**Final Deliverable**

for

**Smart Complaint Management System**

**Version 1.0 – Approved**

**Submitted as:**

**Deliverable 3 – Final Submission**

**CS3009 – Software Engineering**

**Spring 2025**

**Team Name:** TechNova

**Team Members:**

Adeel Mahmood Ansari – 22i-0979

Muhammad Shaif Imran – 22i-1024

Abdul Wasay – 22i-1198

**Prepared by**

National University of Computer and Emerging Sciences (FAST), Islamabad

**Submission Date:** 29/04/2025

# Introduction

## Purpose

This SRS defines the requirements for the Smart Complaint Management System (SCMS), a web-based platform designed to streamline complaint submission, assignment, and resolution. It aims to assist users in reporting issues, staff in resolving them, and administrators/stakeholders in monitoring performance.

## Intended Audience and Reading Suggestions

This document is intended for:

* **Project Stakeholders**: To understand system objectives and functionality.
* **Developers**: To implement the system according to defined requirements.
* **Quality Assurance (QA) Team**: To design and execute test cases.
* **End Users (Staff & Citizens)**: To understand system capabilities.
* **Instructors and Evaluators**: To assess the project based on requirements.

## Project Scope

SCMS enables users (citizens, employees, customers) to submit and track complaints, automatically assigns them to appropriate departments, and allows staff to manage resolutions. It provides real-time tracking, notifications, and reporting.

## References

* IEEE 830-1998 Standard for Software Requirements Specifications
* CS3009 Deliverable 2 Guidelines (March 6, 2025)

# Overall Description

## Product Perspective

The **Smart Complaint Management System (SCMS)** is a **new, standalone web-based application** designed to streamline the complaint submission, tracking, and resolution process. It is **not a follow-up** to an existing system but an independent solution developed to enhance complaint management in organizations such as government agencies, businesses, and customer service providers.

SCMS follows a **MERN (MongoDB, Express.js, React, Node.js) architecture**:

* **Frontend**: Built with **React.js**, providing an interactive user experience.
* **Backend**: Developed using **Node.js with Express.js**, handling API requests and business logic.
* **Database**: Uses **MongoDB**, a NoSQL database for efficient data storage and retrieval.

## Product Features

* Complaint submission with unique IDs and file uploads
* Automatic assignment to departments
* Real-time status tracking
* User/staff account management
* Notifications and feedback
* Time monitoring and reporting

## User Classes and Characteristics

1. **Users**: Submit and track complaints
2. **Staff**: Resolve assigned complaints
3. **Admins**: Oversee system and generate reports

## Operating Environment

* **Client**: Web browser (Chrome, Firefox, Edge, Safari)
* **Server**: Node.js with Express.js backend
* **Database**: MongoDB
* **Frontend**: React.js

## Design and Implementation Constraints

* Must use the MERN stack (MongoDB, Express.js, React, Node.js)
* RESTful API for communication between frontend and backend
* JWT-based authentication for security
* Deployment on a cloud-based server (e.g., AWS, Heroku, or Firebase)

## Assumptions and Dependencies

* Users have internet access
* Email/SMS services are available for notifications
* MongoDB Atlas or a self-hosted MongoDB instance is accessible
* Node.js and related dependencies are properly configured

# System Features

## FR-01: User Authentication

### Description

*The system shall allow users to register and securely log in using their credentials to access the system.*

### Preconditions

* + *The user must not have an existing account for registration.*
  + *The system must be connected to the database.*
  + *The user must have a verified account to log in.*

### Inputs

* ***Email*** *(must be unique and in a valid format)*
* ***Username*** *(must be unique)*
* ***Password*** *(must meet security requirements)*

### Processing Logic

#### Registration Process:

* *Validate the email format.*
* *Check if the email and username are already registered.*
* *Verify that the password meets security requirements.*
* *Encrypt the password before storing it.*
* *Store user details securely in the database.*
* *Send a verification email with an activation link.*

#### Login Process:

* *Receive login credentials from the user.*
* *Validate the email format.*
* *Check if the email exists in the system.*
* *Verify the password against the stored hashed password.*
* *Check if the account is verified and not locked.*
* *Grant access if authentication is successful; otherwise, display an error.*
* *Log the login attempt for security tracking.*

### Outputs

* ***Success Message:***
* *"Registration successful."*
* *“Login Successful. Redirecting to your dashboard”*
* ***Error Messages:***
* *"Email is already in use."*
* *"Invalid email format."*
* *"Username is already taken."*
* *"Password does not meet security requirements."*

### Postconditions

* *The user’s data is securely stored in the database upon successful registration.*
* *The user gains access to their dashboard if login authentication is successful.*

### Acceptance Criteria

* *The system must* ***reject duplicate emails and usernames****.*
* *The system must* ***encrypt passwords*** *before storing them.*

### Exceptions and Constraints

* *If the* ***email is already registered****, the system shall display an* ***error message****.*
* *If the* ***password is too weak****, the system shall reject the registration request.*

## **FR-02: Complaint Registration**

### **Description**

*The system shall allow users to submit complaints by providing details such as category, description, and optional attachments.*

### **Preconditions**

* *The user must be logged into the system.*
* *The system must be connected to the database.*

### **Inputs**

* *Complaint Category (e.g., Technical, Billing, Service).*
* *Complaint Description (mandatory).*
* *Optional Attachments (e.g., images, documents).*

### **Processing Logic**

1. *Validate that all mandatory fields are filled.*
2. *Assign a unique complaint ID to the submission.*
3. *Store complaint details securely in the database.*
4. *Automatically assign the complaint to the relevant department.*
5. *Send a confirmation notification to the user.*

### **Outputs**

* ***Success Message:*** *"Complaint submitted successfully. Your Complaint ID is [XXXX]."*
* ***Error Messages:***
  + *"Complaint description cannot be empty."*
  + *"Invalid file format. Attachments must be in JPG, PNG, or PDF format."*

### **Postconditions**

* *The complaint is stored in the database with a unique ID.*
* *The complaint is assigned to the relevant department for review.*
* *The user receives a confirmation notification.*

### **Acceptance Criteria**

* *The system must generate a unique ID for each complaint.*
* *The system must allow users to attach supporting files.*
* *The system must notify the user upon successful complaint registration.*

### **Exceptions and Constraints**

* *If the system fails to store the complaint, an error message shall be displayed.*
* *If the user uploads an unsupported file type, the system shall reject the attachment.*
* *If the system cannot assign the complaint to a department, it shall flag the complaint for manual assignment.*

## **FR-03: Complaint Status Tracking**

### **Description**

*The system shall allow users to track the status of their submitted complaints using a complaint ID or by viewing their complaint history.*

### **Preconditions**

* *The user must have submitted at least one complaint.*
* *The complaint must be stored in the system with a valid complaint ID.*
* *The system must be connected to the database.*

### **Inputs**

* *Complaint ID (entered manually by the user) OR*
* *User account (to view complaint history).*

### **Processing Logic**

1. *Validate the provided complaint ID or retrieve complaints associated with the user’s account.*
2. *Fetch the current status of the complaint from the database.*
3. *Display the complaint status to the user (e.g., "Received," "In Progress," "Resolved").*

### **Outputs**

* ***Success Message:*** *Display the current complaint status (e.g., "Your complaint is currently being processed and is marked as 'In Progress'.").*
* ***Error Messages:***
  + *"Invalid Complaint ID. Please check and try again."*
  + *"No complaints found for your account."*

### **Postconditions**

* *The user is shown the most recent status of their complaint.*
* *The complaint tracking information remains updated in the system.*

### **Acceptance Criteria**

* *The system must allow users to check complaint statuses using either a complaint ID or their account.*
* *The complaint status must be displayed in real-time.*
* *If a complaint ID is invalid, the system must provide an appropriate error message.*

### **Exceptions and Constraints**

* *If the complaint ID does not exist, the system shall notify the user.*
* *If the system is temporarily unable to fetch complaint details, it shall retry up to three times before displaying an error.*
* *Users should only be able to view the statuses of complaints they have submitted.*

## **FR-04: User Profile Management**

### **Description**

*The system shall allow users to manage and update their profile information, including personal details and account credentials.*

### **Preconditions**

* *The user must be logged into the system.*
* *The system must be connected to the database.*

### **Inputs**

* *Updated user details, such as:* 
  + *Email (must be unique and in a valid format)*
  + *Phone number (optional, must be in a valid format)*
  + *Password (must meet security requirements)*
  + *Profile picture (optional, supported file formats: PNG, JPG)*

### **Processing Logic**

1. *Validate the new email format (if changed).*
2. *Check if the new email is already registered.*
3. *Validate the phone number format (if provided).*
4. *Verify that the new password meets security requirements (if changed).*
5. *Encrypt the new password before storing it in the database.*
6. *Update the user’s profile details in the database.*

### **Outputs­**

* ***Success Message:*** *"Profile updated successfully."*
* ***Error Messages:***
  + *"Invalid email format."*
  + *"Email is already in use."*
  + *"Invalid pho​​​ne number format."*
  + *"Password does not meet security requirements."*

### **Postconditions**

* *The updated user profile information is stored securely in the database.*
* *If the email is changed, a confirmation email is sent to verify the new address.*
* *The user can log in with the updated credentials.*

### **Acceptance Criteria**

* *The system must allow users to update their profile information.*
* *The system must validate new credentials before saving them.*
* *If an email change is made, the system must require verification before finalizing the update.*

### **Exceptions and Constraints**

* *If the new email is already registered, the system shall display an error message.*
* *If the password is too weak, the system shall reject the update.*
* *If a profile picture is uploaded, the system shall allow only supported file formats (PNG, JPG) and enforce a maximum file size limit.*

## **FR-05: Automatic Complaint Assignment**

### **Description**

*The system shall automatically assign complaints to the appropriate department based on predefined rules and complaint categories.*

### **Preconditions**

* *A complaint must be successfully registered in the system.*
* *The system must have predefined rules for assigning complaints to departments.*

### **Inputs**

* *Complaint details, including category and description.*
* *List of predefined rules for complaint assignment.*
* *List of departments and their responsibilities.*

### **Processing Logic**

1. *Retrieve the complaint details and category.*
2. *Match the complaint category with the predefined rules.*
3. *Identify the responsible department.*
4. *Assign the complaint to the appropriate department.*
5. *Notify the department staff about the new complaint.*

### **Outputs**

* ***Success Message:*** *"Your complaint has been assigned to [Department Name]."*
* ***Error Messages:***
  + *"Unable to assign complaint due to missing category information."*
  + *"No matching department found for this complaint category."*

### **Postconditions**

* *The complaint is successfully assigned to the appropriate department.*
* *The assigned department receives a notification about the new complaint.*

### **Acceptance Criteria**

* *The system must automatically assign complaints based on predefined rules.*
* *Complaints must be routed to the correct department without manual intervention.*
* *Department staff must receive notifications about newly assigned complaints.*

### **Exceptions and Constraints**

* *If no predefined rule matches the complaint category, the system shall flag the complaint for manual assignment.*
* *If the department assignment fails, the system shall log the issue and notify an administrator.*
* *The system must support adding or modifying predefined rules for complaint assignment.*

## **FR-06: Staff Complaint Viewing**

### **Description**

*The system shall allow department staff to view a list of complaints assigned to their department through a dashboard or complaint management interface.*

### **Preconditions**

* *The staff member must have a valid account and be logged in.*
* *The staff member must belong to a department responsible for handling complaints.*
* *Complaints must be assigned to the department.*

### **Inputs**

* *Staff login credentials.*
* *Department assignment data.*
* *List of assigned complaints.*

### **Processing Logic**

1. *Authenticate the staff member and retrieve their department information.*
2. *Fetch all complaints assigned to the staff member’s department.*
3. *Display the list of complaints along with relevant details (e.g., complaint ID, category, status, timestamp).*
4. *Allow sorting and filtering of complaints based on status, priority, and submission date.*

### **Outputs**

* ***Success:*** *Display a list of assigned complaints.*
* ***Error Messages:***
  + *"No complaints assigned to your department at this time."*
  + *"Error retrieving complaint data. Please try again later."*

### **Postconditions**

* *The staff member can view complaints assigned to their department.*
* *The system updates the list dynamically when new complaints are assigned.*

### **Acceptance Criteria**

* *The system must display only complaints relevant to the logged-in staff member’s department.*
* *The complaint list must be sortable and filterable.*
* *The staff member must not be able to view complaints from other departments.*

### **Exceptions and Constraints**

* *If the staff member is not assigned to a department, the system shall deny access to the complaint list.*
* *If the database connection fails, the system shall display an error message.*
* *The system must support a large number of complaints and provide efficient searching/filtering.*

## **FR-07: Staff Complaint Response**

### **Description**

*The system shall allow department staff to respond to complaints by updating their status (e.g., "In Progress," "Resolved") and adding comments or resolution details.*

### **Preconditions**

* *The staff member must have a valid account and be logged in.*
* *The staff member must be assigned to the department handling the complaint.*
* *The complaint must exist in the system and be assigned to the staff member’s department.*

#### **Inputs**

* *Complaint ID.*
* *Status update (e.g., "In Progress," "Resolved").*
* *Comments or resolution details.*

#### **Processing Logic**

1. *Authenticate the staff member and verify department access.*
2. *Retrieve the complaint details.*
3. *Allow the staff member to update the complaint status.*
4. *Allow the staff member to add comments or resolution notes.*
5. *Save the updated status and response in the database.*
6. *Notify the user about the status update via email or SMS.*

### **Outputs**

* ***Success:*** *"Complaint status updated successfully."*
* ***Error Messages:***
  + *"You do not have permission to update this complaint."*
  + *"Complaint not found."*
  + *"Error updating complaint. Please try again later."*

### **Postconditions**

* *The complaint status is updated in the system.*
* *The user receives a notification about the update.*
* *The complaint history logs the changes made by the staff.*

### **Acceptance Criteria**

* *Only authorized staff members can update complaints assigned to their department.*
* *The system must log all status updates and comments.*
* *Users must be notified of complaint status changes.*

### **Exceptions and Constraints**

* *If the staff member does not have permission, the system shall deny the update request.*
* *If the database connection fails, the system shall display an error message and retry the operation.*
* *If a complaint is already marked as "Resolved," further updates should require administrative approval.*

## **FR-08: Staff Account Creation and Login**

### **Description**

*The system shall allow department staff to create an account and log in to manage complaints assigned to their department.*

### **Preconditions**

* *The staff member must not already have an account.*
* *The system must be connected to the database.*
* *Only authorized administrators can create staff accounts.*

### **Inputs**

*For Account Creation:*

* *Full Name*
* *Email (must be unique and in a valid format)*
* *Username (must be unique)*
* *Password (must meet security requirements)*
* *Assigned Department*

*For Login:*

* *Email or Username*
* *Password*

### **Processing Logic**

#### **Account Creation**

1. *Validate email format (e.g., staff@example.com).*
2. *Check if the email and username are already registered.*
3. *Verify that the password meets security requirements (minimum 8 characters, at least one uppercase letter, and one number).*
4. *Encrypt the password before storing it in the database.*
5. *Assign the staff member to a department.*
6. *Store staff details securely in the database.*
7. *Send a confirmation email with login credentials.*

#### **Login**

1. *Validate the email/username and password combination.*
2. *Authenticate the staff member against stored credentials.*
3. *Grant access to the staff dashboard upon successful authentication.*
4. *Log the login attempt in the system.*

### **Outputs**

* ***Success Messages:***
  + *"Account successfully created. Login details have been sent to your email."*
  + *"Login successful. Redirecting to the staff dashboard."*
* ***Error Messages:***
  + *"Email is already registered."*
  + *"Invalid email format."*
  + *"Username is already taken."*
  + *"Password does not meet security requirements."*
  + *"Invalid login credentials."*

### **Postconditions**

* *The staff account is securely stored in the system.*
* *The staff member receives a confirmation email with login details.*
* *Successful logins grant access to the staff dashboard.*
* *Failed login attempts are recorded for security purposes.*

### **Acceptance Criteria**

* *The system must reject duplicate staff accounts based on email and username.*
* *The system must encrypt and store passwords securely.*
* *Only authorized administrators can create staff accounts.*
* *Staff members must receive confirmation emails upon successful account creation.*
* *Staff members must be able to log in using their credentials.*

