is responsible for authenticating and maintaining the session of the various users of the PMS. It contains the login function.

`login(staffID,password)`: Logs in to the system

6. HUMAN INTERFACE DESIGN

6.1 Overview of User Interface:

Overview of User Interface The system's UI provides a robust interface for users such as prison staff, administrators etc. Users can browse through the available services such as checking prisoner details, update the details of each prisoner, assign jobs for the prisoners, shift the prisoners to a new cell.

6.2 Screen Images

- Login Page: Fields for username, password, and login button
- Service Listings: Show a list an available service.
- Search Bar: A universal search bar for quick navigation or finding specific inmates, staff, or records.
- Navigation Bar: A menu containing links to different sections (Inmates, Incidents, Visitors, Reports).
- Visitor Log Table: A list showing all past and scheduled visits for inmates, along with visitor details.
- Search: Search by visitor or inmate to locate specific visits or visitors.

6.3 Screen Objects and Actions

- Login Button: Triggers user authentication.
- Search Functionality: Enter search queries to quickly access relevant data (e.g., inmate records, reports).
- Navigation: Click items in the navigation bar to move between modules.
- Add Visitor Button: Button to schedule a new visit.

7. REQUIREMENTS MATRIX

A requirement matrix is a tool used in project management and system development to track and manage requirements throughout a project. It helps ensure that all defined requirements are addressed by the system design, implementation, and testing.

Requirement ID	**System Component**	**Description**
FR-1	User Management	Implements login and registration
FR-2	Inmate Management	Update inmate records (e.g., medical updates, transfer status)
FR-3	Inmate Management	View detailed inmate profiles including personal info, legal status, and medical history
FR-4	Dashboard	Role-based dashboard showing user-specific data
FR-5	Inmate Management	Ability to transfer inmate records to another facility securely