PROCTOR REPOSITORY SYSTEM

Submitted By

Md. Sabbir Ahmed Shanto

ID: 221-35-962

Supervised By

Mr. Khalid Been Badruzzaman Biplob

Lecturer (Senior Scale),

Department of Software Engineering,

Daffodil International University

This project report has been submitted in fulfilment of the requirements for the degree of **Bachelor of Science in Software Engineering**

Department of Software Engineering

Faculty of Science and Information Technology

Supervisor Approval Form

Fall 2025	B.Sc. In SWE	Campus: DSC

Student Name	Student ID
Md. Sabbir Ahmed Shanto	221-35-962

Project/Thesis Information	
Project/Thesis Title	Proctor Repository System
Type of work	Development-Based Project

Supervisor information	
Supervisor Name	Mr. Khalid Been Badruzzaman Biplob
Supervisor Initial	KBB
Completed Credit till now	133
How many credits in this semester	6
Supervisor Consent	☑ Yes ☐ No

Supervisor Signature

DAFFODIL INTERNATIONAL UNIVERSITY

DECLARATION OF PROJECT AND COPYRIGHT

Author's Full Name	:	Md. Sabbir Ahmed Shanto
Date of Birth	:	01-01-2002
Title	:	Proctor Repository System
Academic Session	:	2022-2025

I declare that this project is classified as:

☐	CONFIDENTIAL	(Contains confidential information under the Official Secret Act 1997)*
☐	RESTRICTED	(Contains restricted information as specified by the organization where research was done)*
☐	OPEN ACCESS	I agree that my project to be published as online open access (Full Text)

I acknowledge that Daffodil International University reserves the following rights:

1. The Project is the Property of Daffodil International University.
2. The Library of Daffodil International University has the right to make copies of the Project for the purpose of research only.
3. The Library of Daffodil International University has the right to make copies of the Project for academic exchange.

Certified by:

(Student's Signature)

(Supervisor's Signature)

221-35-962

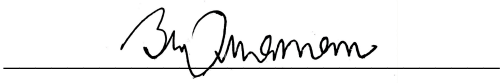
Student ID
Date:

Khalid Been Badruzzaman Biplob

Name of Supervisor
Date:

SUPERVISOR'S DECLARATION

I/We* hereby declare that I/We* have checked this project and in my/our* opinion, this project is adequate in terms of scope and quality for the award of the degree of Bachelor of Science.



(Supervisor's Signature)

Full Name : **Khalid Been Badruzzaman Biplob**
Position : **Lecturer (Senior Scale)**
Date : **13 December, 2025**

STUDENT'S DECLARATION

I hereby declare that the work in this project is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at Daffodil International University or any other institution.

(Student's Signature)

Full Name : **Md. Sabbir Ahmed Shanto**
ID Number : **221-35-962**
Date : **13 December, 2025**

TITLE: PROCTOR REPOSITORY SYSTEM

Md. Sabbir Ahmed Shanto

Project submitted in fulfillment of the requirements
for the award of the degree of
Bachelor of Science/Master of Science

Department of Software Engineering

DAFFODIL INTERNATIONAL UNIVERSITY

DECEMBER 2025

ACKNOWLEDGEMENTS

First and foremost, I am grateful to Almighty Allah, who has given me the strength, patience, and ability to complete this project successfully.

I am particularly thankful to my supervisor, **Mr. Khalid Been Badruzzaman Biplob**, Lecturer (Senior Scale), Department of Software Engineering, for his constant support, suggestions, and encouragement throughout the development process of this project. His inputs to this work have been immensely valuable.

I am also grateful to all those who participated in the survey and interviews relating to this project, whose willingness to share their opinions and experiences was of immense help to the success of my project.

I would like to take this opportunity to sincerely thank all the faculty members of the Department of Software Engineering, Daffodil International University, for their academic guidance, motivation, and support throughout my studies.

Finally, I would like to thank my parents for the unconditional love and support they provide me. Their blessings and motivation are the direct sources of strength and an endless force behind me.

DEDICATION

I therefore declare that I have done this project under the oversight of **"Mr. Khalid Been Badruzzaman Biplob"**, **Lecturer (Senior Scale)**, **Department of Software Engineering**, **Daffodil International University**. Also declare that neither entire record nor any portion of this record has been submitted somewhere else for my degree.

ABSTRACT

The Proctor Repository System is a web-based system developed for Daffodil International University to modernize the current student complaint management processes. In this system, a student can report their complaints against harassment, misconduct, or other issues along with evidence securely. After the submission, the cases will automatically be routed to the Coordination Officer for preliminary screening and followed by the Proctor, Deputy Proctor, and then Assistant Proctor in due order through a workflow. The platform supports role-based dashboards, explanation and evidence management, case file preparation, meeting scheduling, and automated notifications to ensure timely communication. The system improves the disciplinary structure of the university by introducing transparency in the process, faster response times, better reporting, and the concept of accountability. The project seeks to develop a prompt, trusted, and secure environment for addressing complaints and performing student services.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATION	FULL FORM
JWT	JSON Web Token
OAuth	Open Authorization
UI	User Interface
API	Application Programming Interface
ERP	Enterprise Resource Planning
UAT	User Acceptance Testing
SRLC	Software Release Life Cycle

LIST OF APPENDICES

Chapter: 1 Introduction

1.1 Project Planning and Initiation

1.1.1 Phase 1: Preliminary Analysis & Project Scope Definition

Objective:

Design a web-based centralized system for the university proctor's office to manage complaints of student misconduct effectively.

Scope:

- Complaint submission by students
- Explanation collection from both parties
- Assignment of resolution officers
- Case tracking and final decision logging
- Role-based dashboards (Student, Admin, Super Admin)

Expected Outcomes:

- Improved accountability and transparency
- Faster resolution time
- Digital archive for administrative tracking

1.1.2 Phase 2: Market Feasibility Analysis (or Market Research)

Stakeholder Needs: Student needs an easy, secure complaint submission process. Proctors require a centralized mechanism to process and follow through on cases. University Administration needs enhanced visibility, compliance, and audit trails

Market Gap: Currently, complaint handling is mostly manual or half-digital (email-based) and also it has lack of systematic workflow, case tracking, and escalation procedures.

Research Insight: Surveys among students and interviews with proctors showed strong demand for digitalization. Universities globally are shifting toward similar case management systems (e.g., Title IX portals in the U.S.)

1.1.3 Phase 3: Technical Feasibility Analysis

Objective:

In order to rate the capabilities of the new Proctor Repository System, we have to determine the efficiency of the chosen technologies such as HTML, CSS (Bootstrap), Angular, ASP.NET Core, PostgreSQL, and JWT/OAuth implemented via ASP.NET Identity. The main reason for these choices is their ability to fully ensure efficient development and management processes.

System Architecture Overview:

- **Frontend:** Built with **Angular**, offering a responsive and dynamic user interface using component-based architecture. Styled using **Bootstrap**, ensuring a clean and mobile-friendly UI design.

- **Backend:** Utilized **ASP.NET Core** technology, a high-performance cross-platform framework that makes it particularly suited to development of **REST**ful services.

- **Database: PostgreSQL,** it supports transactions that are compliant with the ACID requirements of a database, good indexing, and JSON capabilities that can handle user data, complaint information, and case history effectively.

- **Authentication & Security:** Authentication, registration, and role-based security functionality is provided by the **ASP.NET** Identity library. JSON Web Token and OAuth authorization and authentication functionality also provide role-based security for all web APIs.

Development Tools & Environment:

- **IDE:** Visual Studio 2022 / VS Code
- **Angular CLI** for efficient module and component generation
- **Entity Framework Core** for ORM-based database interaction
- **PostgreSQL** Management Tool: pgAdmin
- **Version Control:** Git + GitHub or Azure Repos

1.1.4 Phase 4: Financial Feasibility Analysis

Development Cost:

- Low (student-developed, open-source tools used)

Operational Cost:

- Hosting (shared or cloud-based)
- Maintenance (bug fixes, updates)

Return on Investment:

- Saves time and administrative resources
- Builds transparency and trust among students

1.2 Target User Profile and Tentative Elicitation Process

1.2.1 Target User Profile:

1.2.1.1 User Profile-01: Student

User Class	Notes on Characteristic	Requirement Implied
Type of User	Student (General User)	Verification
Age Range	18–26	Performance, Acceptance
Frequency of Use	Occasionally (when submitting/viewing complaints)	Performance, Acceptance
Computer Experience	Basic knowledge	Usability, Acceptance
Education	University Level (Undergraduate)	Usability, Acceptance
Language Skills	Bangla and English	Usability, Acceptance
Number of Users	Many	Performance, Scalability, Acceptance
Training	Not required	Usability, Documentation
Other Systems Used	University Portal, Email, LMS	Compatibility, Usability
Ways of Working	Submits complaints and checks status	Usability, Performance

Table 1.2.1.1: User Profile-01: Student

1.2.1.2 User Profile-02: Coordination Officer

User Class	Notes on Characteristic	Requirement Implied
Type of User	Coordination Officer (Admin)	Verification
Age Range	25–40	Performance, Acceptance
Frequency of Use	Daily	Performance, Acceptance
Computer Experience	Intermediate	Usability, Acceptance
Education	Graduate or Postgraduate	Usability, Acceptance
Language Skills	Bangla and English	Usability, Acceptance
Number of Users	Few (1–3 per campus)	Performance, Acceptance
Training	May require system walkthrough	Usability, Documentation
Other Systems Used	University ERP, Official Email	Compatibility, Usability
Ways of Working	Collects evidence, updates case statuses, assigns cases	Usability, Performance

Table 1.2.1.2: User Profile-02: Coordination Officer

1.2.1.3 User Profile-03: Assistant Proctor & Deputy Proctor

User Class	Notes on Characteristic	Requirement Implied
Type of User	Admin (Deputy/Assistant Proctor)	Verification
Age Range	30–50	Performance, Acceptance
Frequency of Use	Regularly (case review, comments)	Performance, Acceptance
Computer Experience	Intermediate	Usability, Acceptance
Education	Postgraduate	Usability, Acceptance
Language Skills	Bangla and English	Usability, Acceptance
Number of Users	Few (2–4 per university)	Performance, Acceptance
Training	Optional, based on familiarity	Usability, Documentation
Other Systems Used	University Systems, Email	Compatibility, Usability
Ways of Working	Review explanations, support decision-making	Usability, Performance

Table 1.2.1.3: User Profile-03: Assistant Proctor & Deputy Proctor

1.2.1.4 User Profile-04: Proctor

User Class	Notes on Characteristic	Requirement Implied
Type of User	Super Admin (Proctor)	Verification
Age Range	35–60	Performance, Acceptance
Frequency of Use	Frequently (to assign, review, finalize cases)	Performance, Acceptance
Computer Experience	Intermediate to advanced	Usability, Acceptance
Education	Postgraduate or PhD	Usability, Acceptance
Language Skills	Bangla and English	Usability, Acceptance
Number of Users	1 (per campus)	Performance, Acceptance
Training	May need orientation	Usability, Documentation
Other Systems Used	ERP, Meeting Platforms, University Dashboards	Compatibility, Usability
Ways of Working	Final decision maker, case assignment, monitoring reports	Usability, Performance

Table 1.2.1.4: User Profile-04: Proctor

1.2.2 Elicitation Process:

- **Stakeholder Interviews:** Carried out with proctors and university personnel on how the current complaint resolution process
- **Surveys:** Distributed to students to identify pain points and expectations
- **Document Analysis:** Analyzed existing manual complaint handling processes
- **Observation:** Shadowed proctorial staff to see current pain points
- **Prototyping & Feedback:** Early UI wireframes were presented to stakeholders before development

1.3 System Requirements

1.3.1 Hardware Requirements

1.3.1.1 Client-Side Hardware

- **Device:** Desktop/Laptop/Smartphone/Tablet
- **Processor:** Minimum Dual-Core 2.0 GHz (Intel/AMD)
- **RAM:** Minimum 4 GB (8 GB recommended for smooth Angular application rendering)
- **Storage:** 200–500 MB free space for browser caching
- **Display:** Minimum resolution 1366 × 768
- **Internet:** Stable connection (minimum 1 Mbps; recommended 5 Mbps+)
- **GPU (Optional):** Basic integrated GPU for rendering Angular animations

1.3.1.2 Server-Side Hardware

- **Processor:** Quad-Core 2.4 GHz or higher
- **RAM:** Minimum 8 GB (16 GB recommended for production)
- **Storage:** 50–100 GB SSD (for application files, logs, evidence uploads)
- **Network:** High-availability connection with static IP
- **Backup Storage:** Cloud or external server for PostgreSQL backups and log archives

1.3.2 Software Requirements

1.3.2.1 Client-Side Software

- **Operating System:**
  - Windows 10/11
  - macOS Catalina or later
  - Linux (Ubuntu/Fedora)
  - Android/iOS (for responsive web access)
- **Web Browsers (latest versions):**
  - Google Chrome
  - Mozilla Firefox
  - Microsoft Edge
  - Safari

1.3.2.2 Server-Side Software

- **Operating System:**
  - Linux
  - Windows Server 2019/2022
- **Backend Framework:**
  - ASP.NET Core (Primary backend technology)

- **Frontend Framework:**
    - Angular (with Angular CLI for module/component generation)
    - Bootstrap (for UI styling)

- **Database:**
    - PostgreSQL
    - pgAdmin (for admin and DB management)

- **Authentication:**
    - ASP.NET Identity
    - JWT + OAuth for secure API access

- **ORM:**
    - Entity Framework Core for database communication

- **Development Tools:**
    - Visual Studio 2022 / Visual Studio Code
    - Git with GitHub or Azure Repos

- **Runtime/SDK Requirements:**
    - .NET SDK 6/7
    - Node.js + npm (required for Angular)
    - Angular CLI

1.3.3 Constraints and Dependencies

1.3.3.1 Constraints

- **Technology Dependency:** System must use Angular for frontend, ASP.NET Core for backend, and PostgreSQL for data management.
- **Cross-Browser Requirement:** Must run on modern browsers that support Angular's latest rendering engine.
- **Authentication Rules:** Protected endpoints must require valid JWT tokens generated via ASP.NET Identity.
- **Security Constraints:** All communication must use HTTPS. Passwords must follow ASP.NET Identity complexity rules
- **File Upload Limitations:** Maximum size restrictions on evidence uploads (set by server configuration)
- **Server Scalability Constraints:** PostgreSQL performance depends on server RAM and disk I/O.
- **University Policy Constraints:** All workflows must follow DIU's proctorial and complaint-handling regulations.

1.3.3.2 Dependencies

- **Framework Dependencies:** Angular components depend on Node.js and Angular CLI. ASP.NET Core depends on .NET runtime and EF Core ORM.
- **Database Dependency:** PostgreSQL database must be online and accessible for complaint, evidence, and case operations.
- **Third-Party Authentication Services:** OAuth provider availability affects login and token generation.
- **Notification System Dependency:** Email/SMS gateway must be active for sending system alerts.
- **Version Control Dependency:** Git hosting services (GitHub/Azure) required for project tracking and collaboration.
- **Hosting Environment:** Deployment depends on stable Linux/Windows server with proper configuration.

Chapter 2: Design and Implementation

2.1 Functional Requirements

FR01	**Registration**
Description	Allows students and staff (proctor, admin, coordination officer) to register with full name, email, password, and role-specific information (e.g., department, student ID).
Stakeholder	Student, Admin, Proctor

FR02	**Login**
Description	Enables users to securely log in using their email and password. Role-based dashboards will be loaded based on their identity (student/admin/proctor).
Stakeholder	Student, Admin, Proctor, Coordination Officer

FR03	**Forgot Password**
Description	Users who forget their password can request a reset link via their registered email. The system will generate a secure token and guide users through resetting.
Stakeholder	All users

FR04	**Complaint Submission**
Description	Enables students to submit a complaint form detailing their issue. Includes input fields for description, incident type, and attachments.
Stakeholder	Student

FR05	**Complaint Forwarding**
Description	Automatically forwards the submitted complaint to the Coordination Officer and Proctor for review. Ensures the complaint reaches the appropriate authorities instantly.
Stakeholder	Coordination Officer, Proctor

FR06	Explanation Upload
Description	Allows both the complainant and the accused to upload their written explanations. These are stored and attached to the respective complaint case file.
Stakeholder	Student, Coordination Officer

FR07	Evidence Upload
Description	Supports uploading of evidence such as images, documents, and videos. Helps in building a strong and valid case file.
Stakeholder	Student, Coordination Officer

FR08	Complaint Assignment
Description	The Proctor reviews the complaint and assigns it to a designated Proctorial Team member. Assignment helps track progress and responsibility.
Stakeholder	Proctor

FR09	Case File Preparation
Description	Coordination Officer compiles all materials (explanation, evidence, etc.) into a structured case file. This file is then submitted to Proctor and deputies.
Stakeholder	Coordination Officer, Proctor

FR10	Meeting Scheduling
Description	Proctorial Team sets a meeting date and time to hear the case with both parties. The schedule is stored and notified to concerned users.
Stakeholder	Proctorial Team, Proctor

FR11	Notification System
Description	Notifies all users via dashboard and email about updates, meeting dates, and status changes. Helps keep users informed and on schedule.
Stakeholder	All Users

FR12	Dashboard Access
Description	Each user role has a unique dashboard showing relevant statistics and actions. E.g., students see their complaints, Coordination Officer sees case statuses.
Stakeholder	Student, Coordination Officer, Admin, Proctor

FR13	Audit Logging
Description	Logs key user actions such as complaint submission, evidence upload, and assignments. Useful for transparency and future audits.
Stakeholder	Admin

FR14	Logout
Description	Allows users to safely end their session and logout from the system. Ensures session clearance to protect user privacy.
Stakeholder	All Users

2.2 Non Functional Requirements

2.2.1 Security:

- Use JWT/OAuth and HTTPS for secure access.
- Role-based API and UI protection.
- Passwords encrypted with industry-standard hashing.

2.2.2 Performance:

- Asynchronous complaint handling and optimized queries.
- System should handle up to 500 concurrent users efficiently.

2.2.3 Scalability:

- Designed with modular backend and REST API.
- PostgreSQL and ASP.NET Core ensure scalable architecture.

2.2.4 Usability:

- Responsive UI (Bootstrap, Angular) for all devices.
- Clear workflows for each role

2.2.5 Maintainability:

- Modular codebase with ASP.NET Core best practices.
- Role, status, and category management configurable via admin panel.

2.2.6 Reliability:

- Daily backups and fail-safe mechanisms for PostgreSQL.
- Robust error handling with user-friendly messages.

2.2.7 Compatibility:

- Supports latest versions of Chrome, Firefox, Edge, Safari.
- Deployable in Linux-based Docker environments.

2.3 Object-oriented System design using UML

2.3.1 Use Case Diagram

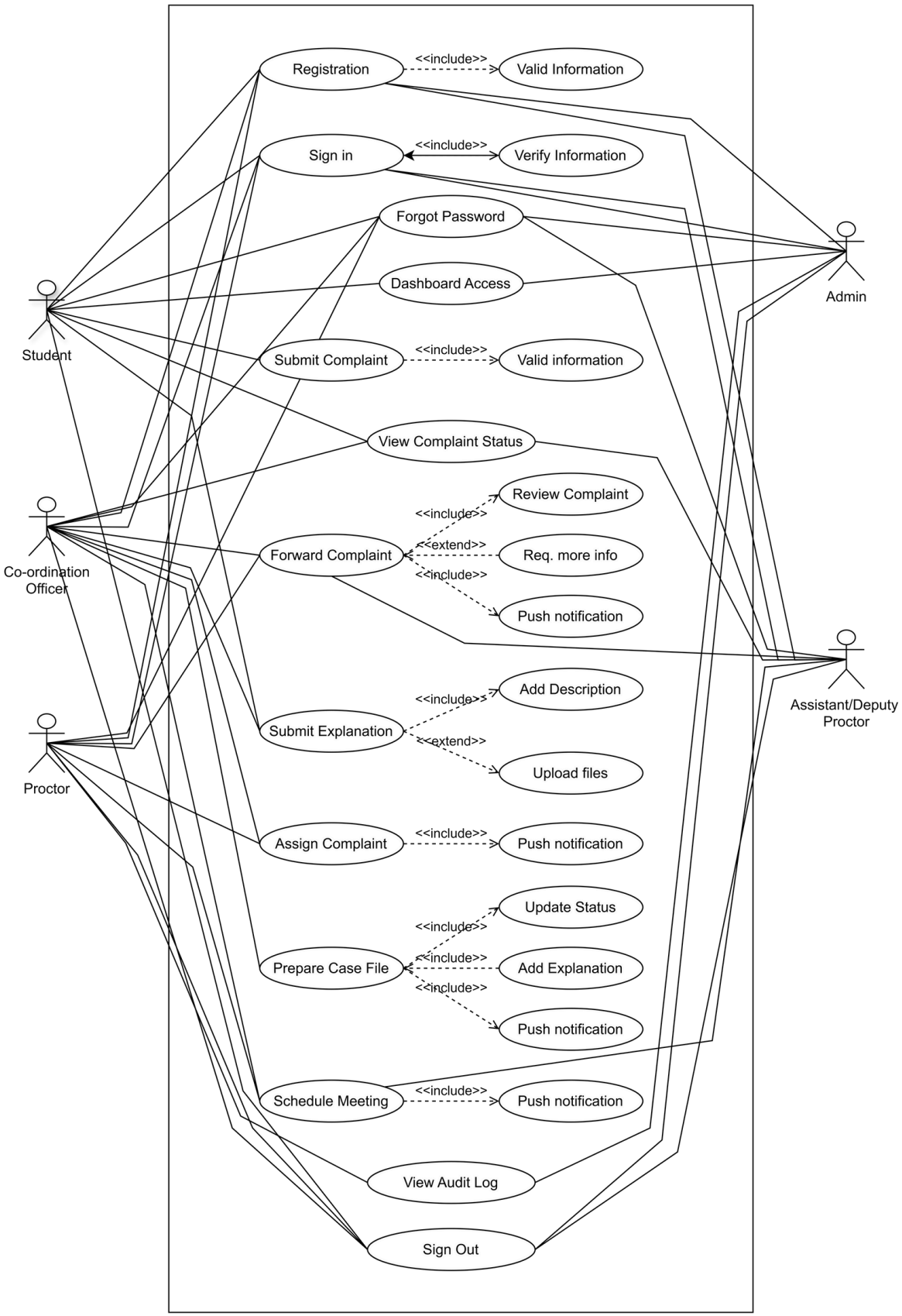


Figure 2.3.1: Use Case Diagram

2.3.2 Use Case Description

2.3.2.1 Case Description-01: Registration

<table>
<tr><td>Use Case</td><td>Registration</td></tr>
<tr><td>Goal</td><td>Users can register to sign in to the system.</td></tr>
<tr><td>Precondition</td><td>The system must be accessible.</td></tr>
<tr><td>Success End Condition</td><td>Notification: “Successfully Registered!”</td></tr>
<tr><td>Failed End Condition</td><td>Notification: “Registration Failed!”</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>Student, Proctor, Coordination Officer