### **Exceptions and Constraints**

* *If the email is already registered, the system shall prevent account creation.*
* *If the password is too weak, the system shall reject the registration request.*
* *If the confirmation email fails to send, the system shall retry up to three times before notifying the administrator.*
* *If a staff member enters incorrect login credentials three consecutive times, the system shall temporarily lock the account for security reasons.*

## **FR-09: Complaint Resolution Time Monitoring**

### **Description**

*The system shall track the time taken to resolve complaints and flag overdue cases for administrative review.*

### **Preconditions**

* *The complaint must be registered in the system.*
* *The complaint must be assigned to a department.*
* *The system must have predefined resolution time thresholds based on complaint categories.*

### **Inputs**

* *Complaint ID*
* *Complaint Category*
* *Assigned Department*
* *Complaint Submission Time*
* *Complaint Status Updates*

### **Processing Logic**

1. *Capture the timestamp when the complaint is submitted.*
2. *Determine the expected resolution time based on the complaint category.*
3. *Track the time elapsed since complaint submission.*
4. *Update the system when the complaint status changes (e.g., "In Progress," "Resolved").*
5. *If the complaint exceeds the predefined resolution time, flag it as overdue.*
6. *Notify administrators and department staff about overdue complaints.*
7. *Generate reports on resolution times for analysis.*

### **Outputs**

* ***Success Messages:***
  + *"Complaint resolution time updated successfully."*
* ***Error Messages:***
  + *"Unable to retrieve complaint details."*
  + *"System error while tracking complaint time."*
* ***Alerts & Notifications:***
  + *"Complaint [Complaint ID] has exceeded the resolution time and requires urgent attention."*
  + *"Complaint resolved within the expected time."*

### **Postconditions**

* *The system logs and updates the resolution time for each complaint.*
* *Overdue complaints are flagged and highlighted for review.*
* *Administrators and staff receive notifications for overdue complaints.*

### **Acceptance Criteria**

* *The system must correctly calculate the elapsed resolution time for each complaint.*
* *The system must flag overdue complaints based on predefined thresholds.*
* *Administrators must receive alerts for overdue complaints.*
* *Reports on complaint resolution trends must be available.*

### **Exceptions and Constraints**

* *If a complaint is not assigned to a department, resolution time tracking shall not begin.*
* *If the system fails to calculate resolution time due to a technical issue, an error message shall be displayed, and an administrator must be notified.*
* *The system shall allow authorized personnel to adjust resolution thresholds if necessary.*

## **FR-10: Stakeholder Complaint Status Overview**

### **Description**

*The system shall allow stakeholders (e.g., administrators, department heads) to view the status of all complaints within the system to monitor progress and identify bottlenecks.*

### **Preconditions**

* *The stakeholder must have an authorized account with appropriate permissions.*
* *Complaints must be registered in the system.*

### **Inputs**

* *Stakeholder login credentials*
* *Filters (e.g., department, complaint category, status, date range)*

### **Processing Logic**

1. *Authenticate the stakeholder's login credentials.*
2. *Retrieve all complaints based on user-defined filters (e.g., pending, resolved, overdue).*
3. *Display complaint details such as Complaint ID, category, assigned department, status, and resolution time.*
4. *Allow stakeholders to sort and filter complaints by various attributes.*
5. *Generate a summary view of complaint trends and overall system performance.*

### **Outputs**

* ***Success Messages:***
  + *"Complaint status overview retrieved successfully."*
* ***Error Messages:***
  + *"Insufficient permissions to access complaint data."*
  + *"No complaints found matching the selected filters."*
* ***Displayed Information:***
  + *Complaint ID, category, submission date, assigned department, current status, resolution time.*

### **Postconditions**

* *The stakeholder successfully views the status of complaints based on the applied filters.*
* *The system generates summary reports for monitoring complaint resolution trends.*

### **Acceptance Criteria**

* *The system must retrieve and display complaint details within 3 seconds.*
* *The system must allow filtering by department, status, and date range.*
* *Stakeholders must be able to generate summary reports of complaint trends.*

### **Exceptions and Constraints**

* *If the stakeholder has insufficient permissions, access shall be denied with an error message.*
* *If no complaints match the selected filters, the system shall display a "No complaints found" message.*
* *If there is a system failure while fetching complaint data, an appropriate error message shall be displayed, and an administrator shall be notified.*

## **FR-11: Complaint Reports Generation**

### **Description**

*The system shall allow administrators and authorized stakeholders to generate reports on complaint trends, resolution times, and department performance for analysis and decision-making.*

### **Preconditions**

* *The user must be an authorized administrator or stakeholder.*
* *Complaints must exist in the system for reporting.*

### **Inputs**

* *User login credentials*
* *Report parameters (e.g., date range, complaint category, department, status)*

### **Processing Logic**

1. *Authenticate the user's credentials and verify permissions.*
2. *Retrieve complaint data based on selected report parameters.*
3. *Process data to generate summary statistics (e.g., number of complaints per category, average resolution time).*
4. *Format the data into a structured report (e.g., table, chart, downloadable file).*
5. *Allow users to export reports in multiple formats (PDF, CSV, Excel).*

### **Outputs**

* ***Success Messages:***
  + *"Report generated successfully."*
* ***Error Messages:***
  + *"No data available for the selected report parameters."*
  + *"Insufficient permissions to generate reports."*
* ***Displayed Information:***
  + *Complaint volume by category and department*
  + *Resolution times and overdue complaints*
  + *Trend analysis of complaints over time*

### **Postconditions**

* *The generated report is displayed and available for download.*
* *The system logs the report generation for audit purposes.*

### **Acceptance Criteria**

* *The system must generate reports within 5 seconds.*
* *Reports must be exportable in at least three formats (PDF, CSV, Excel).*
* *Users must be able to filter reports based on date range, category, and department.*

### **Exceptions and Constraints**

* *If the user has insufficient permissions, access shall be denied with an appropriate message.*
* *If no data matches the selected parameters, the system shall notify the user with a "No data available" message.*
* *If report generation fails due to a system error, an administrator shall be notified.*

## **FR-12: Notification System**

### **Description**

*The system shall send notifications to users and staff regarding complaint status updates, new complaint assignments, and important actions required. Notifications can be sent via email or SMS.*

### **Preconditions**

* *The user must have a registered account with a valid email or phone number.*
* *The complaint must exist in the system.*
* *The system must be connected to the email/SMS service provider.*

### **Inputs**

* *Complaint status updates (e.g., "In Progress," "Resolved")*
* *New complaint assignments for staff*
* *User preference for notification type (email/SMS)*

### **Processing Logic**

1. *Detect complaint status changes or new assignments.*
2. *Retrieve user or staff contact details from the database.*
3. *Format the notification message based on the update type.*
4. *Send the notification via the selected method (email/SMS).*
5. *Log the notification delivery status.*

### **Outputs**

* ***Success Messages:***
  + *"Notification sent successfully."*
* ***Error Messages:***
  + *"Failed to send notification. Please try again."*
  + *"User has not enabled notifications."*
* ***Displayed Information:***
  + *Complaint status update message*
  + *Complaint ID and details in notifications*
  + *Time and date of notification*

### **Postconditions**

* *The user receives a notification about complaint status updates.*
* *The assigned staff member is notified of new complaints.*
* *The system logs the notification for tracking and auditing.*

### **Acceptance Criteria**

* *Notifications must be sent within 5 seconds of a status change.*
* *Users must be able to choose between email and SMS notifications.*
* *The system must retry notifications up to 3 times in case of failure.*

### **Exceptions and Constraints**

* *If the email/SMS service is unavailable, the system shall retry and log the failure.*
* *If the user has opted out of notifications, no messages shall be sent.*
* *If the contact details are missing or invalid, an error shall be logged.*

## **FR-13: Complaint Feedback**

### **Description**

*The system shall allow users to provide feedback on the resolution of their complaints by submitting ratings and comments.*

### **Preconditions**

* *The complaint must have a "Resolved" status.*
* *The user must have a registered account and be logged in.*

### **Inputs**

* *Complaint ID*
* *Rating (e.g., 1 to 5 stars)*
* *Comments (optional)*

### **Processing Logic**

1. *Validate that the complaint exists and is marked as "Resolved."*
2. *Accept user input for rating and optional comments.*
3. *Store the feedback securely in the database.*
4. *Update the complaint record to indicate that feedback has been provided.*
5. *Generate reports for administrators on feedback trends.*

### **Outputs**

* ***Success Messages:***
  + *"Thank you for your feedback!"*
* ***Error Messages:***
  + *"You can only provide feedback for resolved complaints."*
  + *"Invalid rating value. Please select a rating between 1 and 5."*

### **Postconditions**

* *The user's feedback is stored in the system.*
* *The complaint record is updated with the feedback.*
* *The feedback is available for administrators in reports.*

### **Acceptance Criteria**

* *Users can only submit feedback for complaints marked as "Resolved."*
* *Feedback must include a rating, with comments being optional.*
* *Administrators can access reports summarizing user feedback.*

### **Exceptions and Constraints**

* *If the user attempts to provide feedback on an unresolved complaint, an error message shall be displayed.*
* *If the database connection fails, the system shall retry storing feedback up to three times before displaying an error.*
* *Feedback submissions shall be limited to one per complaint.*

## **FR-14: Search/Filter Complaints**

### **Description**

*The system shall allow users to search or filter their past complaints by date, category, or status to quickly find specific cases without manually scrolling through the entire history.*

### **Preconditions**

* *The user must be logged in.*
* *The user must have at least one complaint recorder in their account .*

### **Inputs**

* *S****earch Query*** *(optional) – A keyword or phrase to match against complaint details.*
* ***Date Range*** *(optional) – Start and end dates to filter complaints.*
* ***Category*** *(optional) – Complaint category selection.*
* **Status** (optional) – Complaint status selection (e.g., Open, In Progress, Resolved).

### **Processing Logic**

1. *Validate that the user is logged in.*
2. *Retrieve the user’s complaint records from the database.*
3. *If a search query is provided, filter complaints based on matching keywords.*
4. *If filters (date range, category, or status) are applied, refine the results accordingly.*
5. *Display the filtered complaint list to the user.*

### **Outputs**

* ***Filtered Complaint List*** *– A dynamically updated list showing only complaints matching the search criteria.*
* ***Success Messages:***
  + *"Complaints filtered successfully."*
* ***Error Messages:***
  + *"No complaints found matching your criteria."*
  + *"Invalid date range. Please select a valid start and end date."*

### **Postconditions**

* *The system displays only complaints that match the applied search or filter criteria.*

### **Acceptance Criteria**

* *Users can filter complaints by date range, category, and status individually or in combination.*
* *If no complaints match the filter, an appropriate message is displayed.*
* *The system returns results within two seconds for typical complaint records.*

### **Exceptions and Constraints**

* *If the database connection fails, the system shall retry retrieving data up to three times before displaying an error.*
* *Filters shall not apply to complaints deleted or archived by the user.*
* *The system shall support at least 10,000 complaints per user without performance degradation.*

## **FR-15: Appeal Complaint Resolution**

### **Description**

*The system shall allow users to appeal a resolved complaint if they are unsatisfied with the outcome, triggering a re-evaluation by a supervisor.*

### **Preconditions**

* *The user must be logged in.*
* *The complaint must have a status of* ***"Resolved"*** *but not* ***"Closed"****.*

### **Inputs**

* ***Complaint ID*** *– The unique identifier of the complaint being appealed.*
* ***Appeal Reason*** *– A required text field where the user provides justification for the appeal.*
* ***Supporting Documents (optional)*** *– Users may upload files (e.g., PDFs, images) as evidence.*

### **Processing Logic**

1. *Validate that the user is logged in.*
2. *Verify that the complaint is marked as* ***"Resolved"*** *but not* ***"Closed"****.*
3. *Accept user input for the* ***appeal reason*** *and optional supporting documents.*
4. *Change the complaint status to* ***"Under Review"****.*
5. *Store the appeal request and any attached documents in the database.*
6. *Notify a supervisor about the appeal request for further action.*

### **Outputs**

* ***Success Messages:***
  + "Your appeal has been submitted successfully. A supervisor will review it soon."
* ***Error Messages:***
  + "You can only appeal complaints that are marked as 'Resolved' and not 'Closed'."
  + "Appeal reason is required. Please provide a valid justification."
  + "File upload failed. Please try again with a valid document format (PDF, JPG, PNG)."

### **Postconditions**

* *The complaint status is updated to* ***"Under Review"****.*
* *The appeal reason and any supporting documents are recorded in the system.*
* *A supervisor is notified to review the appeal.*

### **Acceptance Criteria**

* *Users can only appeal complaints that are* ***"Resolved"*** *but not* ***"Closed"****.*
* *Users must provide an appeal reason to proceed.*
* *The system must notify a supervisor upon receiving an appeal.*

### **Exceptions and Constraints**

* *If the database connection fails, the system shall retry storing the appeal up to three times before displaying an error.*
* *Users cannot submit multiple appeals for the same complaint.*
* *Supporting documents must not exceed* ***10MB*** *in total size.*

## **FR-16: Pick Complaint to Resolve**

### **Description**

*The system shall allow staff members to select a complaint from the list of available complaints assigned to their department, enabling them to take ownership of cases based on their expertise or workload.*

### **Preconditions**

* *The staff member must be logged in.*
* *The staff member must belong to a department.*
* *Complaints must be assigned to the staff member’s department but not yet assigned to an individual.*
* *Complaints must have a status that allows assignment (e.g.,* ***"Unassigned"*** *or* ***"Pending"****).*

### **Inputs**

* ***Complaint ID*** *– The unique identifier of the complaint selected by the staff member.*

### **Processing Logic**

1. *Validate that the staff member is logged in.*
2. *Verify that the complaint is assigned to the staff member’s department but not yet assigned to an individual.*
3. *Verify that the complaint is in a valid state for assignment (****"Unassigned"*** *or* ***"Pending"****).*
4. *Assign the complaint to the staff member and update the status to* ***"In Progress"****.*
5. *Store the assignment timestamp in the database.*
6. *Link the staff member’s name to the complaint for accountability.*
7. *Remove the complaint from the available list for other staff members.*
8. *Notify the user who submitted the complaint about the assignment.*

### **Outputs**

* ***Success Messages:***
  + "Complaint successfully assigned to you and marked as 'In Progress'."
* ***Error Messages:***
  + "This complaint has already been assigned to another staff member."
  + "You can only pick complaints assigned to your department."
  + "Invalid complaint selection. Please try again."

### **Postconditions**

* *The complaint status is updated to* ***"In Progress"****.*
* *The staff member’s name is linked to the complaint record.*
* *The complaint is removed from the available list for other staff members.*
* *The complaint submitter is notified about the assignment.*

### **Acceptance Criteria**

* *Only staff members belonging to the complaint’s assigned department can select it.*
* *Once assigned, the complaint is removed from the available list.*
* *The system correctly updates the complaint’s status to* ***"In Progress"*** *and records the assignment timestamp.*
* *The original complaint submitter receives a notification upon assignment.*

### **Exceptions and Constraints**

* *If the complaint is already assigned to another staff member before selection, an error is displayed.*
* *If the database connection fails, the system shall retry updating the complaint status up to three times before displaying an error.*
* *Complaints marked as* ***"Closed"*** *cannot be selected.*

## **FR-17: Prioritize Urgent Complaints**

### **Description**

*The system shall allow administrators to mark complaints as* ***"High Priority"*** *based on severity (e.g., safety issues) to ensure critical cases are addressed first.*

### **Preconditions**

* *The* ***admin*** *must be logged in.*
* *The complaint must exist in the system.*

### **Inputs**

* ***Complaint ID*** *– The unique identifier of the complaint to be prioritized.*
* ***Priority Level*** *– Admin selects* ***"High Priority"*** *to flag urgent complaints.*
* ***Reason for Prioritization (optional)*** *– Admin can provide a justification for marking the complaint as urgent.*

### **Processing Logic**

1. *Validate that the admin is logged in.*
2. *Verify that the complaint exists in the system.*
3. *Allow the admin to mark the complaint as* ***"High Priority"****.*
4. *Store the updated priority status in the database.*
5. *Ensure the priority status is reflected in dashboards and complaint lists.*
6. *Notify relevant staff members about the urgent complaint.*

### **Outputs**

* ***Success Messages:***
  + "Complaint successfully marked as High Priority."
* ***Error Messages:***
  + "You must be an admin to set complaint priority."
  + "Invalid complaint ID. Please select a valid complaint."

