CivicFix

Software Design and Requirement Specification

Session: 2021 – 2025

Submitted by:

Muhammad Husnain 2021-SE-07

Hammad Ul Hassan 2021-SE-36

Faqeed Hassan 2021-SE-54

Supervised by:

Ms. Drakhshan Bokhat

Department of Computer Science, New Campus

University of Engineering and Technology

Lahore, Pakistan

Declaration

Contents

List of Figures

List of Tables

Chapter 1

Software Requirement Specification

The Software Requirement Specification (SRS) for CivicFix outlines the functional and non-functional requirements essential for the successful development, deployment, and maintenance of a robust and user-friendly complaint management system. These requirements ensure the system aligns with its core objectives: automating complaint routing, streamlining communication between users and service providers, and enabling efficient tracking and resolution of utility-related issues. The SRS provides a structured framework for developers, testers, and stakeholders to understand the functionality, performance expectations, and constraints of the CivicFix system.

1.1 Functional Requirements

Functional requirements define the basic system behaviour. These are essential features and functionalities that allow the CivicFix system to work as intended. Each functional requirement is categorized and numbered, ensuring traceability throughout the project.

1.1.1 Business Requirements

- **FR-01-01**: The system shall allow users to submit complaints via the CivicFix mobile app. [**Priority 1**]

- **FR-01-02**: The system must categorize complaints using image recognition with computer vision models. [**Priority 1**]
- **FR-01-03**: The system must automatically route complaints to the relevant departments (e.g., WAPDA, SNGPL). [**Priority 1**]

1.1.2 Administrative Functions

- **FR-02-01**: Administrators shall be able to view all submitted complaints on a centralized dashboard. [**Priority 1**]
- **FR-02-02**: Administrators must assign teams to address reported issues. [**Priority 1**]
- **FR-02-03**: Administrators must mark complaints as resolved after receiving resolution proof. [**Priority 1**]

1.1.3 User Requirements

- **FR-03-01**: Users shall be able to track the status of their complaints in real time. [**Priority 2**]
- **FR-03-02**: Users must be able to upload images of the issue from their mobile phones. [**Priority 1**]
- **FR-03-03**: Users shall be notified when the complaint is resolved or a team is assigned. [**Priority 2**]

1.1.4 System Requirements

- **FR-04-01**: The system must provide secure user authentication and maintain a history of previous complaints for users. [**Priority 1**]
- **FR-04-02**: The system must support multi-department integration for complaint forwarding (e.g., SNGPL, WAPDA, LWMC). [**Priority 1**]

1.2 Non-Functional Requirements

Non-functional requirements describe how the CivicFix system will operate, including usability, performance, reliability, and security considerations. These are essential for ensuring a smooth and efficient user experience.

1.2.1 Usability

- **NR-01-01**: The system must have an intuitive user interface, ensuring ease of use across different devices and screen sizes. [**Priority 1**]
- **NR-01-02**: All system interfaces must be responsive and optimized for both mobile and web platforms. [**Priority 2**]

1.2.2 Reliability / Availability

- **NR-02-01**: CivicFix must be available 24/7, ensuring users can submit complaints at any time. [**Priority 1**]

1.2.3 Scalability

- **NR-03-01**: The system architecture must be designed to scale as the user base grows, accommodating more users and complaint types. [**Priority 2**]

1.2.4 Performance

- **NR-04-01**: The system must maintain fast response times for all API calls to provide real-time updates to users. [**Priority 1**]

1.2.5 Supportability

- **NR-05-01**: System support must include remote accessibility for troubleshooting and management. [**Priority 3**]
- **NR-05-02**: The system must include detailed documentation for developers and administrators. [**Priority 2**]

1.2.6 Security

- **NR-06-01**: The system must implement secure user authentication and data security to protect user information. [**Priority 1**]

1.3 Use Case Description

Table 1.1: Use Case 1:Login

UseCase Name	**Login**
Actor	Sub Administrator, Administrator, User
Goal	To authenticate the user into the system.
Precondition	The actor must be registered in the system.

Postcondition	The actor is logged in and can access the relevant functionalities based on their role.
Main Success Scenario	1. Actor navigates to the login page. 2. Actor enters username and password. 3. System verifies credentials. 4. Actor is logged into the system.
Alternative Path	If credentials are invalid, the system displays an error message.

Table 1.2: Use Case 2: Register

UseCase Name	**Register**
Actor	User
Goal	To create a new account in the system.
Precondition	User must provide required details (Name, CNIC, etc.).
Postcondition	User account is created, and the user can log in.
Main Success Scenario	1. User navigates to the registration page. 2. User enters the required details (Name, CNIC, Email, etc.). 3. System validates the details. 4. System creates a new user account. 5. User receives a confirmation of successful registration.

Alternative Path	• If required information is missing or invalid, the system shows an error. • If CNIC or email already exists in the system, the system displays an error like *"This email is already registered. Please use a different email."*.
Exceptions	• **Network Failure:** – *Scenario:* The system is unable to save the data due to a network issue. – *System Response:* Displays an error message like *"Unable to process your registration at the moment. Please try again later."*. • **System Validation Error:** – *Scenario:* Internal validation logic malfunctions (e.g., due to a software bug). – *System Response:* Displays a generic error like *"An error occurred. Please contact support."*. • **Database Error:** – *Scenario:* The system cannot save the data due to a database issue (e.g., out of storage). – *System Response:* Displays an error message like *"Registration cannot be completed. Please try again later."*.

Qualities (Non-Functional Requirements)	• **Usability:** – Registration form should be user-friendly with clear labels, hints, and error messages for invalid inputs. – Mandatory fields should be marked with an asterisk (*) and explained in tooltips. • **Performance:** – The system should validate and create the account within 2-3 seconds of submission. • **Security:** – All passwords must be stored in a hashed and salted format. – User inputs should be sanitized to prevent SQL injection or XSS attacks. – Email verification should be optional for additional security. • **Scalability:** – The system should support concurrent registrations without performance degradation. • **Accessibility:** – The form should be accessible to users with disabilities, adhering to WCAG standards.

Table 1.3: Use Case 3: Submit Complaint

Use Case Name	**Submit Complaint**
Actor	User
Goal	To submit a complaint for a specific issue.
Precondition	The user must be logged in.

Postcondition	Complaint is recorded in the system, and the status is set to "submitted."
Main Success Scenario	1. User selects the "Submit Complaint" option. 2. User provides details about the complaint (Complaint Type, Description, Image). 3. System saves the complaint details. 4. Complaint status is set to "submitted." 5. User receives a confirmation of successful complaint submission.
Alternative Path	If complaint details are incomplete, the system requests more information.

Table 1.4: Use Case 4: Team

Use Case Name	**Assign Team**
Actor	Sub Administrator
Goal	To assign a team to resolve a complaint.
Precondition	The complaint must be submitted, and the team must be available.
Postcondition	The team is assigned to the complaint.

Main Success Scenario	1. Sub Administrator views the list of submitted complaints. 2. Sub Administrator selects a complaint. 3. Sub Administrator assigns a team to handle the complaint. 4. System updates the complaint with the assigned team. 5. The team is notified of the assignment.
Alternative Path	If no team is available, the system notifies the Sub Administrator.

Table 1.5: Use Case 5: Monitor Complaint

Use Case Name	**Monitor Complaint**
Actor	Sub Administrator
Goal	To track the progress of a complaint.
Precondition	The complaint must be in the system.
Postcondition	Sub Administrator views the current status and updates.
Main Success Scenario	1. Sub Administrator logs in to the system. 2. Sub Administrator selects a complaint to monitor. 3. System displays the current status and progress of the complaint. 4. Sub Administrator takes necessary actions based on the updates.
Alternative Path	N/A

Table 1.6: Use Case 6: Track Complaint Status

Use Case Name	**Track Complaint Status**
Actor	User, Sub Administrator, Administrator, Team
Goal	To view the current status of a complaint.
Precondition	A complaint must be submitted and present in the system.
Postcondition	The actor views the complaint's current status.
Main Success Scenario	1. Actor logs into the system. 2. Actor selects the "Track Complaint Status" option. 3. System displays the current status and updates of the complaint.
Alternative Path	1. If no updates are available, the system notifies the actor.

Table 1.7: Use Case 7: Add New Team

Use Case Name	**Add New Team**
Actor	Sub Administrator
Goal	To add a new team into the system.
Precondition	The Sub Administrator must be logged in.
Postcondition	A new team is created and available for assignment.

Main Success Scenario	1. Sub Administrator navigates to the team management section. 2. Sub Administrator provides details for the new team (Team Name, Members). 3. System saves the new team in the system. 4. The team is available for complaint assignments.

Table 1.8: Use Case 8: Provide Feedback

Use Case Name	**Provide Feedback**
Actor	User
Goal	To provide feedback on the resolution of the complaint.
Precondition	The complaint must be resolved.
Postcondition	Feedback is submitted and recorded in the system.
Main Success Scenario	1. User logs in to the system. 2. User views the resolved complaints. 3. User selects a complaint and provides feedback (Rating, Comments). 4. System saves the feedback and notifies the administrator.

Table 1.9: Use Case 9: Check Feedback

Use Case Name	**Check Feedback**
Actor	Administrator
Goal	To review the feedback provided by users.
Precondition	Feedback must be submitted by the user.

Postcondition	Administrator reviews the feedback.
Main Success Scenario	1. Administrator logs into the system. 2. Administrator views the feedback section. 3. Administrator reviews feedback for resolved complaints.

Table 1.10: Use Case 10: Computer Vision Analysis (Include)

Use Case Name	**Computer Vision Analysis (Include)**
Actor	Internal System
Goal	To analyze images submitted with complaints using computer vision.
Precondition	The user has submitted an image with the complaint.
Postcondition	The system analyzes the image and provides insights.
Main Success Scenario	1. User submits a complaint with an image. 2. System triggers the ”Computer Vision Analysis” process. 3. System analyzes the image and categorizes the issue based on the analysis.

Chapter 2

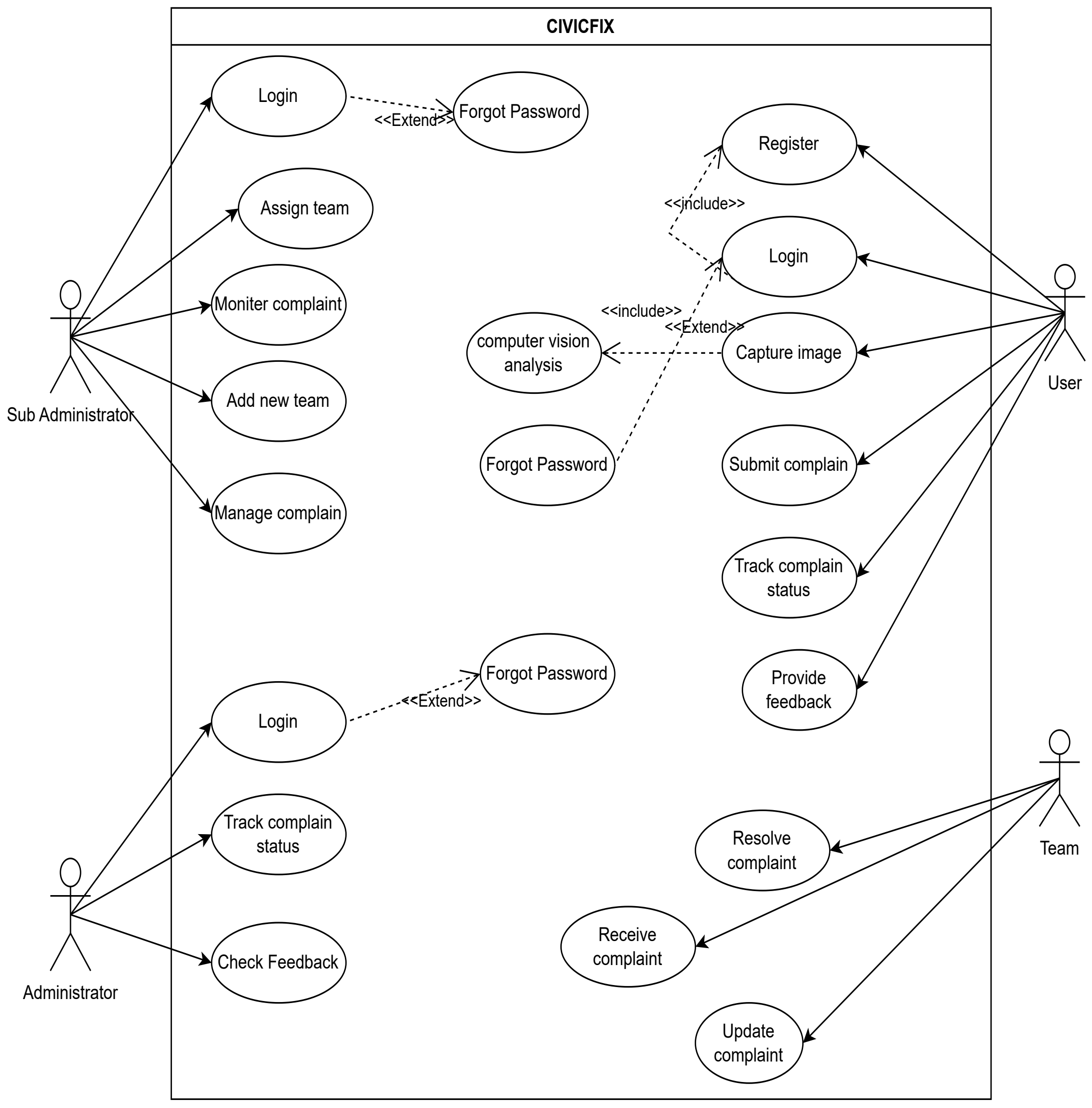
Design Specification

2.1 System Behavioral Design

Behavioral diagrams portray a dynamic view of a system, illustrating how it behaves and functions over time. They describe the interactions and processes within the system, providing insight into its operational aspects.

2.1.1 Use Case Diagram

User: Registers, logs in, submits complaints, tracks status, provides feedback, and resets passwords.
Sub-Administrator: Assigns teams, monitors complaints, manages teams, tracks status, and resets passwords.
Administrator: Tracks complaints, checks feedback, and resets passwords.
Team: Receives, resolves, and updates complaints.
Includes Computer Vision Analysis for complaint image processing.



2.1.2 Activity Diagram

The activity diagram for *CivicFix* shows the process flow.
In figure 2.2 the Diagram illustrates the step-by-step Login Process of Administrator within the CivicFix system.