Admin</td></tr>
<tr><td>Trigger</td><td>User clicks “Register” button.</td></tr>
<tr><td>Description / Main Success Scenario</td><td>
<table>
<tr><td>1.</td><td>Press “Registration” Button</td></tr>
<tr><td>2.</td><td>Provide registration form</td></tr>
<tr><td>3.</td><td>Enter Information</td></tr>
<tr><td>4.</td><td>Press “Submit” Button</td></tr>
<tr><td>5.</td><td>Information saved</td></tr>
<tr><td>6.</td><td>System registers user and shows success message</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows</td><td>
<table>
<tr><td>1.</td><td>Required fields are empty</td></tr>
<tr><td></td><td>1.1.a. Notification: “Please fill all fields!”</td></tr>
<tr><td>2.</td><td>Email already registered</td></tr>
<tr><td></td><td>2.1.a. Notification: “Email already exists. Try logging in.”</td></tr>
<tr><td>3.</td><td>System error during submission</td></tr>
<tr><td></td><td>3.1.a. Notification: “Registration failed. Try again later.”</td></tr>
<tr><td>4.</td><td>The system did not respond</td></tr>
<tr><td></td><td>4.1.a. Show Error Message.</td></tr>
<tr><td>5.</td><td>The system Doesn’t save the details.</td></tr>
<tr><td></td><td>5.1.a. Notification: “Details did not Saved”</td></tr>
</table>
</td></tr>
<tr><td>Quality Requirements</td><td>The user Will fill up all the details in 30 minutes.</td></tr>
</table>

Table 2.3.2.1: Case Description-01: Registration

2.3.2.2 Case Description-02: Login

<table>
<tr><td>Use Case</td><td>Login</td></tr>
<tr><td>Goal</td><td>Users log in to access their dashboard</td></tr>
<tr><td>Precondition</td><td>User must be registered</td></tr>
<tr><td>Success End Condition</td><td>Notification: “Login Successful”</td></tr>
<tr><td>Failed End Condition</td><td>Notification: “Login Failed!!”</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>Student, Proctor, Coordination Officer

Admin</td></tr>
<tr><td>Trigger</td><td>User clicks “Login” button.</td></tr>
<tr><td>Description / Main Success Scenario</td><td><table>
<tr><td>1.</td><td>Press “Login” Button</td></tr>
<tr><td>2.</td><td>Provide login interface</td></tr>
<tr><td>3.</td><td>Provide login credentials</td></tr>
<tr><td>4.</td><td>Press "Login"</td></tr>
<tr><td>5.</td><td>System verifies credentials</td></tr>
<tr><td>6.</td><td>System redirects to respective dashboard</td></tr>
</table></td></tr>
<tr><td>Alternative Flows</td><td><table>
<tr><td>1.</td><td>Email not registered</td></tr>
<tr><td></td><td>1.1.a. Notification: “Account not found”</td></tr>
<tr><td>2.</td><td>Incorrect password</td></tr>
<tr><td></td><td>2.1.a. Notification: “Incorrect password”</td></tr>
<tr><td>3.</td><td>The user did not fill up the details!</td></tr>
<tr><td></td><td>3.1.a. Checked By the system & Notify by “Please! Fill Up the required fields!”.</td></tr>
<tr><td>4.</td><td>The system did not respond</td></tr>
<tr><td></td><td>4.1.a. Show Error Message.</td></tr>
<tr><td>5.</td><td>System is down</td></tr>
<tr><td></td><td>Notification: “System unavailable. Try again later.”</td></tr>
</table></td></tr>
<tr><td>Quality Requirements</td><td>Users fill up the login info within 10 minutes.</td></tr>
</table>

Table 2.3.2.2: Case Description-02: Login

2.3.2.3 Case Description-03: Forgot Password

<table>
<tr><td>Use Case</td><td>Forgot Password</td></tr>
<tr><td>Goal</td><td>Reset login credentials</td></tr>
<tr><td>Precondition</td><td>User email must exist in the system</td></tr>
<tr><td>Success End Condition</td><td>Email sent with reset link</td></tr>
<tr><td>Failed End Condition</td><td>Email not recognized</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>Student, Proctor, Coordination Officer

Admin</td></tr>
<tr><td>Trigger</td><td>User clicks “Forgot Password” button.</td></tr>
<tr><td>Description / Main Success Scenario</td><td><table>
<tr><td>1.</td><td>Press “Forgot Password” Button</td></tr>
<tr><td>2.</td><td>User enters email</td></tr>
<tr><td>3.</td><td>System validates email</td></tr>
<tr><td>4.</td><td>Sends reset link</td></tr>
<tr><td>5.</td><td>Set new password</td></tr>
<tr><td>6.</td><td>Press “Save & Apply”.</td></tr>
</table></td></tr>
<tr><td>Alternative Flows</td><td><table>
<tr><td>2.</td><td>Email doesn’t exist</td></tr>
<tr><td></td><td>2.1.a. Notification: “No account associated with this email”</td></tr>
<tr><td>4.</td><td>Server error</td></tr>
<tr><td></td><td>4.1.a. Notification: “Couldn’t send reset link. Try again.”</td></tr>
<tr><td>6.</td><td>System is down</td></tr>
<tr><td></td><td>6.1.aNotification: “System unavailable. Try again later.”</td></tr>
</table></td></tr>
<tr><td>Quality Requirements</td><td>Reset link must be valid for 15 minutes.</td></tr>
</table>

Table 2.3.2.3: Case Description-03: Forgot Password

2.3.2.4 Case Description-04: Complaint Submission

<table>
<tr><td>Use Case</td><td>Complaint Submission</td></tr>
<tr><td>Goal</td><td>Submit complaint with details</td></tr>
<tr><td>Precondition</td><td>User must be logged in</td></tr>
<tr><td>Success End Condition</td><td>Complaint ID or Case ID generated</td></tr>
<tr><td>Failed End Condition</td><td>Form not submitted</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>Student

System</td></tr>
<tr><td>Trigger</td><td>User clicks “File a Complaint” button.</td></tr>
<tr><td>Description / Main Success Scenario</td><td><table>
<tr><td>1.</td><td>Press “File a Complaint” Button</td></tr>
<tr><td>2.</td><td>Fill in complaint details</td></tr>
<tr><td>3.</td><td>Attach optional files</td></tr>
<tr><td>4.</td><td>Click on “Submit” button</td></tr>
<tr><td>5.</td><td>Complaint is saved in database</td></tr>
<tr><td>6.</td><td>System notifies Coordination Officer</td></tr>
</table></td></tr>
<tr><td>Alternative Flows</td><td><table>
<tr><td>3.</td><td>Attachment type/size is invalid</td></tr>
<tr><td></td><td>3.1.a. Notification: “Invalid file type or size exceeded”</td></tr>
<tr><td>4.</td><td>Required fields are empty</td></tr>
<tr><td></td><td>4.1.a. Notification: “Please fill all required fields”</td></tr>
<tr><td>5.</td><td>Server timeout</td></tr>
<tr><td></td><td>5.1.a Notification: “Submission failed. Please try again”</td></tr>
</table></td></tr>
<tr><td>Quality Requirements</td><td>Submission takes less than 30 minutes.</td></tr>
</table>

Table 2.3.2.4: Case Description-04: Complaint Submission

2.3.2.5 Case Description-05: Complaint Forwarding

<table>
<tr><td>Use Case</td><td>Complaint Forwarding</td></tr>
<tr><td>Goal</td><td>Forward complaint to Coordination Officer</td></tr>
<tr><td>Precondition</td><td>Complaint submitted by the user</td></tr>
<tr><td>Success End Condition</td><td>Complaint is forwarded and visible to designated staff</td></tr>
<tr><td>Failed End Condition</td><td>Forwarding fails</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>Coordination Officer

Assistant Proctor, Deputy Proctor</td></tr>
<tr><td>Trigger</td><td>Officer clicks “Forward” on a pending complaint</td></tr>
<tr><td>Description / Main Success Scenario</td><td><table>
<tr><td>1.</td><td>Press “Manage Cases” Button</td></tr>
<tr><td>2.</td><td>Officer views new complaint</td></tr>
<tr><td>3.</td><td>Clicks “Forward”</td></tr>
<tr><td>4.</td><td>Selects recipient (Assistant or Deputy Proctor)</td></tr>
<tr><td>5.</td><td>System sends to selected recipient and updates status</td></tr>
</table></td></tr>
<tr><td>Alternative Flows</td><td><table>
<tr><td>4.</td><td>No recipient selected</td></tr>
<tr><td></td><td>4.1.a. Notification: “Please select a staff member to forward”</td></tr>
<tr><td>5.</td><td>Network issue during forwarding</td></tr>
<tr><td></td><td>5.1.a Notification: “Forwarding failed. Try again later”</td></tr>
</table></td></tr>
<tr><td>Quality Requirements</td><td>Forwarding should happen instantly.</td></tr>
</table>

Table 2.3.2.5: Case Description-05: Complaint Forwarding

2.3.2.6 Case Description-06: Explanation Upload

<table>
<tr><td>Use Case</td><td>Explanation Upload</td></tr>
<tr><td>Goal</td><td>Accused and complainant can upload their explanation</td></tr>
<tr><td>Precondition</td><td>Complaint assigned</td></tr>
<tr><td>Success End Condition</td><td>Explanation saved</td></tr>
<tr><td>Failed End Condition</td><td>Upload failed</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>Student (Accused, Complainant)

Coordination Officer</td></tr>
<tr><td>Trigger</td><td>Staff clicks on “Upload Explanation”</td></tr>
<tr><td>Description / Main Success Scenario</td><td><table>
<tr><td>1.</td><td>Staff clicks on “Upload Explanation”</td></tr>
<tr><td>2.</td><td>Attaches file or types explanation</td></tr>
<tr><td>3.</td><td>Presses “Submit”</td></tr>
<tr><td>4.</td><td>System saves the file and links it with complaint</td></tr>
<tr><td>5.</td><td>Notify the proctor</td></tr>
</table></td></tr>
<tr><td>Alternative Flows</td><td><table>
<tr><td>2.</td><td>File format is invalid</td></tr>
<tr><td></td><td>2.1.a. Notification: “Only PDF or DOCX allowed”</td></tr>
<tr><td>3.</td><td>No file or text provided</td></tr>
<tr><td></td><td>3.1.a. Notification: “Explanation cannot be empty”</td></tr>
<tr><td>4.</td><td>Database error</td></tr>
<tr><td></td><td>4.1.a. Notification: “Upload failed. Please retry”</td></tr>
</table></td></tr>
<tr><td>Quality Requirements</td><td>Only PDF/DOC under 5MB accepted.</td></tr>
</table>

Table 2.3.2.6: Case Description-06: Explanation Upload

2.3.2.7 Case Description-07: Evidence Upload

<table>
<tr><td>Use Case</td><td>Evidence Upload</td></tr>
<tr><td>Goal</td><td>Students can attach supporting evidence to a complaint</td></tr>
<tr><td>Precondition</td><td>Complaint must be created or accessible</td></tr>
<tr><td>Success End Condition</td><td>Evidence is uploaded successfully</td></tr>
<tr><td>Failed End Condition</td><td>Upload fails</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>Student

Coordination Officer</td></tr>
<tr><td>Trigger</td><td>User selects “Add Evidence”</td></tr>
<tr><td>Description / Main Success Scenario</td><td><table>
<tr><td>1.</td><td>User opens complaint details</td></tr>
<tr><td>2.</td><td>Clicks “Add Evidence”</td></tr>
<tr><td>3.</td><td>Attaches file (image, document, audio)</td></tr>
<tr><td>4.</td><td>System validates and saves file</td></tr>
<tr><td>5.</td><td>Notify the Coordination Officer</td></tr>
</table></td></tr>
<tr><td>Alternative Flows</td><td><table>
<tr><td>3.</td><td>File type not supported</td></tr>
<tr><td></td><td>3.1.a. Notification: “Unsupported file format”</td></tr>
<tr><td></td><td>3.2.a. Notification: “File size too large”</td></tr>
<tr><td>4.</td><td>Upload fails due to system error</td></tr>
<tr><td></td><td>4.1.a. Notification: “Upload unsuccessful. Try again later”</td></tr>
</table></td></tr>
<tr><td>Quality Requirements</td><td>Accept only JPG, PDF, DOCX.</td></tr>
</table>

Table 2.3.2.7: Case Description-07: Evidence Upload

2.3.2.8 Case Description-08: Complaint Assignment

<table>
<tr><td>Use Case</td><td>Complaint Assignment</td></tr>
<tr><td>Goal</td><td>Proctor assigns unresolved complaints to appropriate staff for resolution</td></tr>
<tr><td>Precondition</td><td>Complaint must be forwarded but unassigned</td></tr>
<tr><td>Success End Condition</td><td>Complaint is successfully assigned and logged</td></tr>
<tr><td>Failed End Condition</td><td>Assignment fails or is unrecorded</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>Proctor

Coordination Officer, Deputy Proctor, Assistant Proctor</td></tr>
<tr><td>Trigger</td><td>Proctor clicks “Assign” on an unassigned complaint</td></tr>
<tr><td>Description / Main Success Scenario</td><td>
<table>
<tr><td>1.</td><td>Proctor views list of unassigned complaints</td></tr>
<tr><td>2.</td><td>Clicks on “Assign” button</td></tr>
<tr><td>3.</td><td>Selects an available staff member</td></tr>
<tr><td>4.</td><td>System logs assignment and notifies selected staff</td></tr>
<tr><td>5.</td><td>Notify the Coordination Officer, Deputy Proctor, Assistant Proctor</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows</td><td>
<table>
<tr><td>3.</td><td>No staff selected</td></tr>
<tr><td></td><td>3.1.a. Notification: “Select someone to assign this complaint”</td></tr>
<tr><td>4.</td><td>Staff already overloaded</td></tr>
<tr><td></td><td>4.1.a. Notification: “User cannot handle more cases currently”</td></tr>
<tr><td>5.</td><td>Server issue</td></tr>
<tr><td></td><td>5.1.a. Notification: “Assignment failed. Try again later”</td></tr>
</table>
</td></tr>
<tr><td>Quality Requirements</td><td>Assignment completed in under 10 mins.</td></tr>
</table>

Table 2.3.2.8: Case Description-08: Complaint Assignment

2.3.2.9 Case Description-09: Case File Preparation

<table>
<tr><td>Use Case</td><td>Case File Preparation</td></tr>
<tr><td>Goal</td><td>Staff compiles evidence, explanation, and reports into a single case file</td></tr>
<tr><td>Precondition</td><td>Complaint must have all required documents uploaded</td></tr>
<tr><td>Success End Condition</td><td>Case file prepared and saved</td></tr>
<tr><td>Failed End Condition</td><td>Case file incomplete or not saved</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>Assistant Proctor, Deputy Proctor

Proctor</td></tr>
<tr><td>Trigger</td><td>Staff clicks “Prepare Case File” on assigned complaint</td></tr>
<tr><td>Description / Main Success Scenario</td><td><table>
<tr><td>1.</td><td>Staff opens complaint details</td></tr>
<tr><td>2.</td><td>Clicks “Prepare Case File”</td></tr>
<tr><td>3.</td><td>Selects included items (evidence, notes, explanation)</td></tr>
<tr><td>4.</td><td>System compiles into one document and saves</td></tr>
<tr><td>5.</td><td>Notify Proctor</td></tr>
</table></td></tr>
<tr><td>Alternative Flows</td><td><table>
<tr><td>3.</td><td>Some required content is missing</td></tr>
<tr><td></td><td>3.1.a. Notification: “Incomplete complaint – cannot prepare file”</td></tr>
<tr><td>4.</td><td>File generation error</td></tr>
<tr><td></td><td>4.1.a. Notification: “Failed to generate file. Try again”</td></tr>
<tr><td>5.</td><td>Server issue</td></tr>
<tr><td></td><td>5.1.a. Notification: “Try again later”</td></tr>
</table></td></tr>
<tr><td>Quality Requirements</td><td>File size must be under 10MB.</td></tr>
</table>

Table 2.3.2.9: Case Description-09: Case File Preparation

2.3.2.10 Case Description-10: Meeting Scheduling

<table>
<tr><td>Use Case</td><td>Meeting Scheduling</td></tr>
<tr><td>Goal</td><td>Proctor schedules hearing or discussion meetings with stakeholders</td></tr>
<tr><td>Precondition</td><td>Complaint must be in "Ready for Hearing" status</td></tr>
<tr><td>Success End Condition</td><td>Meeting scheduled and participants notified</td></tr>
<tr><td>Failed End Condition</td><td>Meeting not scheduled</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>Proctor

Student, Coordination Officer, Assigned Staff</td></tr>
<tr><td>Trigger</td><td>Proctor clicks “Schedule Meeting”</td></tr>
<tr><td>Description / Main Success Scenario</td><td><table>
<tr><td>1.</td><td>Proctor opens complaint</td></tr>
<tr><td>2.</td><td>Clicks “Schedule Meeting”</td></tr>
<tr><td>3.</td><td>Selects date, time, and participants</td></tr>
<tr><td>4.</td><td>System confirms meeting</td></tr>
<tr><td>5.</td><td>Notify all</td></tr>
</table></td></tr>
<tr><td>Alternative Flows</td><td><table>
<tr><td>3.</td><td>Date/time not selected</td></tr>
<tr><td></td><td>3.1.a. Notification: “Please choose date and time”</td></tr>
<tr><td>4.</td><td>Server issue</td></tr>
<tr><td></td><td>4.1.a. Notification: “Try again later”</td></tr>
<tr><td>5.</td><td>Notification system fails</td></tr>
<tr><td></td><td>5.1.a. Notification: “Meeting scheduled but notification delivery failed”</td></tr>
</table></td></tr>
<tr><td>Quality Requirements</td><td>Meetings must be set 48 hrs prior.</td></tr>
</table>

Table 2.3.2.10: Case Description-10: Meeting Scheduling

2.3.2.11 Case Description-11: Push Notification

<table>
<tr><td>Use Case</td><td>Push Notification</td></tr>
<tr><td>Goal</td><td>Deliver real-time notifications to users about relevant actions</td></tr>
<tr><td>Precondition</td><td>Trigger event occurs (e.g., complaint update, assignment, meeting)</td></tr>
<tr><td>Success End Condition</td><td>Notification is delivered</td></tr>
<tr><td>Failed End Condition</td><td>Notification is not sent or fails to reach user</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>System

All user roles (Student, Proctor, Staff)</td></tr>
<tr><td>Trigger</td><td>System detects an action requiring notification</td></tr>
<tr><td>Description / Main Success Scenario</td><td><table>
<tr><td>1.</td><td>Action occurs (e.g., complaint is forwarded or explanation uploaded)</td></tr>
<tr><td>2.</td><td>System generates notification</td></tr>
<tr><td>3.</td><td>Notification is sent to appropriate user(s)</td></tr>
<tr><td>4.</td><td>User views the notification</td></tr>
</table></td></tr>
<tr><td>Alternative Flows</td><td><table>
<tr><td>3.</td><td>Network/server failure</td></tr>
<tr><td></td><td>3.1.a. Notification: “Could not send notification. Try again later”</td></tr>
<tr><td>4.</td><td>User does not check dashboard</td></tr>
<tr><td></td><td>4.1.a. Notification remains unseen but marked unread</td></tr>
</table></td></tr>
<tr><td>Quality Requirements</td><td>All notifications must be sent within 5 seconds.</td></tr>
</table>

Table 2.3.2.11: Case Description-11: Push Notification

2.3.2.12 Case Description-12: Dashboard Access

<table>
<tr><td>Use Case</td><td>Dashboard Access</td></tr>
<tr><td>Goal</td><td>Users access a personalized dashboard displaying relevant data and actions</td></tr>
<tr><td>Precondition</td><td>User must be logged in</td></tr>
<tr><td>Success End Condition</td><td>Dashboard loads successfully with real-time data</td></tr>
<tr><td>Failed End Condition</td><td>Dashboard fails to load or shows incorrect data</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>Student, Proctor, Coordination Officer, Deputy/Assistant Proctor

Admin</td></tr>
<tr><td>Trigger</td><td>User logs in and accesses dashboard</td></tr>
<tr><td>Description / Main Success Scenario</td><td><table>
<tr><td>1.</td><td>User logs in</td></tr>
<tr><td>2.</td><td>System authenticates and identifies user role</td></tr>
<tr><td>3.</td><td>Dashboard loads based on role-specific components</td></tr>
<tr><td>4.</td><td>User interacts with dashboard features</td></tr>
</table></td></tr>
<tr><td>Alternative Flows</td><td><table>
<tr><td>2.</td><td>Session timeout or token error</td></tr>
<tr><td></td><td>2.1.a. Notification: “Session expired. Please log in again”</td></tr>
<tr><td>3.</td><td>Data retrieval fails</td></tr>
<tr><td></td><td>3.1.a. Notification: “Error loading dashboard. Try refreshing”</td></tr>
</table></td></tr>
<tr><td>Quality Requirements</td><td>Dashboard loads under 10 seconds.</td></tr>
</table>