### **Postconditions**

* *The complaint is flagged as* ***"High Priority"*** *in the system.*
* *The priority status is reflected in dashboards and complaint lists.*
* *Relevant staff members are notified about the urgent complaint.*

### **Acceptance Criteria**

* *Only admins can mark complaints as* ***"High Priority"****.*
* *The updated priority status appears in the admin and staff dashboards.*
* *Staff members receive notifications about urgent complaints.*

### **Exceptions and Constraints**

* *If the database update fails, the system shall retry saving the priority status up to three times before displaying an error.*
* *Admins cannot mark* ***"Closed"*** *complaints as high priority.*
* *The number of high-priority complaints may be limited to prevent abuse of the feature.*

## **FR-18: Assign Complaint to Specific Staff Member**

### **Description**

*The system shall allow an admin to manually assign a specific complaint to a selected staff member within their department to ensure fair workload distribution and proper handling of complex cases.*

### **Preconditions**

* *The* ***admin*** *must be logged in with department-level permissions.*
* *The* ***staff member*** *must be registered in the system and belong to the admin’s department.*
* *The* ***complaint*** *must be unassigned.*

### **Inputs**

* ***Complaint ID*** *– The unique identifier of the complaint to be assigned.*
* ***Staff Member ID*** *– The unique identifier of the staff member who will handle the complaint.*

### **Processing Logic**

1. *Validate that the admin is logged in with department-level permissions.*
2. *Verify that the selected complaint exists and is currently unassigned.*
3. *Check that the selected staff member is registered and belongs to the admin’s department.*
4. *Assign the complaint to the selected staff member.*
5. *Update the complaint status to* ***"Assigned"****.*
6. *Store the assignment details in the database, including the timestamp and assigned staff member.*
7. *Notify the assigned staff member about the complaint assignment.*
8. *Remove the complaint from the list of available complaints for other staff members.*

### **Outputs**

* ***Success Messages:***
  + "Complaint successfully assigned to [Staff Member Name]."
* ***Error Messages:***
  + "Invalid complaint ID. Please select a valid unassigned complaint."
  + "Selected staff member does not belong to your department."
  + "Complaint has already been assigned to another staff member."

### **Postconditions**

* *The* ***complaint is assigned*** *to the selected staff member.*
* *The* ***complaint status changes to "Assigned"****.*
* *The* ***staff member is notified*** *of the assignment.*
* *The* ***complaint is no longer available*** *for other staff members to pick.*

### **Acceptance Criteria**

* *Only* ***admins*** *with department-level permissions can assign complaints.*
* *The* ***selected staff member must belong*** *to the admin’s department.*
* *Once assigned, the complaint’s* ***status updates to "Assigned"*** *and is no longer available to other staff.*
* *The assigned staff member receives a* ***notification*** *of the new complaint.*

### **Exceptions and Constraints**

* *If the complaint is already assigned, the system shall prevent reassignment unless explicitly changed by an admin.*
* *If the database update fails, the system shall retry storing the assignment up to three times before displaying an error.*
* *Complaints that are marked as* ***"Closed"*** *cannot be reassigned.*

Below is a functional requirement for the scenario where an Admin verifies a Staff member of their respective department, formatted as requested:

## FR-19: Verify Staff Member by Admin

### Description

### *The system shall allow an admin to verify a registered staff member within their department to ensure that only authorized personnel can access and handle complaints, maintaining departmental security and accountability.*

### Preconditions

* *The admin must be logged in with department-level permissions and verified.*
* *The staff member must be registered in the system with a pending verification status (is\_verified: false, verification\_status: 'Pending') and belong to the admin’s department*
* *The system must be connected to the database.*

### Inputs

* *Staff Member ID – The unique identifier of the staff member to be verified (\_id from Staff collection).*
* *Admin ID – The unique identifier of the admin performing the verification (derived from JWT token).*

### Processing Logic

1. *Validate that the admin is logged in with department-level permissions (via JWT token).*
2. *Verify that the selected staff member exists in the system and has a pending verification status.*
3. *Check that the staff member belongs to the admin’s department (compare Staff.department\_id with Admin.department\_id).*
4. *Update the staff member’s verification status to is\_verified: true and verification\_status: 'Verified'.*
5. *Set the verified\_by field of the staff member to the admin’s ID.*
6. *Store the updated staff member details in the database, including the timestamp of the update (updated\_at).*
7. *Notify the staff member about their successful verification (e.g., via email or system notification).*
8. *Grant the staff member access to complaint-handling features.*

### Outputs

• ***Success Messages****:*

* *"Staff member [Staff Member Name] successfully verified."*

*•* ***Error Messages****:*

* *"Invalid staff member ID. Please select a valid staff member."*
* *"Staff member does not belong to your department."*
* *"Staff member is already verified."*

### Postconditions

* *The staff member’s is\_verified status is updated to true, and verification\_status is set to 'Verified'.*
* *The verified\_by field reflects the admin who performed the verification.*
* *The staff member is notified of their verification.*

### Acceptance Criteria

* *Only admins with department-level permissions can verify staff members.*
* *The selected staff member must belong to the admin’s department.*
* *Once verified, the staff member’s status updates to 'Verified', and they can log in and access complaint-related features.*
* *The staff member receives a notification upon successful verification.*

### Exceptions and Constraints

* *If the staff member is already verified (is\_verified: true), the system shall prevent re-verification and display an error.*
* *If the database update fails, the system shall retry storing the verification details up to three times before displaying an error.*
* *Staff members from other departments cannot be verified by an admin outside their department.*

# Other Nonfunctional Requirements

## Performance

*The system shall be able to handle up to 1000 complaints per day.*

## Security

*The system shall restrict access to authorized users through authentication mechanisms.*

## Scale ability

*The system shall support multiple departments without performance degradation.*

## Availability

*The system shall have an uptime of at least 99.9%.*

# Use Case Diagram

A diagram of a diagram

AI-generated content may be incorrect.

# User Stories

## User Authentication

*As a* ***User****, I want to register an account and log in using my credentials so that I can access the system to submit and track complaints.*

* ***Pre-conditions:*** *The user must not have an existing account to register; The system must be connected to the database.*
* ***Post-conditions:*** *Upon successful registration, the user’s data is stored securely; Upon successful login, the user gains access to their dashboard.*

## Register Complaint

*As a* ***User****, I want to submit a complaint with details like category,, description, photos so I can report issues and get them resolved.*

* ***Pre-conditions****: User is logged in; system is connected to the database.*
* ***Post-conditions:*** *Complaint is stored with a unique ID; assigned to a department; user receives a confirmation notification.*

## Track Complaint Status

*As a* ***User****, I want to track the status of my complaints using a complaint ID or history so I know the progress.*

* ***Pre-conditions:*** *User has submitted at least one complaint; complaint has a valid ID; system is connected to the database.*
* ***Post-conditions:*** *User views the current status; system updates tracking information in real-time.*

## Manage User Profile

*As a* ***User****, I want to update my profile information so it remains up-to-date and secure.*

* ***Pre-conditions:*** *User is logged in; system is connected to the database.*
* ***Post-conditions:*** *Updated profile information is stored; if email changes, a verification email is sent.*

## Search/Filter Complaints

*As a* ***User****, I want to search or filter my past complaints by date, category, or status so I can quickly find specific cases without scrolling through my entire history.*

* + ***Pre-conditions:*** *User is logged in; complaints exist in their account.*
  + ***Post-conditions:*** *Filtered results are displayed*

## Appeal Complaint Resolution

*As a* ***User****, I want to appeal a resolved complaint if I’m unsatisfied with the outcome so that it can be re-evaluated by a supervisor.*

* + ***Pre-conditions:*** *Complaint is marked "Resolved" but not "Closed"; user is logged in.*
  + ***Post-conditions:*** *Complaint status reverts to "Under Review"; escalation notes are added; supervisor is alerted.*

## Submit Feedback

*As a* ***User****, I want to provide feedback on resolved complaints so I can express satisfaction and help improve the system.*

* ***Pre-conditions:*** *Complaint is marked as "Resolved"; user is logged in.*
* ***Post-conditions:*** *Feedback is stored; complaint record is updated; feedback is available for reports.*

## View Assigned Complaints

*As a* ***Staff member****, I want to view a list of complaints assigned to my department through a dashboard so I can manage and respond to them.*

* ***Pre-conditions:*** *Staff is logged in and assigned to a department; complaints are assigned.*
* ***Post-conditions:*** *Staff views a list of assigned complaints; list updates dynamically with new assignments.*

## Update Complaint Status

*As a* ***Staff member****, I want to update the status of complaints and add comments so I can resolve them effectively.*

* ***Pre-conditions:*** *Staff is logged in and authorized; complaint is assigned to their department.*
* ***Post-conditions:*** *Complaint status is updated; user is notified of the change.*

## Register Staff

*As a* ***Staff member****, I want to create and log in to an account so I can manage department complaints.*

* ***Pre-conditions:*** *Staff does not have an account; system is connected to the database; admin authorization is required.*
* ***Post-conditions:*** *Staff account is created and stored; login credentials are emailed.*

## Monitor Complaint

*As a* ***Staff member****, I want to monitor complaint progress in real-time so I can proactively manage my workload.*

* ***Pre-conditions:*** *Staff is logged in; complaints are assigned to their department.*
* ***Post-conditions:*** *Staff views real-time complaint progress; system flags overdue cases.*

## Pick Complaint to Resolve

*As a* ***Staff Member,*** *I want to select a complaint from a list of available complaints assigned to my department so that I can proactively choose cases based on my expertise or current workload.*

* + ***Pre-conditions:*** *Staff member is logged in and belongs to a department; Complaints are assigned to the staff member’s department but not yet assigned to an individual; Complaints are in a state that allows assignment (e.g., “Unassigned” or “Pending”).*
  + ***Post-conditions:*** *Staff member selects a complaint from the list; Complaint status changes to “In Progress” with a timestamp; Staff member’s name is linked to the complaint for accountability; The complaint is removed from the available list for other staff members; User who submitted the complaint is notified of the assignment.*

## View System-wide Complaints

*As an* ***Admin****, I want to view the status of all complaints across the system so I can monitor progress and identify bottlenecks.*

* ***Pre-conditions:*** *Admin is logged in with appropriate permissions.*
* ***Post-conditions:*** *Admin views all complaint statuses; system provides filtered views*

## Generate Complaint Reports

*As an* ***Admin****, I want to generate reports on complaint trends and performance so I can analyze and improve efficiency.*

* ***Pre-conditions:*** *Admin is logged in; complaints exist in the system.*
* ***Post-conditions:*** *Report is generated and stored; available for download in multiple formats*

## Prioritize Urgent Complaints

*As a* ***Admin****, I want to mark complaints as “High Priority” based on severity (e.g., safety issues) so that my team can address critical cases first.*

* + ***Pre-conditions:*** *Admin is logged in*
  + ***Post-conditions:*** *Complaint is flagged as urgent; priority is reflected in dashboards.*

## Assign Complaint to Specific Staff Member

*As an* ***Admin****, I want to manually assign a specific complaint to a selected staff member within my department so that workloads are distributed fairly and complex cases are handled by the most qualified personnel.*

* + ***Pre-conditions:*** *Admin is logged in with department-level permissions; Staff member is registered and belongs to the admin’s department; Complaint is unassigned*
  + ***Post-conditions:*** *Complaint is assigned to the selected staff member; Complaint status changes to “Assigned”*

## Send Notifications

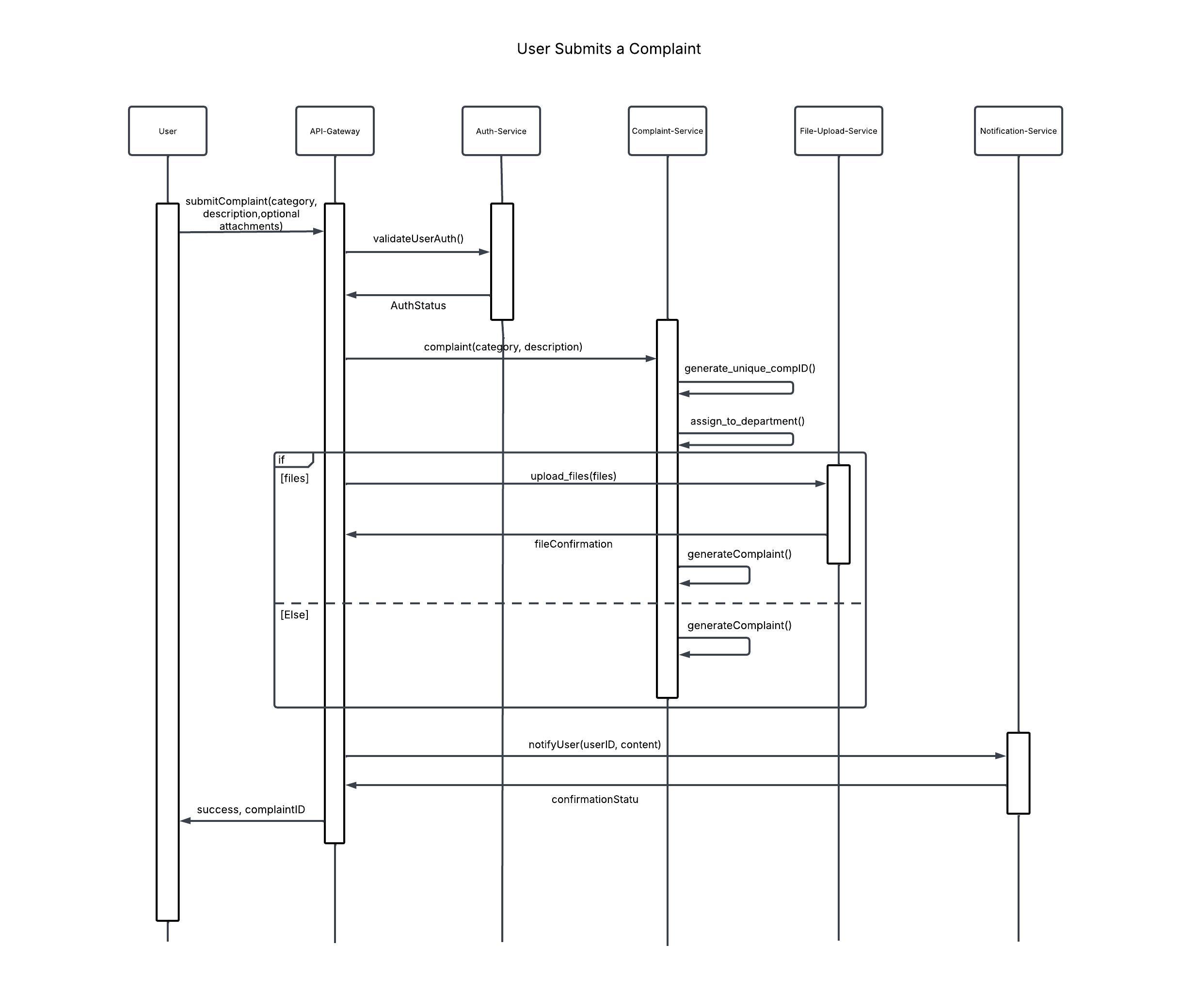
*As an* ***Admin****, I want to send notifications to users and staff about updates and actions so they can stay informed and manage tasks.*

* ***Pre-conditions:*** *User or staff has a valid contact method; system is connected to email service.*
* ***Post-conditions:*** *Notification is sent and logged; recipient is informed of updates.*

## Staff Verification by Admin

*As an Admin, I want to verify a Staff member of my department after their registration so that only authorized personnel can access and handle complaints within my department.*  
• ***Pre-conditions****: The Admin must be logged in and verified; The Staff member must be registered with a pending verification status and assigned to the Admin’s department; The system must be connected to the database.  
•* ***Post-conditions****: Upon successful verification, the Staff member’s is\_verified status is updated to true, verification\_status is set to 'Verified', and verified\_by is set to the Admin’s ID; The Staff member gains access to their dashboard and complaint-handling features.*