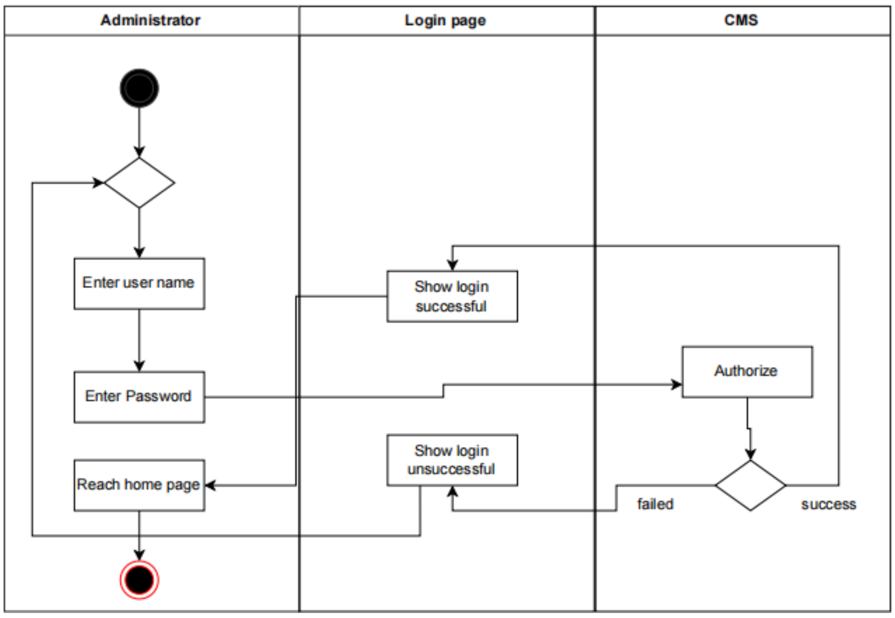


FIGURE 2.2: Activity Diagram 1 for CivicFix

In figure 2.3 the Diagram illustrates the step-by-step Login Process of Sub Administrator within the CivicFix system.

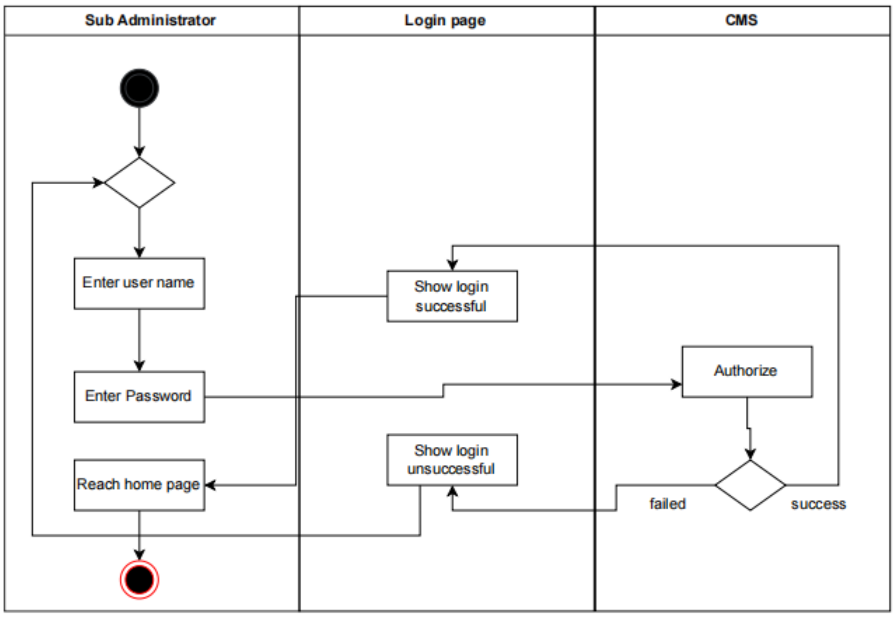


FIGURE 2.3: Activity Diagram 2 for CivicFix

In figure 2.4 the Diagram illustrates the step-by-step Login Process of User within the CivicFix system.

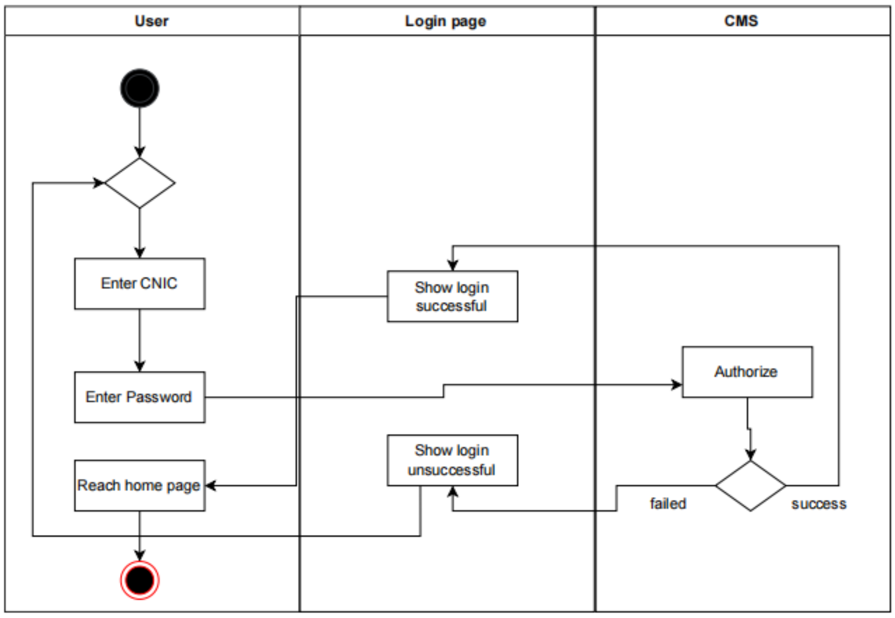


FIGURE 2.4: Activity Diagram 3 for CivicFix

In figure 2.5 the Diagram illustrates the step-by-step Login Process of Team within the CivicFix system.

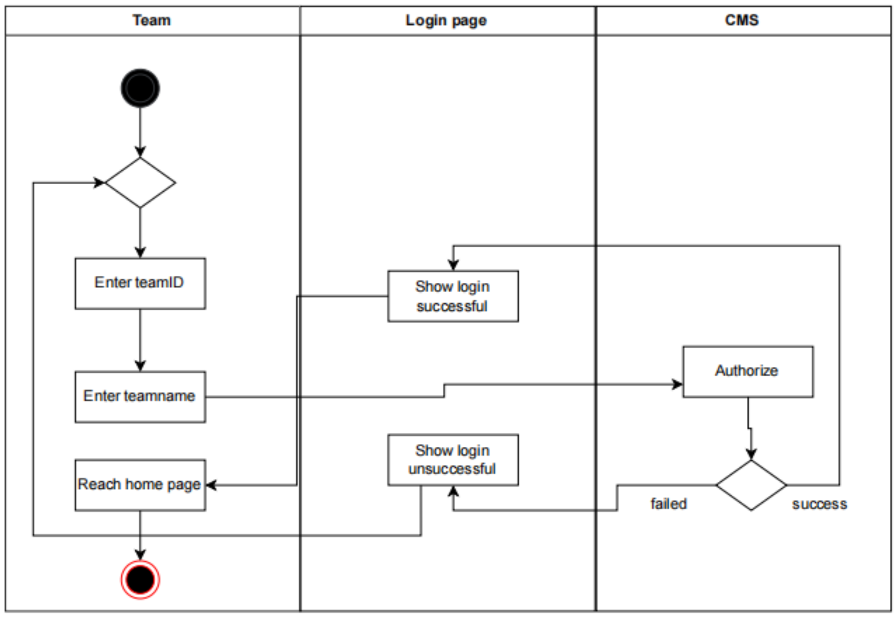


FIGURE 2.5: Activity Diagram 4 for CivicFix

In figure 2.6 the Diagram illustrates the step-by-step Login Process of System Notification within the CivicFix system.

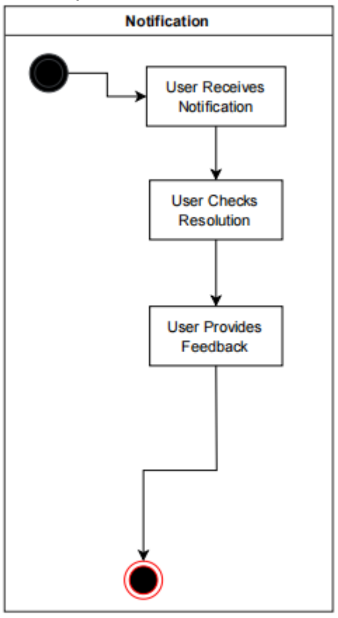


FIGURE 2.6: Activity Diagram 9 for CivicFix

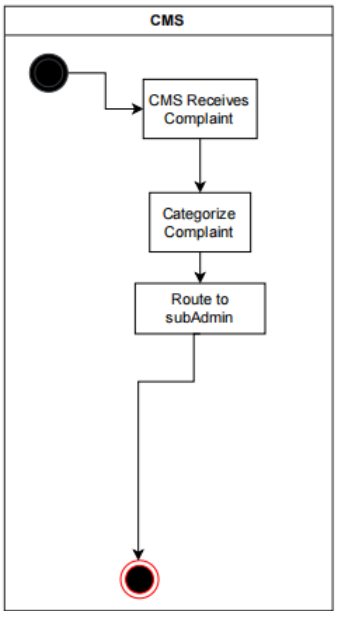


FIGURE 2.7: Activity Diagram 6 for CivicFix

In figure 2.8 the Diagram illustrates the step-by-step Sub Administrator Activity within the CivicFix system.

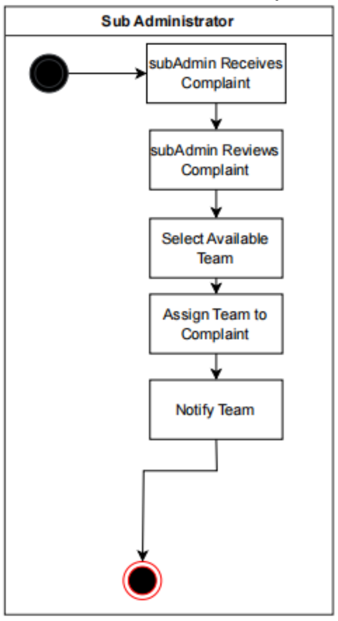


FIGURE 2.8: Activity Diagram 7 for CivicFix

In figure 2.9 the Diagram illustrates the step-by-step Team's Activity within the CivicFix system.

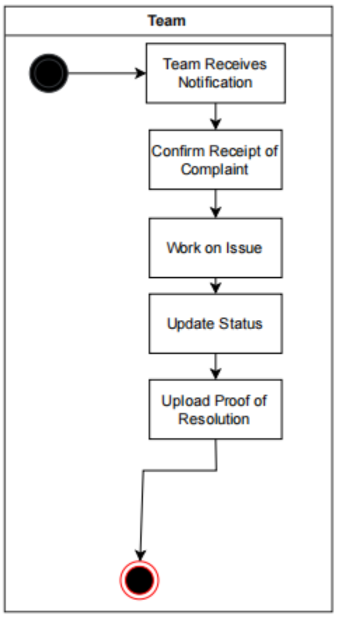

Team
Team Receives Notification
Confirm Receipt of Complaint
Work on Issue
Update Status
Upload Proof of Resolution

FIGURE 2.9: Activity Diagram 8 for CivicFix

In figure 2.10 the Diagram illustrates the step-by-step User's System within the CivicFix system.

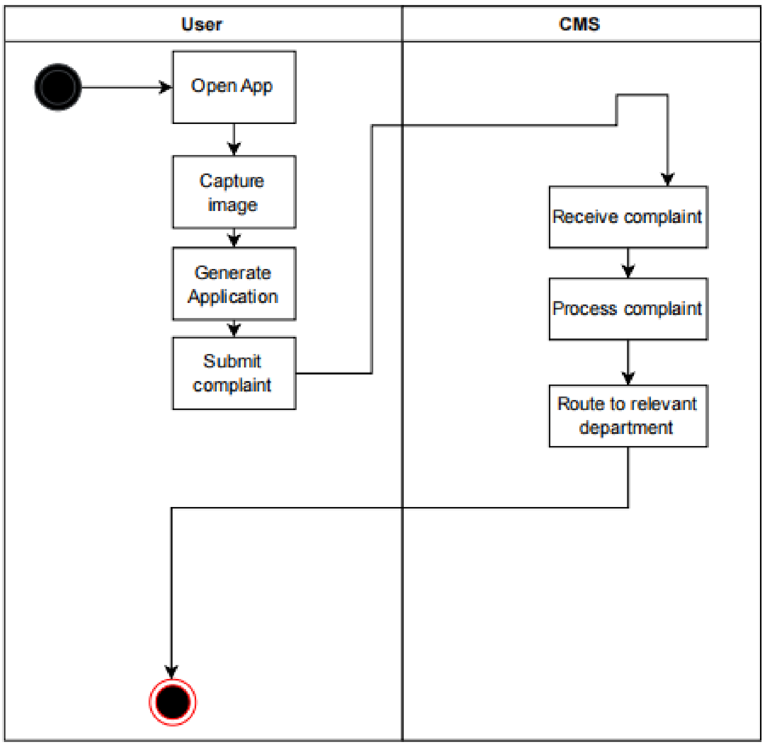


Figure 2.10: Activity Diagram 9 for CivicFix

2.1.3 State Diagram

In this Figure 2.11 the complaint begins in the Submitted state when the User submits it through the system.
It moves to the Under Review state when the SubAdmin reviews the details of the complaint.
After a Team is assigned, the complaint transitions to the In Progress state, where the team works on resolving the issue.Once the complaint is resolved, it moves to the Resolved state, where the User is notified.
The complaint can then transition to the Closed state after the User provides feedback or confirms resolution.