Table 2.3.2.12: Case Description-12: Dashboard Access

2.3.2.13 Case Description-13: Audit Logging

<table>
<tr><td>Use Case</td><td>Audit Logging</td></tr>
<tr><td>Goal</td><td>Record all system interactions for tracking and security</td></tr>
<tr><td>Precondition</td><td>Action must be performed by a registered user</td></tr>
<tr><td>Success End Condition</td><td>Log is saved in the database</td></tr>
<tr><td>Failed End Condition</td><td>Action not logged</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>System

Admin, Proctor (for viewing logs)</td></tr>
<tr><td>Trigger</td><td>Any user action (login, complaint submission, assignment, etc.)</td></tr>
<tr><td>Description / Main Success Scenario</td><td><table><tr><td>1.</td><td>User performs an action (e.g., submits complaint)</td></tr><tr><td>2.</td><td>System creates a log entry</td></tr><tr><td>3.</td><td>Entry is stored with timestamp and user ID</td></tr></table></td></tr>
<tr><td>Alternative Flows</td><td><table><tr><td>2.</td><td>Log table is unavailable</td></tr><tr><td></td><td>2.1.a. Notification: “Audit record could not be saved”</td></tr><tr><td>3.</td><td>Database write error</td></tr><tr><td></td><td>3.1.a. System retries logging in 5 seconds or logs to backup</td></tr></table></td></tr>
<tr><td>Quality Requirements</td><td>Logs must be stored for 12 months.</td></tr>
</table>

Table 2.3.2.13: Case Description-13: Audit Logging

2.3.2.14 Case Description-14: Logout

<table>
<tr><td>Use Case</td><td>Logout</td></tr>
<tr><td>Goal</td><td>Users safely exit the system</td></tr>
<tr><td>Precondition</td><td>User must be logged in</td></tr>
<tr><td>Success End Condition</td><td>Redirected to login page</td></tr>
<tr><td>Failed End Condition</td><td>Session is not ended</td></tr>
<tr><td>Primary Actors:

Secondary Actors:</td><td>All registered users

Admin</td></tr>
<tr><td>Trigger</td><td>User clicks “Logout”</td></tr>
<tr><td>Description / Main Success Scenario</td><td><table>
<tr><td>1.</td><td>User clicks logout</td></tr>
<tr><td>2.</td><td>User ensures to confirm logout</td></tr>
<tr><td>3.</td><td>System ends session</td></tr>
<tr><td>4.</td><td>Redirect to login screen</td></tr>
</table></td></tr>
<tr><td>Alternative Flows</td><td><table>
<tr><td>2.</td><td>Server not responding</td></tr>
<tr><td></td><td>2.1.a. Notification: “Logout failed. Please try again.”</td></tr>
<tr><td>3.</td><td>Server down</td></tr>
<tr><td></td><td>3.1.a. Try again</td></tr>
</table></td></tr>
<tr><td>Quality Requirements</td><td>Session must end within 5 second.</td></tr>
</table>