# Sequence Diagrams



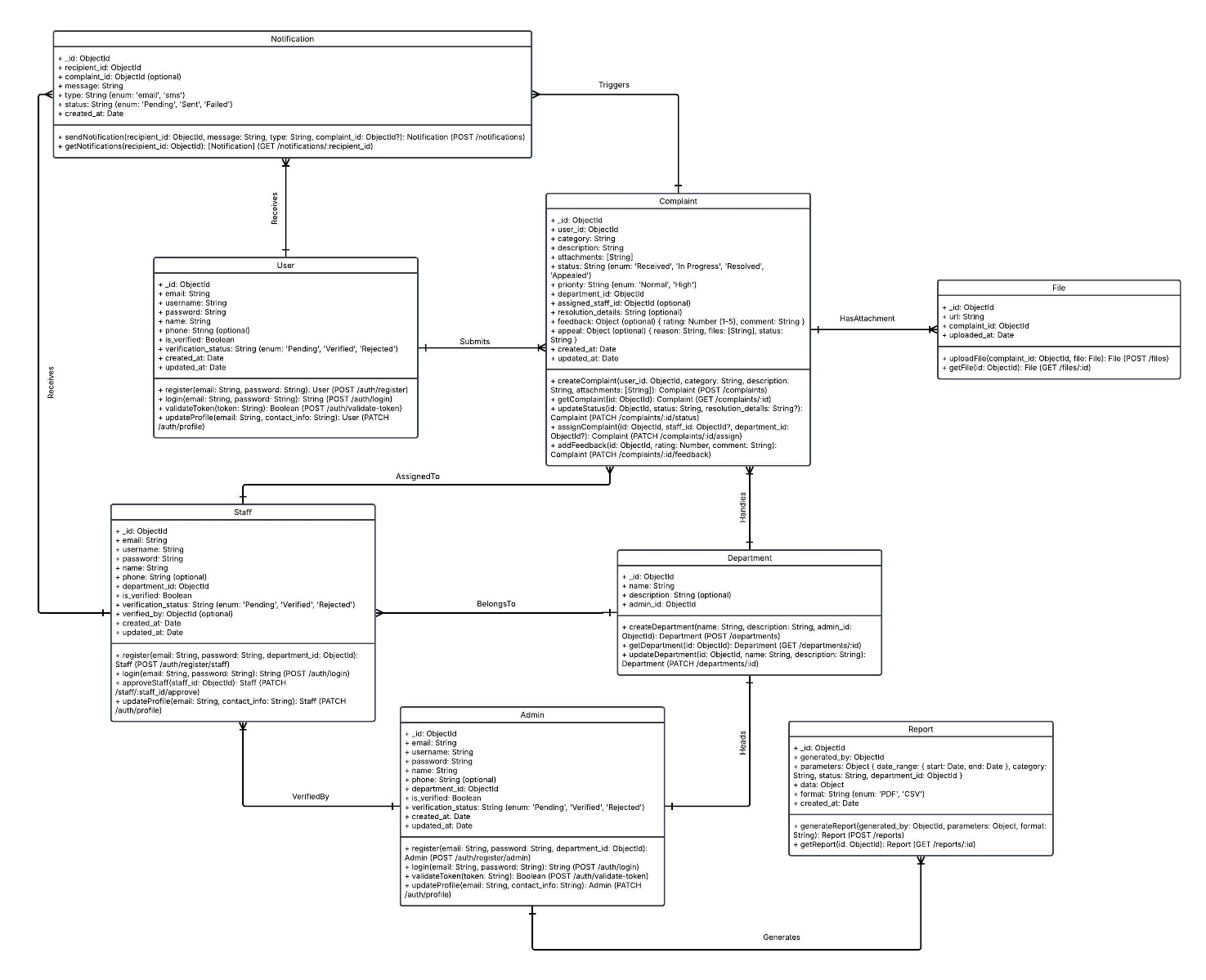
A diagram of a process flow

AI-generated content may be incorrect.

A diagram of a company

AI-generated content may be incorrect.

# Class Diagram



# Product Backlog

| ID | User Story | Priority |
| --- | --- | --- |
| **1.1** | As a User, I want to register an account and log in using my credentials so that I can access the system to submit and track complaints. | High |
| **1.2** | As a User, I want to submit a complaint with details like category, description, photos so I can report issues and get them resolved. | High |
| **1.3** | As a User, I want to track the status of my complaints using a complaint ID or history so I know the progress. | High |
| **1.4** | As a User, I want to update my profile information so it remains up-to-date and secure. | Medium |
| **1.5** | As a User, I want to search or filter my past complaints by date, category, or status so I can quickly find specific cases without scrolling through my entire history. | Medium |
| **1.6** | As a User, I want to appeal a resolved complaint if I’m unsatisfied with the outcome so that it can be re-evaluated by a supervisor. | Low |
| **1.7** | As a User, I want to provide feedback on resolved complaints so I can express satisfaction and help improve the system. | Low |
| **1.8** | As a Staff member, I want to view a list of complaints assigned to my department through a dashboard so I can manage and respond to them. | High |
| **1.9** | As a Staff member, I want to update the status of complaints and add comments so I can resolve them effectively. | High |
| **1.10** | As a Staff member, I want to create and log in to an account so I can manage department complaints. | High |
| **1.11** | As a Staff member, I want to monitor complaint progress in real-time so I can proactively manage my workload. | Medium |
| **1.12** | As a Staff Member, I want to select a complaint from a list of available complaints assigned to my department so that I can proactively choose cases based on my expertise or current workload. | Medium |
| **1.13** | As an Admin, I want to view the status of all complaints across the system so I can monitor progress and identify bottlenecks. | High |
| **1.14** | As an Admin, I want to generate reports on complaint trends and performance so I can analyze and improve efficiency. | Medium |
| **1.15** | As an Admin, I want to mark complaints as “High Priority” based on severity (e.g., safety issues) so that my team can address critical cases first. | High |
| **1.16** | As an Admin, I want to manually assign a specific complaint to a selected staff member within my department so that workloads are distributed fairly and complex cases are handled by the most qualified personnel. | Medium |
| **1.17** | As an Admin, I want to send notifications to users and staff about updates and actions so they can stay informed and manage tasks. | High |
| **1.18** | As an Admin, I want to verify a Staff member of my department after their registration so that only authorized personnel can access and handle complaints within my department. | High |

# Spring Backlog

**Sprint 1: User Features**

**User Story 1.1: User Registration and Login**

**Description**: As a User, I want to register an account and log in using my credentials so that I can access the system to submit and track complaints.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.1-T1 | Create Registration Form (Frontend) | Develop a React form for user registration with email, username, and password fields. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.1-T2 | Create Login Form (Frontend) | Develop a React form for user login with email/username and password fields. | Adeel Mahmood Ansari | 2 hours | To-Do |
| SCMS-1.1-T3 | Set Up Authentication API (Backend) | Implement Express routes for registration and login with JWT authentication, storing data in MongoDB. | Muhammad Shaif Imran | 4 hours | To-Do |
| SCMS-1.1-T4 | Integrate Frontend with Backend | Connect React forms to Express APIs using Axios for registration and login. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.1-T5 | Testing and Bug Fixing | Test registration and login flows, ensure error handling (e.g., duplicate email), and fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.2: Submit a Complaint**

**Description**: As a User, I want to submit a complaint with details like category, description, photos so I can report issues and get them resolved.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.2-T1 | Create Complaint Submission Form (Frontend) | Develop a React form with category dropdown, description field, and file upload for photos. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.2-T2 | Set Up Complaint Submission API (Backend) | Implement Express route to save complaint details and files to MongoDB, auto-assign department. | Muhammad Shaif Imran | 4 hours | To-Do |
| SCMS-1.2-T3 | Integrate Frontend with Backend | Connect React form to Express API using Axios, handle file uploads. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.2-T4 | Testing and Bug Fixing | Test complaint submission, validate mandatory fields, ensure file upload works, and fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.3: Track Complaint Status**

**Description**: As a User, I want to track the status of my complaints using a complaint ID or history so I know the progress.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.3-T1 | Create Complaint Tracking Page (Frontend) | Develop a React page to input complaint ID or view complaint history. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.3-T2 | Set Up Status Retrieval API (Backend) | Implement Express route to fetch complaint status from MongoDB based on ID or user. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.3-T3 | Integrate Frontend with Backend | Connect React page to Express API using Axios, display status. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.3-T4 | Testing and Bug Fixing | Test status tracking with valid/invalid IDs, ensure user-specific access, and fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.4: Update Profile Information**

**Description**: As a User, I want to update my profile information so it remains up-to-date and secure.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.4-T1 | Create Profile Edit Page (Frontend) | Develop a React page with fields to update email, phone, and password. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.4-T2 | Set Up Profile Update API (Backend) | Implement Express route to update user profile in MongoDB, validate inputs. | Muhammad Shaif Imran | 3 hours | To-Do |
| SCMS-1.4-T3 | Integrate Frontend with Backend | Connect React page to Express API using Axios, handle updates. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.4-T4 | Testing and Bug Fixing | Test profile updates, ensure validation (e.g., unique email), and fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.5: Search or Filter Past Complaints**

**Description**: As a User, I want to search or filter my past complaints by date, category, or status so I can quickly find specific cases.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.5-T1 | Create Search/Filter Interface (Frontend) | Develop a React interface with search bar and filter options (date, category, status). | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.5-T2 | Set Up Filtering API (Backend) | Implement Express route to filter complaints in MongoDB based on user criteria. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.5-T3 | Integrate Frontend with Backend | Connect React interface to Express API using Axios, display filtered results. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.5-T4 | Testing and Bug Fixing | Test filtering by various criteria, ensure user-specific results, and fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.6: Appeal a Resolved Complaint**

**Description**: As a User, I want to appeal a resolved complaint if I’m unsatisfied with the outcome so that it can be re-evaluated.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.6-T1 | Create Appeal Form (Frontend) | Develop a React form to submit appeal reason and optional files. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.6-T2 | Set Up Appeal API (Backend) | Implement Express route to handle appeal requests, update status in MongoDB. | Muhammad Shaif Imran | 3 hours | To-Do |
| SCMS-1.6-T3 | Integrate Frontend with Backend | Connect React form to Express API using Axios, handle notifications. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.6-T4 | Testing and Bug Fixing | Test appeal submission, ensure status updates and notifications work, fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.7: Provide Feedback on Resolved Complaints**

**Description**: As a User, I want to provide feedback on resolved complaints so I can express satisfaction and help improve the system.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.7-T1 | Create Feedback Form (Frontend) | Develop a React form with rating and comment fields for resolved complaints. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.7-T2 | Set Up Feedback API (Backend) | Implement Express route to store feedback in MongoDB. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.7-T3 | Integrate Frontend with Backend | Connect React form to Express API using Axios, display confirmation. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.7-T4 | Testing and Bug Fixing | Test feedback submission, ensure it links to resolved complaints, fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**Sprint 2: Staff Features**

**User Story 1.8: View Assigned Complaints**

**Description**: As a Staff member, I want to view a list of complaints assigned to my department through a dashboard so I can manage and respond to them.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.8-T1 | Create Staff Dashboard (Frontend) | Develop a React dashboard to display department-assigned complaints. | Adeel Mahmood Ansari | 4 hours | To-Do |
| SCMS-1.8-T2 | Set Up Complaint Fetch API (Backend) | Implement Express route to fetch complaints by department from MongoDB. | Muhammad Shaif Imran | 3 hours | To-Do |
| SCMS-1.8-T3 | Integrate Frontend with Backend | Connect React dashboard to Express API using Axios, display list. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.8-T4 | Testing and Bug Fixing | Test dashboard, ensure department-specific access, fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.9: Update Complaint Status**

**Description**: As a Staff member, I want to update the status of complaints and add comments so I can resolve them effectively.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.9-T1 | Create Status Update Form (Frontend) | Develop a React form to update complaint status and add comments. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.9-T2 | Set Up Status Update API (Backend) | Implement Express route to update complaint status/comments in MongoDB. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.9-T3 | Integrate Frontend with Backend | Connect React form to Express API using Axios, handle notifications. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.9-T4 | Testing and Bug Fixing | Test status updates, ensure notifications work, fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.10: Create and Log in to Staff Account**

**Description**: As a Staff member, I want to create and log in to an account so I can manage department complaints.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.10-T1 | Create Staff Registration Form (Frontend) | Develop a React form for staff registration with role assignment. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.10-T2 | Create Staff Login Form (Frontend) | Develop a React form for staff login. | Adeel Mahmood Ansari | 2 hours | To-Do |
| SCMS-1.10-T3 | Set Up Staff Authentication API (Backend) | Implement Express routes for staff registration/login with role-based JWT. | Muhammad Shaif Imran | 4 hours | To-Do |
| SCMS-1.10-T4 | Integrate Frontend with Backend | Connect React forms to Express API using Axios. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.10-T5 | Testing and Bug Fixing | Test staff account creation/login, ensure role restrictions, fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.11: Monitor Complaint Progress in Real-Time**

**Description**: As a Staff member, I want to monitor complaint progress in real-time so I can proactively manage my workload.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.11-T1 | Create Real-Time Monitor (Frontend) | Develop a React component to display real-time complaint updates. | Adeel Mahmood Ansari | 4 hours | To-Do |
| SCMS-1.11-T2 | Set Up Real-Time API (Backend) | Implement Node.js WebSocket or polling to push complaint updates. | Muhammad Shaif Imran | 4 hours | To-Do |
| SCMS-1.11-T3 | Integrate Frontend with Backend | Connect React component to real-time API for live updates. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.11-T4 | Testing and Bug Fixing | Test real-time updates, ensure stability, fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.12: Select Complaint from Available List**

**Description**: As a Staff Member, I want to select a complaint from a list of available complaints assigned to my department.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.12-T1 | Create Complaint Selection Interface (Frontend) | Develop a React interface to list and select unassigned complaints. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.12-T2 | Set Up Complaint List API (Backend) | Implement Express route to fetch unassigned complaints by department. | Muhammad Shaif Imran | 3 hours | To-Do |
| SCMS-1.12-T3 | Integrate Frontend with Backend | Connect React interface to Express API, update status on selection. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.12-T4 | Testing and Bug Fixing | Test complaint selection, ensure status updates, fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**Sprint 3: Admin Features**

**User Story 1.13: View Status of All Complaints**

**Description**: As an Admin, I want to view the status of all complaints across the system so I can monitor progress and identify bottlenecks.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.13-T1 | Create Admin Dashboard (Frontend) | Develop a React dashboard to display all complaints with filters. | Adeel Mahmood Ansari | 4 hours | To-Do |
| SCMS-1.13-T2 | Set Up All Complaints API (Backend) | Implement Express route to fetch all complaints from MongoDB. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.13-T3 | Integrate Frontend with Backend | Connect React dashboard to Express API using Axios, restrict to admins. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.13-T4 | Testing and Bug Fixing | Test dashboard, ensure admin-only access, fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.14: Generate Reports on Complaint Trends**

**Description**: As an Admin, I want to generate reports on complaint trends and performance so I can analyze and improve efficiency.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.14-T1 | Create Report Interface (Frontend) | Develop a React interface to select report parameters and view results. | Adeel Mahmood Ansari | 4 hours | To-Do |
| SCMS-1.14-T2 | Set Up Report Generation API (Backend) | Implement Express route with MongoDB aggregation for report data. | Adeel Mahmood Ansari | 4 hours | To-Do |
| SCMS-1.14-T3 | Integrate Frontend with Backend | Connect React interface to Express API, enable PDF/CSV download. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.14-T4 | Testing and Bug Fixing | Test report generation, ensure accurate data and downloads, fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.15: Mark Complaints as High Priority**

**Description**: As an Admin, I want to mark complaints as “High Priority” based on severity so that my team can address critical cases first.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.15-T1 | Create Priority Marking Feature (Frontend) | Develop a React feature to mark complaints as high priority. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.15-T2 | Set Up Priority Update API (Backend) | Implement Express route to update complaint priority in MongoDB. | Muhammad Shaif Imran | 3 hours | To-Do |
| SCMS-1.15-T3 | Integrate Frontend with Backend | Connect React feature to Express API, handle notifications. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.15-T4 | Testing and Bug Fixing | Test priority marking, ensure staff notifications, fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.16: Manually Assign Complaint to Staff**

**Description**: As an Admin, I want to manually assign a specific complaint to a selected staff member within my department.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.16-T1 | Create Assignment Feature (Frontend) | Develop a React interface to select staff and assign complaints. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.16-T2 | Set Up Assignment API (Backend) | Implement Express route to assign complaints to staff in MongoDB. | Muhammad Shaif Imran | 3 hours | To-Do |
| SCMS-1.16-T3 | Integrate Frontend with Backend | Connect React interface to Express API, handle notifications. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.16-T4 | Testing and Bug Fixing | Test assignment, ensure staff notifications, fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.17: Send Notifications**

**Description**: As an Admin, I want to send notifications to users and staff about updates and actions so they can stay informed.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.17-T1 | Set Up Notification System (Backend) | Implement Node.js service for email/SMS notifications using a provider (e.g., Nodemailer). | Adeel Mahmood Ansari | 4 hours | To-Do |
| SCMS-1.17-T2 | Create Notification Trigger API (Backend) | Implement Express route to trigger notifications based on updates. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.17-T3 | Integrate with Existing Features | Connect notification API to complaint updates across all roles. | Abdul Wasay | 3 hours | To-Do |
| SCMS-1.17-T4 | Testing and Bug Fixing | Test notification delivery, ensure reliability, fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

**User Story 1.18: Staff Verification by Admin**

**Description**: As an Admin, I want to verify a Staff member of my department after their registration so that only authorized personnel can access and handle complaints within my department.