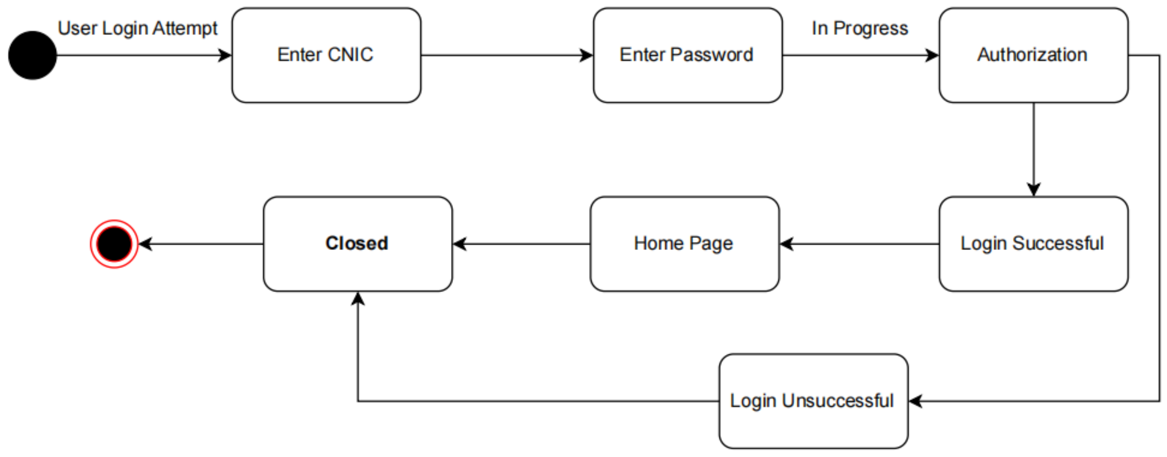


FIGURE 2.11: State Diagram 1 for CivicFix

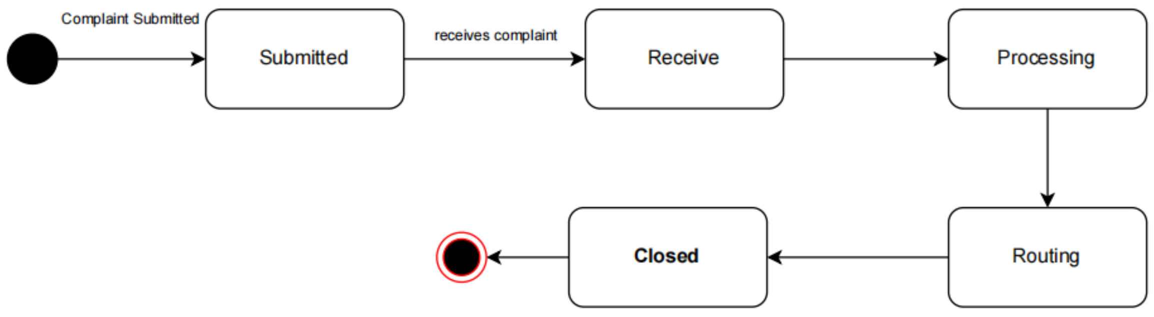


FIGURE 2.12: State Diagram 2 for CivicFix

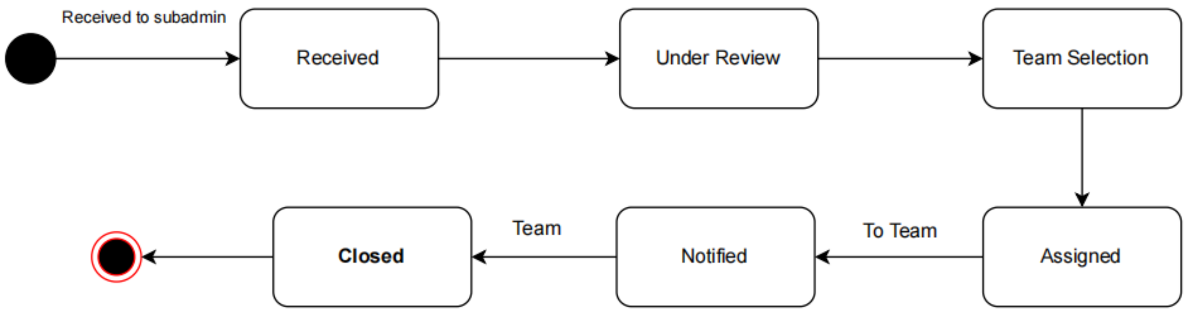


FIGURE 2.13: State Diagram 3 for CivicFix

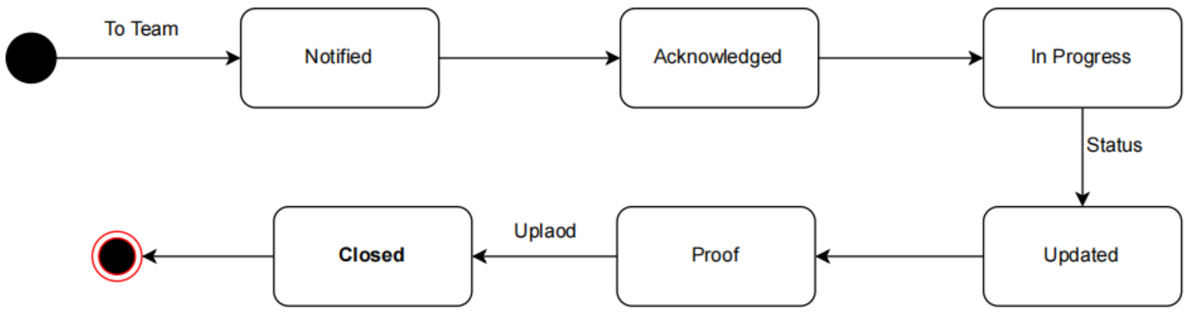


FIGURE 2.14: State Diagram 4 for CivicFix

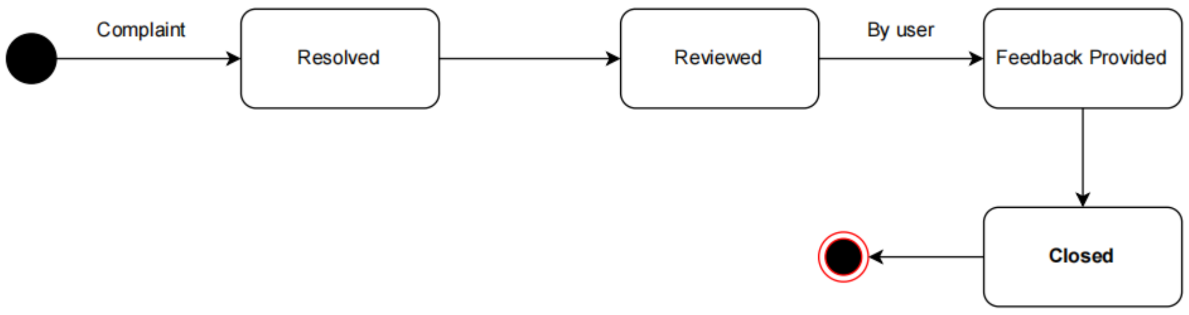


FIGURE 2.15: State Diagram 5 for CivicFix

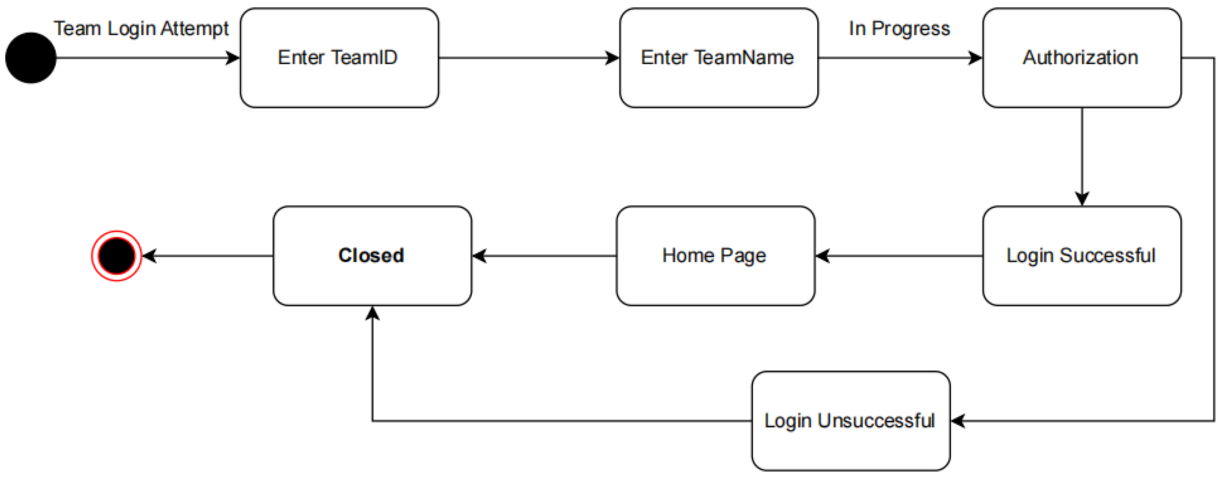


FIGURE 2.16: State Diagram 5 for CivicFix

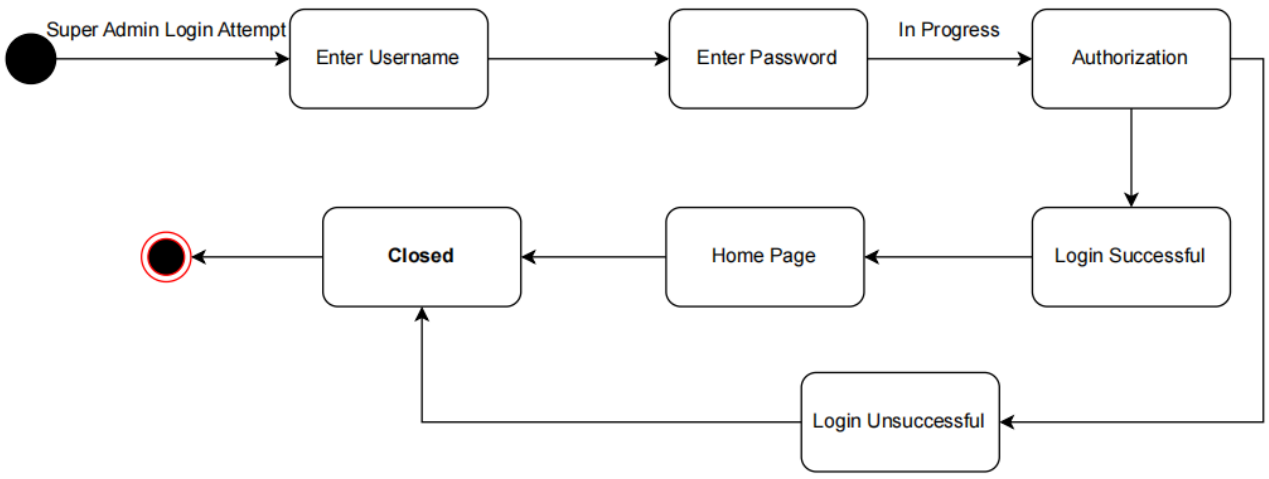


FIGURE 2.17: State Diagram 6 for CivicFix

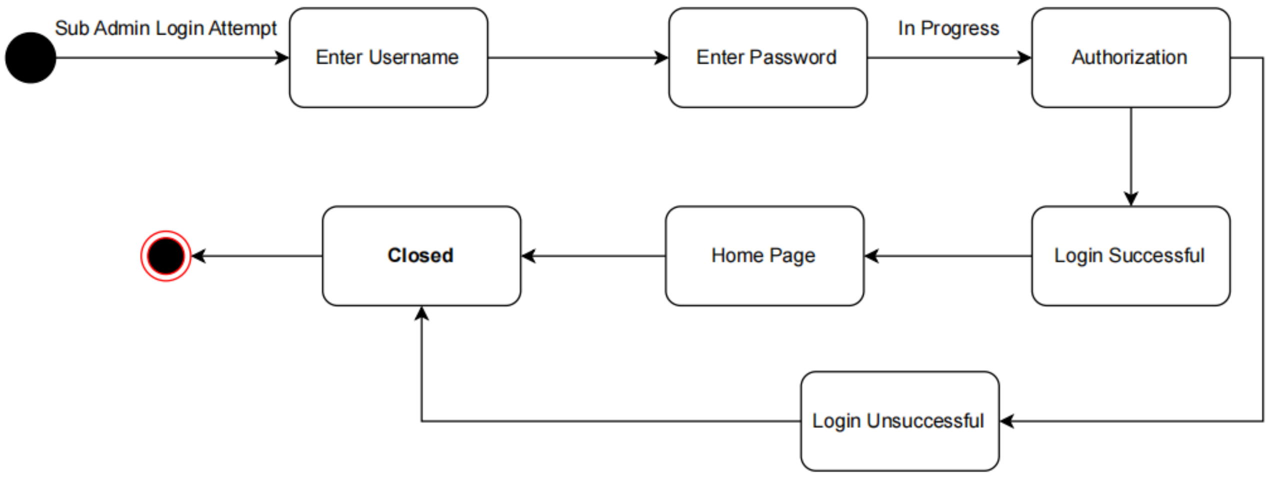


FIGURE 2.18: State Diagram 7 for CivicFix

2.1.4 Sequence Diagram

In figure 2.19 the Diagram illustrates the step-by-step Login Process of User within the CivicFix system.

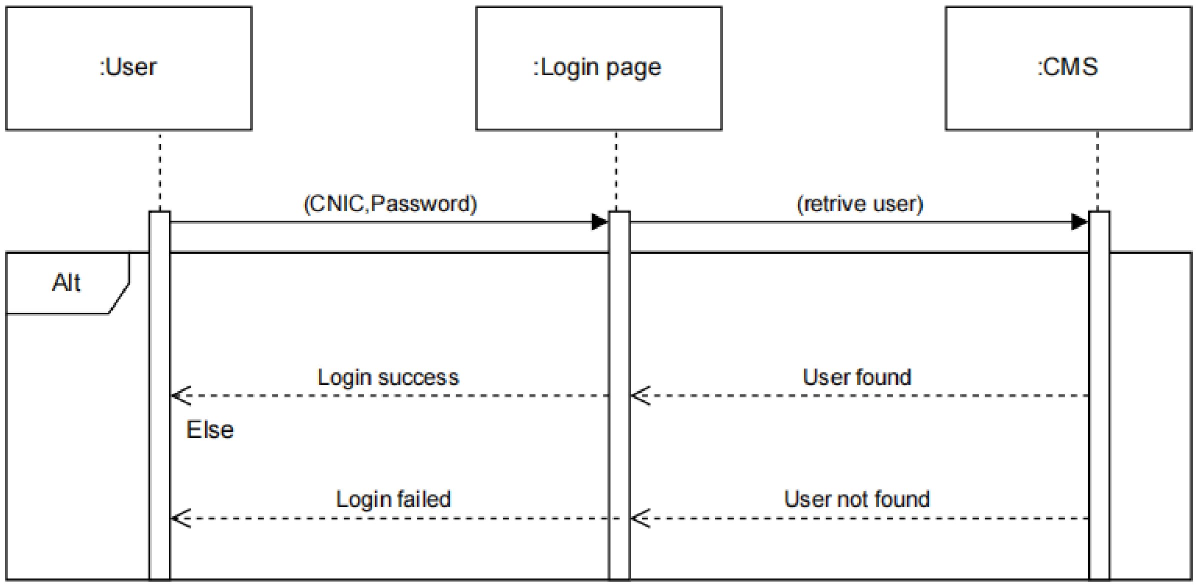


Figure 2.19: Sequence Diagram 1 for CivicFix

In figure 2.20 the Diagram illustrates the step-by-step User 's Complain Submission within the CivicFix system.