Table 2.3.2.14: Case Description-14: Logout

2.3.3. Activity Diagram

2.3.3.1 Activity Diagram-01: Registration

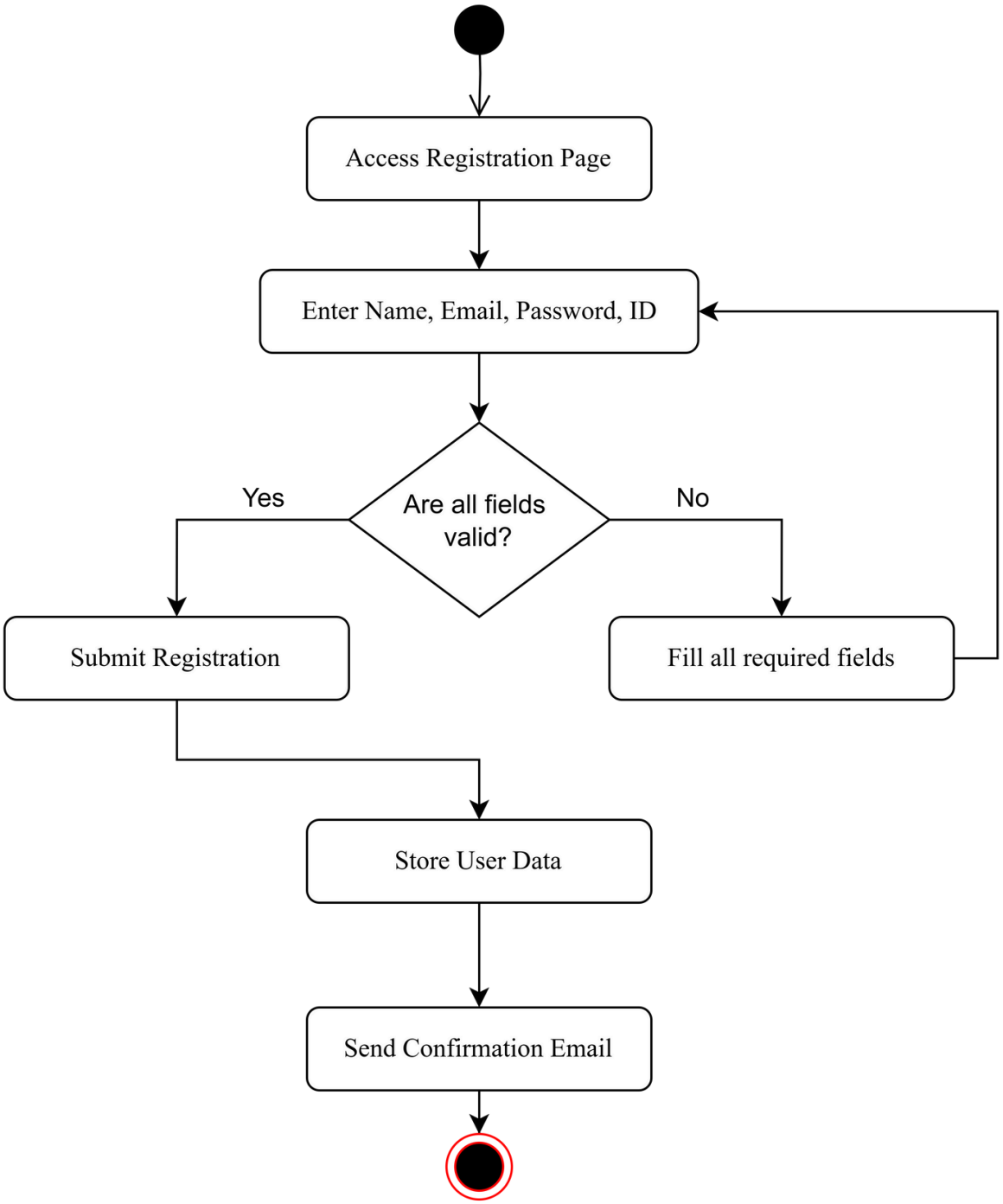


Figure 2.3.3.1: Activity Diagram-01: Registration

2.3.3.2 Activity Diagram-02: Sign in

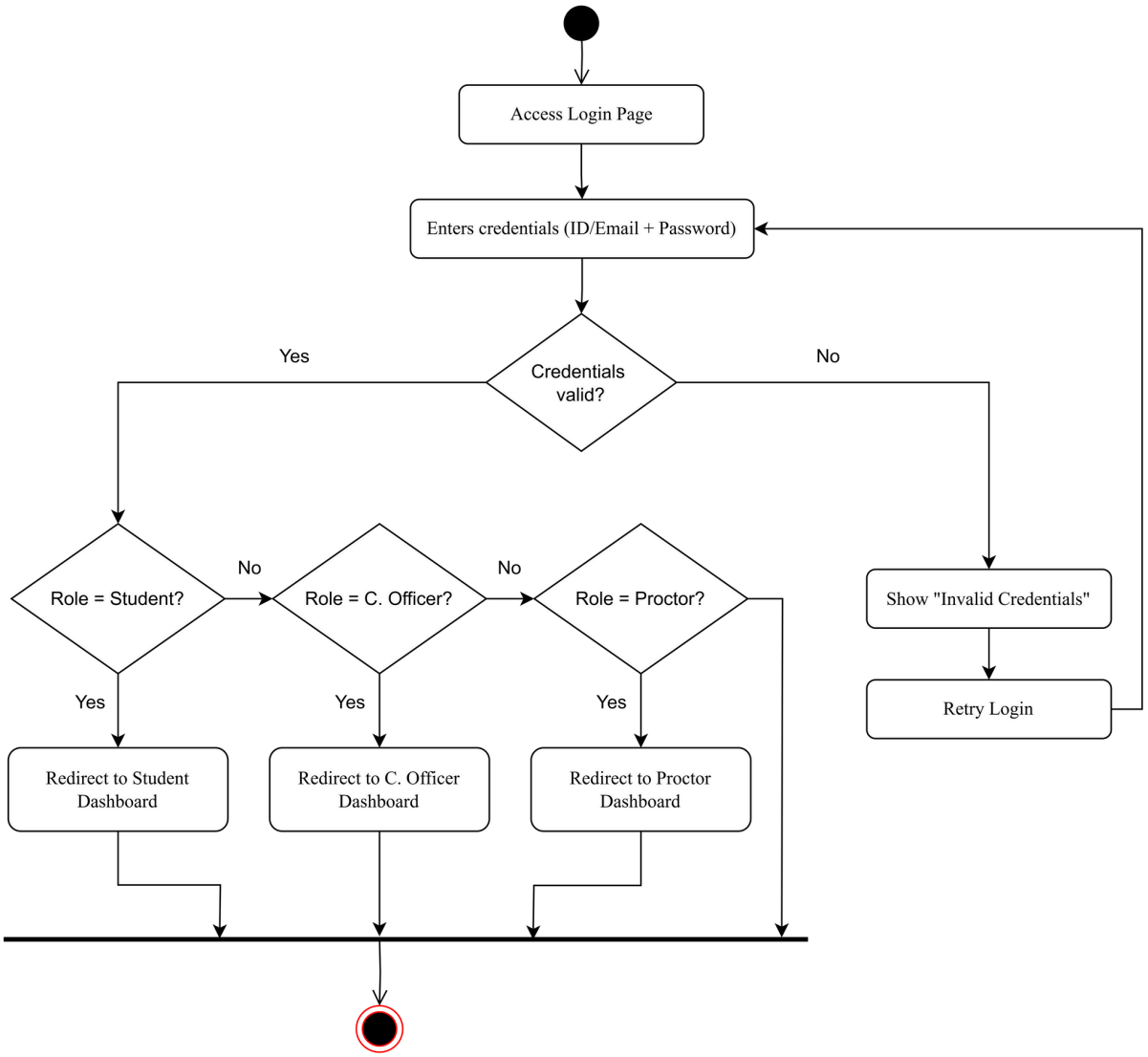


Figure 2.3.3.2: Activity Diagram-02: Sign in

2.3.3.3 Activity Diagram-03: Forgot Password

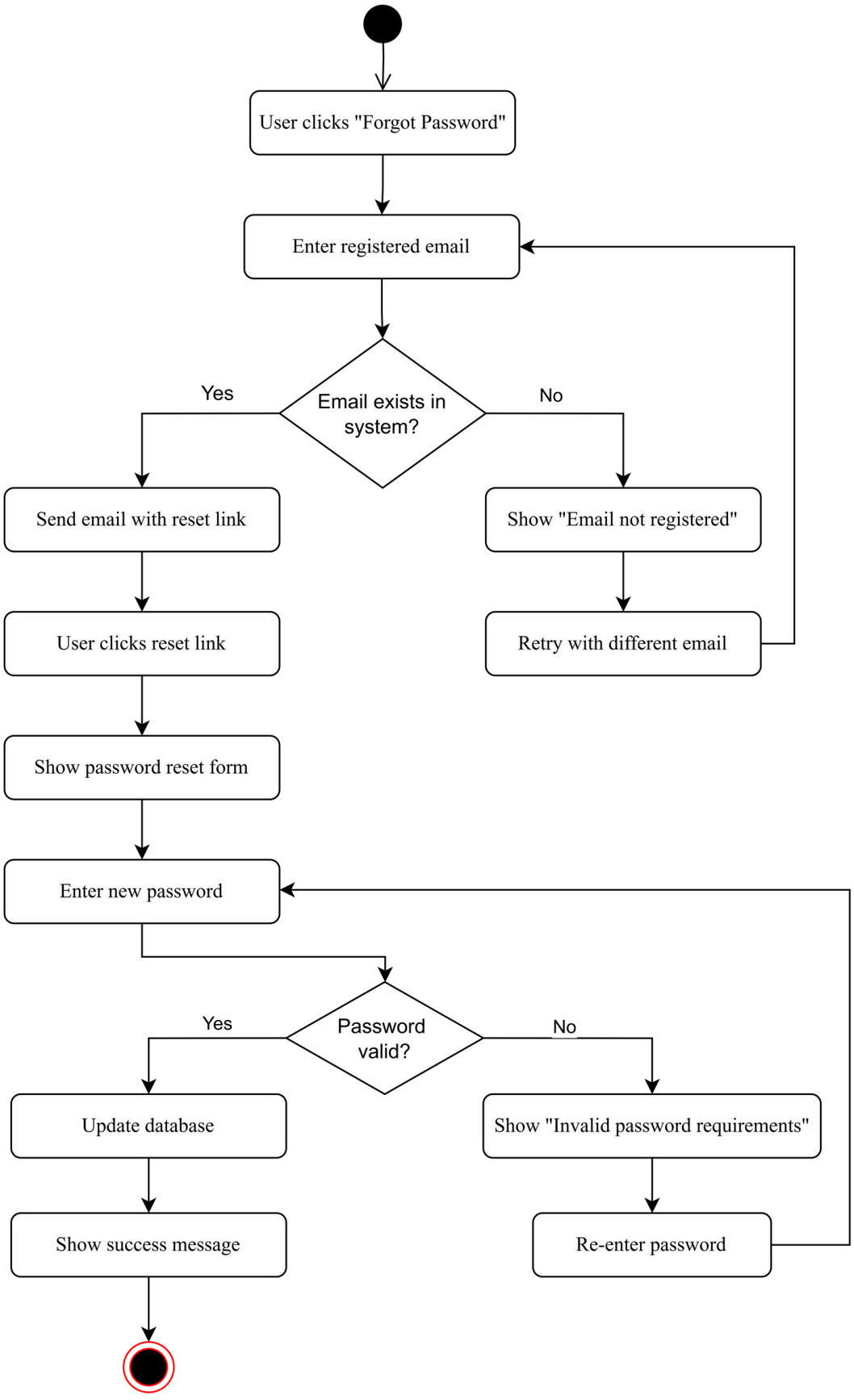


Figure 2.3.3.3: Activity Diagram--03: Forgot Password

2.3.3.4 Activity Diagram-04: Complaint Submission

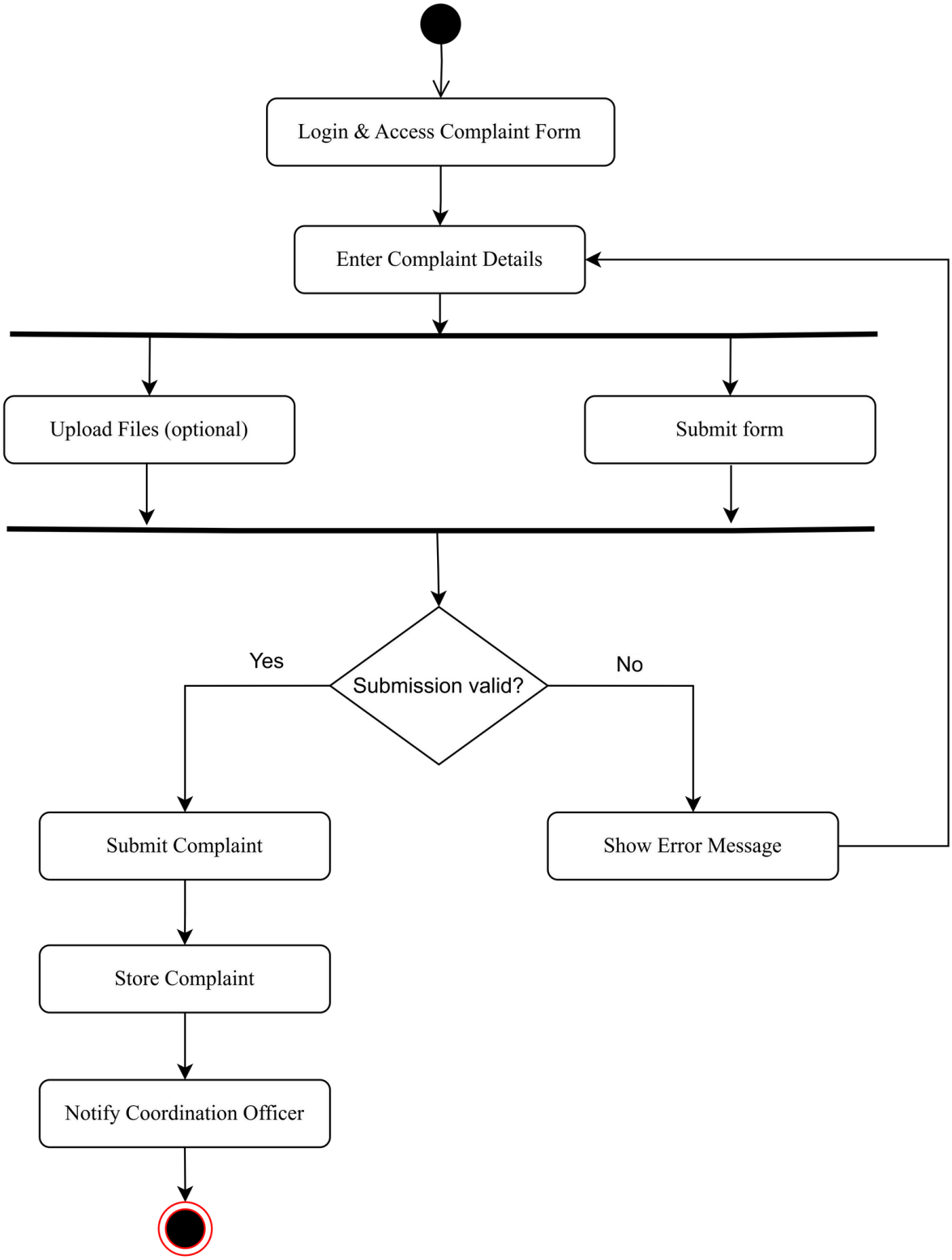


Figure 2.3.3.4: Activity Diagram--04: Complaint Submission

2.3.3.5 Activity Diagram-05: Complaint Forwarding

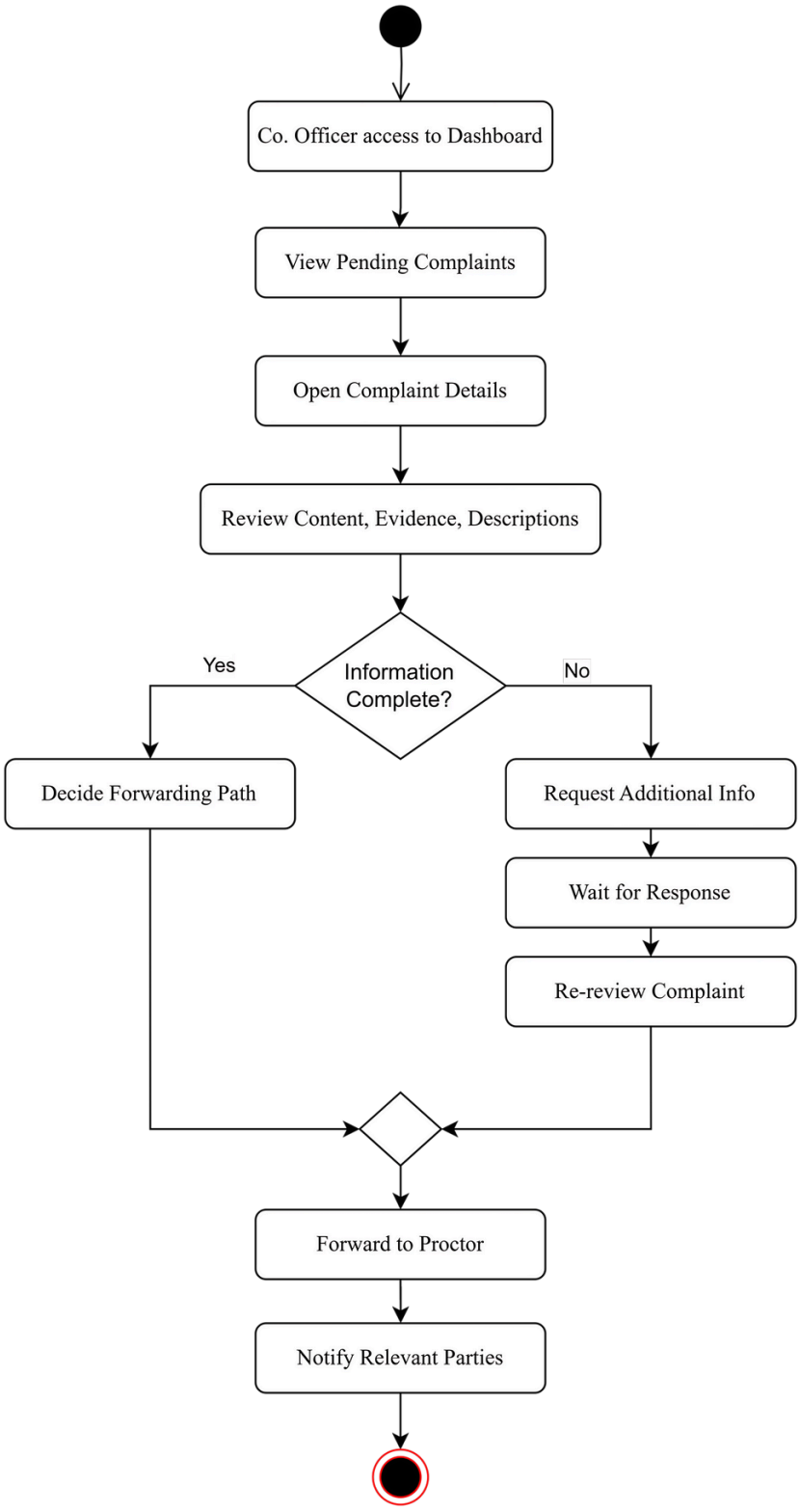


Figure 2.3.3.5: Activity Diagram--05: Complaint Forwarding

2.3.3.6 Activity Diagram-06: Explanation Upload

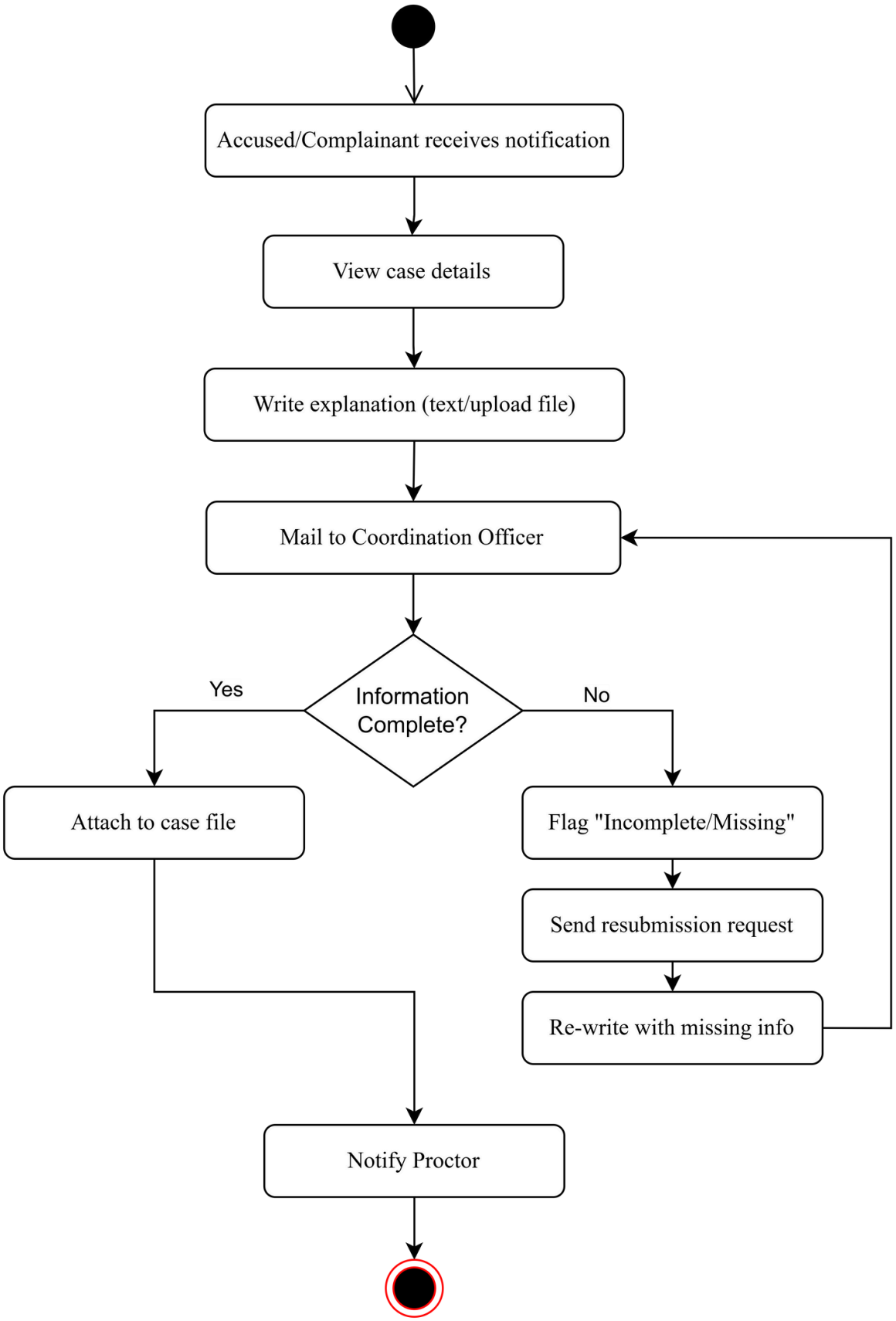


Figure 2.3.3.6: Activity Diagram--06: Explanation Upload

2.3.3.7 Activity Diagram-07: Evidence Upload

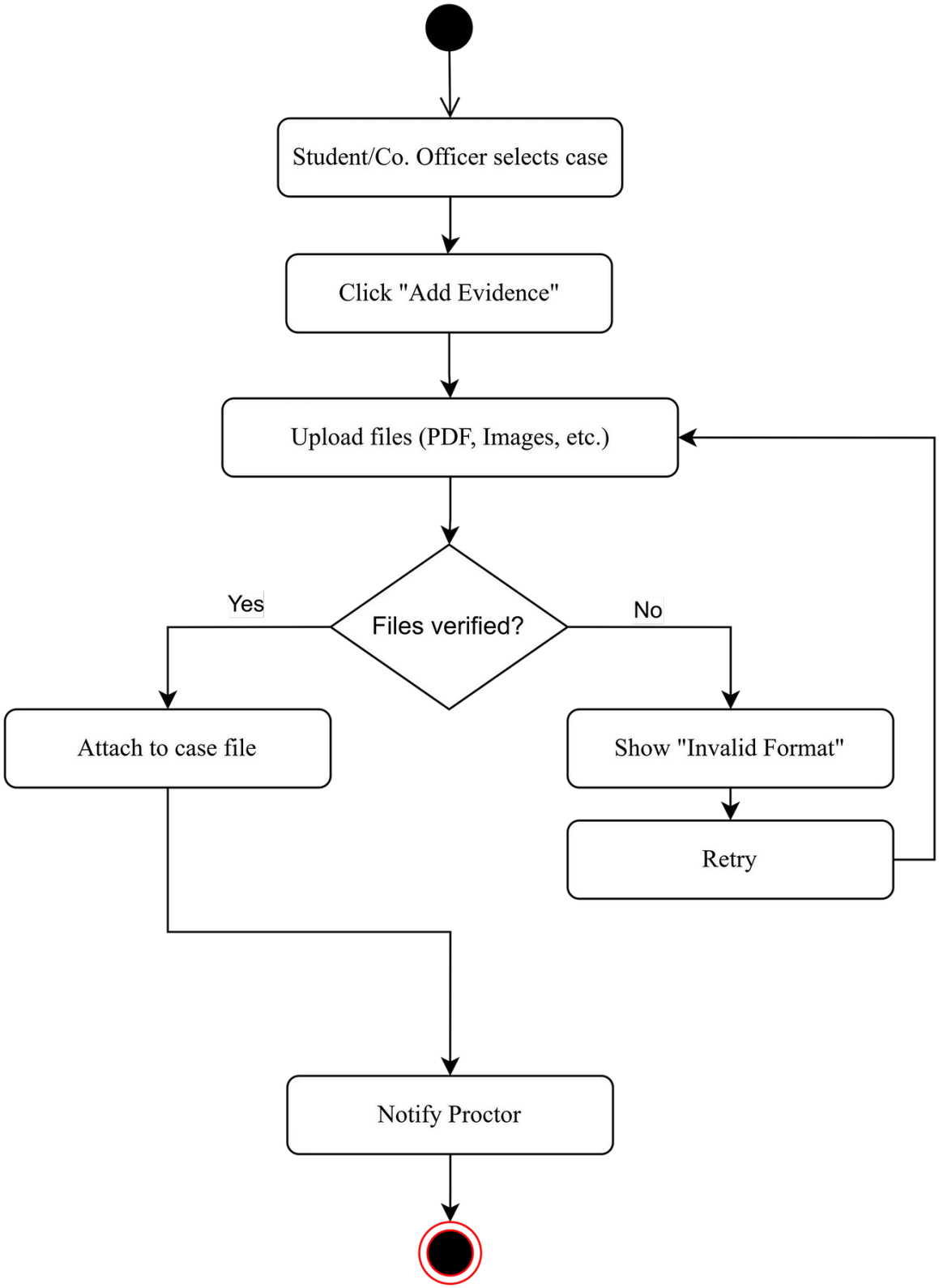


Figure 2.3.3.7: Activity Diagram--07: Evidence Upload

2.3.3.8 Activity Diagram-08: Complaint Assignment

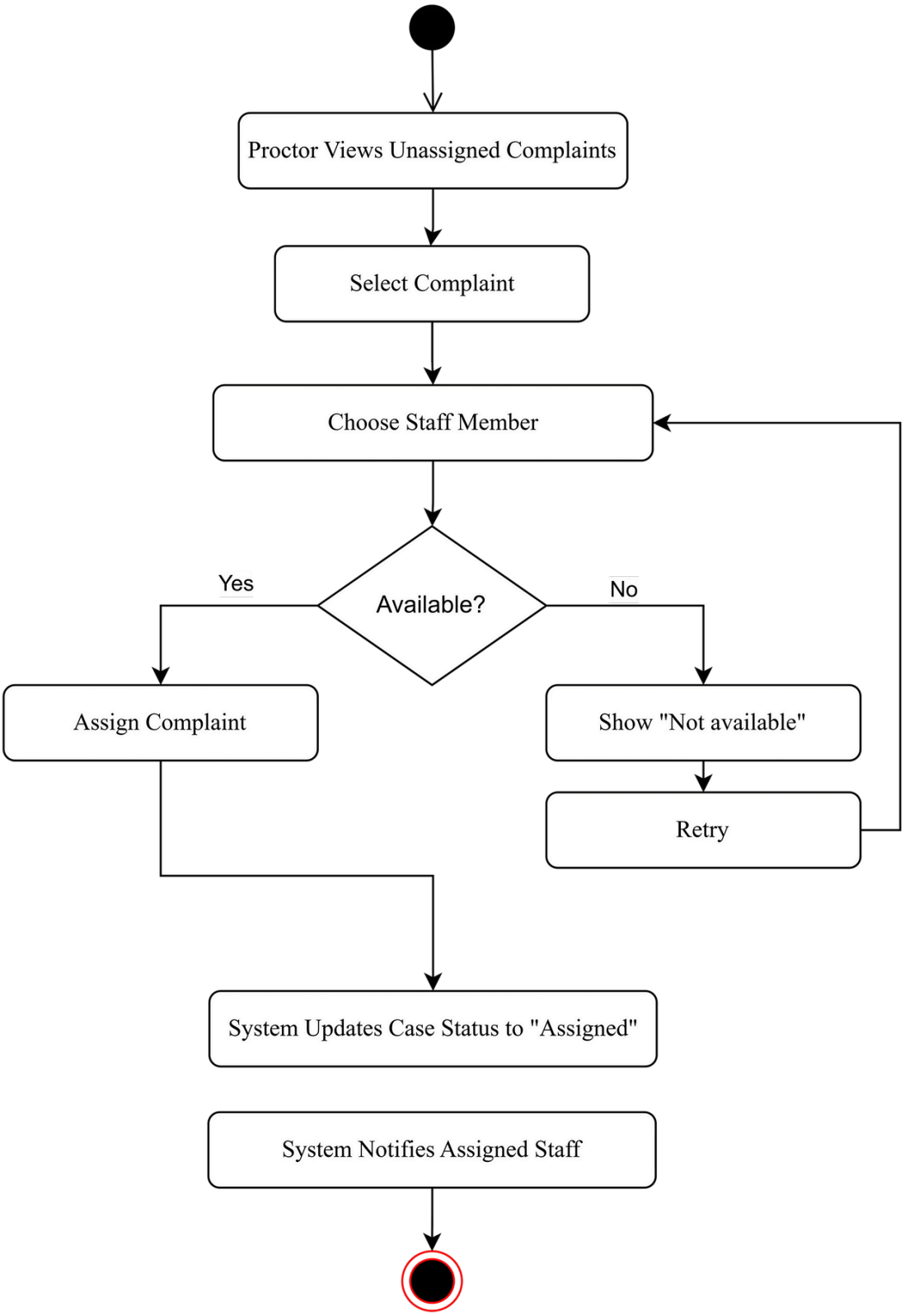


Figure 2.3.3.8: Activity Diagram--08: Complaint Assignment

2.3.3.9 Activity Diagram-09: Case File Preparation

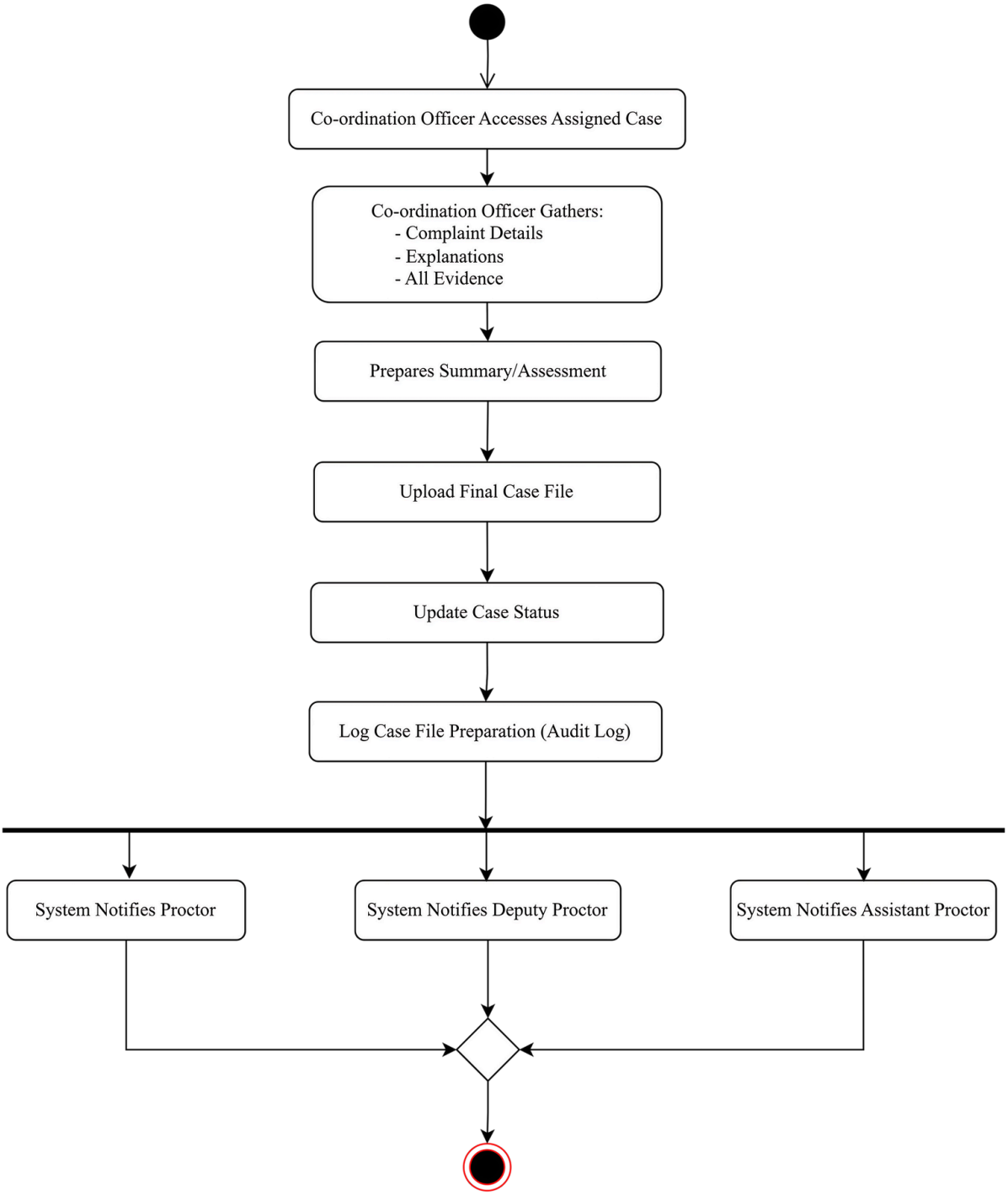


Figure 2.3.3.9: Activity Diagram--09: Case File Preparation

2.3.3.10 Activity Diagram-10: Meeting Scheduling

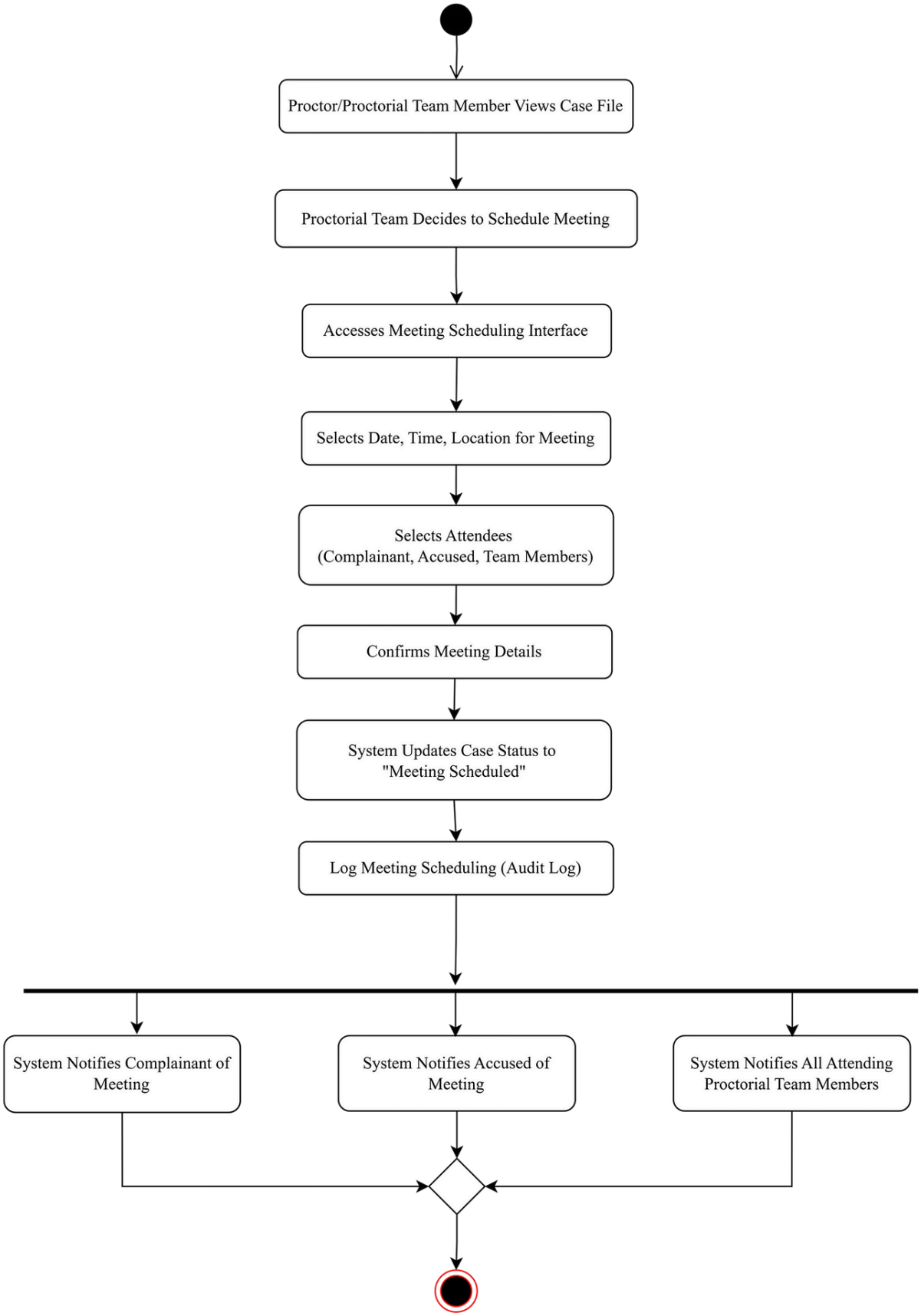


Figure 2.3.3.10: Activity Diagram--10: Meeting Scheduling

2.3.3.11 Activity Diagram-11: Notification System

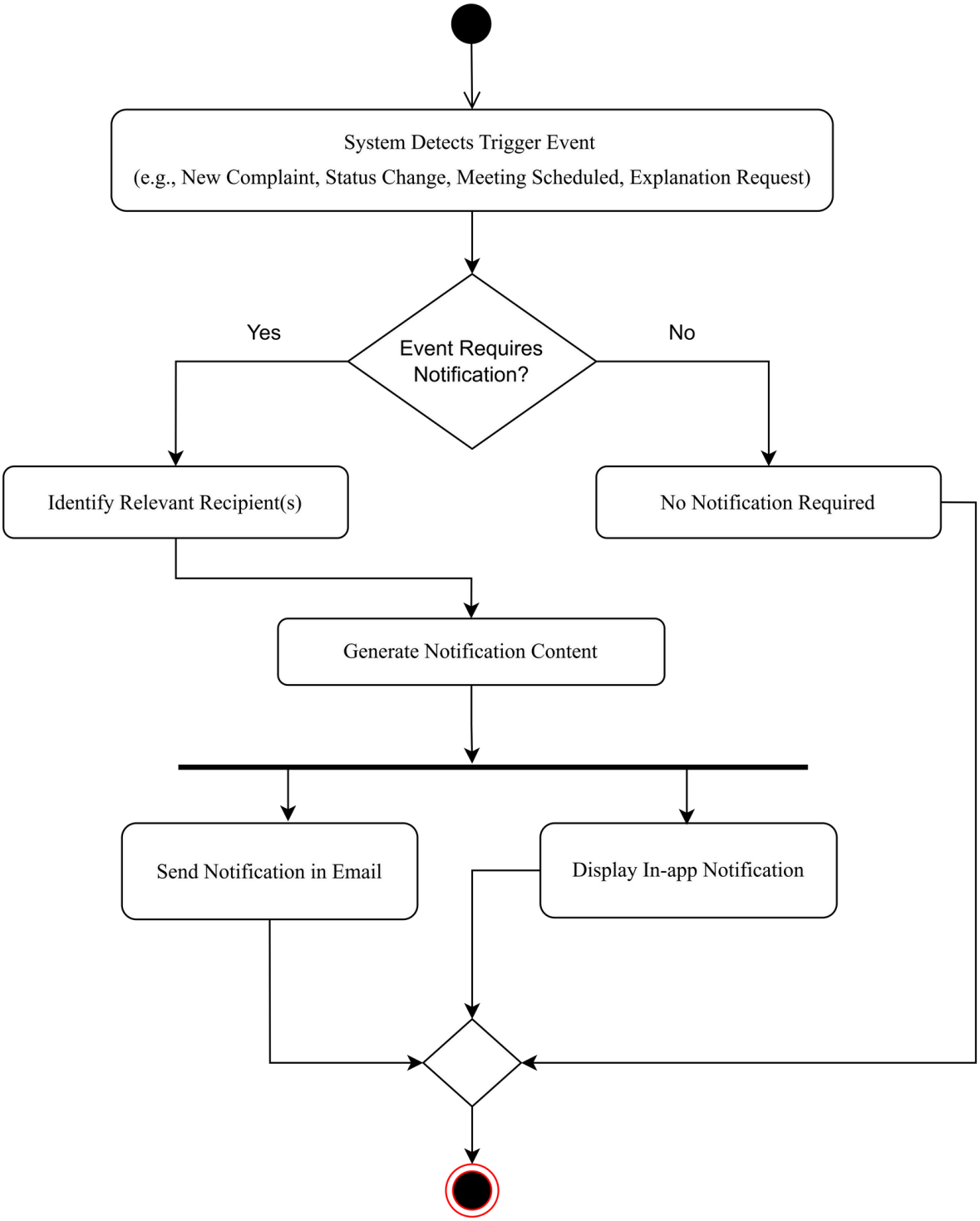


Figure 2.3.3.11: Activity Diagram--11: Notification System

2.3.3.12 Activity Diagram-12: Dashboard Access

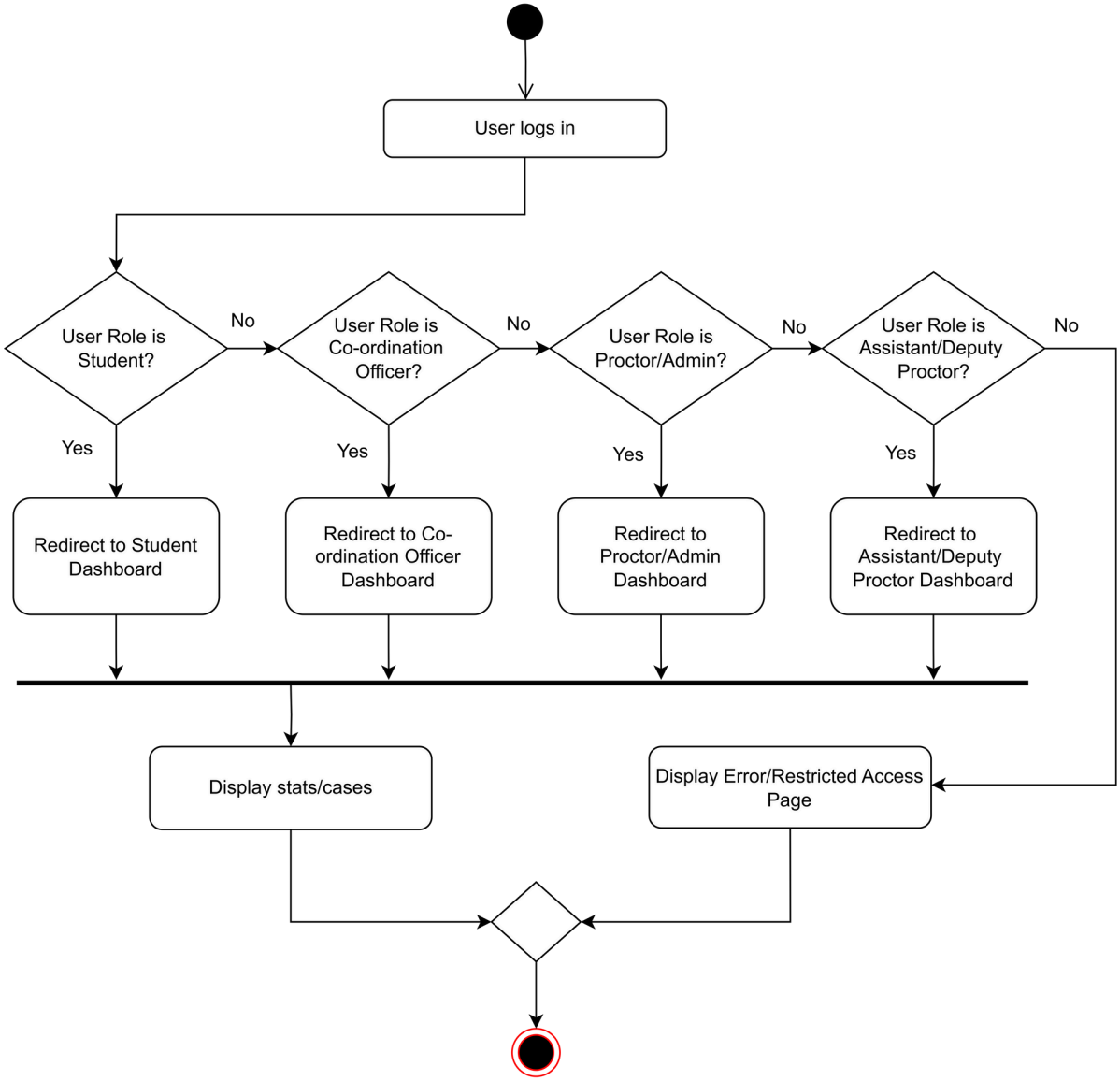


Figure 2.3.3.12: Activity Diagram--12: Dashboard Access

2.3.3.13 Activity Diagram-13: Audit Logging

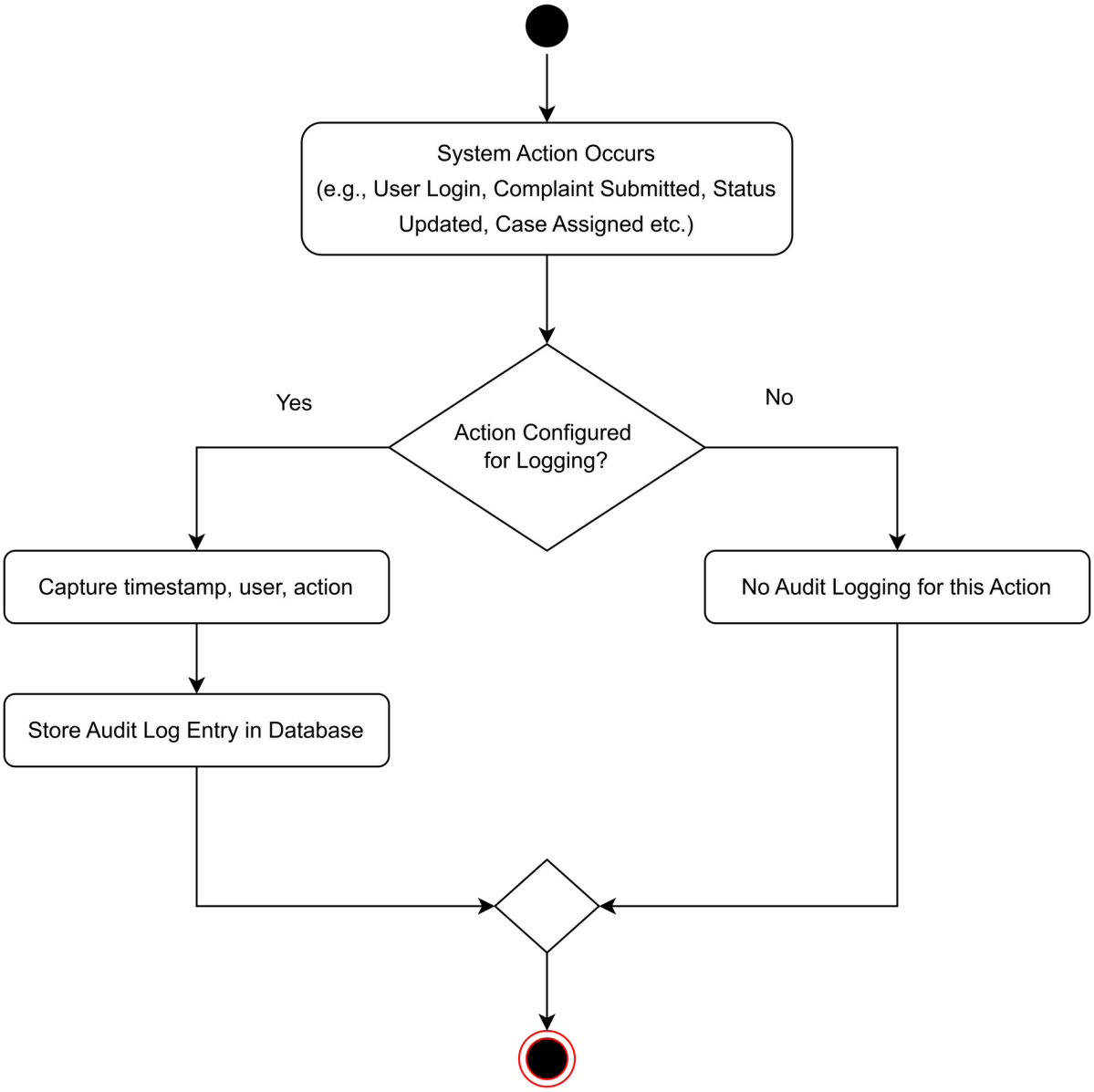


Figure 2.3.3.13: Activity Diagram--13: Audit Logging

2.3.3.14 Activity Diagram-14: Sign out

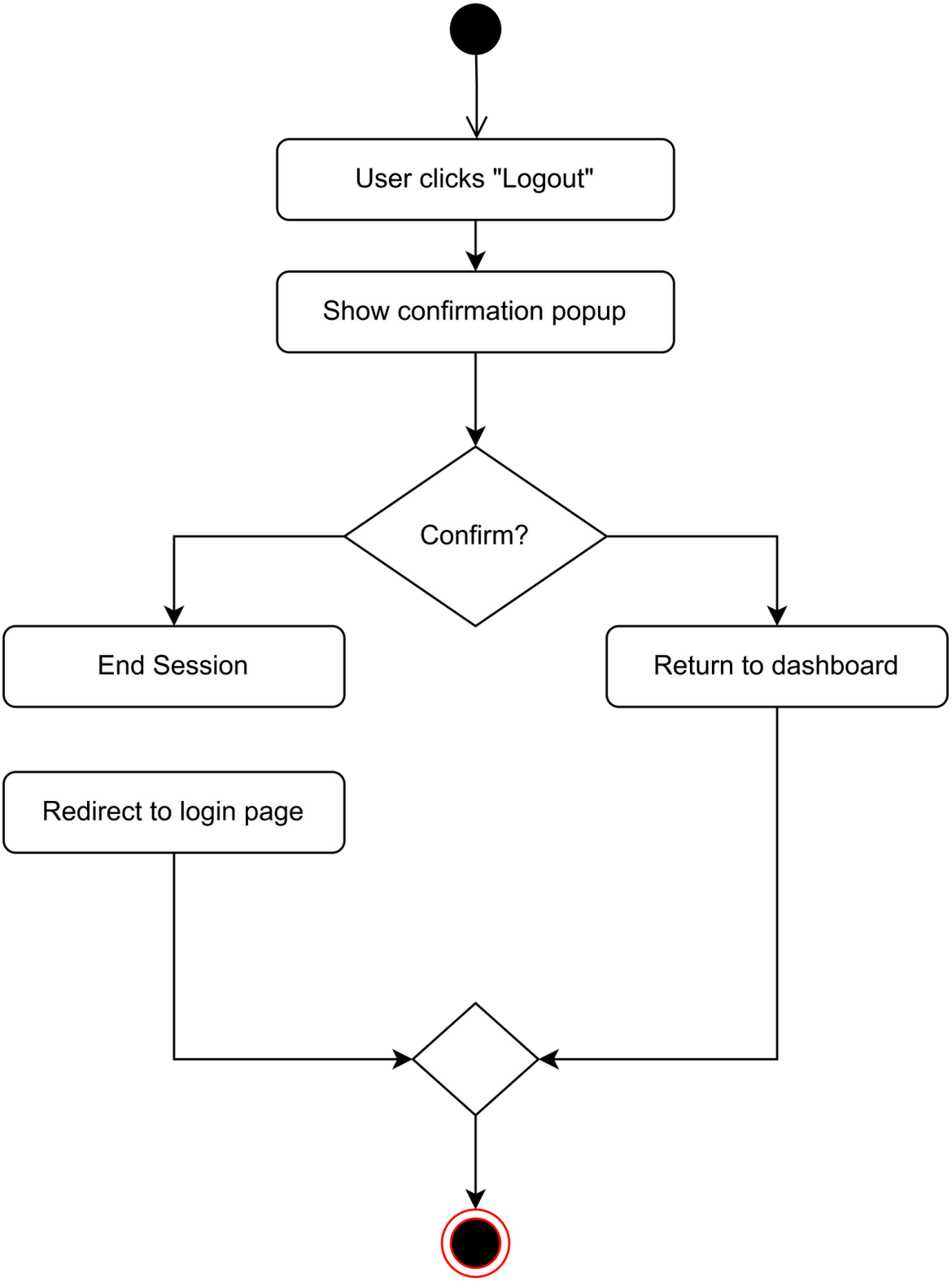


Figure 2.3.3.14: Activity Diagram--13: Audit Logging

2.3.4. Sequence Diagram

2.3.4.1 Sequence Diagram-01: Registration

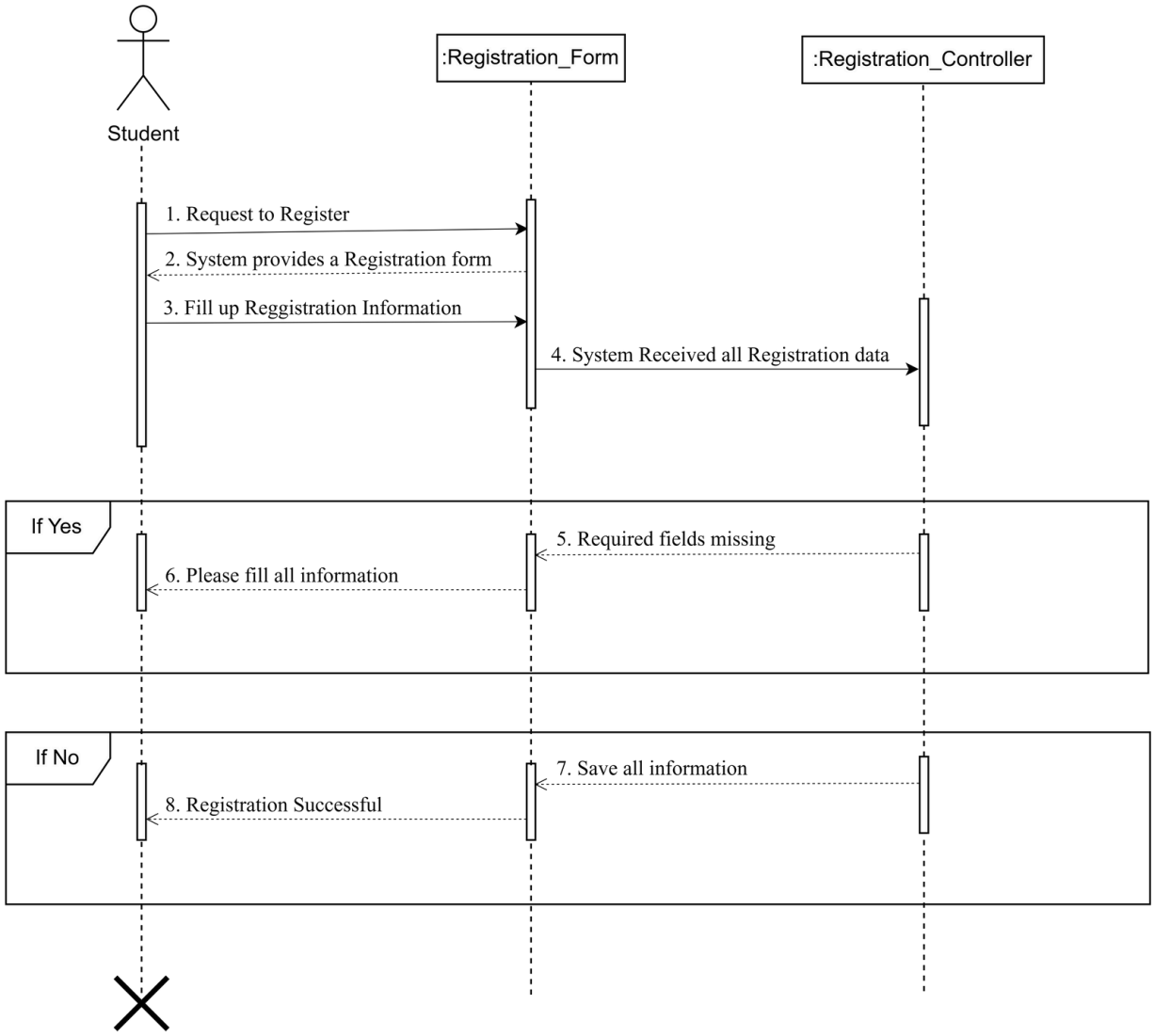


Figure 2.3.4.1: Sequence Diagram-01: Registration

2.3.4.2 Sequence Diagram-02: Sign in

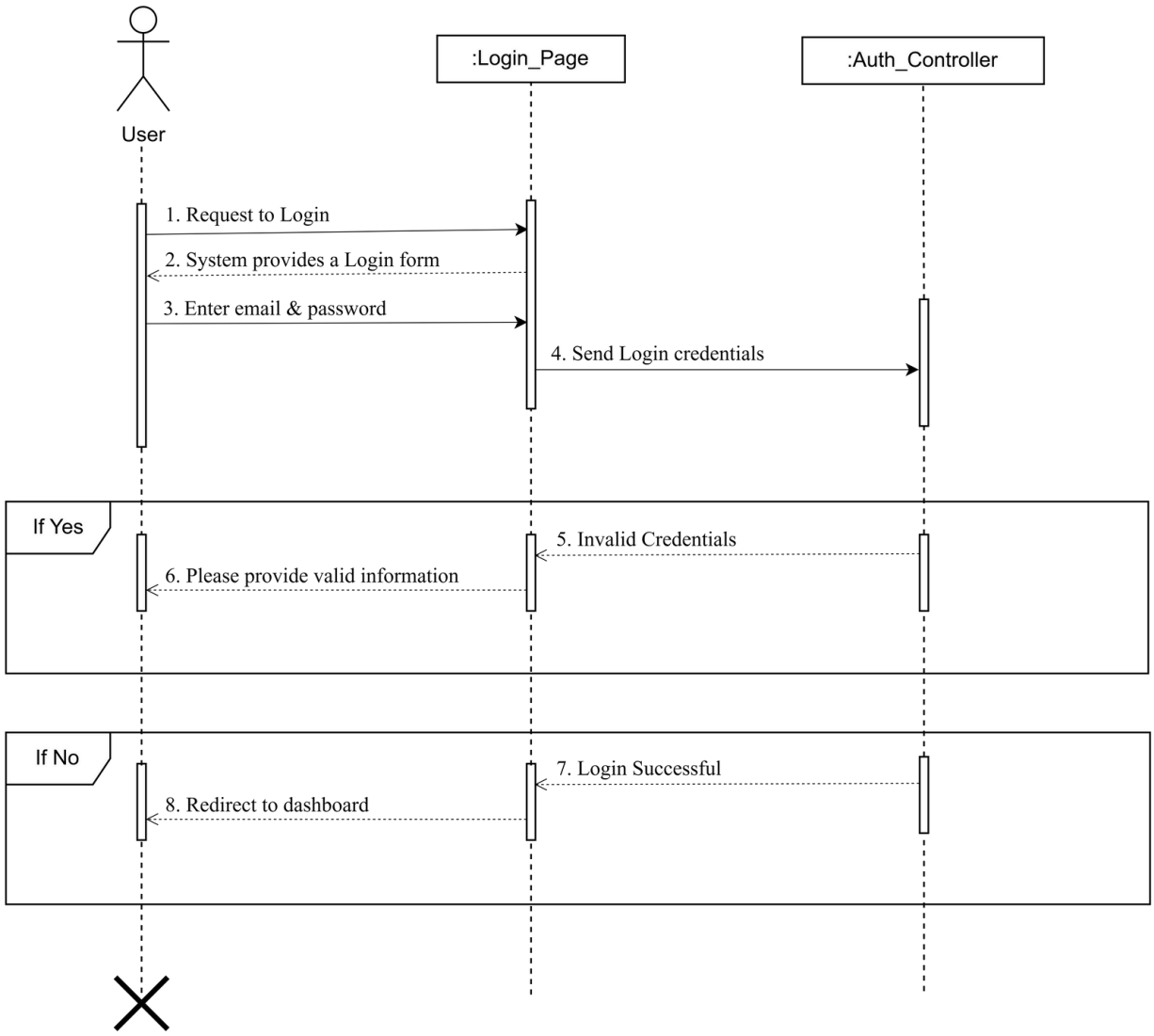


Figure 2.3.4.2: Sequence Diagram-02: Sign in

2.3.4.3 Sequence Diagram-03: Forgot Password

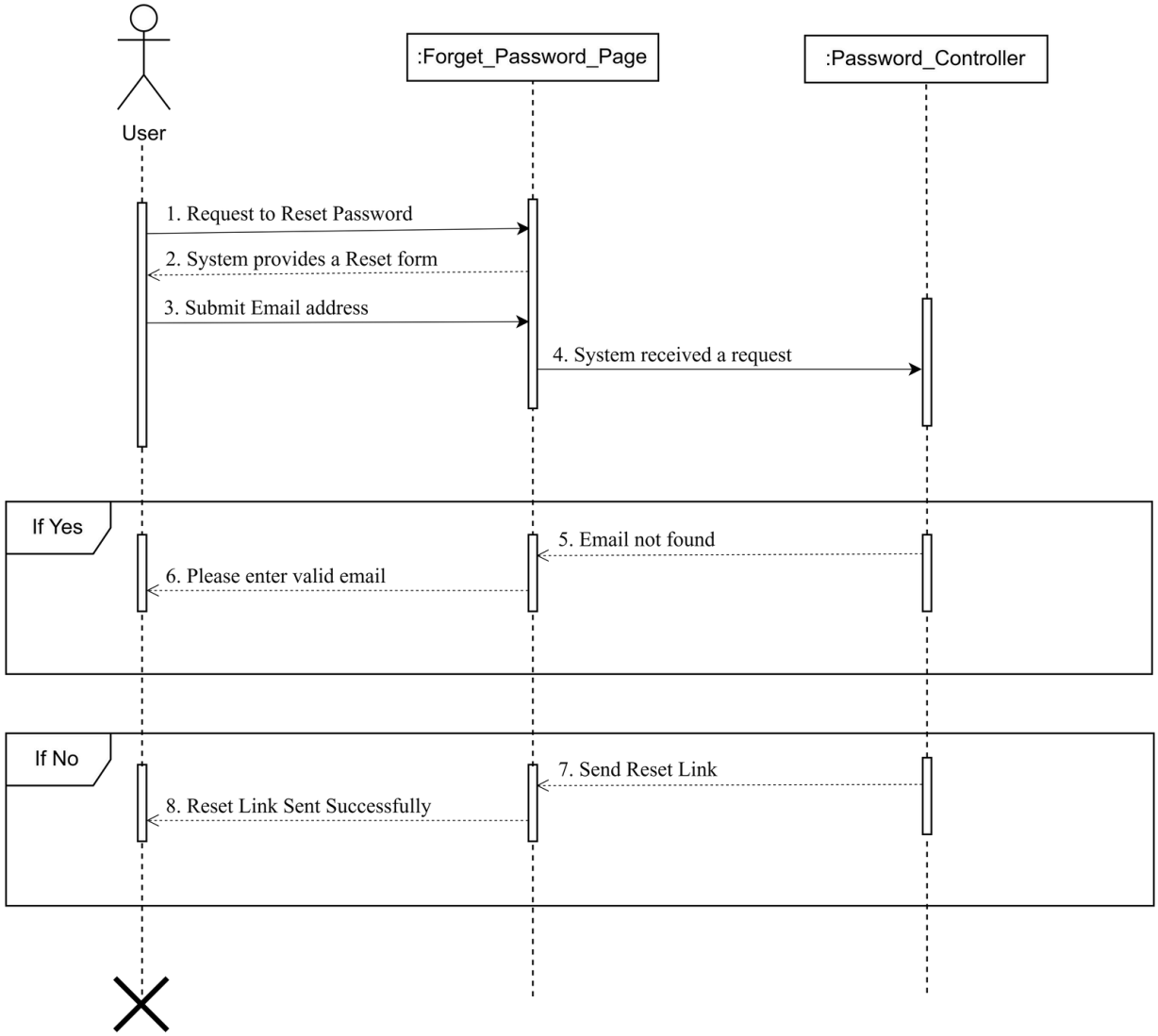


Figure 2.3.4.3: Sequence Diagram-03: Forgot Password

2.3.4.4 Sequence Diagram-04: Complaint Submission

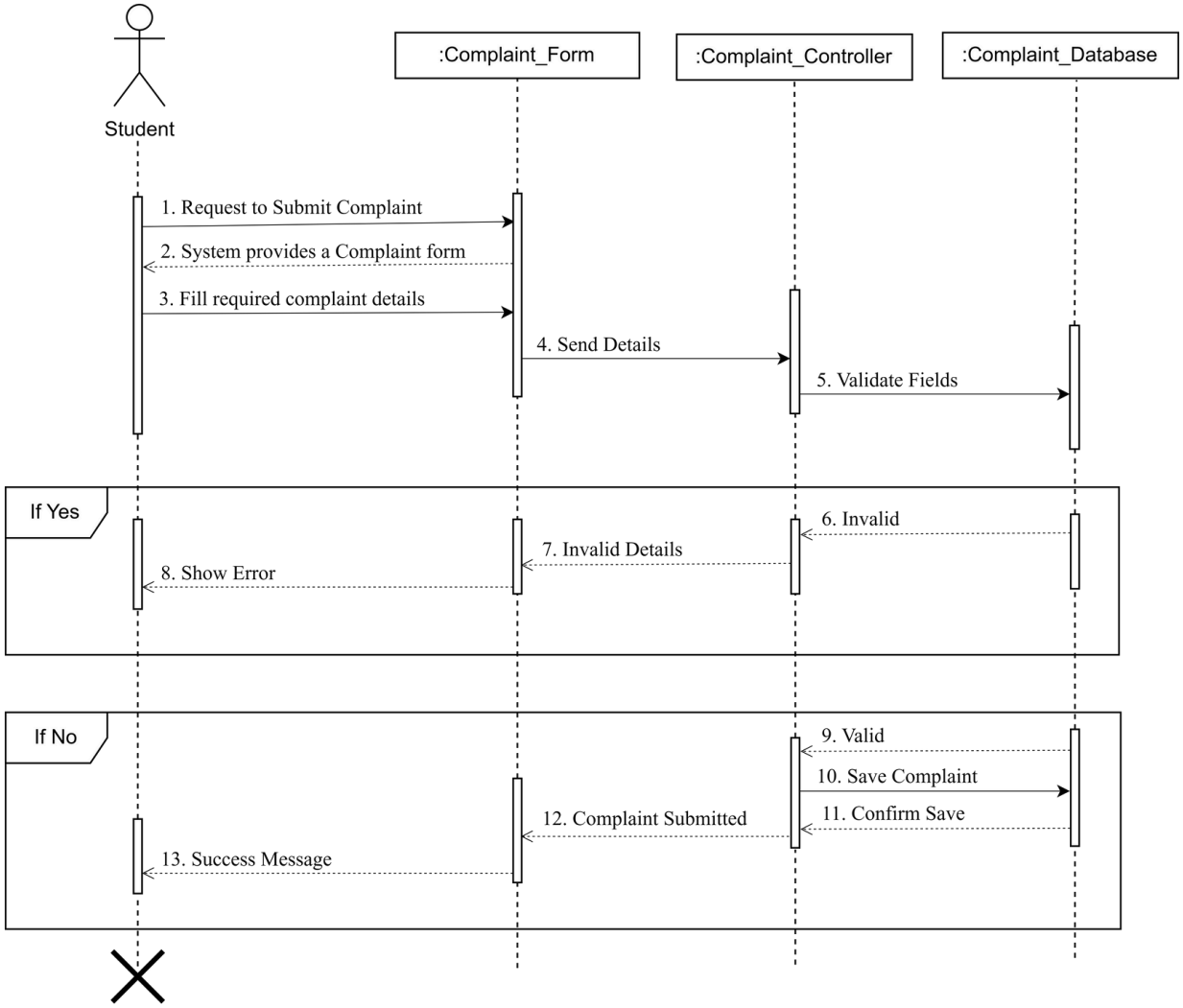


Figure 2.3.4.4: Sequence Diagram-04: Complaint Submission

2.3.4.5 Sequence Diagram-05: Complaint Forwarding

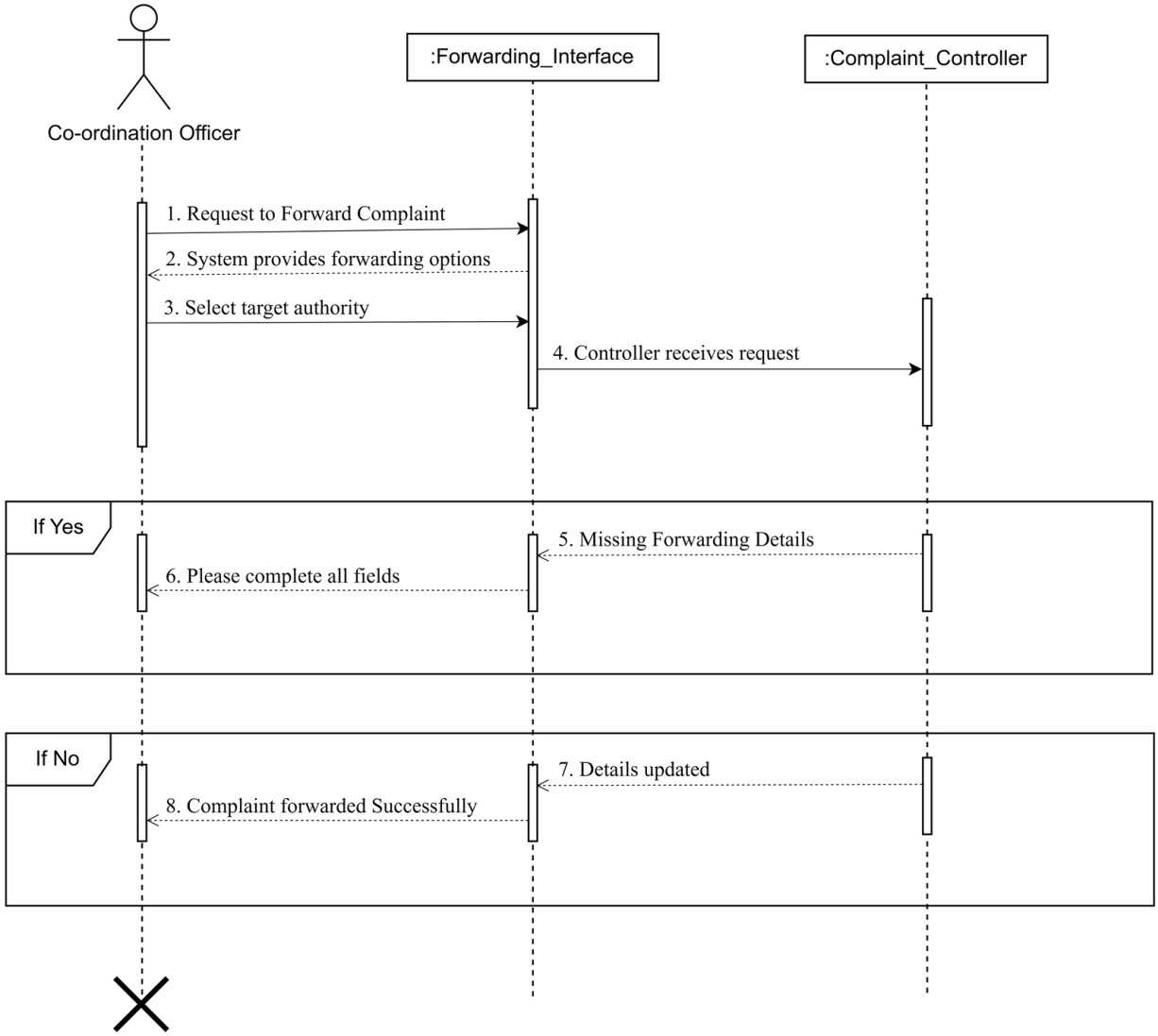


Figure 2.3.4.5: Sequence Diagram-05: Complaint Forwarding

2.3.4.6 Sequence Diagram-06: Explanation Upload

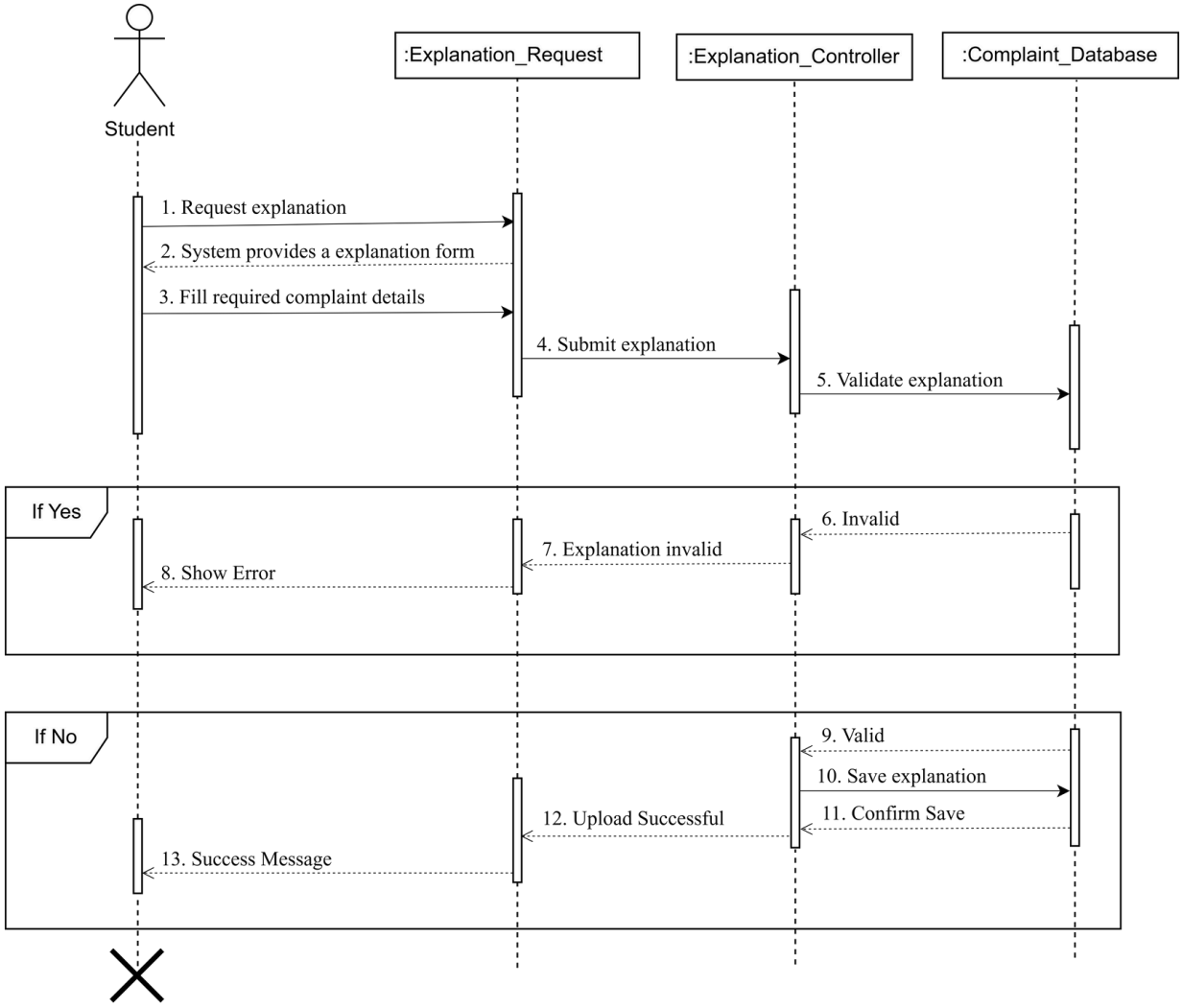


Figure 2.3.4.6: Sequence Diagram-06: Explanation Upload

2.3.4.7 Sequence Diagram-07: Evidence Upload

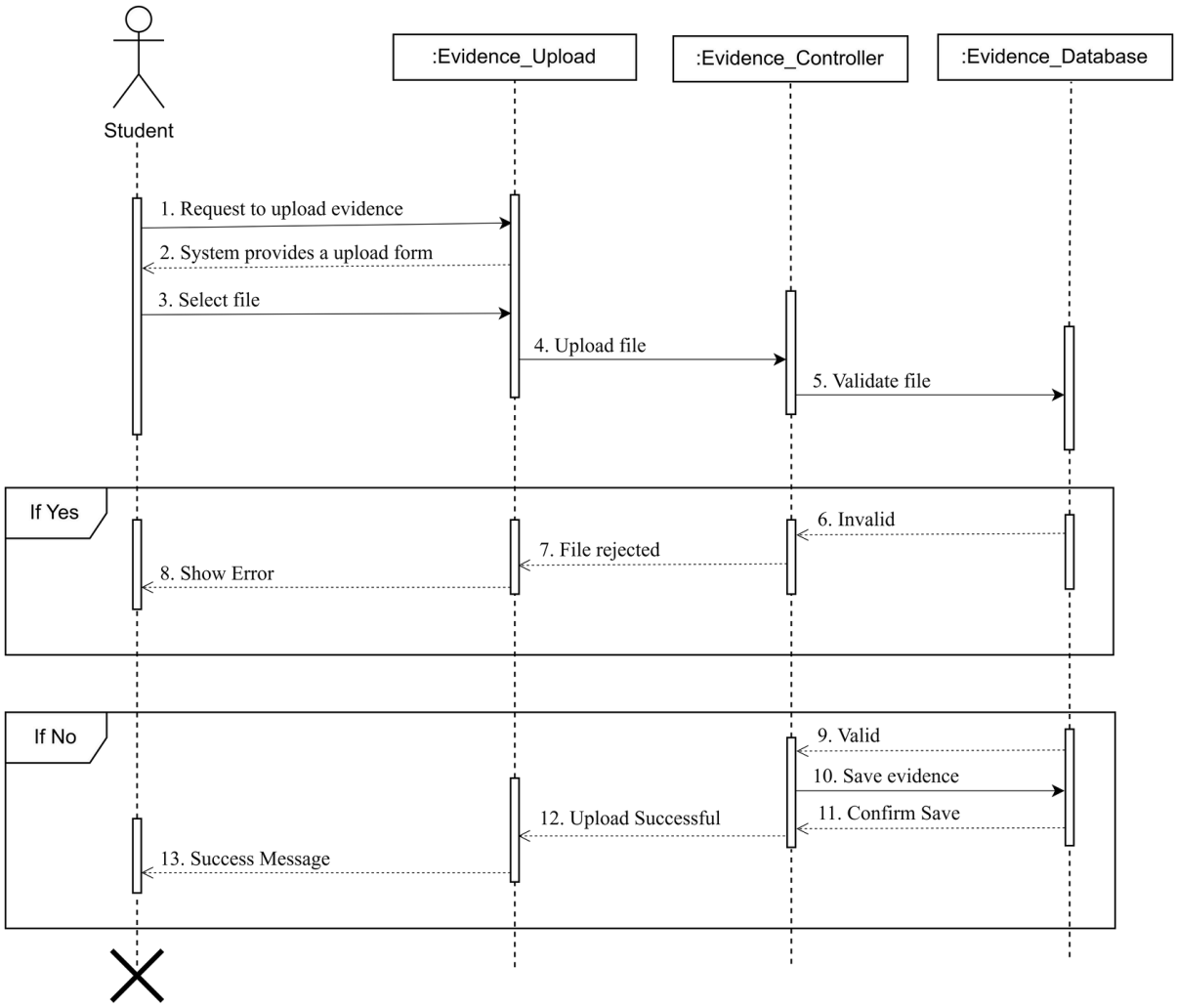


Figure 2.3.4.7: Sequence Diagram-07: Evidence Upload

2.3.4.8 Sequence Diagram-08: Complaint Assignment