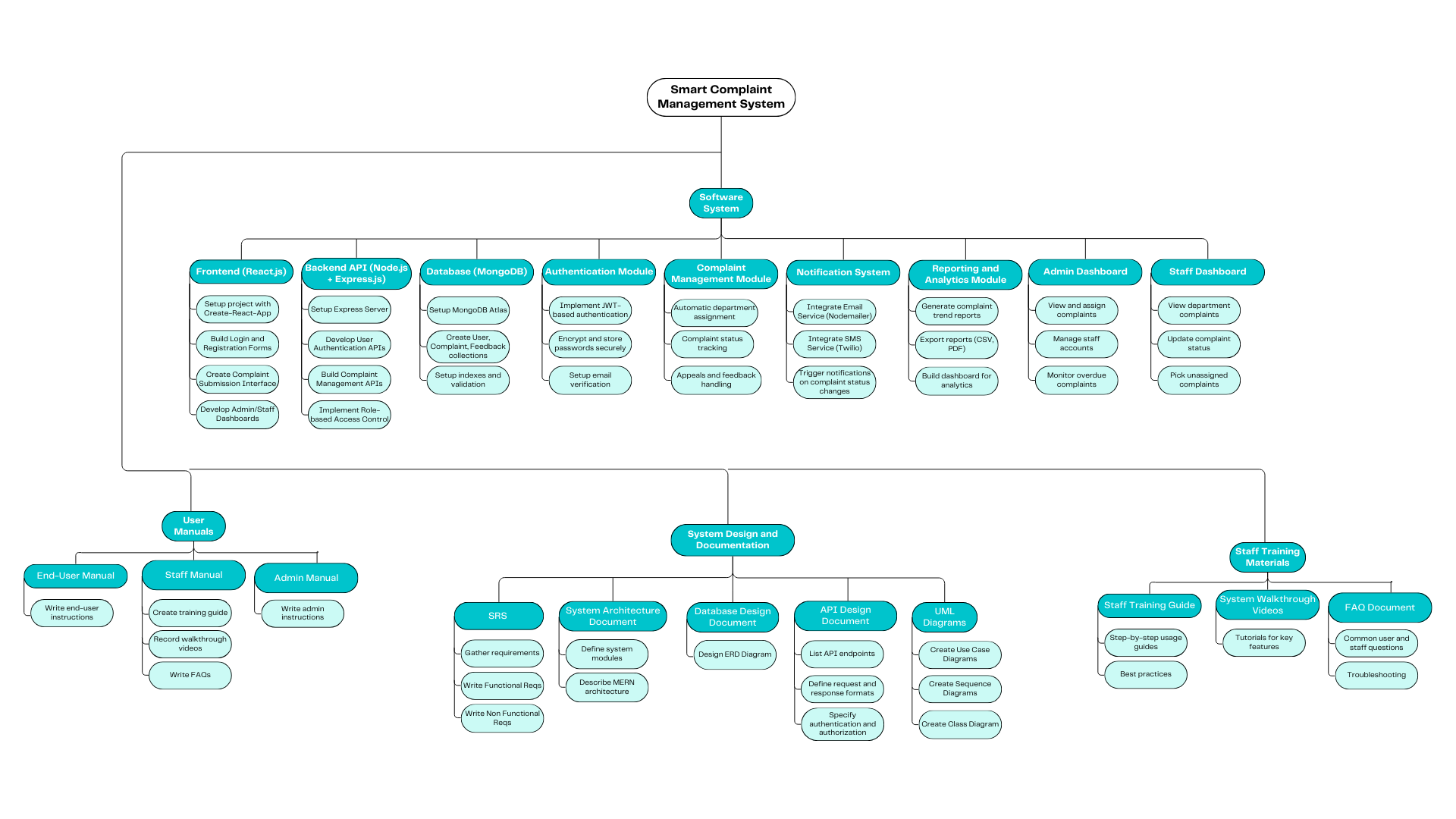
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task ID | Title | Description | Assignee | Estimated Time | Status |
| SCMS-1.18-T1 | Create Staff Verification Interface (Frontend) | Develop a React interface to list pending staff members and allow verification selection. | Adeel Mahmood Ansari | 3 hours | To-Do |
| SCMS-1.18-T2 | Set Up Staff Verification API (Backend) | Implement Express route to fetch pending staff and update verification status in MongoDB. | Adeel Mahmood Ansari | 4 hours | To-Do |
| SCMS-1.18-T3 | Integrate Frontend with Backend | Connect React interface to Express API using Axios, handle verification and notifications. | Abdul Wasay | 2 hours | To-Do |
| SCMS-1.18-T4 | Testing and Bug Fixing | Test verification process, ensure department restrictions and notifications work, fix bugs. | Muhammad Shaif Imran | 2 hours | To-Do |

# Project Plan

**1. Work Breakdown Structure (WBS)**

The Work Breakdown Structure (WBS) visually represents the division of the project into manageable sections. Each task is divided into subtasks to simplify project planning and resource allocation.

**Figure 1**: *Work Breakdown Structure of Smart Complaint Management System*

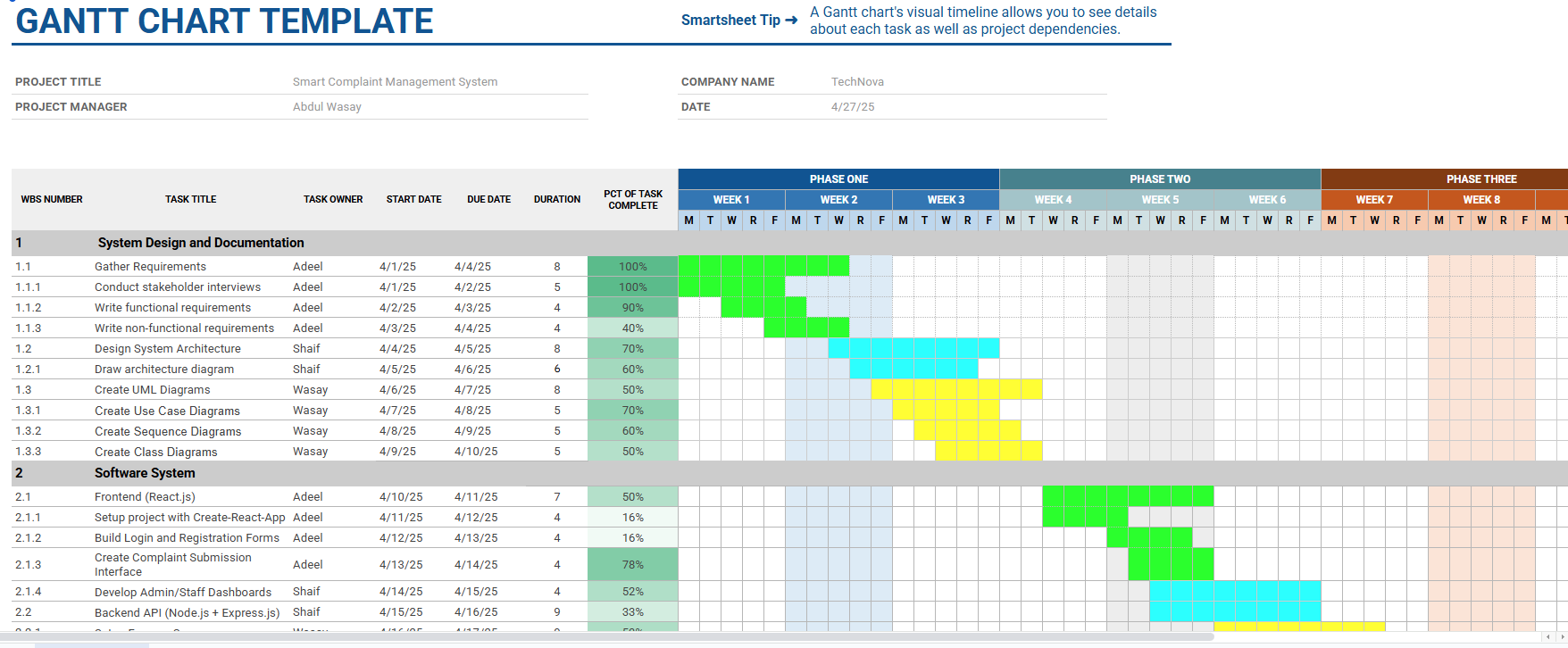
**

**Link of Design:** <https://www.canva.com/design/DAGlvkxSSJ4/50rFbD9wWHoTwsmHOWPhgw/edit?utm_content=DAGlvkxSSJ4&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton>

**2. Project Gantt Chart**

The Gantt Chart outlines the timeline of the project, indicating the start and end dates of all major activities and milestones. It is essential for tracking progress and ensuring timely delivery.

**Figure 2**: *Gantt Chart of Smart Complaint Management System Project*

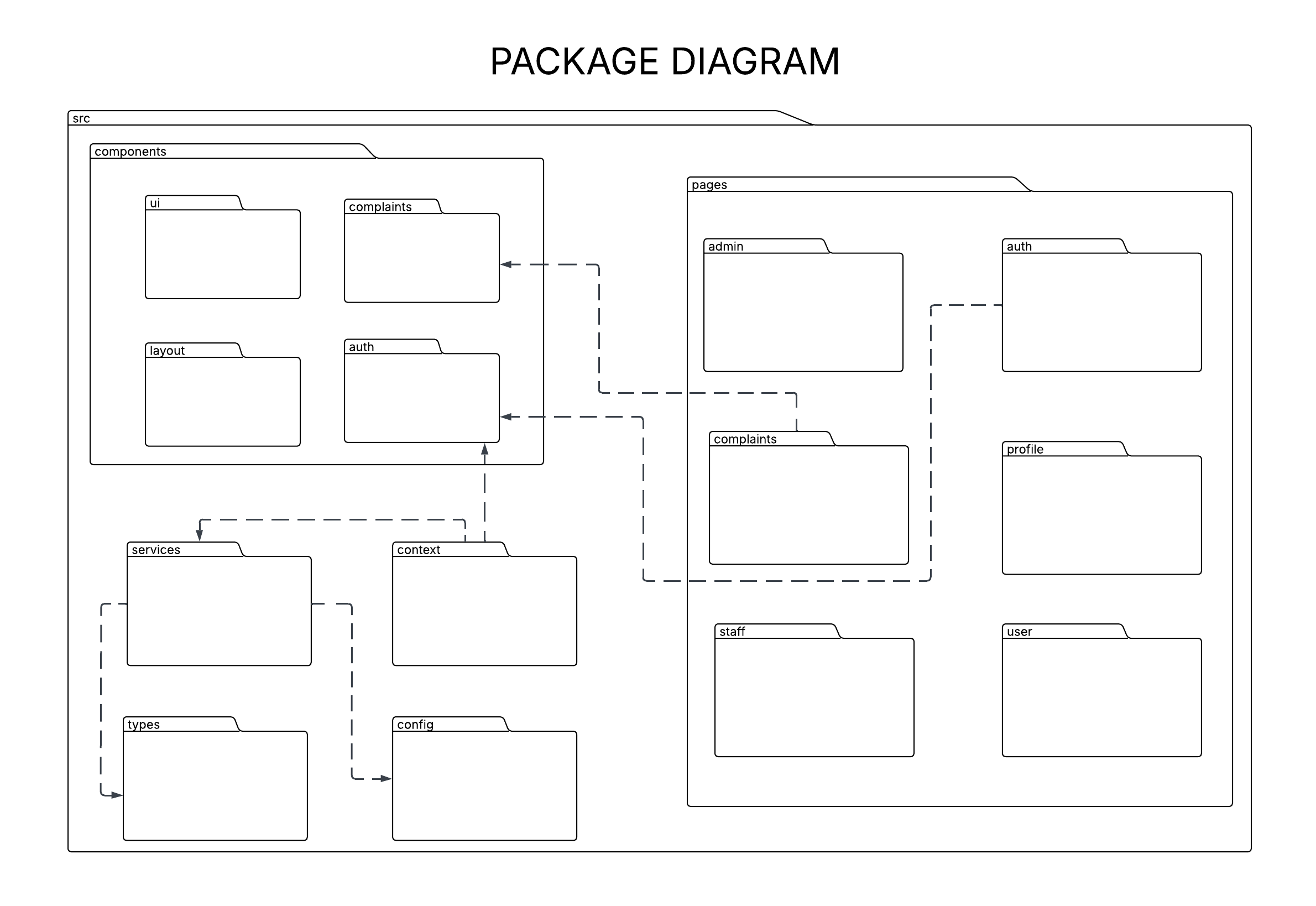


**Link of Design:**

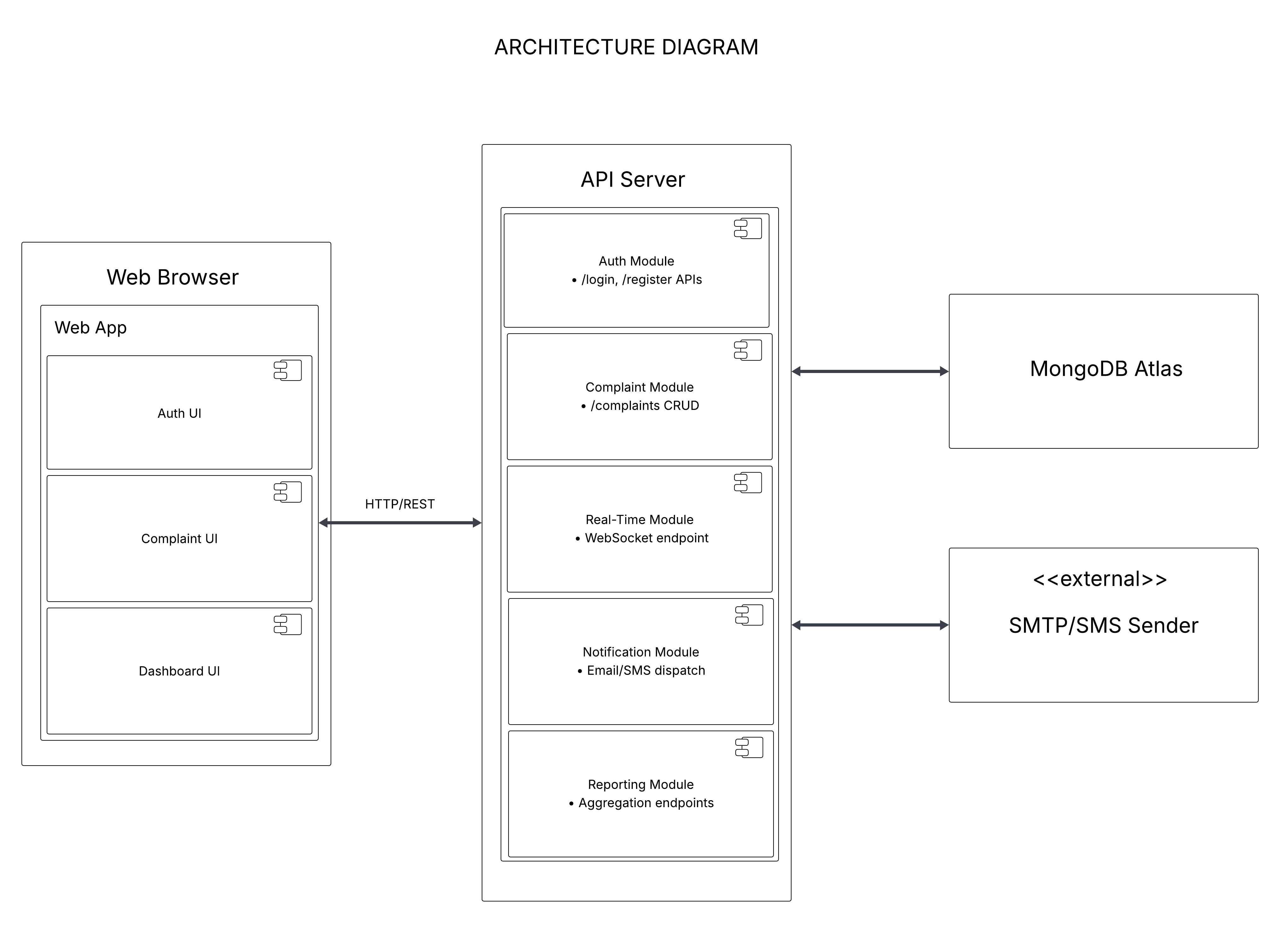
https://docs.google.com/spreadsheets/d/1Y0W04dFA-dAtgDcWWRwiklOOPw\_arHTKI1su3HWEz6E/edit?gid=1115838130#gid=11158381

# Diagrams

**1. UML Package Diagram**

****   
Each subsystem is represented as a package. Arrows show dependencies, for example, the Reporting and Notification modules depend on Complaint data.