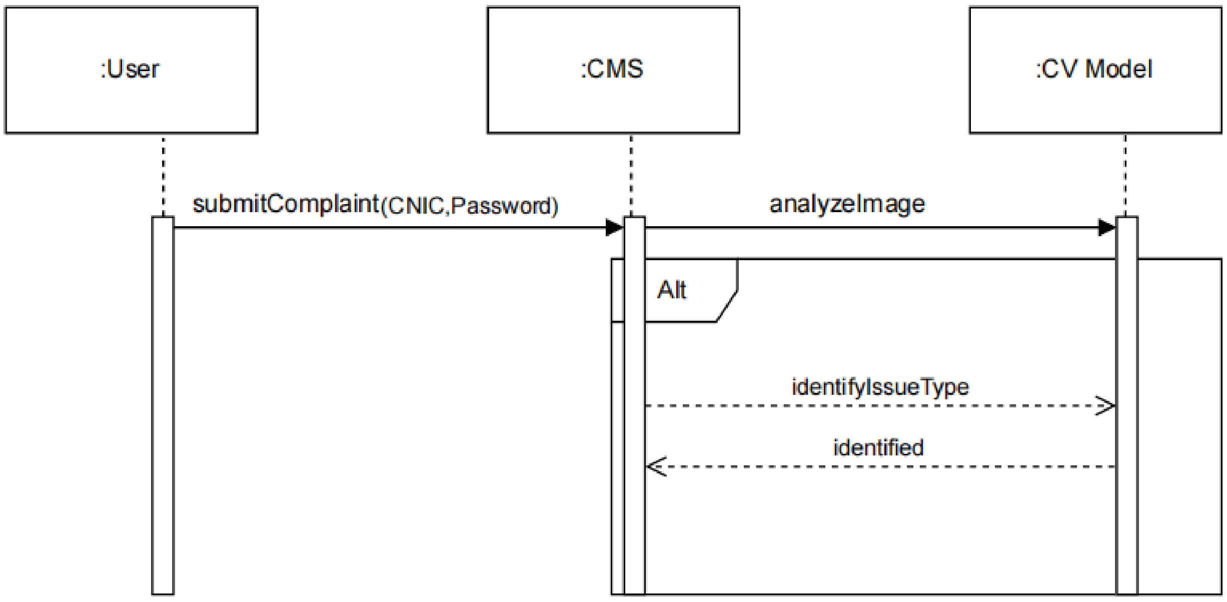


Figure 2.20: Sequence Diagram 2 for CivicFix

In figure 2.21 the Diagram illustrates the connection between CMS and Sub Administrator within the CivicFix system.

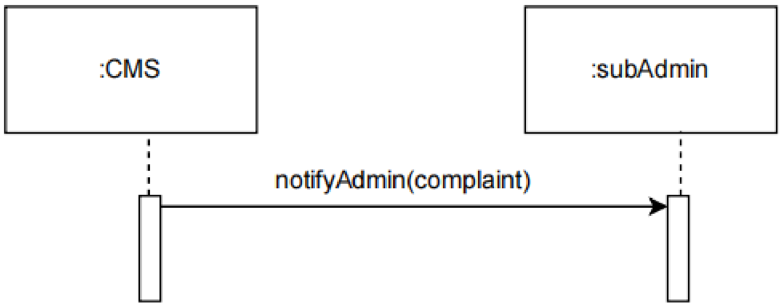


FIGURE 2.21: Sequence Diagram 3 for CivicFix

In figure 2.22 the Diagram illustrates the work flow of Sub Administrator within the CivicFix system.

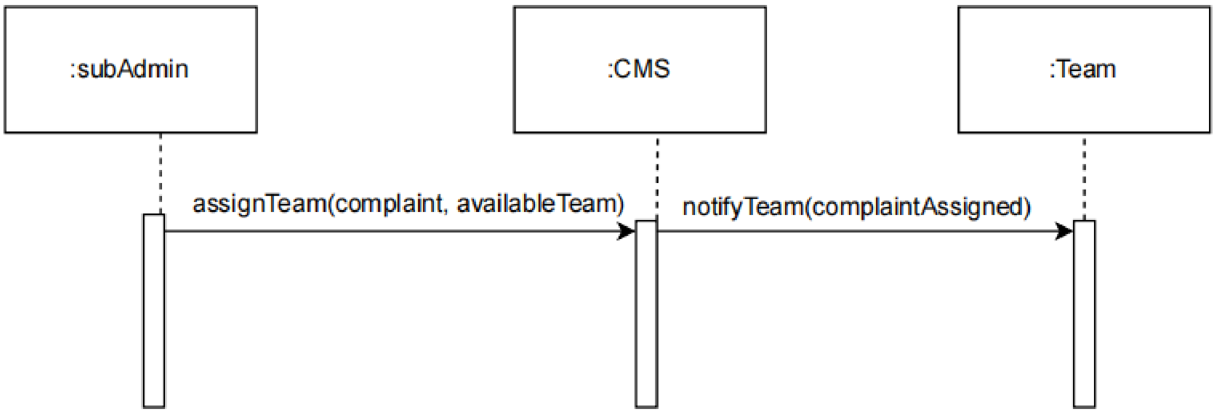

:subAdmin
:CMS
:Team
assignTeam(complaint, availableTeam)
notifyTeam(complaintAssigned)

FIGURE 2.22: Sequence Diagram 4 for CivicFix

In figure 2.23 the Diagram illustrates the Notification connection between Team and user within the CivicFix system.

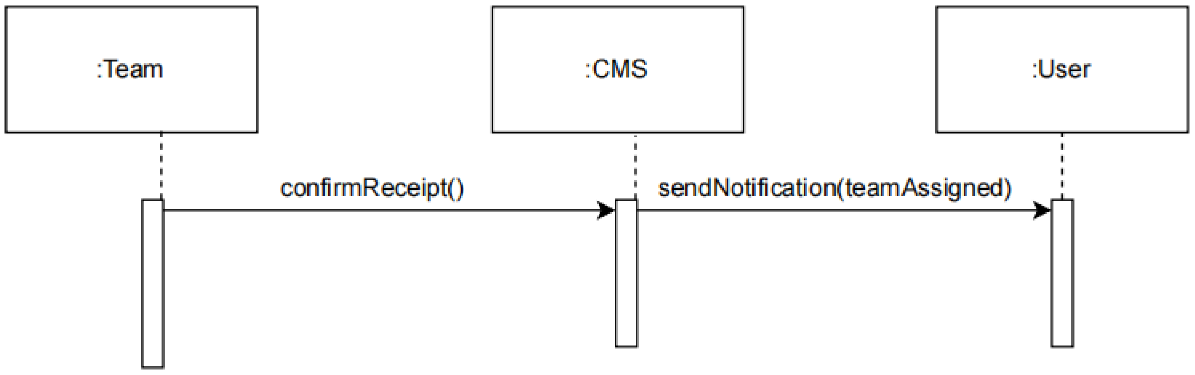


FIGURE 2.23: Sequence Diagram 5 for CivicFix

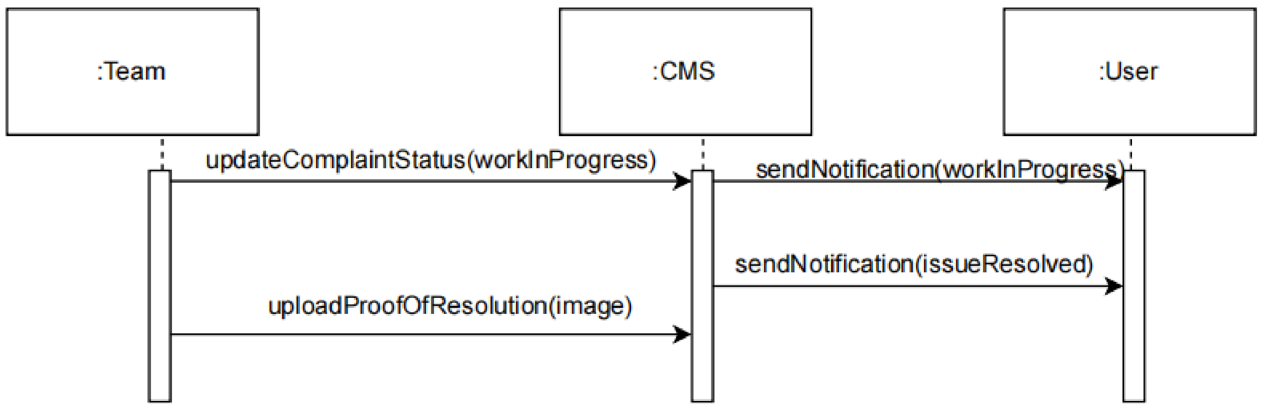

:Team
:CMS
:User
updateComplaintStatus(workInProgress)
sendNotification(workInProgress)
sendNotification(issueResolved)
uploadProofOfResolution(image)

FIGURE 2.24: Sequence Diagram 6 for CivicFix

In figure 2.25 the Diagram illustrates the Feedback system between user and super Administrator within the CivicFix system.

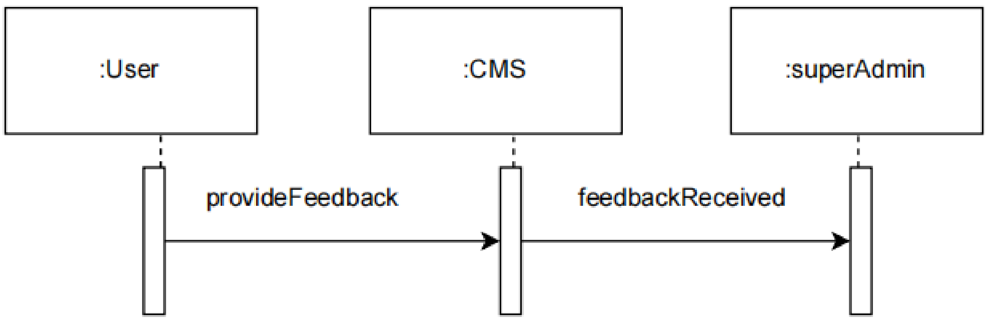


FIGURE 2.25: Sequence Diagram 7 for CivicFix

In figure 2.26 the Diagram illustrates the Login Process of Super Administrator within the CivicFix system.

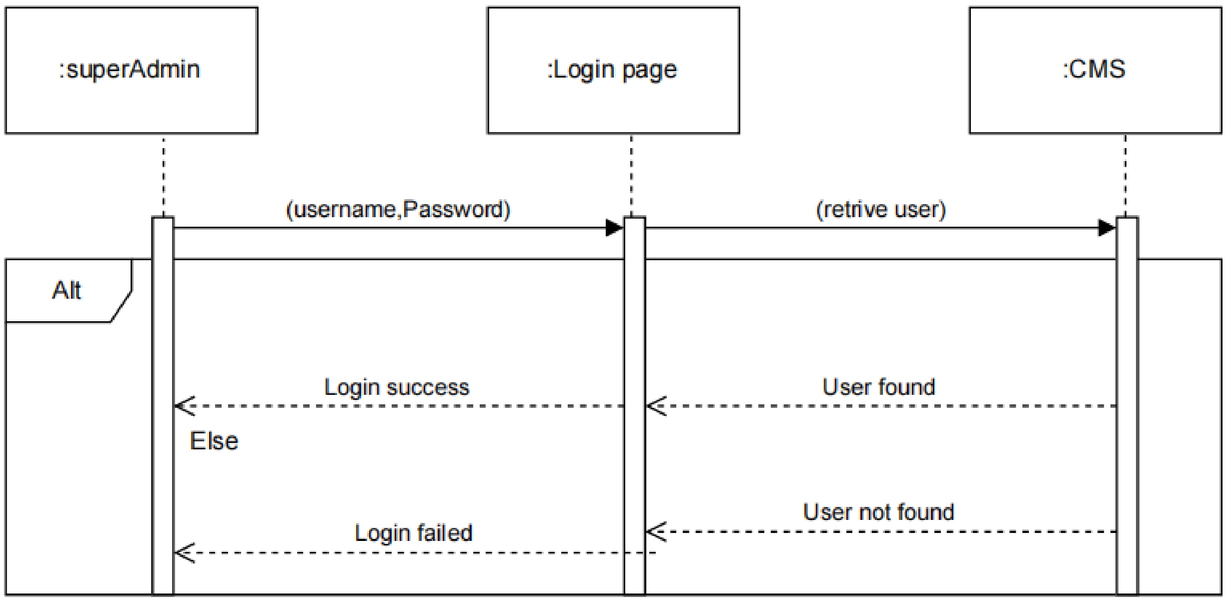


FIGURE 2.26: Sequence Diagram 8 for CivicFix

In figure 2.27 the Diagram illustrates the Login Process of Sub Administrator within the CivicFix system.

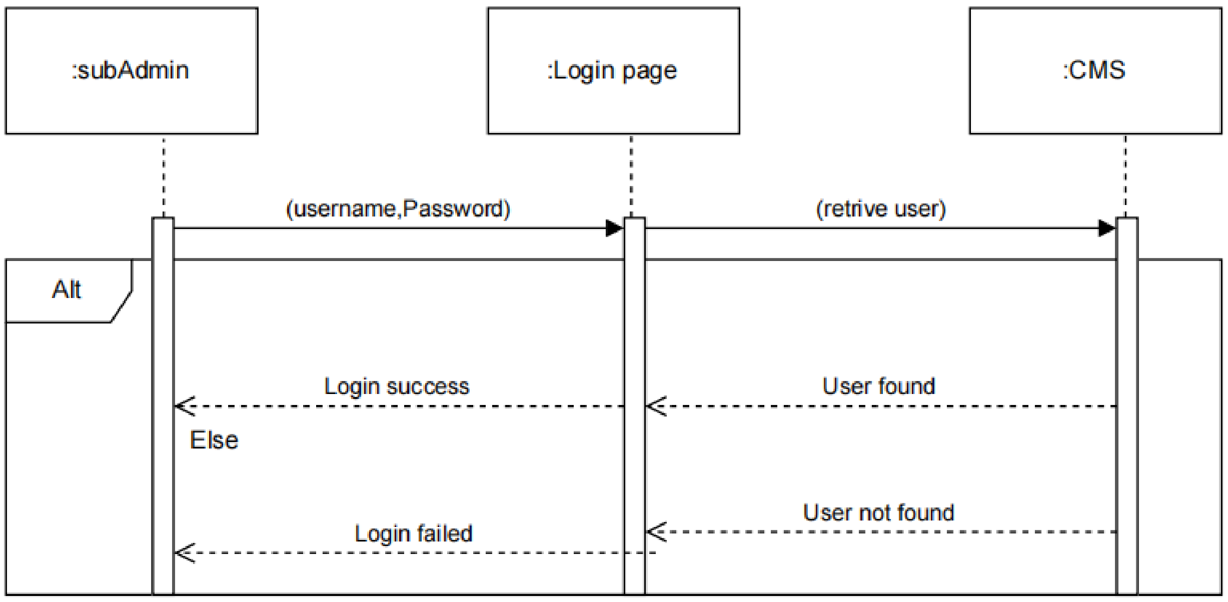


FIGURE 2.27: Sequence Diagram 9 for CivicFix

In figure 2.28 the Diagram illustrates the Login Process of Team within the CivicFix system.