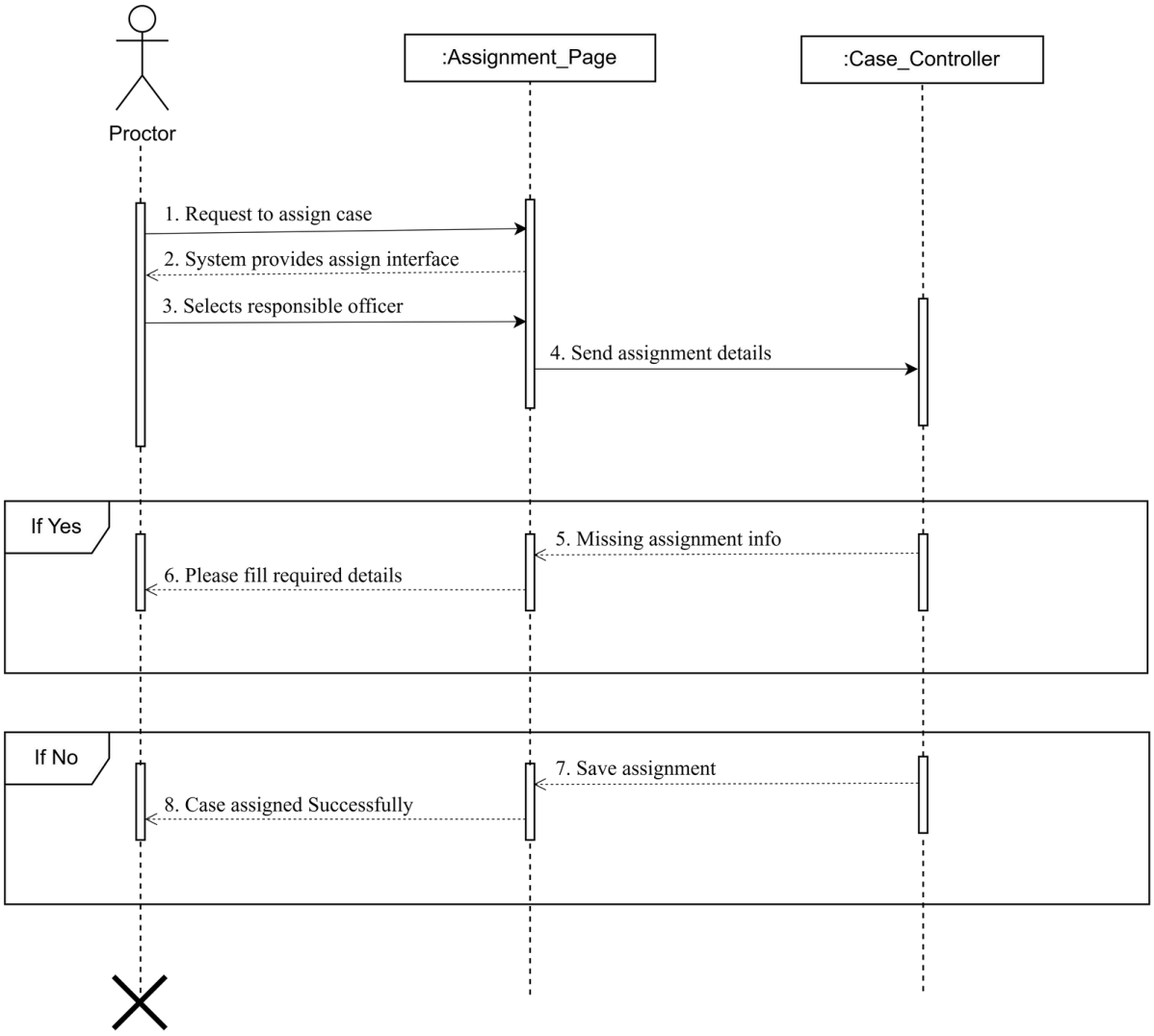


Figure 2.3.4.8: Sequence Diagram-08: Complaint Assignment

2.3.4.9 Sequence Diagram-09: Case File Preparation

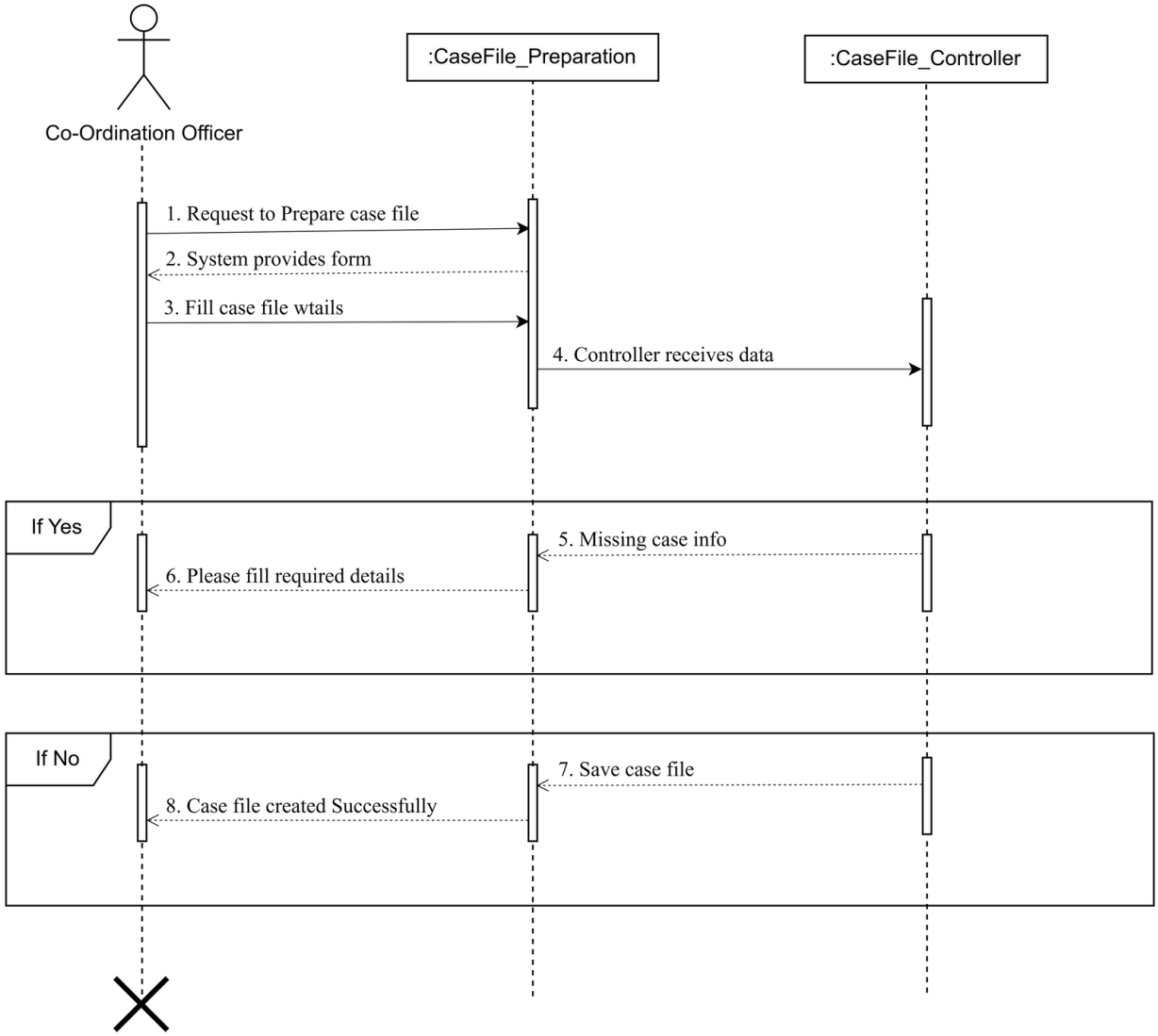


Figure 2.3.4.9: Sequence Diagram-09: Case File Preparation

2.3.4.10 Sequence Diagram-10: Meeting Scheduling

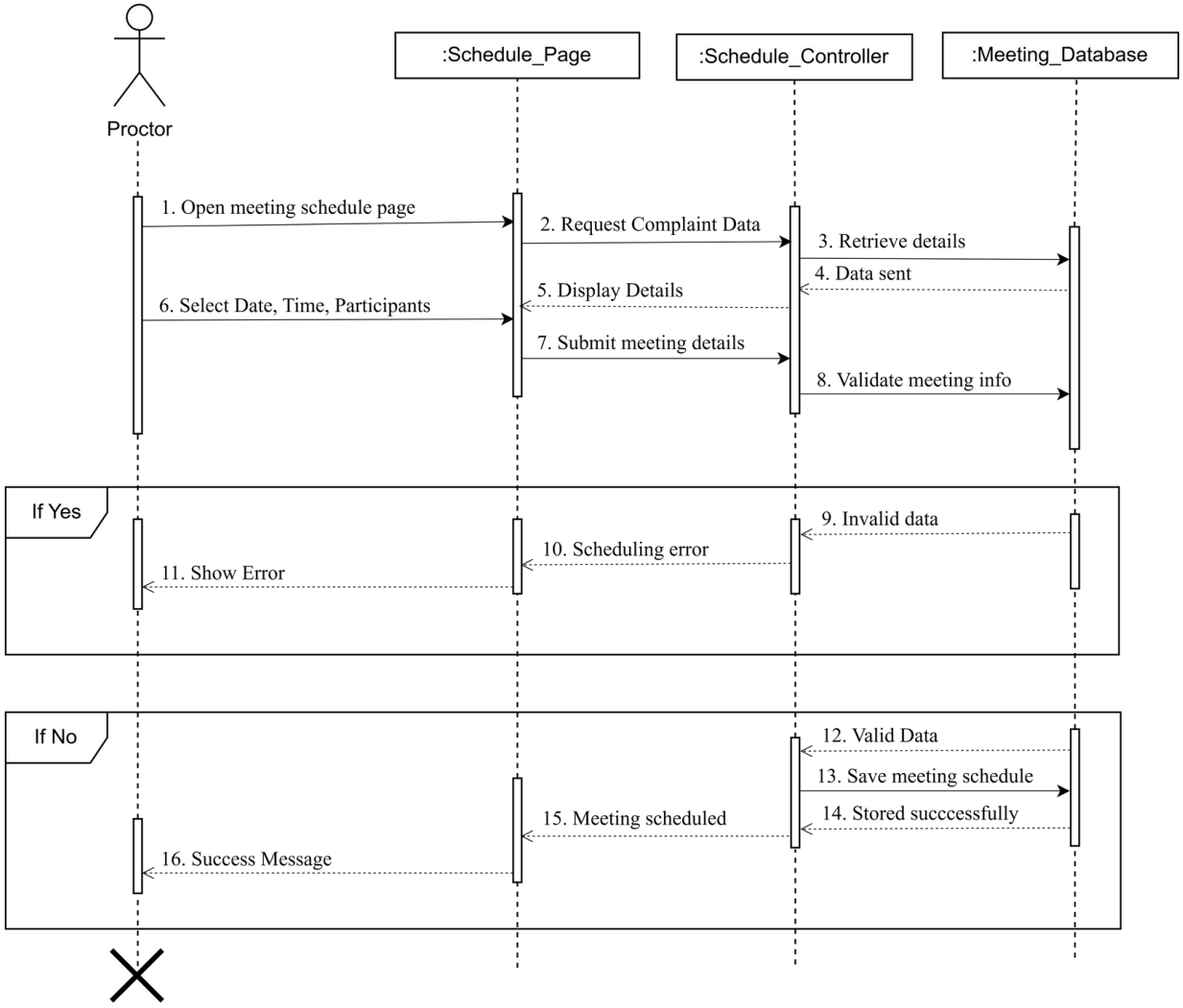


Figure 2.3.4.10: Sequence Diagram-10: Meeting Scheduling

2.3.4.11 Sequence Diagram-11: Notification System

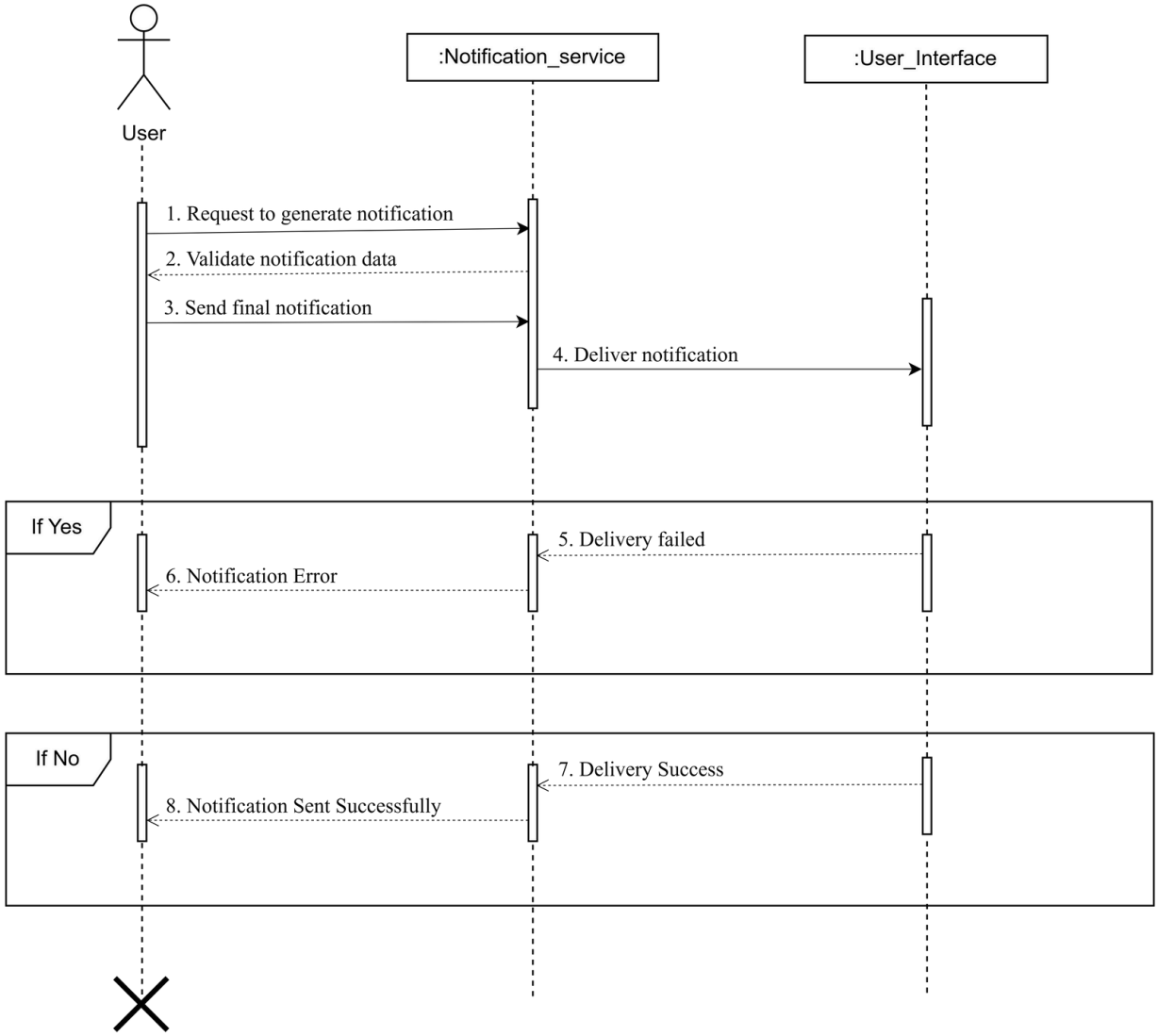


Figure 2.3.4.11: Sequence Diagram-11: Notification System

2.3.4.12 Sequence Diagram-12: Dashboard Access

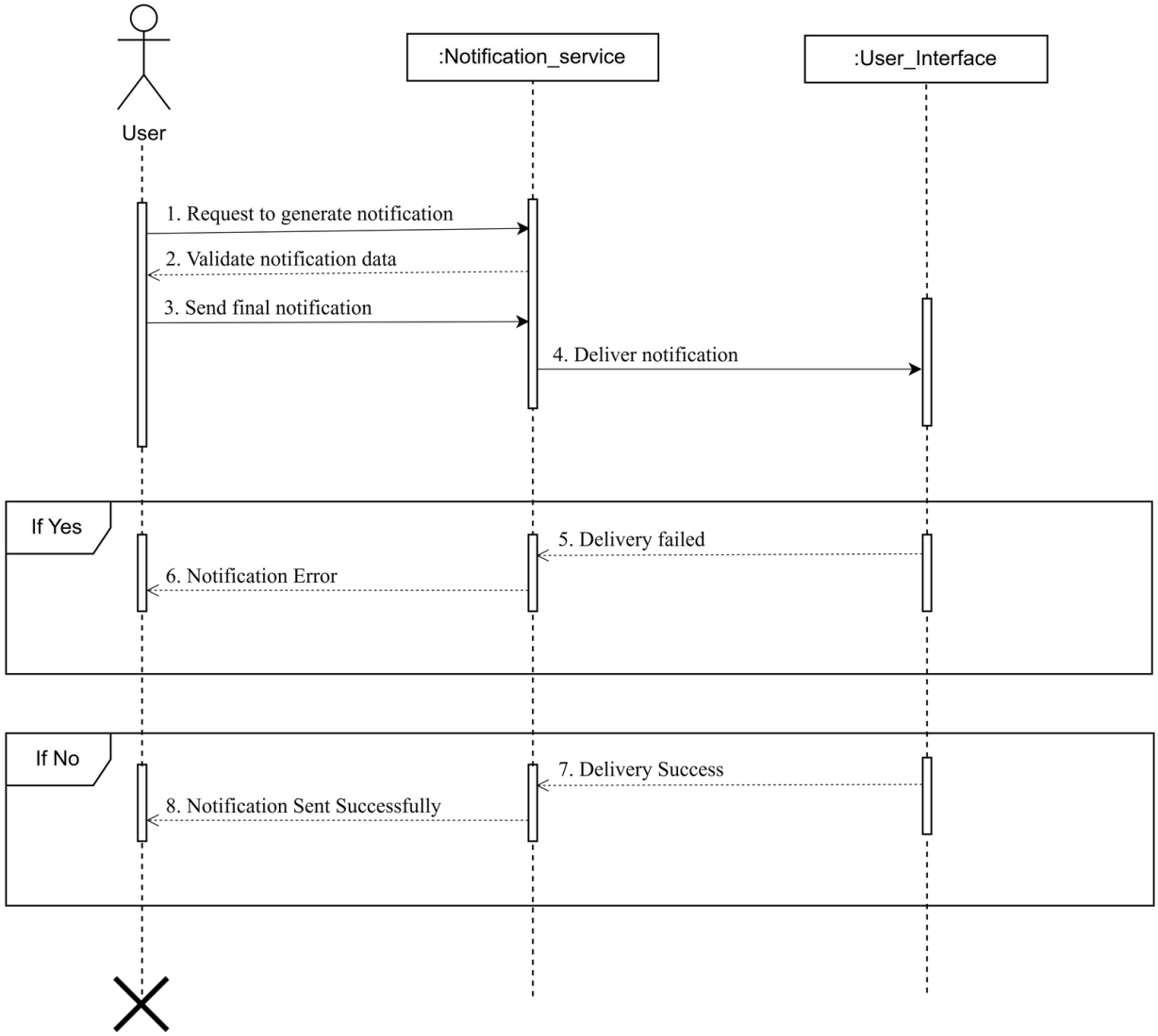


Figure 2.3.4.12: Sequence Diagram-12: Dashboard Access

2.3.4.13 Sequence Diagram-13: Audit Logging

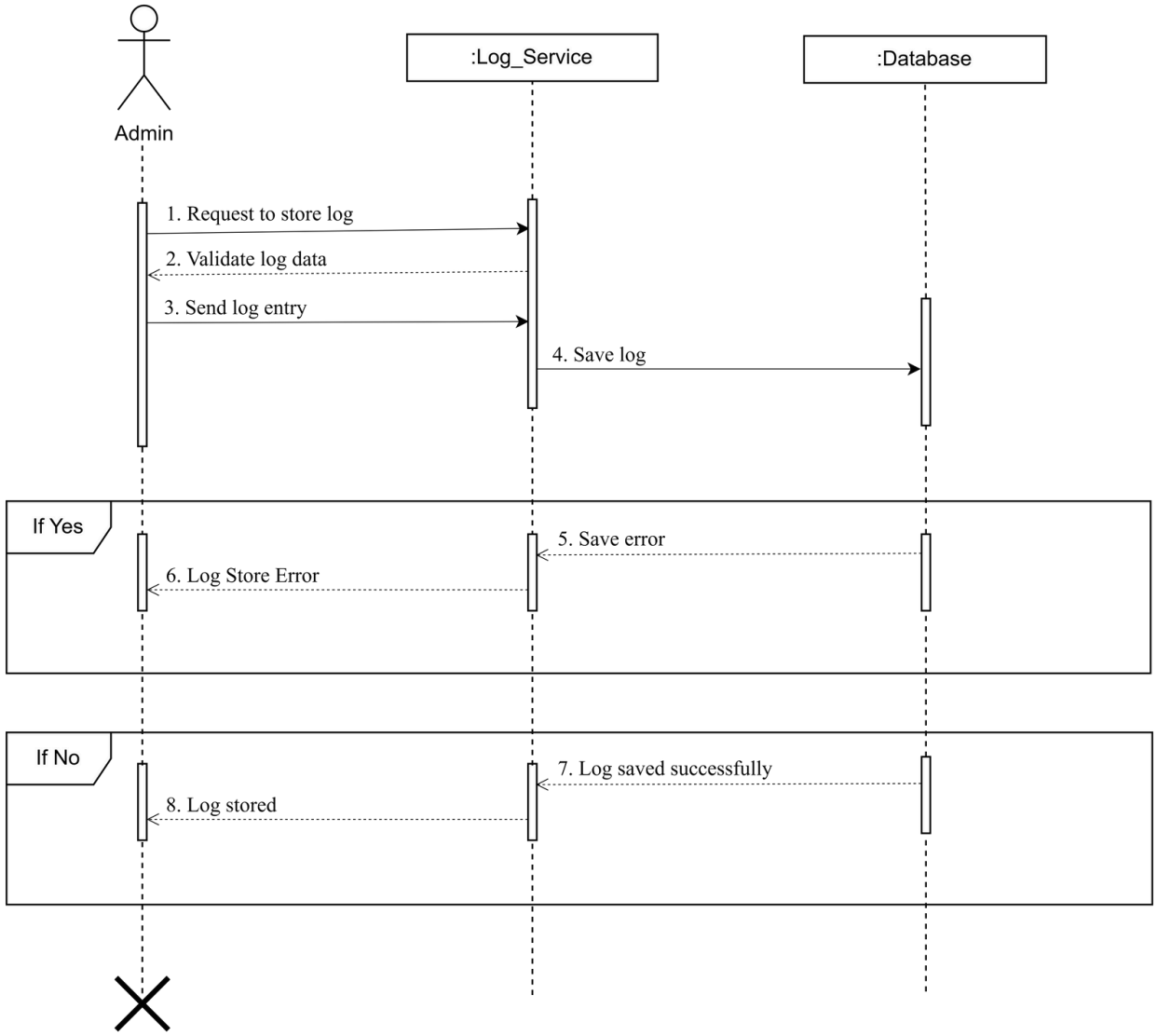


Figure 2.3.4.13: Sequence Diagram-13: Audit Logging

2.3.4.14 Sequence Diagram-14: Sign out

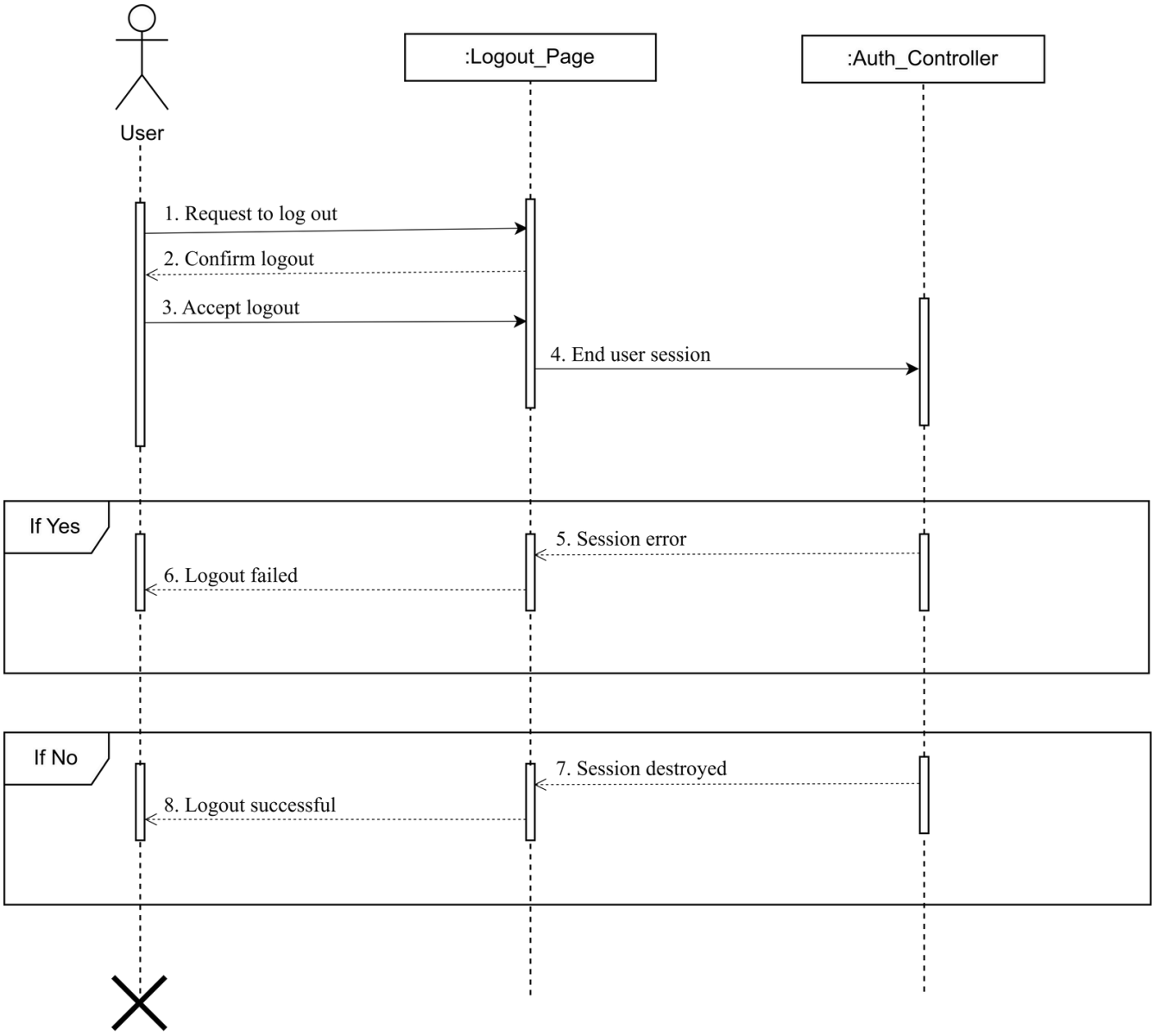


Figure 2.3.4.14: Sequence Diagram-14: Sign out

2.3.5 Class Diagram

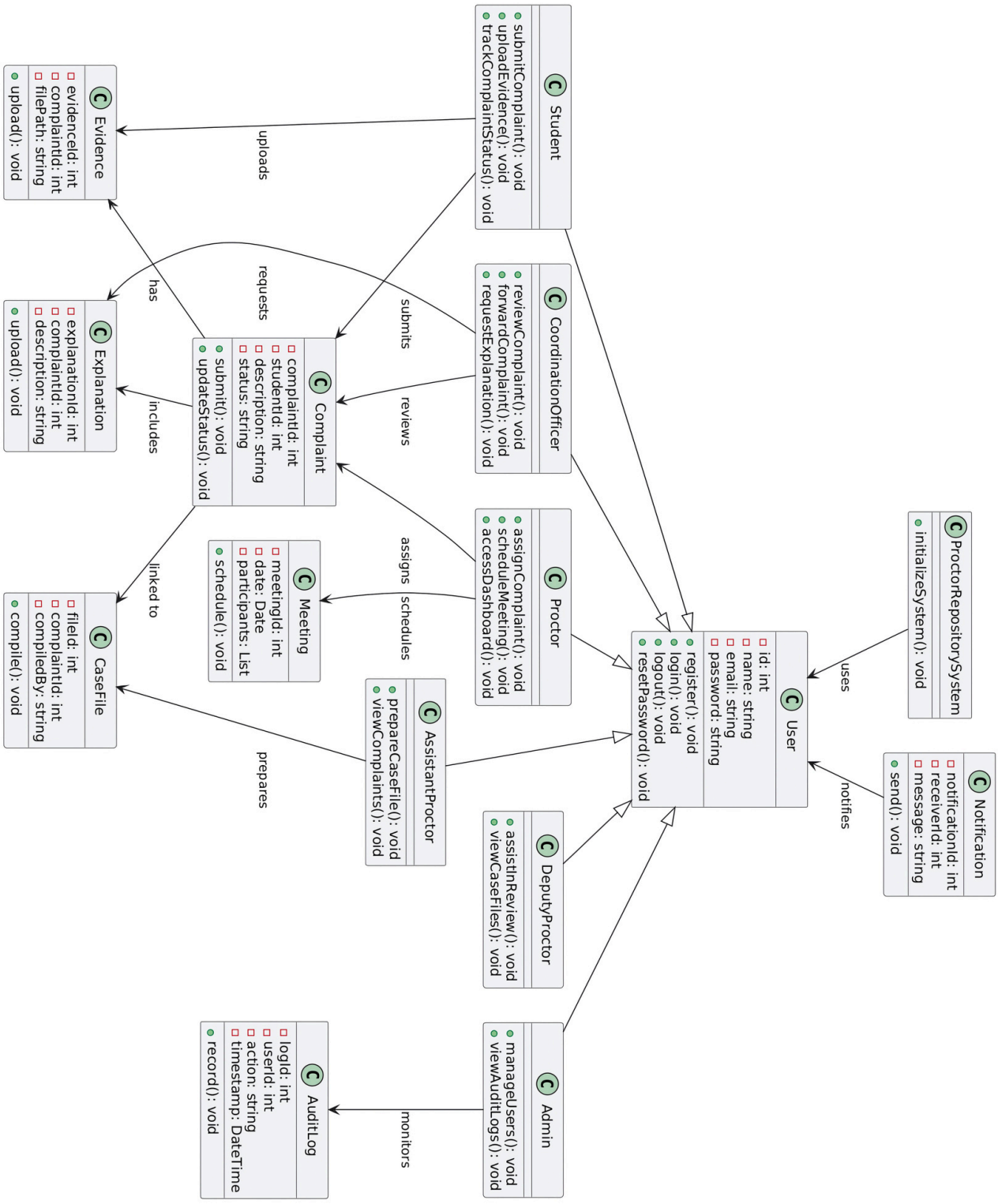


Figure 2.3.5: Class Diagram

2.3.6 ER Diagram

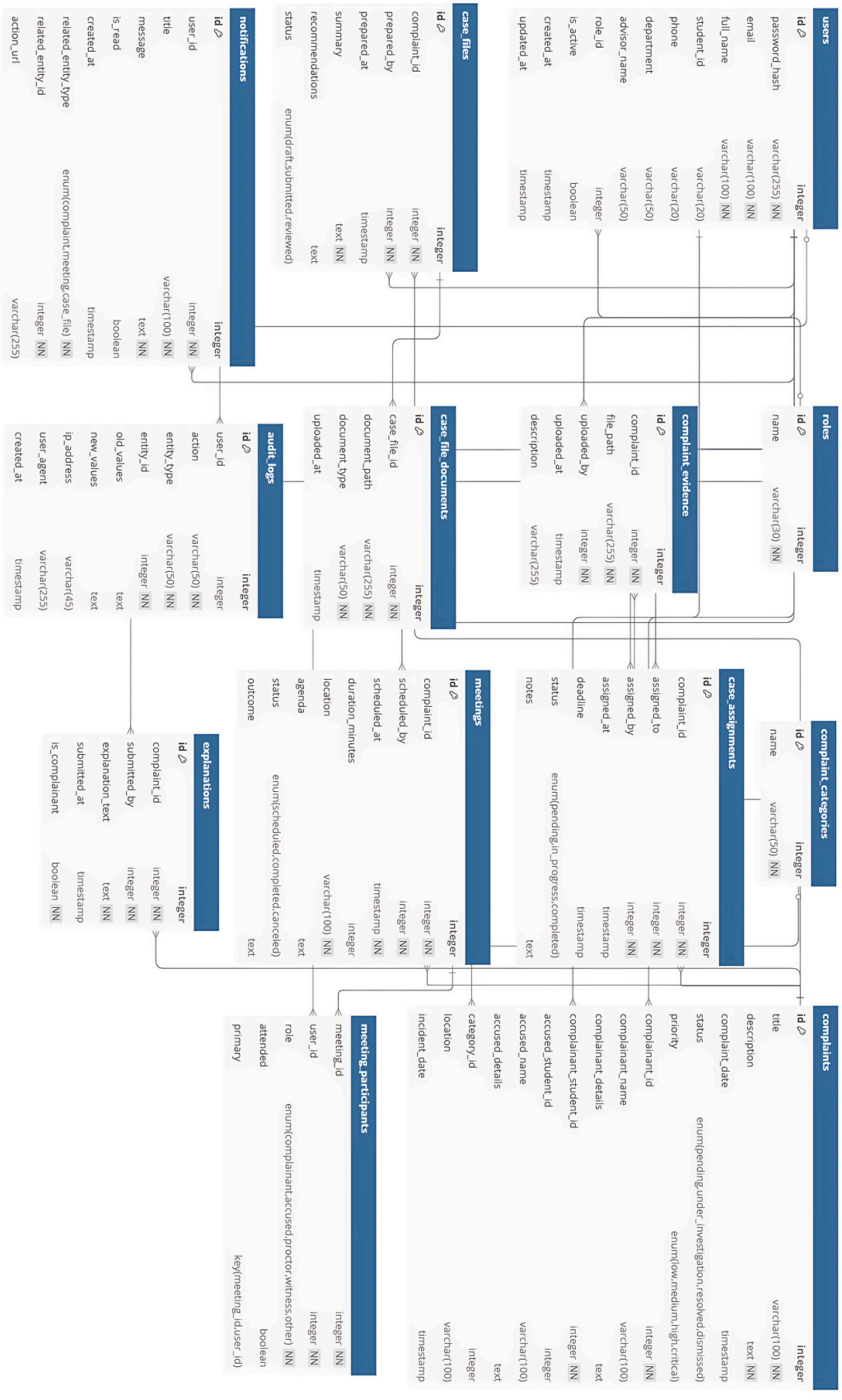


Figure 2.3.6: ER Diagram

Chapter 3: Software Testing

3.1 Introduction

Testing is considered a very vital phase of the development lifecycle for the Proctor Repository System, wherein testing ensures all the feature functionalities, including workflows, work as expected and meet the desired expectations for users. The aim of testing in this project is to check that the digital complaint-handling process, which covers complaint submission, complaint forwarding, explanation management, case preparation, meeting scheduling, and delivering notifications, works correctly, safely, and efficiently for all stakeholders, like students, Coordination Officers, Proctors, and administrative personnel.

3.2 Testing Features

3.2.1 Feature to Be Tested

a. User Registration
b. User Login
c. Forget Password
d. Complaint Submission
e. Explanation Upload
f. Complaint Assignment
g. Meeting Scheduling
h. Notification System
i. Audit Logging
j. System Performance & Load

3.3 Testing Strategies

3.3.1 Black-box Testing

This testing method is performed by validating the system without viewing the internal code. It includes:

- Testing of form validations: registration, login, complaint submission.
- Ensuring notifications are triggered correctly.
- Confirming that dashboards reflect correct case counts and statuses.
- Checking system responses for valid/invalid inputs.

3.3.2 White-box Testing

Performed code-level testing to verify:

- Logical flow of the complaint life cycle.
- Throwing and catching exceptions.
- Authentication and authorization logic.
- Internal controllers and service functions.

3.3.3 User Acceptance Testing (UAT)

UAT sessions involved:

- Students submitting test complaints.
- Coordination Officers forwarding and requesting explanations.
- Proctors assigning and managing cases. Admin checks the activity log.

3.3.4 Pass / Fail Criteria