**2. Architecture Styles**



 **Client-Server Architecture**

* The system follows a **client-server model**, where a web browser (client) interacts with an API server (backend) through HTTP/REST.
* The client comprises multiple UI modules: Auth UI, Complaint UI, and Dashboard UI.

 **Modular Monolith / Microservices-Inspired**

* The API Server is structured into **independent functional modules**:
  + Auth Module
  + Complaint Module
  + Real-Time Module
  + Notification Module
  + Reporting Module
* This mimics microservices in modularity while retaining a single deployment unit.

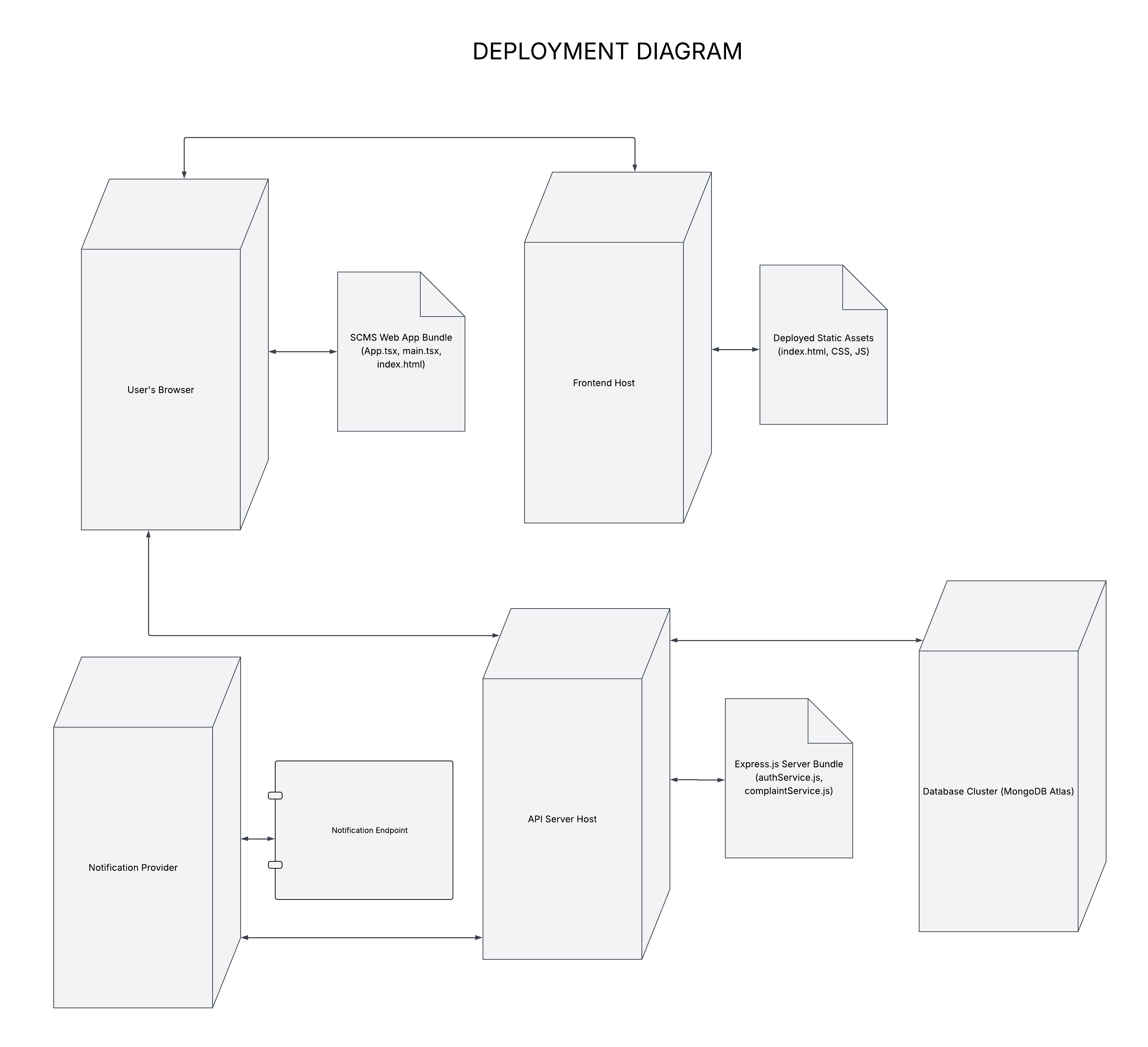
 **Layered Architecture**

* Each module within the API server follows a **layered approach**:
  + **Controller Layer**: Handles incoming requests.
  + **Service Layer**: Contains business logic.
  + **Repository/Data Layer**: Interacts with MongoDB Atlas.

 **Event-Driven Integration (for Notifications)**

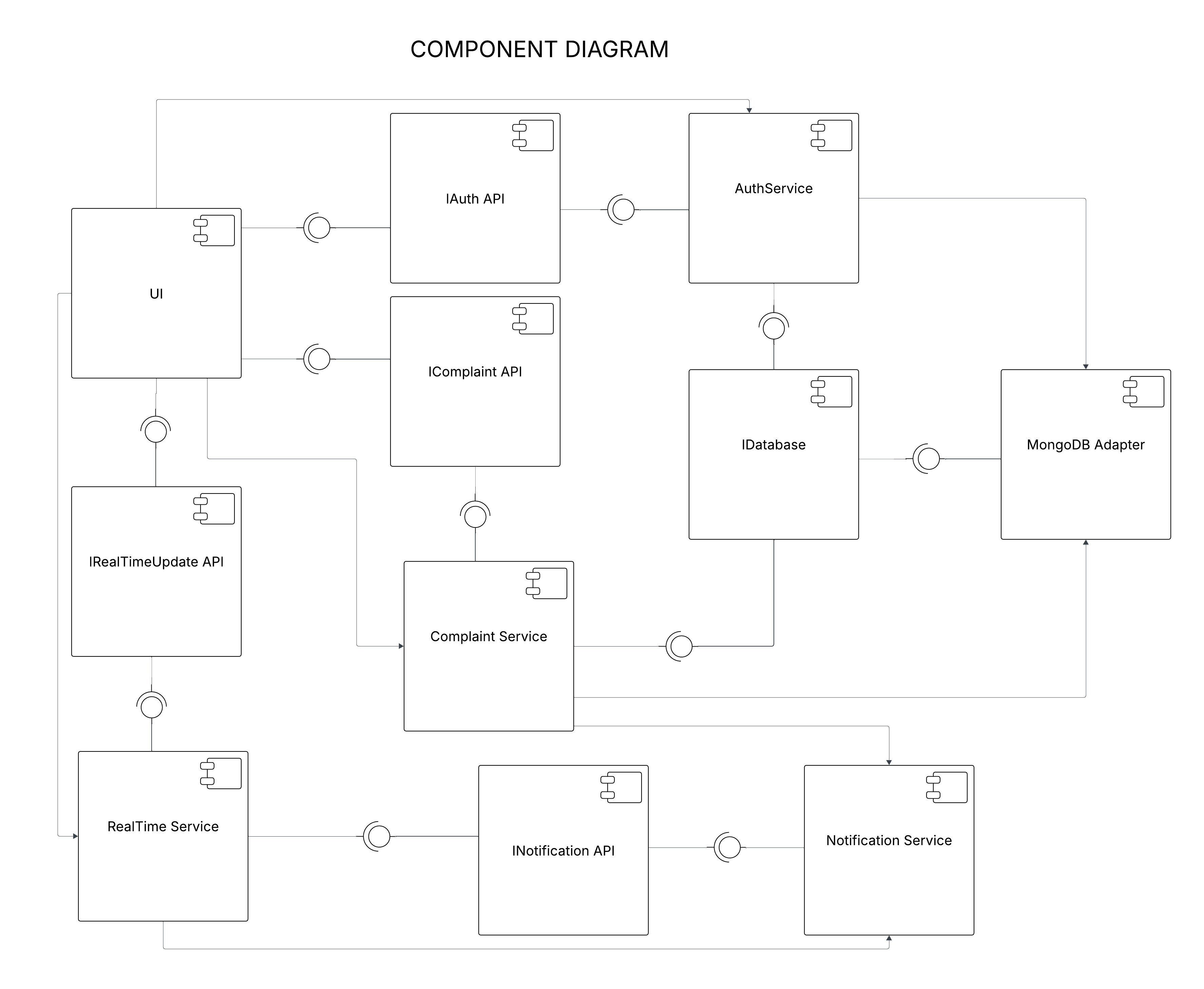
* The Notification Module is **loosely coupled** using an event-driven model (e.g., triggered on complaint status change).
* Integration with external services like SMTP/SMS is also abstracted.

**3. Deployment Diagram**

****

* The **Web Browser** hosts the Web App with UI components.
* The **API Server** hosts the logic:
  + Auth Module
  + Complaint Module
  + Real-Time WebSocket handling
  + Notifications
  + Reporting
* **MongoDB Atlas** stores application data.
* External **SMTP/SMS service** is used to send alerts.