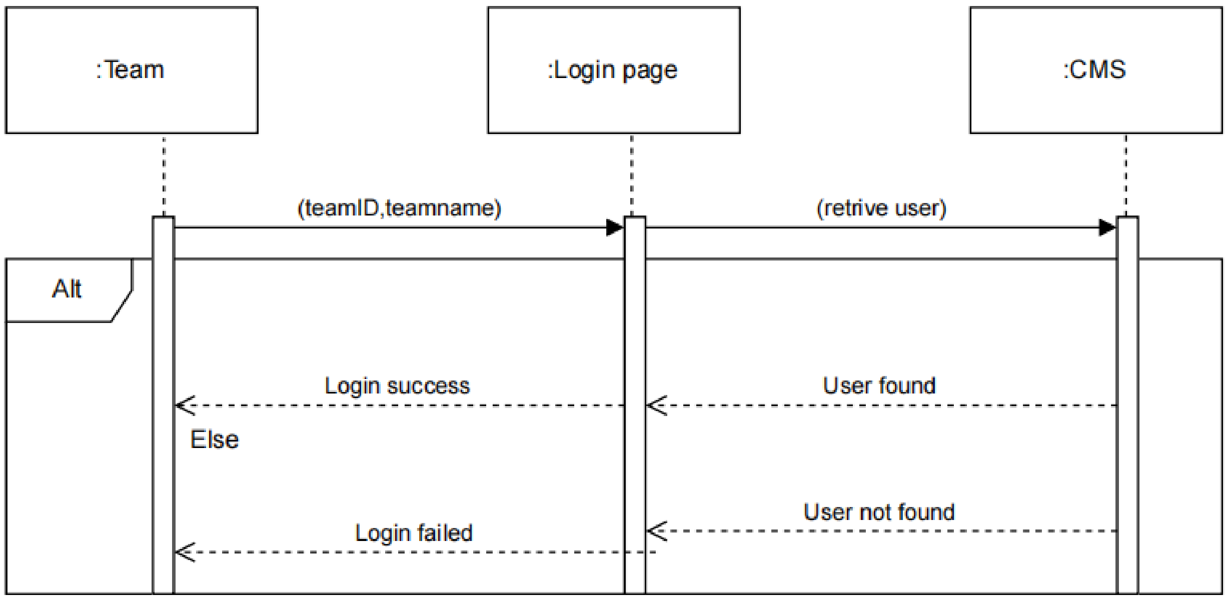


FIGURE 2.28: Sequence Diagram 10 for CivicFix

2.1.5 Collaboration Diagram

The collaboration diagram shows how different objects (user, system, department, maintenance team) interact to resolve an issue in *CivicFix*.

In figure 2.29 the Diagram illustrates the Login Process of User within the CivicFix system.

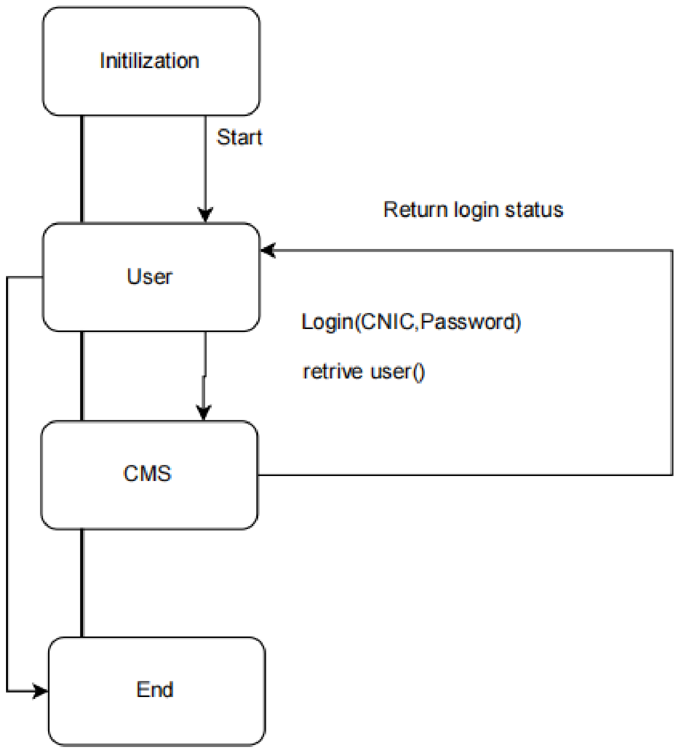


FIGURE 2.29: Collaboration Diagram 1 for CivicFix

In figure 2.30 the Diagram illustrates the Login Process of SuprAdmin within the CivicFix system.

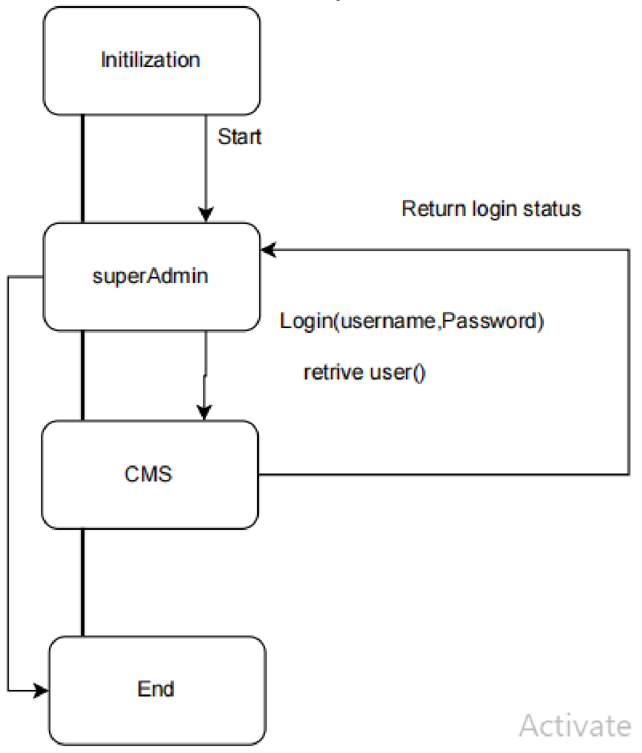


FIGURE 2.30: Collaboration Diagram 2 for CivicFix

In figure 2.31 the Diagram illustrates the Login Process of SubAdmin within the CivicFix system.

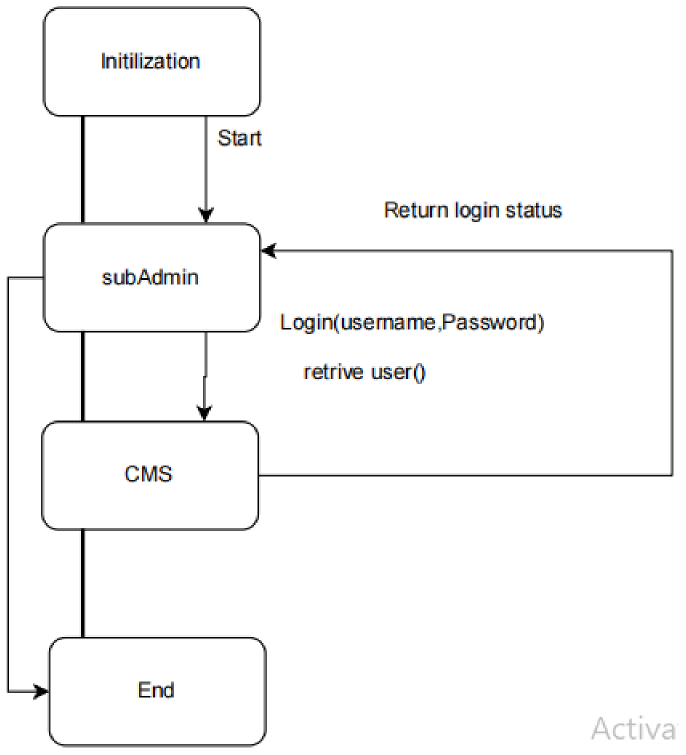


FIGURE 2.31: Collaboration Diagram 3 for CivicFix

In figure 2.32 the Diagram illustrates the User's submit complain within the CivicFix system.

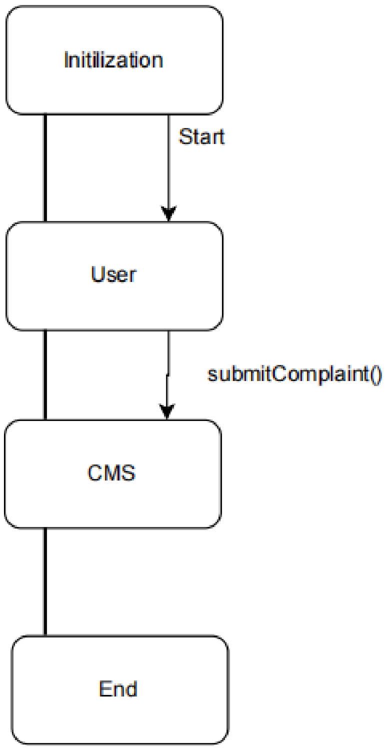


FIGURE 2.32: Collaboration Diagram 4 for CivicFix

In figure 2.33 the Diagram illustrates the Login Process of Team within the CivicFix system.

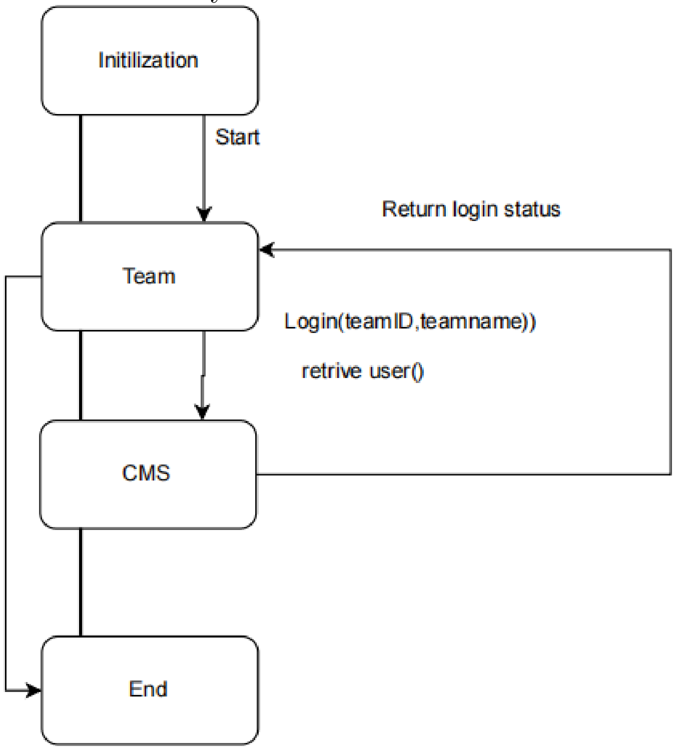


FIGURE 2.33: Collaboration Diagram 5 for CivicFix

In figure 2.34 the Diagram illustrates the subadmin wrok within the CivicFix system.

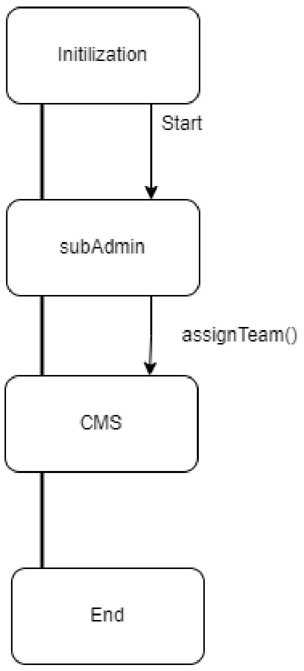


Figure 2.34: Collaboration Diagram 6 for CivicFix

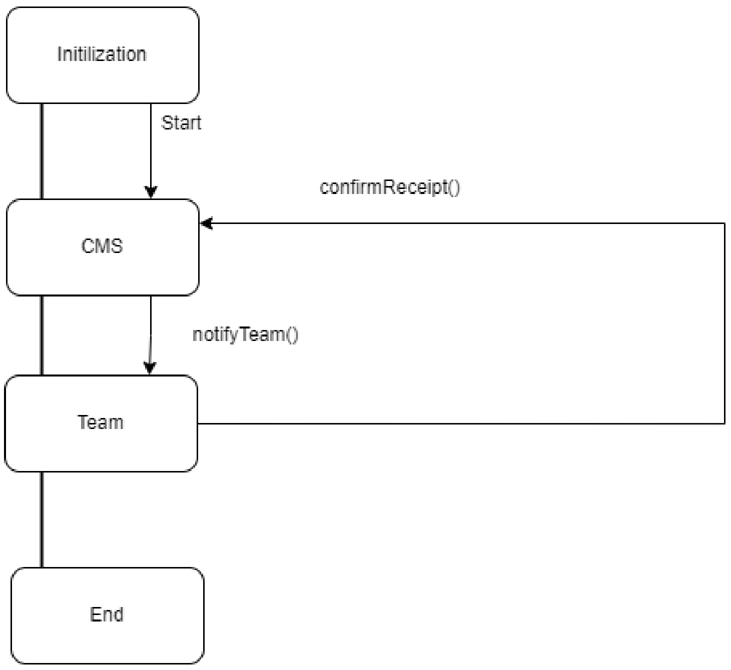


Figure 2.35: Collaboration Diagram 7 for CivicFix

In figure 2.36 the Diagram illustrates the User's Notification complain within the CivicFix system.