The pass/fail criteria are the standards based on which each test case of the Proctor Repository System will be deemed to have satisfied its expected result successfully. The pass/fail criteria will ensure the correct functionality of the system according to its modules whenever the system's functionality has been tested through the various test cases. The pass/fail criteria will be considered based on the following guidelines:

- A test case will be said to be passed if the system delivers the output entirely as expected. Anything less will be termed as fail.
- A test case will pass only if the system's behaviors match perfectly with the functional requirement. In case of deviation or non-availability of functionality, the test case will fail.
- A test case passes when valid inputs are accepted and invalid inputs are correctly rejected; acceptance of invalid data or crashes results in a fail.
- A test case passes if the system responds within acceptable time limits, while unresponsiveness, freezing, or unexpected errors result in a fail.
- A test case will pass when data submitted is stored and retrieved correctly; missing, corrupted, or mismatched data constitutes a fail.
- A test case passes if, and only if, restricted features are accessed by only authorized roles and fails otherwise.
- A test case passes when the workflow proceeds correctly through all defined steps; any workflow break or jump results in a fail.
- A test case passes when notifications reach correct users at correct times; delays, missing notifications, or incorrect recipients constitute a fail.
- It would pass if UI elements load well and match the design. If the layout is broken or elements are missing, it will fail.
- A test case passes when sensitive data is kept secure and sanitized; any vulnerability or exposure equates to a fail.

3.4 System Testing

3.4.1 Test case: User Registration

Test case- 01				**Test case Name: User Registration**			
System: Form Fill Up				**Subsystem:**			
Designed by: Md. Sabbir Ahmed Shanto				**Designed date:** 05/10/25			
Executed by:				**Executed Date:**			
Short description: It's the system's registration section.							
Pre-condition: The user accesses the system's public-facing pages (Register, Login, Forgot Password).							
Step	**Name**	**Email**	**ID**	**Password**	**Retype Password**	**Response**	**Pass/Fail**
1	Sabbir Ahmed	sabbir@email.com		Sab12345	Sab12345	Fill up blank field	Fail
2	Sabbir Ahmed	sabbir@email.com	221-35-962	Sab12345	Sab12345	Registration Successful	Pass
3	Sabbir Ahmed	sabbir@email.com	221-35-962	Sab12345	sa040545	Passwords do not match	Fail
4	Sabbir Ahmed	sabbir@email	221-35-962	Sab12345	Sab12345	Invalid Email	Fail
5	Sabbir Ahmed	sabbir@email.com	221-35-962	Sab12345	Sab12345	Email already exists.	Fail
Post-condition: The user is successfully registered with valid information. Invalid inputs are rejected.							

Table 3.4.1: Test case: User Registration

3.4.2 Test case: User Login

Test case- 02	**Test case Name: User Login**
System: User Authentication	**Subsystem:**
Designed by: Md. Sabbir Ahmed Shanto	**Designed date:** 05/10/25
Executed by:	**Executed Date:**
Short description: This feature allows a registered user to gain secure access to the system by authenticating their identity using their email and password.	
Pre-condition: The user must have an existing, registered account and must be on the system's Login page.	