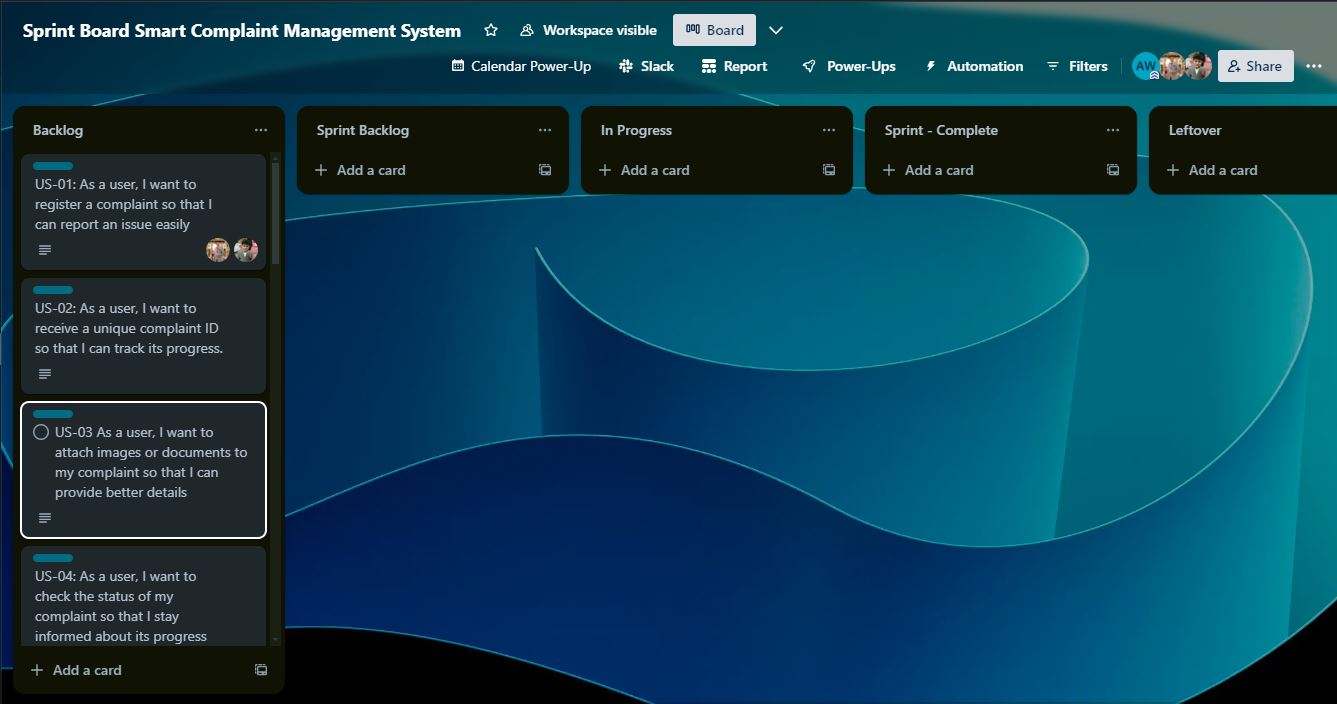
**4. Component Diagrams**

****

# Trello Screenshots

**Scrum Board**

* **Snapshot 1**: Backlog

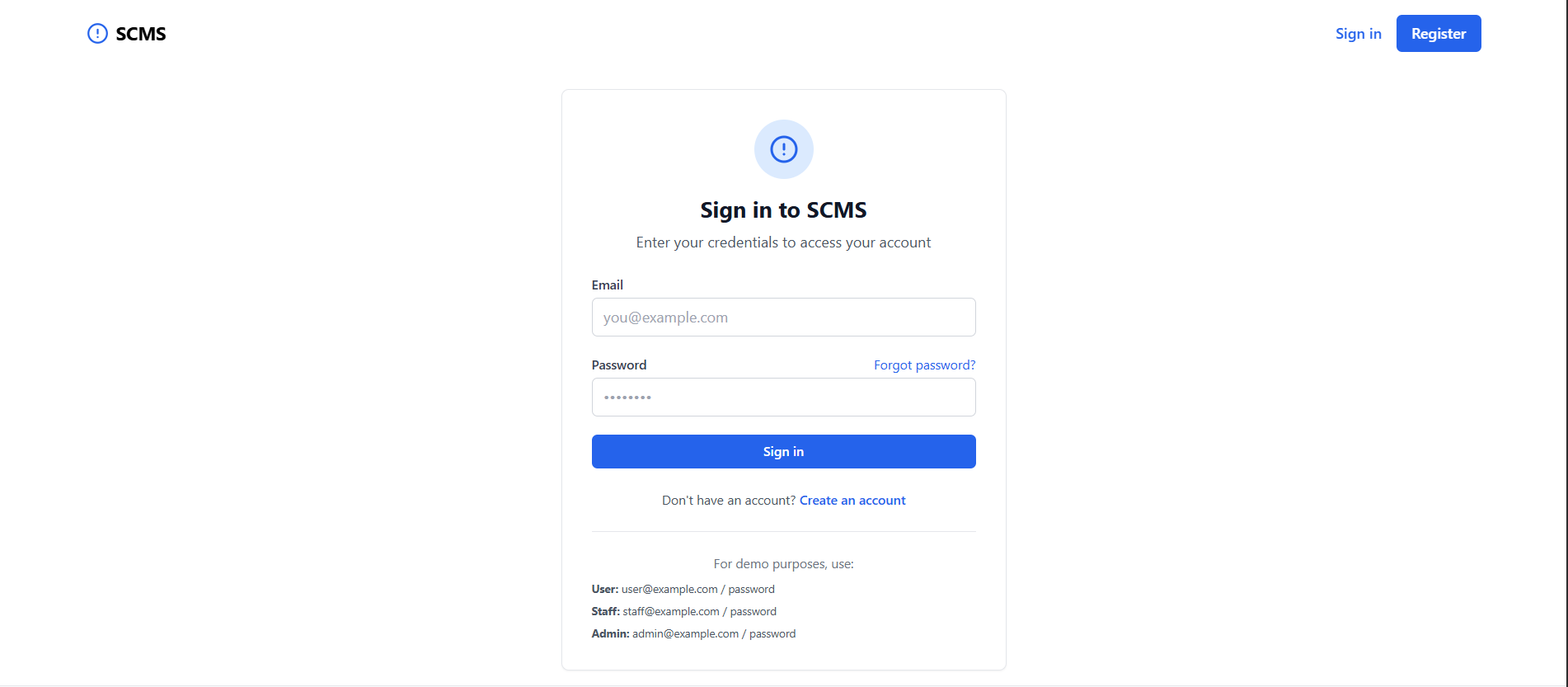
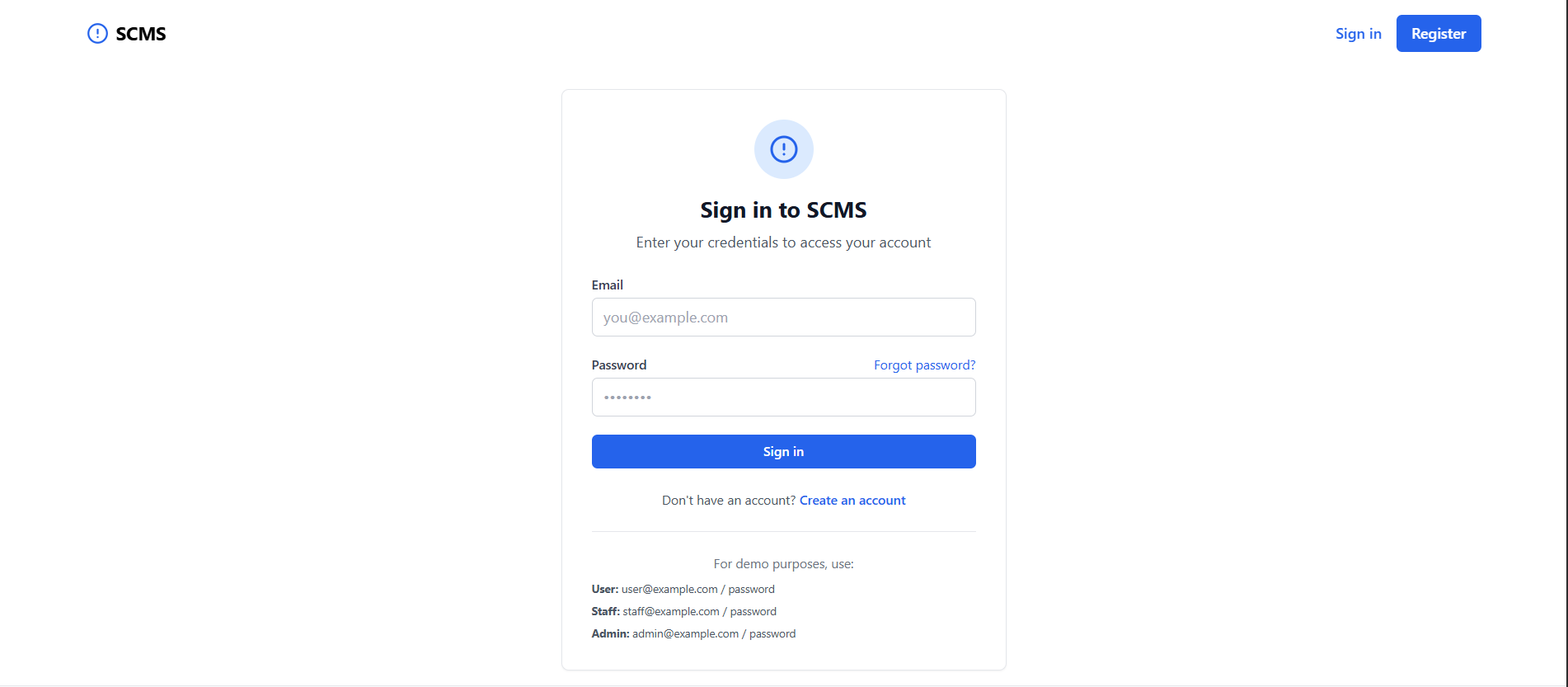
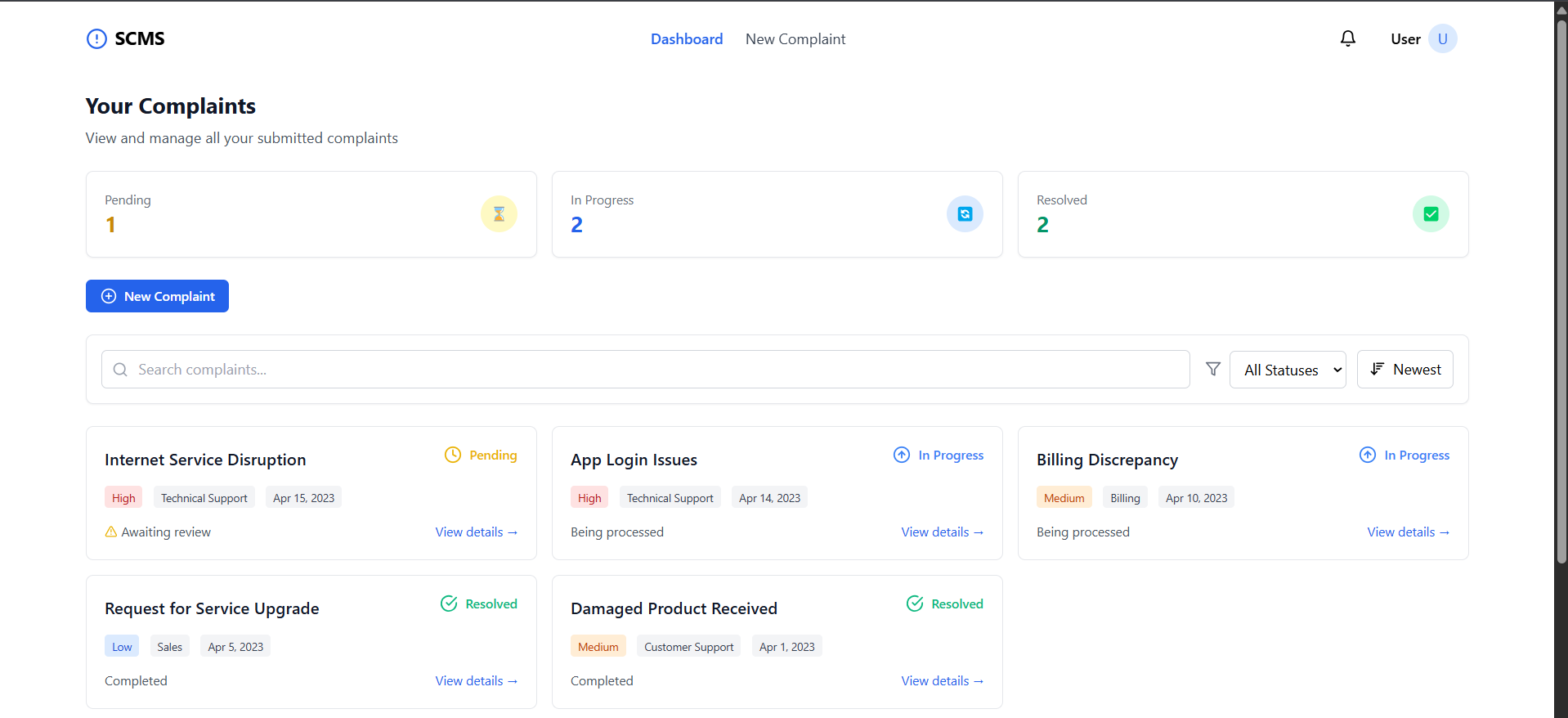
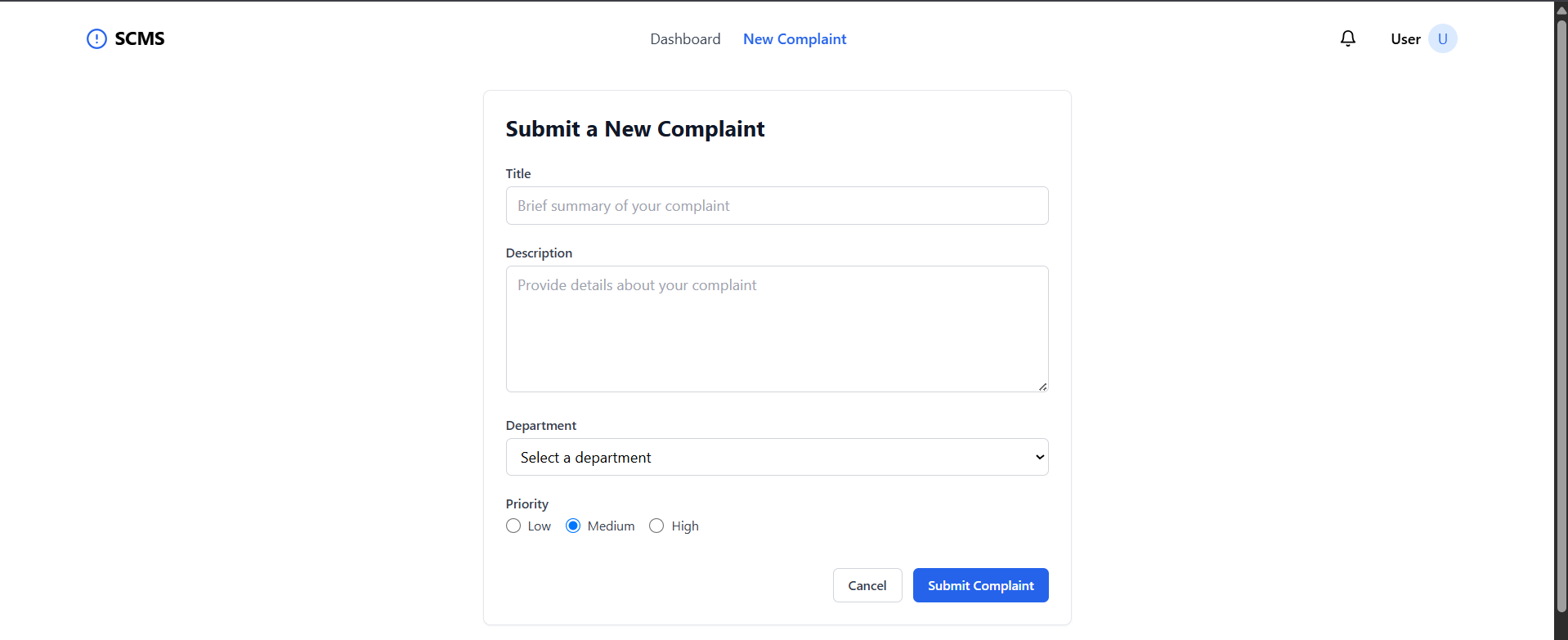
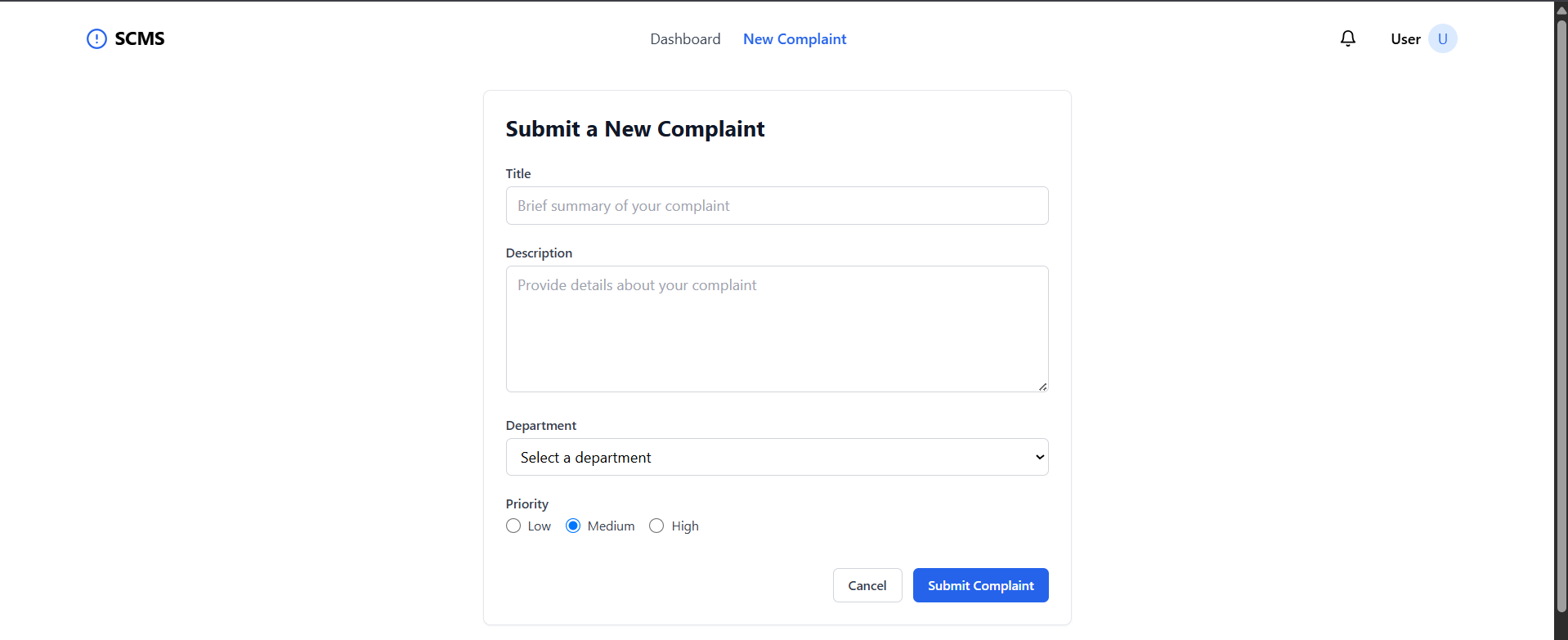
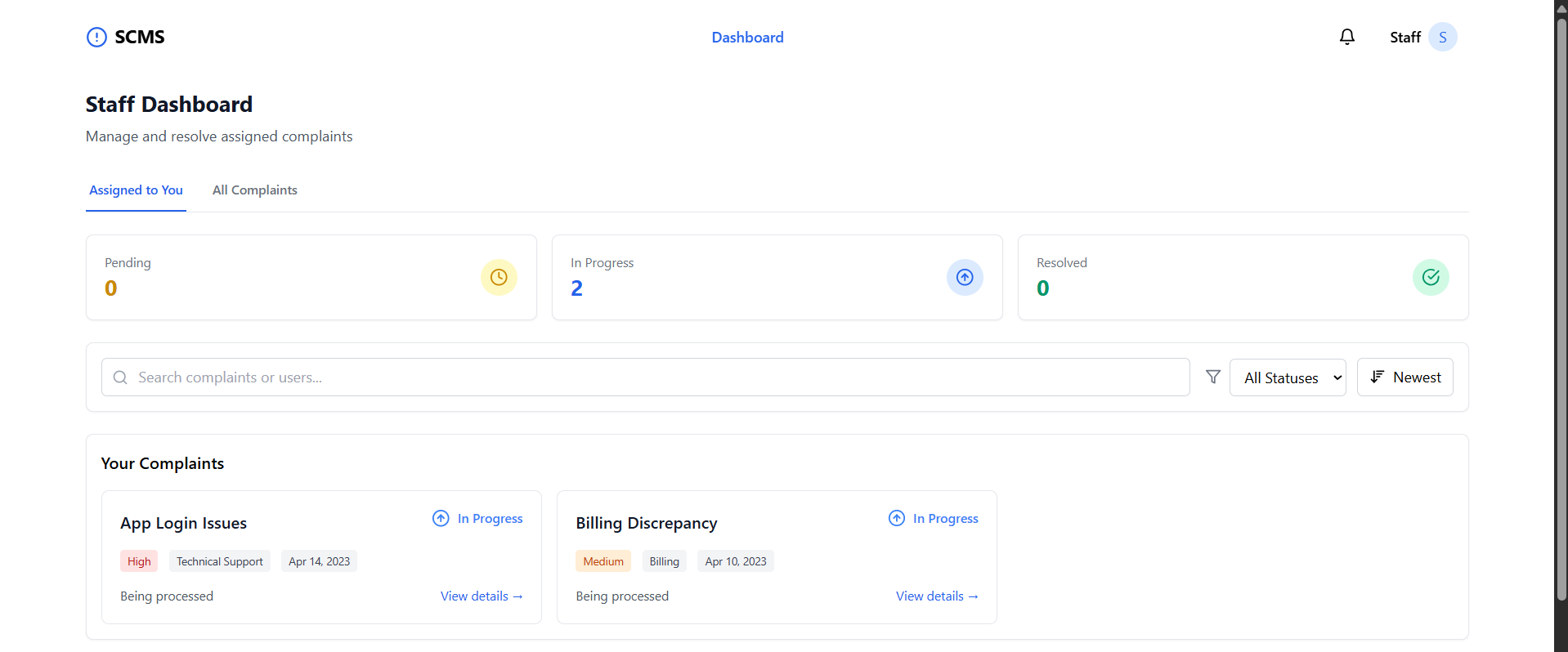
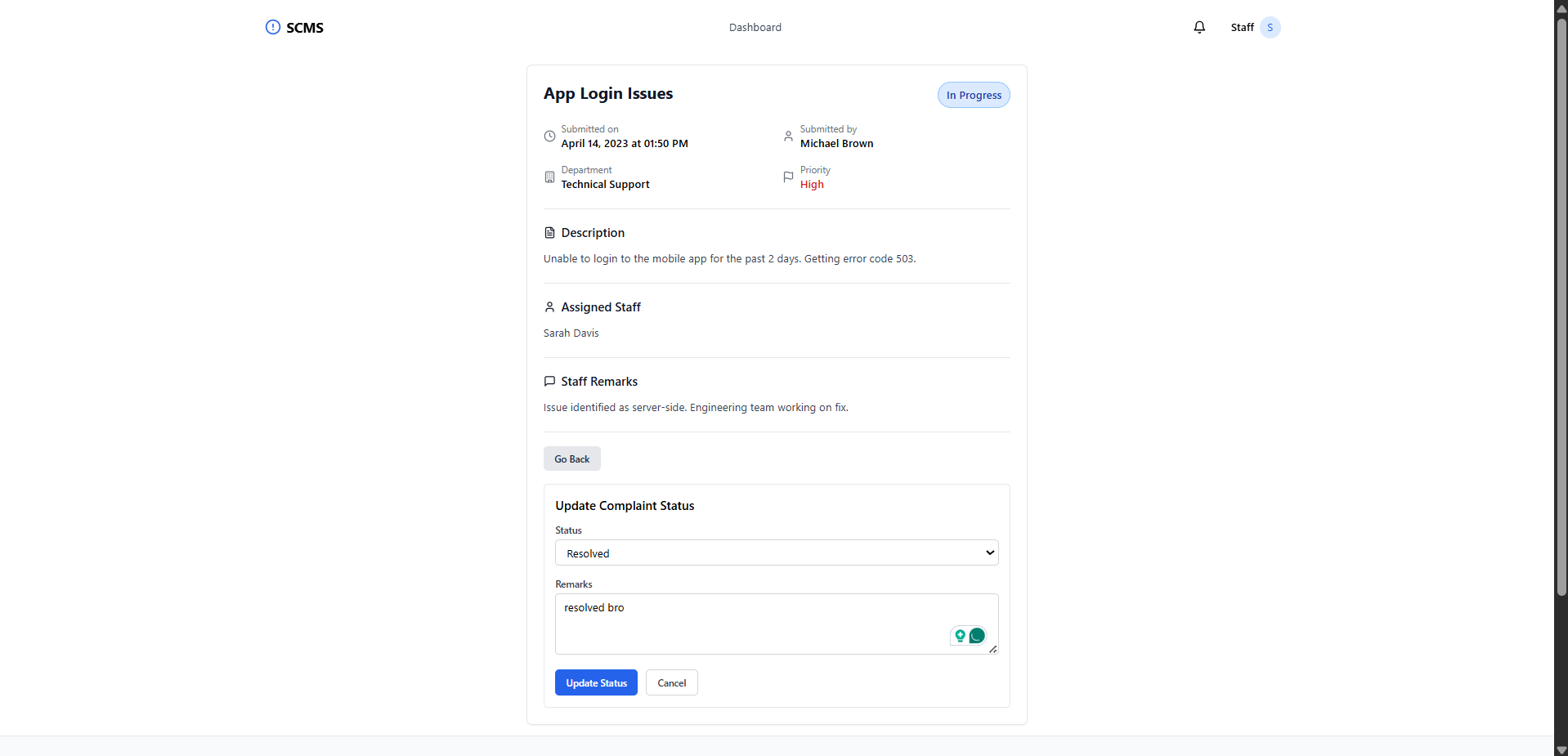
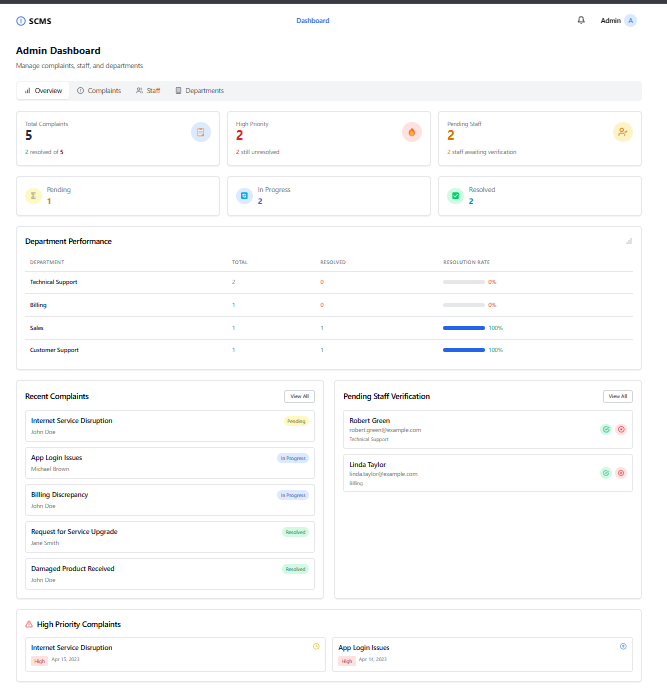
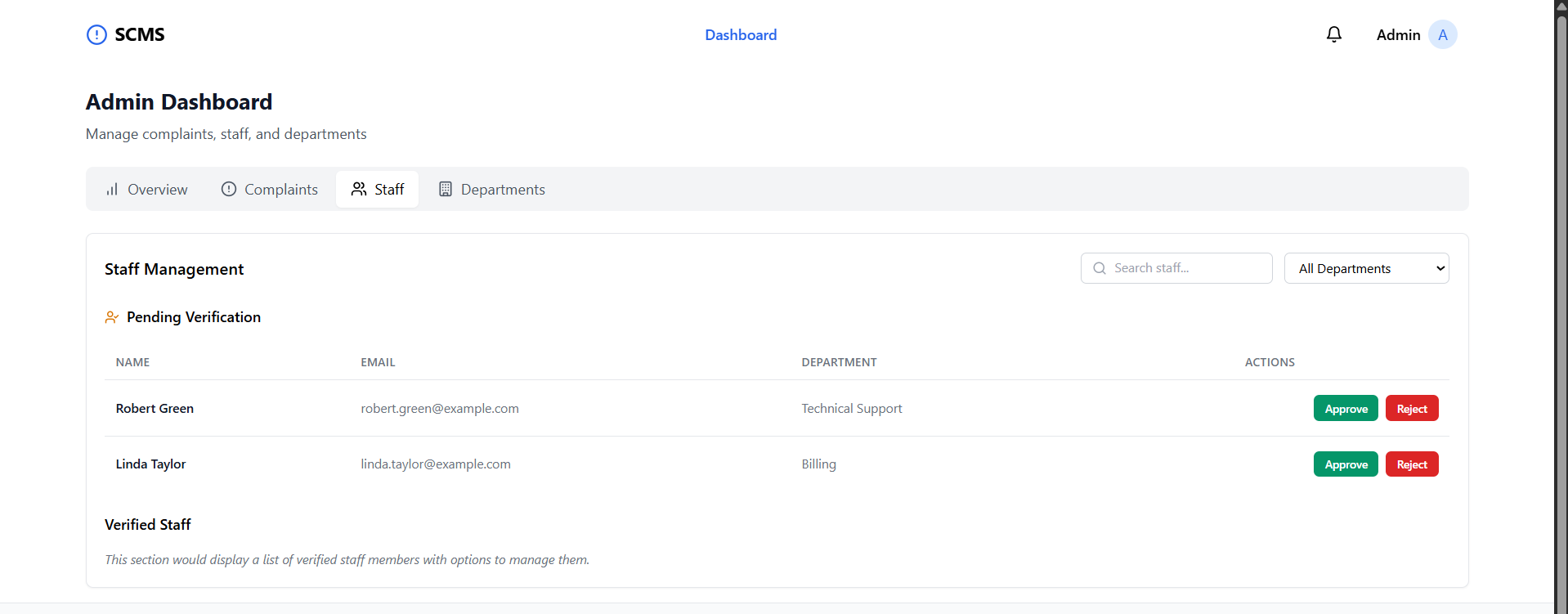