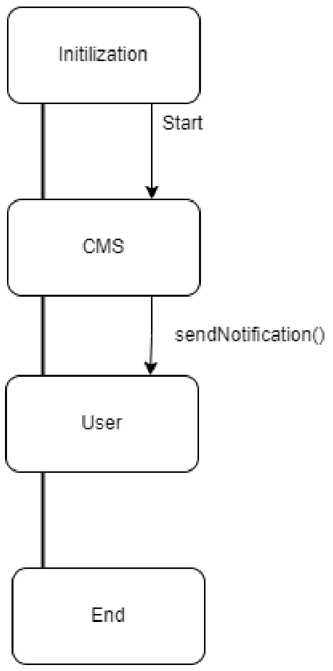


FIGURE 2.36: Collaboration Diagram 8 for CivicFix

In figure 2.37 the Diagram illustrates the superadmin's Notification within the CivicFix system.

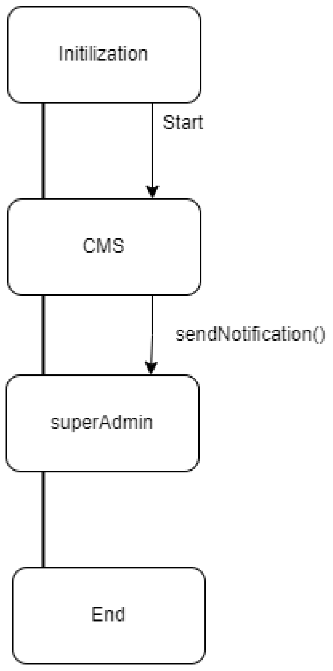


FIGURE 2.37: Collaboration Diagram 9 for CivicFix

2.2 System Structure Design

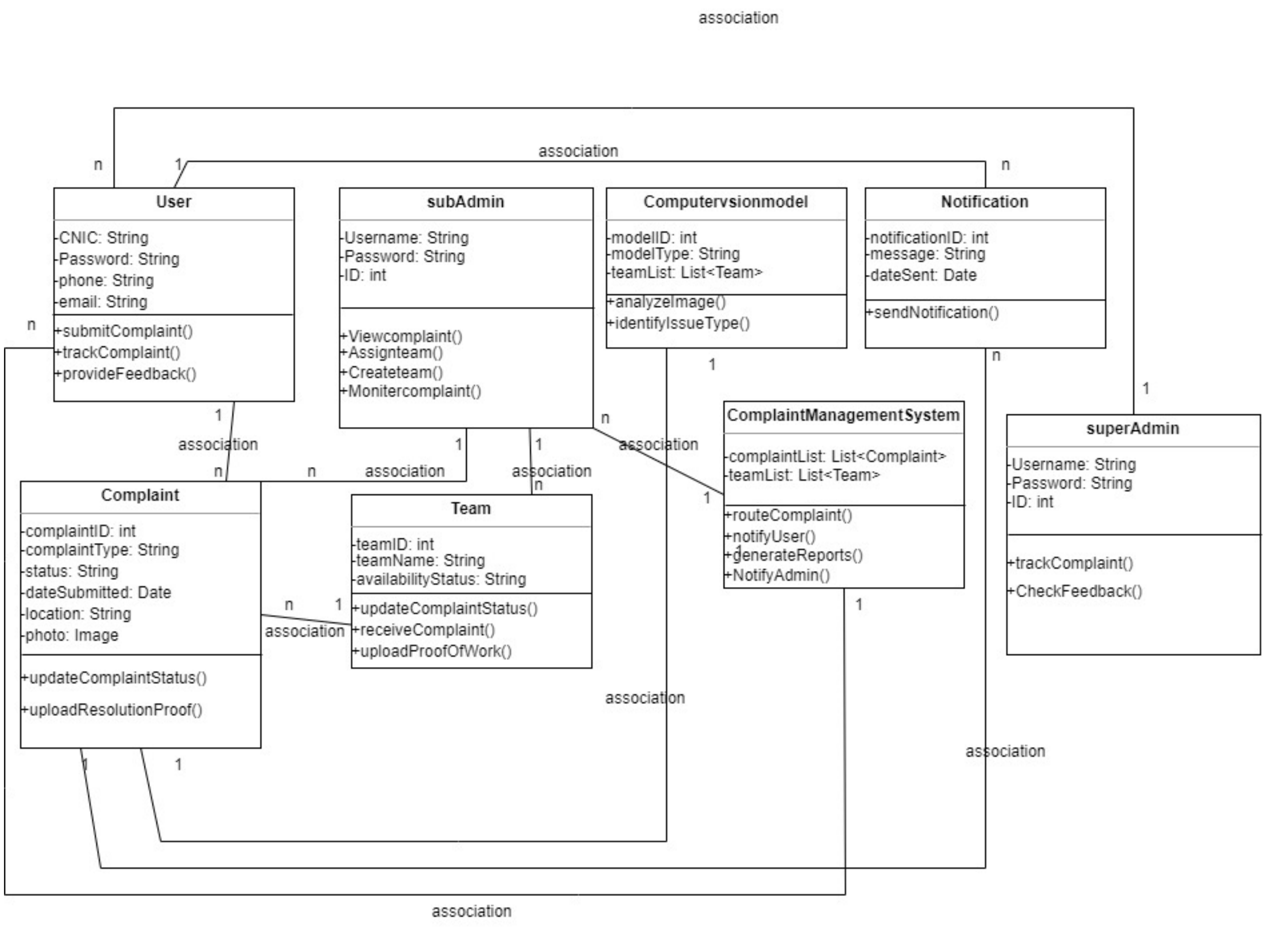
Structural diagrams depict the static aspects or structure of a system, providing a detailed outline of the system's architecture and its components.

2.2.1 Class Diagram

The Class Diagram illustrates the relationships between core entities such as User, Complaint, SubAdmin, Team, and CMS.
In 2.38 diagram:
The User class has attributes such as CNIC, Name, and Email, and is associated with multiple Complaint instances.

The Complaint class represents each complaint and contains details like Complaint Type, Status, and Image, with a one-to-many relationship to User and Team.

SubAdmin manages the assignment of complaints to Team, while SuperAdmin oversees SubAdmin activities. Each Team can handle multiple complaints but is managed by one SubAdmin.



2.2.2 Component Diagram

A component diagram illustrates the different software components used in *CivicFix* and their relationships.

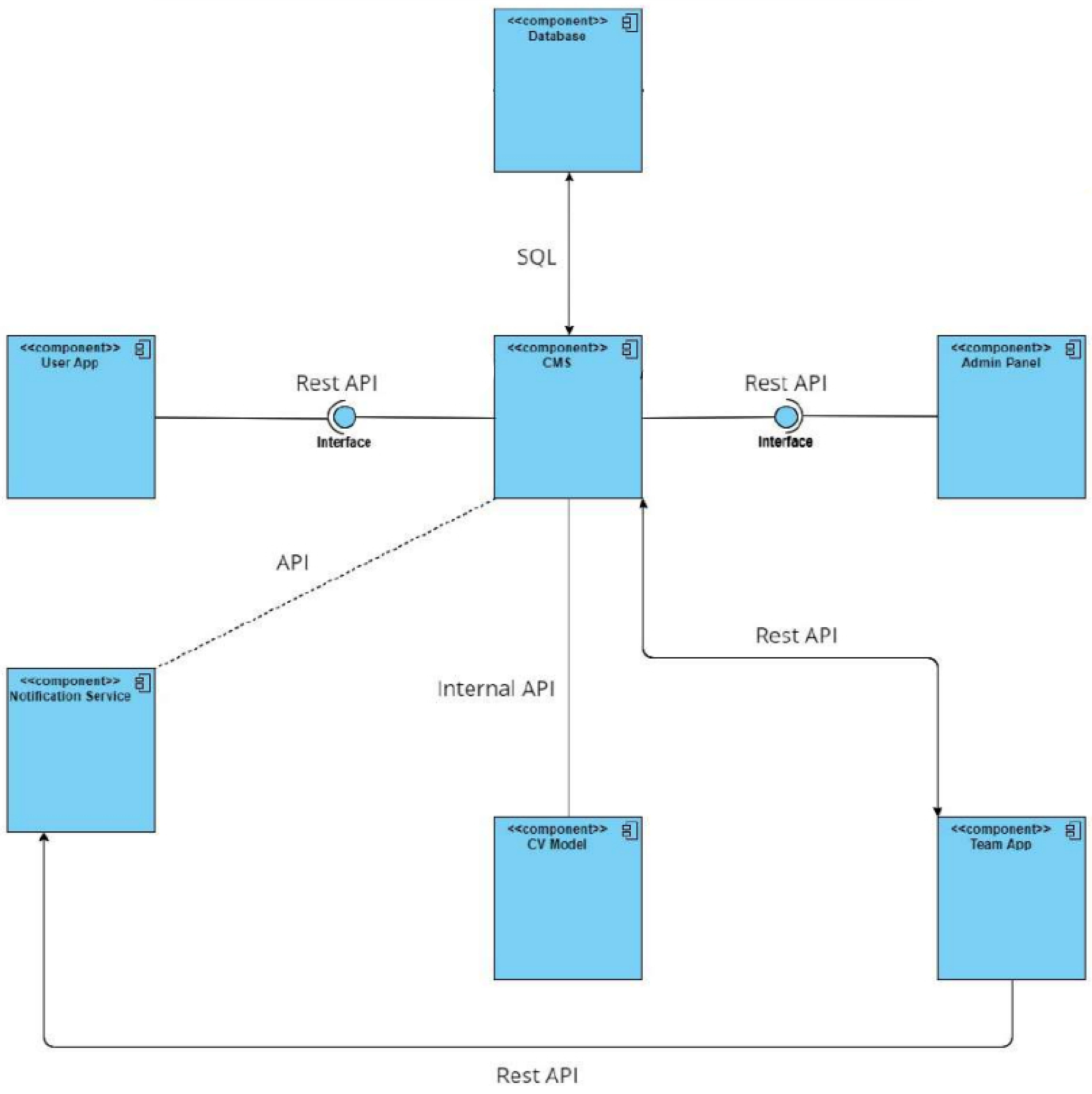


FIGURE 2.39: Component Diagram for CivicFix

2.2.3 Deployment Diagram

The deployment diagram shows the hardware used to run *CivicFix*.

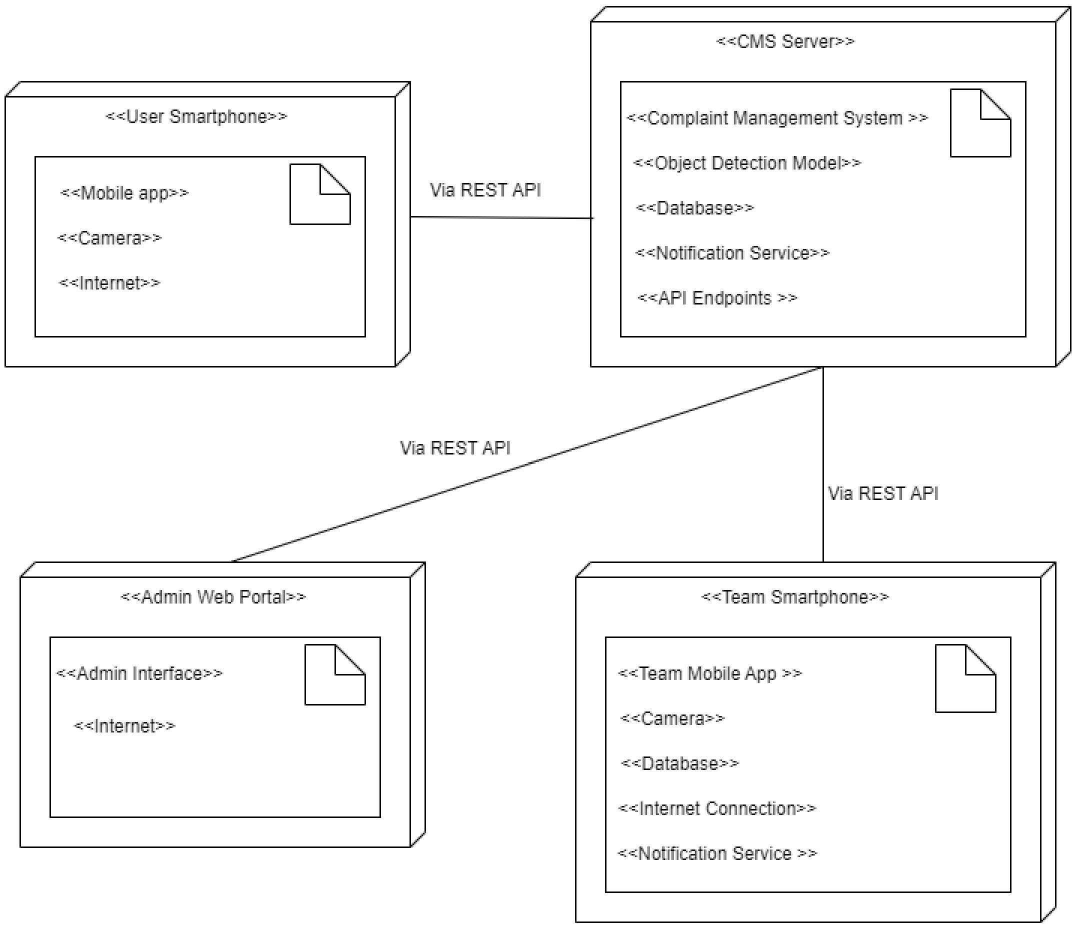


Figure 2.40: Deployment Diagram for CivicFix

2.3 User Interface Design

User interface (UI) design is essential for shaping how users interact with the *CivicFix* app. The design should be intuitive and user-friendly to ensure easy navigation for reporting issues.

2.3.1 Wireframes

Wireframes for *CivicFix* represent the layout and structure of the mobile app interface.