Step	**Email**	**Password**	**Response**	**Pass/Fail**	
1	sabbir@email.com	Sab12345	Login successful	Pass	
2	sabbir@	Sab12345	Invalid Email Address	Fail	
3	sabbir@email.com	hhcdxxsjffku	Wrong Password	Fail	
4	sabbir@email.com		Input Password	Fail	
Post-condition: The user is successfully logged in with valid information. Invalid inputs are rejected.					

Table 3.4.2: Test case: User Login

3.4.3 Test case: Forget Password

Test case- 03	**Test case Name: Forget Password**
System: Reset Password	**Subsystem:**
Designed by: Md. Sabbir Ahmed Shanto	**Designed date:** 05/10/25
Executed by:	**Executed Date:**
Short description: This feature provides a way for a user who has forgotten their password to initiate a secure password reset process.	
Pre-condition: The user must be on the Login page or the dedicated "Forgot Password" page and must know the email address associated with their account.	

Step	Action	Email	Password	Retype Password	Response	Pass/Fail
1	Submit Request	student@email.com			Password reset link sent to your email.	Pass
2	Submit Request	student@			Please enter a valid email address.	Fail
3	Click Reset Link	student@email.com			Redirects to "Reset Password" page.	Pass
4	Reset Password	student@email.com	newpass123	newpass123	Password successfully reset. Please log in.	Pass
5	Reset Password	student@email.com	newpass123	newpass456	Passwords do not match.	Fail

Post-condition: The user successfully resets their password and can log in with the new password. Invalid attempts are blocked.

Table 3.4.3: Test case: Forget Password

3.4.4 Test case: Complaint Submission

Test case- 04	**Test case Name: Complaint Submission**
System: Submit Complaint	**Subsystem:**
Designed by: Md. Sabbir Ahmed Shanto	**Designed date:** 05/10/25
Executed by:	**Executed Date:**
Short description: A student submits a new complaint with details and evidence.	
Pre-condition: User is logged in as a 'Student'.	

Step	Complaint Title	Complaint Details	Evidence File	Response	Pass/Fail
1	Valid Title	Full details of the incident	evidence.pdf	Complaint successfully submitted. ID: [C-1001]	Pass
2	Valid Title	Full details of the incident	evidence.jpg	Complaint successfully submitted. ID: [C-1002]	Pass
3		Full details of the incident	evidence.pdf	Title field is required.	Fail
4	Valid Title		script.exe	Details field is required.	Fail
5	Valid Title	Full details of the incident	evidence.pdf	Invalid file type. Only PDF, JPG, PNG allowed.	Fail

Post-condition: The complaint is saved correctly with a unique CaseID, or rejected if inputs are invalid.

Table 3.4.4: Test case: Complaint Submission

3.4.5 Test case: Explanation Upload

Test case- 05	**Test case Name: Explanation Upload**
System: Upload Explanation	**Subsystem:**
Designed by: Md. Sabbir Ahmed Shanto	**Designed date:** 05/10/25
Executed by:	**Executed Date:**
Short description: A user (complainant or accused) uploads their explanation in response to a request.	
Pre-condition: User is logged in. A request for explanation is pending for their case (e.g., C-1001).	

Step	Case ID	Explanation Text	Explanation File	Response	Pass/Fail
1	C-1001	This is my detailed explanation...		Explanation submitted.	Pass
2	C-1001		statement.pdf	Explanation submitted.	Pass