* **Snapshot 2**: Sprint Backlog

A screenshot of a computer

AI-generated content may be incorrect.

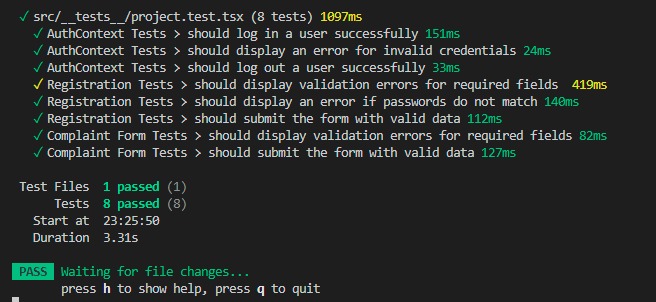
# Screenshots of Implementation



# Test Cases BlackBox

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario / Description** | **Test Steps** | **Test Data / Input** | **Expected Result** | **Actual Result** | **Status** |
| TC-EQ-001 | Register user with valid input | 1. Open registration form 2. Enter valid details 3. Click submit | Email: valid@example.com Username: user123 Password: ValidPass1 | Registration successful | Registration successful | Pass |
| TC-EQ-002 | Register user with duplicate email | 1. Open registration form 2. Enter existing email 3. Click submit | Email: existing@example.com Username: user456 Password: ValidPass1 | Error: Email is already in use | Error: Email is already in use | Pass |
| TC-EQ-003 | Submit complaint with valid category and description | 1. Login 2. Go to complaint page 3. Fill valid fields 4. Submit | Category: Technical Description: 'Internet not working' | Complaint submitted successfully. Your Complaint ID is [XXXX] | Complaint submitted successfully. Your Complaint ID is CMP2025 | Pass |
| TC-EQ-004 | Submit complaint with unsupported file format | 1. Login 2. Go to complaint page 3. Upload .exe file 4. Submit | Category: Billing Description: 'Wrong charge' Attachment: unsupported.exe | Error: Invalid file format. Attachments must be JPG, PNG, or PDF | Error: Invalid file format | Pass |
| TC-BVA-001 | Password length = 5 (below minimum) | 1. Open registration 2. Enter short password 3. Submit | Password: Abc12 | Error: Password does not meet security requirements | Error: Password does not meet security requirements | Pass |
| TC-BVA-002 | Password length = 8 (minimum valid) | 1. Open registration 2. Enter valid password of 8 chars 3. Submit | Password: Abc12345 | Registration successful | Registration successful | Pass |
| TC-BVA-003 | Rating value = 6 (above max limit) | 1. Login 2. Attempt to give rating 6 3. Submit | Rating: 6 | Error: Invalid rating value. Please select a rating between 1 and 5 | Error: Invalid rating value | Pass |
| TC-BVA-004 | Rating value = 5 (max valid limit) | 1. Login 2. Give rating 5 3. Submit | Rating: 5 | Thank you for your feedback! | Thank you for your feedback! | Pass |
| TC-FUNC-001 | User logs in with correct credentials | 1. Open login page 2. Enter correct email/password 3. Click login | Email: user@example.com Password: CorrectPass123 | Login Successful. Redirecting to your dashboard | Login Successful. Redirecting to your dashboard | Pass |
| TC-FUNC-002 | Track complaint with valid ID | 1. Login 2. Go to track complaint page 3. Enter valid ID 4. Submit | Complaint ID: CMP12345 | Status: In Progress | Status: In Progress | Pass |
| TC-FUNC-003 | Update profile with already registered email | 1. Login 2. Go to profile 3. Enter an existing email 4. Save | Email: existing@example.com | Error: Email is already in use | Error: Email is already in use | Pass |
| TC-FUNC-004 | Appeal a resolved complaint | 1. Login 2. Go to resolved complaints 3. Select one 4. Click 'Appeal' 5. Submit | Complaint ID: CMP54321 Appeal Reason: Still unresolved | Your appeal has been submitted successfully | Your appeal has been submitted successfully | Pass |

# Test Cases WhiteBox



# Work division between Group Members

The work was distributed across three team members based on their strengths in frontend, backend, and integration tasks. Here's a summary of responsibilities:

**Adeel Mahmood Ansari (22i-1024)**

* Frontend development of key interfaces:
  + User registration, login, complaint submission, feedback, dashboards
  + Staff and admin forms and dashboards

**Muhammad Shaif Imran (22i-0979)**

* Backend development using Express.js and MongoDB:
  + APIs for authentication, complaints, feedback, filtering, and reporting
  + Staff and admin-specific APIs, including real-time updates and role-based access
* Testing and debugging across multiple sprints

**Abdul Wasay (22i-1198)**

* Full integration of frontend and backend
* API wiring for complaint submission, tracking, updates, and notifications
* Handled communication flows and assisted in testing and bug fixing

**Workload was balanced** across all three sprints. Each member took lead roles in their areas and contributed to testing, UI reviews, and documentation

# Lesson Learnt by Group

1. **Agile Planning Matters**  
   We learned the importance of breaking work into sprints and user stories. This structure helped us maintain clarity and stay focused on user needs.
2. **Communication is Critical**  
   Regular meetings and progress updates were essential. Miscommunication in early sprints led to delays that we learned to avoid in Sprint 3.
3. **Version Control Discipline**  
   Consistent use of Git branches and commits helped manage code effectively. Merge conflicts taught us to communicate before pushing changes.
4. **Backend-Frontend Sync Is Key**  
   We underestimated the effort in syncing APIs with UI components. Frequent integration helped us catch mismatches early in later sprints.
5. **Testing Saves Time**  
   Incorporating black box and white box testing allowed us to catch critical bugs before final submission, boosting confidence in stability.
6. **Flexibility and Adaptability**  
   Changing priorities and added features (like admin report generation) forced us to adapt quickly and improve time estimation.