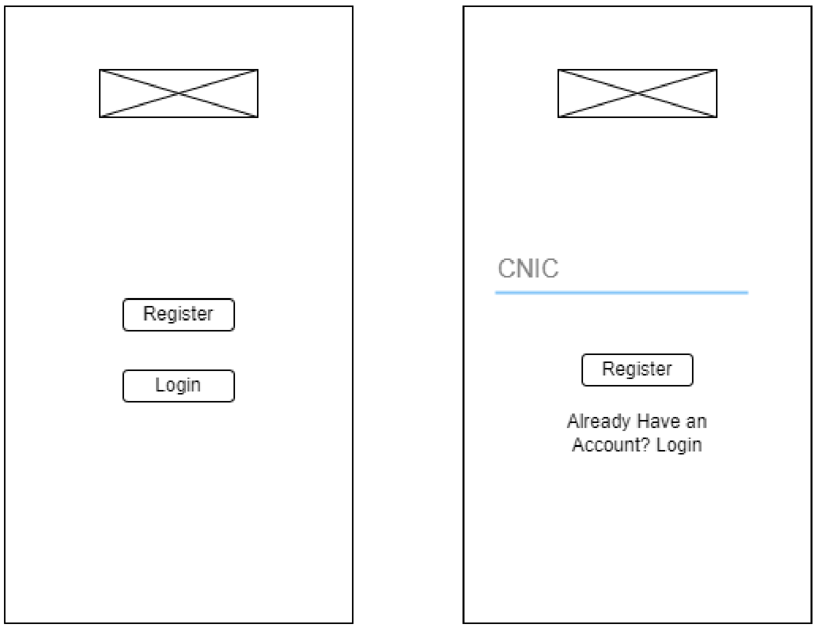


FIGURE 2.41: Android Wireframe User 1 for CivicFix

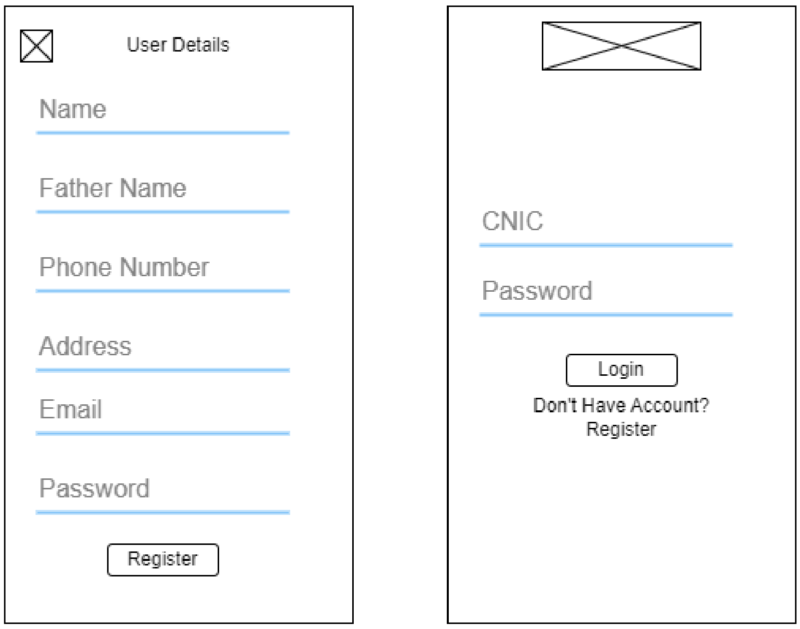


FIGURE 2.42: Android Wireframe User 2 for CivicFix

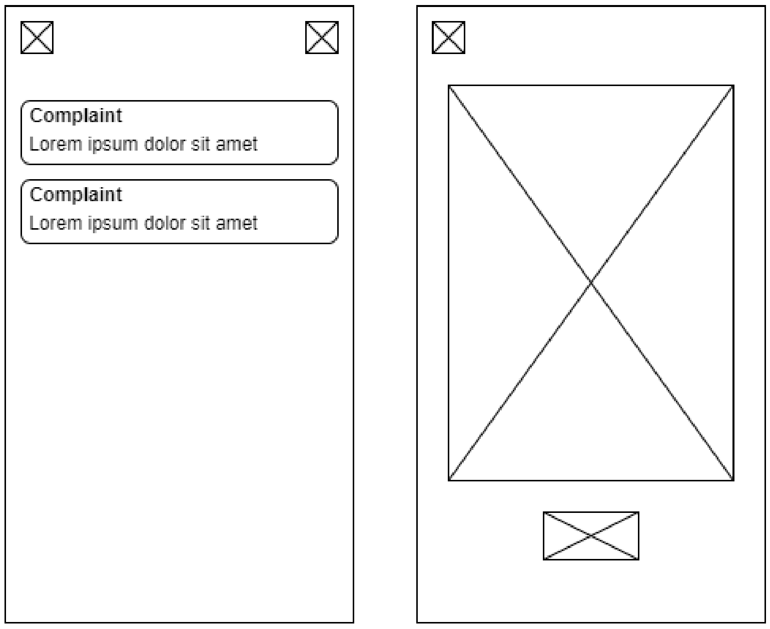


FIGURE 2.43: Android Wireframe User 3 for CivicFix

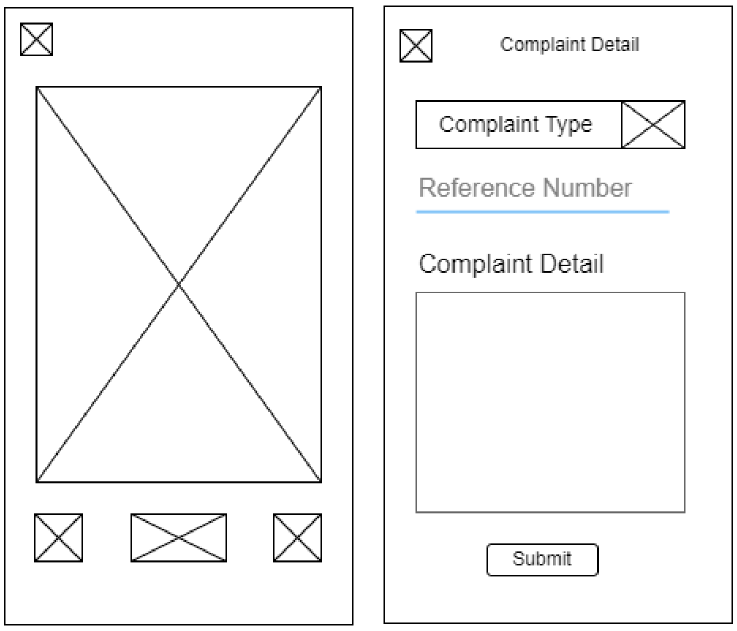


FIGURE 2.44: Android Wireframe User 4 for CivicFix

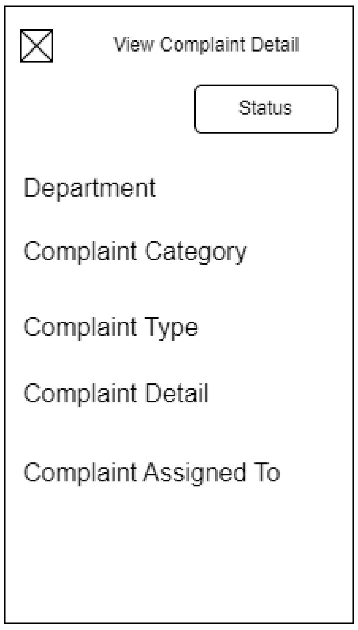


FIGURE 2.45: Android Wireframe User 5 for CivicFix

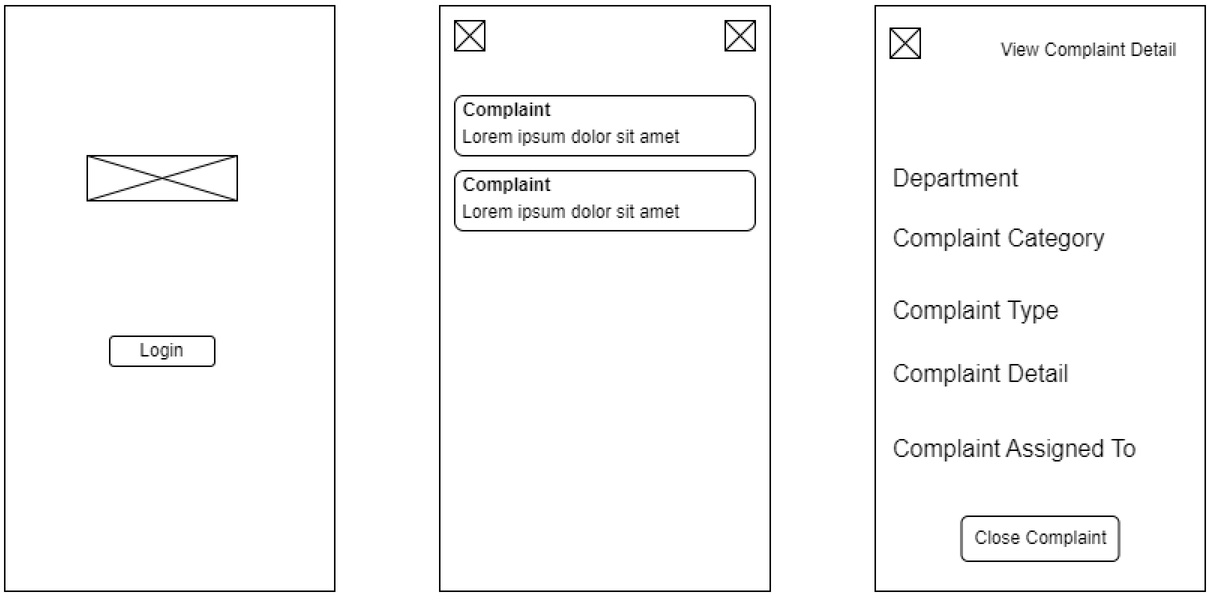


FIGURE 2.46: Android Wireframe Team 1 for CivicFix

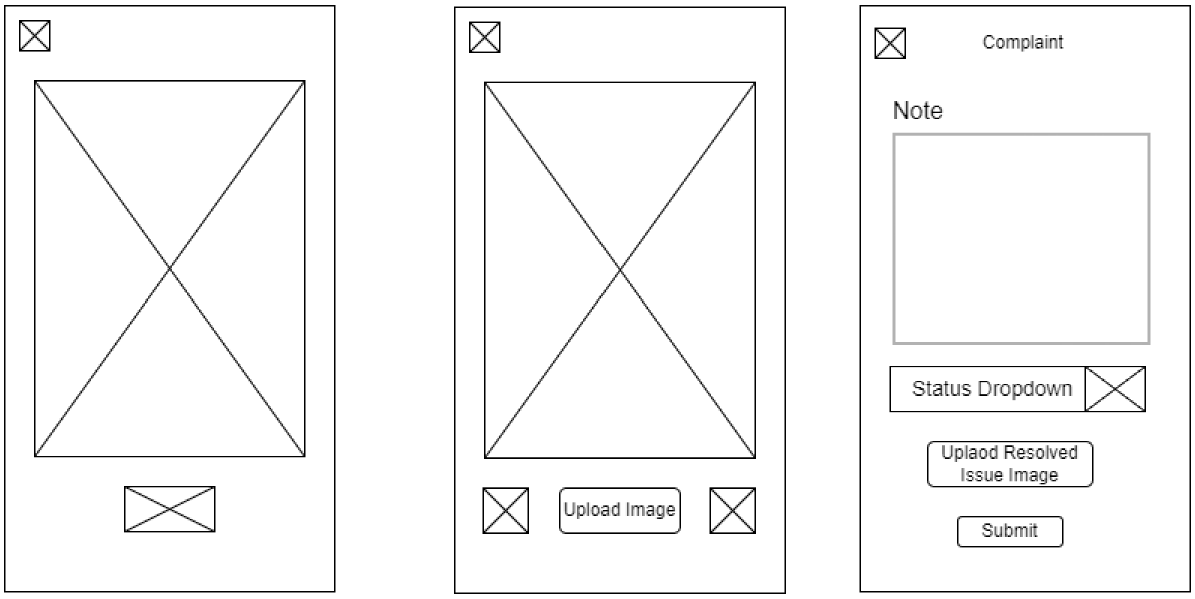


FIGURE 2.47: Android Wireframe Team 2 for CivicFix

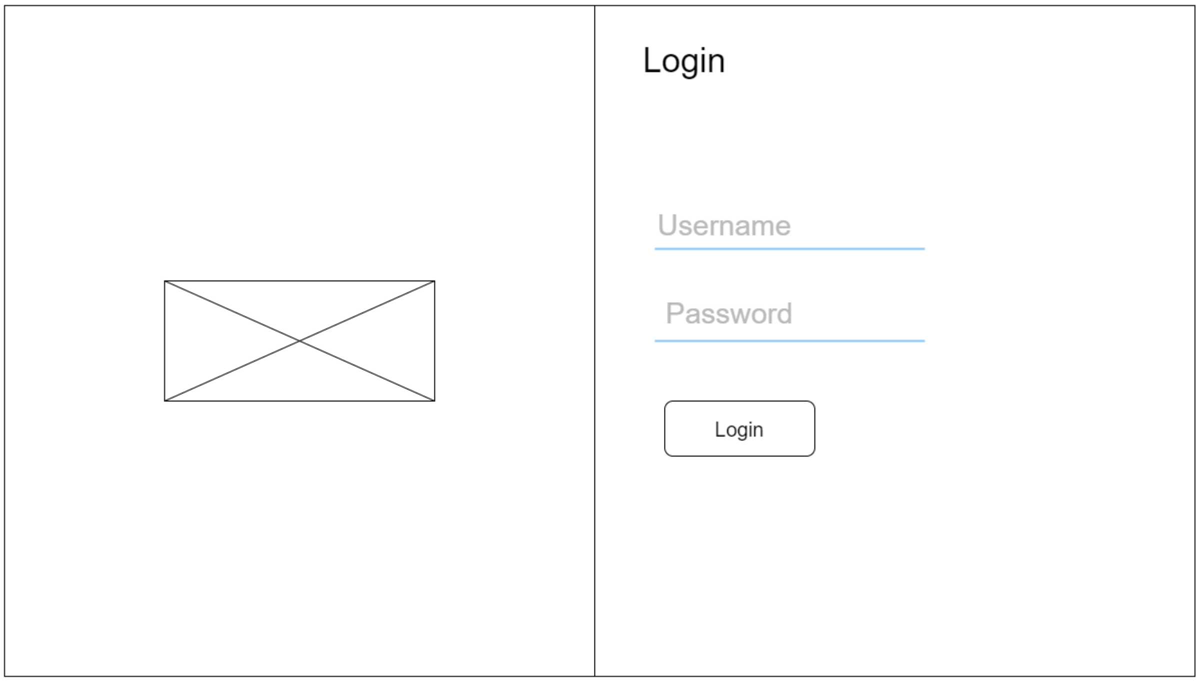


FIGURE 2.48: Web Wireframe Administrator 1 for CivicFix

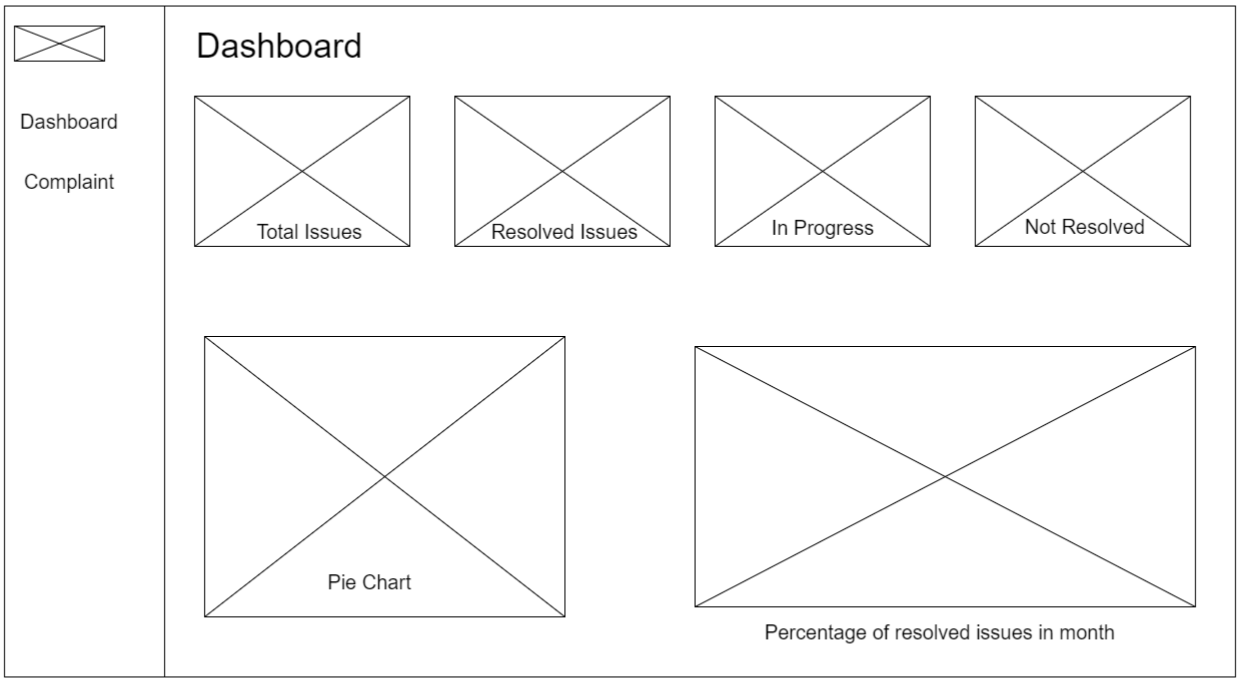


FIGURE 2.49: Web Wireframe Administrator 2 for CivicFix

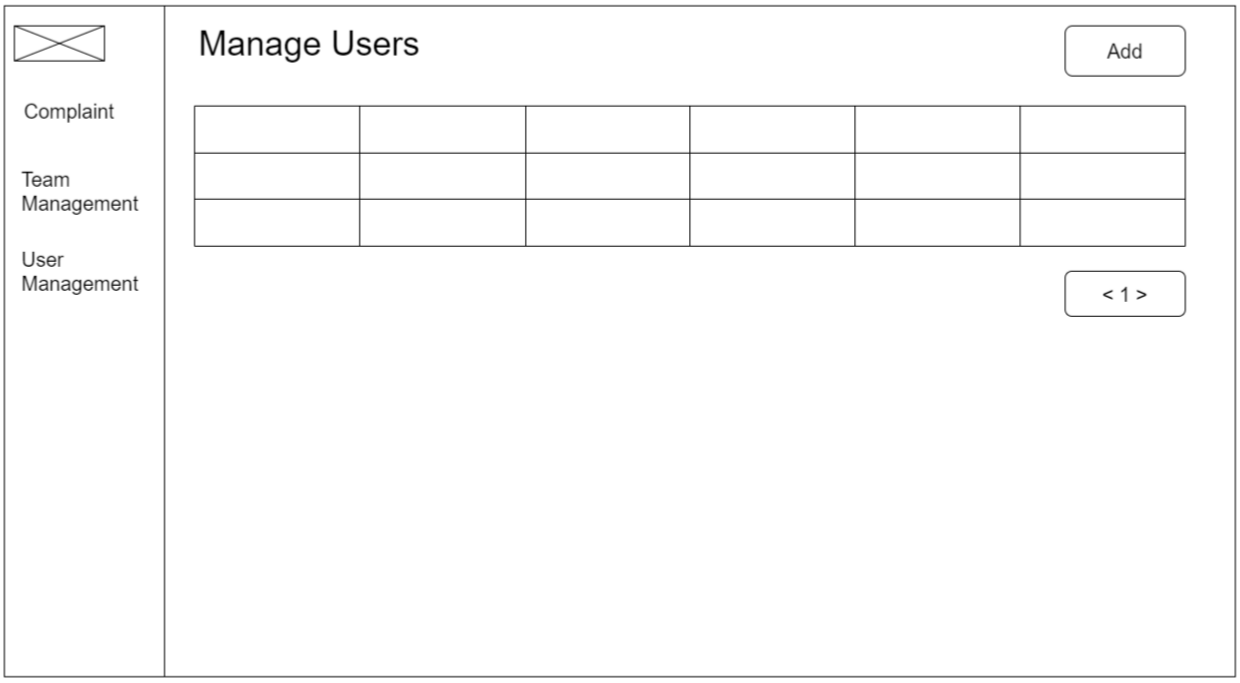


FIGURE 2.50: Web Wireframe Sub Admin 1 for CivicFix

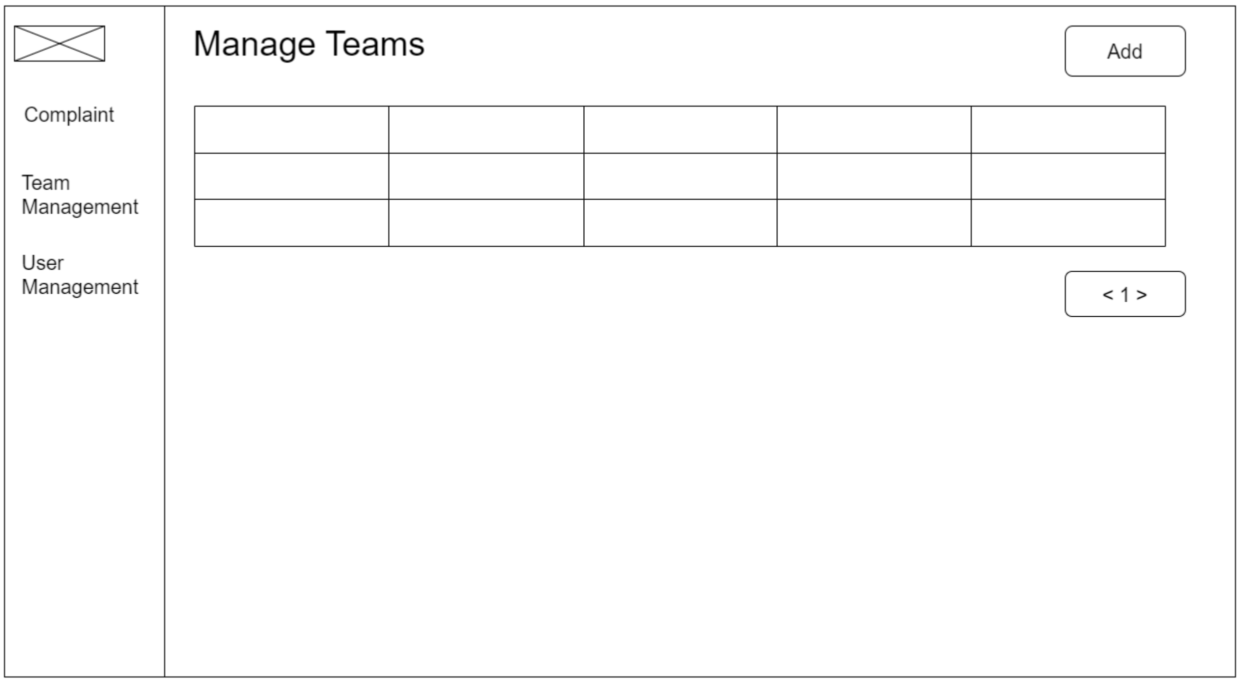


FIGURE 2.51: Web Wireframe Sub Admin 2 for CivicFix

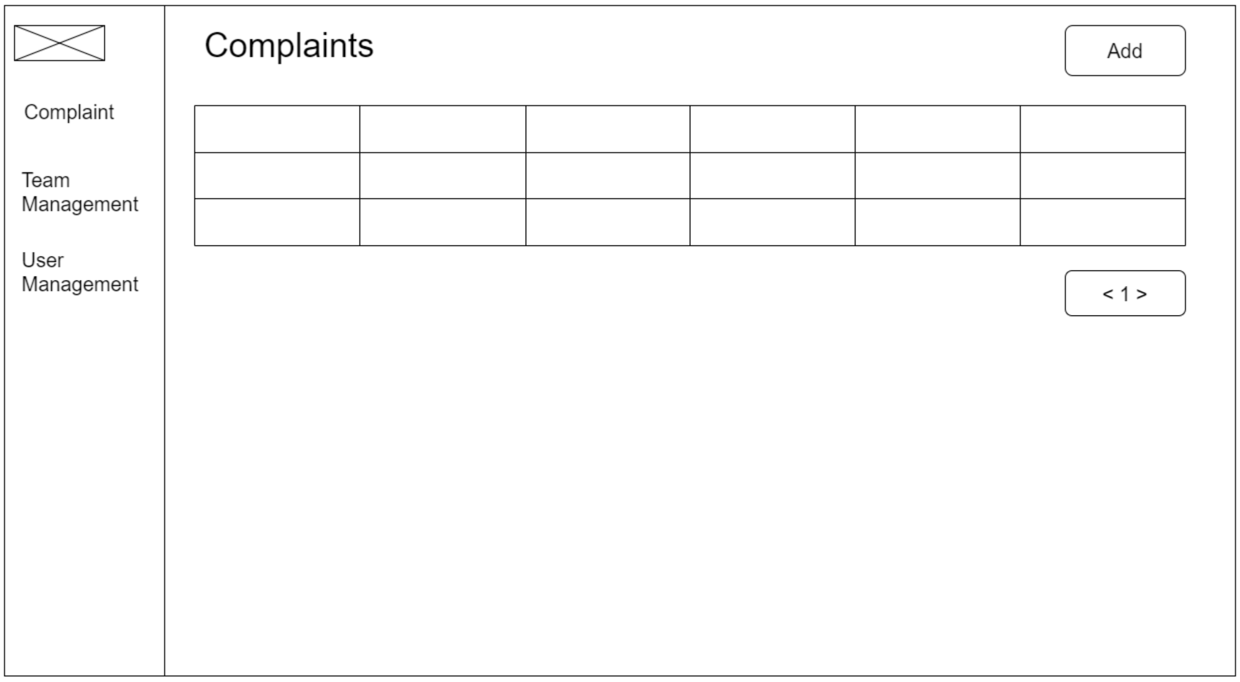


FIGURE 2.52: Web Wireframe Sub Admin 3 for CivicFix

2.4 Database Design

A well-designed database is crucial for ensuring accurate data storage and retrieval in *CivicFix*.

2.4.1 ER Diagram

The ER diagram represents the relationships between key entities in the system.

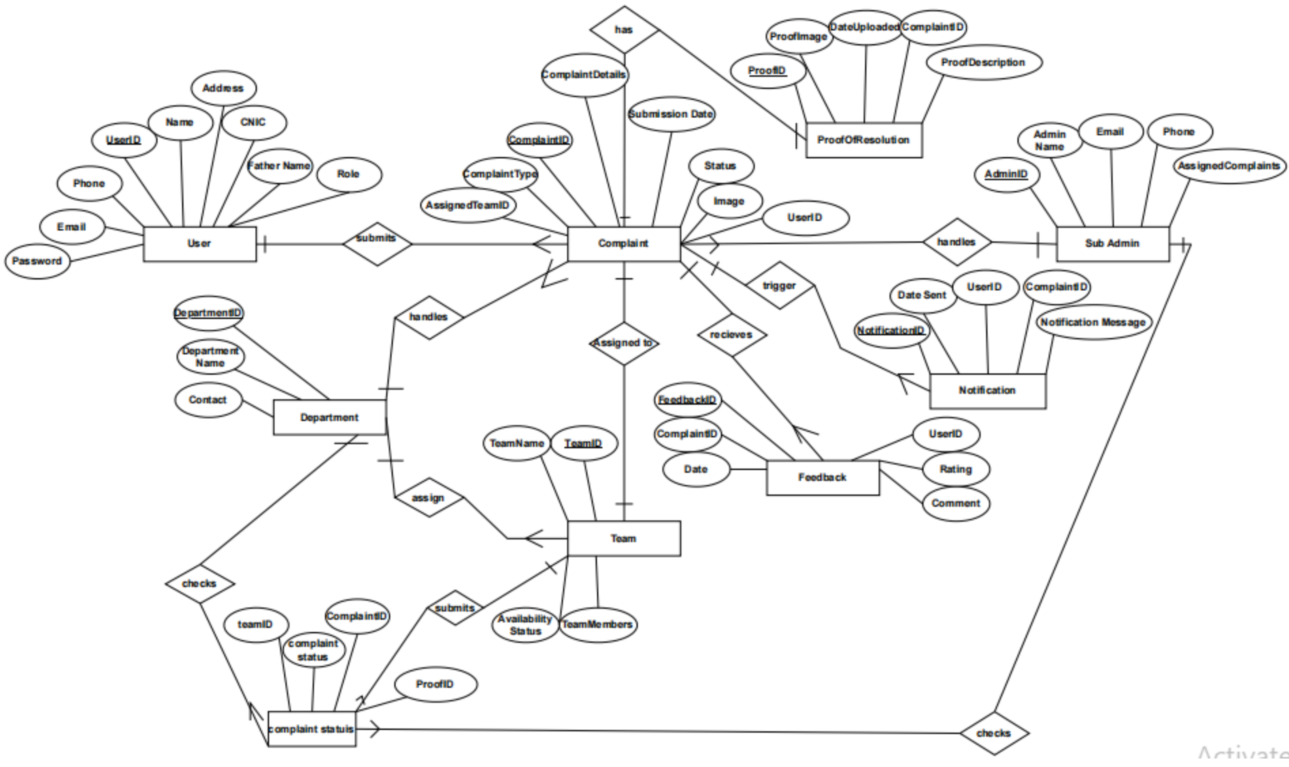


FIGURE 2.54: ER Diagram for CivicFix

References

[1] Draw.io - online diagram software. https://www.draw.io. Accessed: September 26, 2024.

[2] John Doe and Jane Smith. Cityzen: A smart city project for real-time issue reporting. *Journal of Smart City Development*, 2018. URL https://cityzen.com.

[3] Ben Green and David Walker. Seeclickfix: Bridging the gap between citizens and government services. *Urban Technology Journal*, 2015. URL https://seeclickfix.com/.

[4] S. Thomson and J. Maddox. Fixmystreet: A citizen-centric platform for reporting infrastructure issues. *Public Service Review*, 2010. URL https://www.fixmystreet.com/.

[2] [1] [4] [3]