3	C-1001		video.exe	Invalid file type.	Fail
4	C-1001			Please provide an explanation or upload a file.	Fail

Post-condition: The explanation (text or file) is successfully linked to the correct case file.

Table 3.4.5: Test case: Explanation Upload

3.4.6 Test case: Complaint Assignment

Test case- 06	**Test case Name: Complaint Assignment**
System: Case Management	**Subsystem:**
Designed by: Md. Sabbir Ahmed Shanto	**Designed date:** 05/10/25
Executed by:	**Executed Date:**
Short description: Proctor assigns a prepared case file to a member of the proctorial team for investigation.	
Pre-condition: User is logged in as 'Proctor'. A prepared case file (e.g., C-1001) is ready for assignment.	

Step	**Case ID**	**Assign To (Team Member)**	**Member Availability**	**Response**	**Pass/Fail**
1	C-1001	Mr. David (Asst. Proctor)	Available	Case C-1001 assigned. Notification sent.	Pass
2	C-1001	Ms. Sarah (Asst. Proctor)	On Leave	Member is unavailable. Cannot assign.	Fail
3	C-1001		Available	Please select a team member.	Fail
Post-condition: The case is assigned to the selected team member and a notification is sent.					

Table 3.4.6: Test case: Complaint Assignment

3.4.7 Test case: Meeting Scheduling

Test case- 07	**Test case Name: Meeting Scheduling**
System: Meeting Scheduler	**Subsystem:**
Designed by: Md. Sabbir Ahmed Shanto	**Designed date:** 05/10/25
Executed by:	**Executed Date:**
Short description: A proctorial team member schedules a meeting with the complainant, accused, and other parties.	
Pre-condition: User is logged in (e.g., 'Proctor'). User is handling a case (e.g., C-1001).	

Step	Case ID	Date	Time	Participants	Response	Pass/Fail
1	C-1001	Future Date	10:00 AM	Student, Accused, Self	Meeting scheduled. Notifications sent.	Pass
2	C-1001	Past Date	10:00 AM	Student, Accused, Self	Cannot schedule a meeting in the past.	Fail
3	C-1001	Future Date	12:00 PM		Please select participants.	Fail
4	C-1001	Future Date	08:00 PM	Student, Accused, Self	Please select time within working hours.	Fail

Post-condition: The meeting is saved in the system and notifications are sent to all participants.

Table 3.4.7: Test case: Meeting Scheduling

3.4.8 Test case: Notification System

Test case- 08	**Test case Name: Notification System**
System: Verify Notification Delivery	**Subsystem:**
Designed by: Md. Sabbir Ahmed Shanto	**Designed date:** 05/10/25
Executed by:	**Executed Date:**
Short description: Test that all notifications are triggered by correct actions and sent to the correct users.	
Pre-condition: An action is performed in the system (e.g., submitting a complaint).	

Step	Triggering Action	Expected Recipient (Role)	Expected Notification	Response	Pass/Fail
1	Student submits complaint	Coordination Officer	New complaint [C-1001] submitted.	Notification received.	Pass
2	AP requests explanation	Student (Complainant)	Explanation requested for [C-1001].	Notification received.	Pass
3	Proctor assigns case	Asst. Proctor (Mr. David)	You are assigned to Case [C-1001].	Notification received.	Pass
4	Meeting scheduled	Student, Accused	Meeting scheduled for [C-1001] on [Date].	Notification received.	Pass
Post-condition: The intended recipient receives the correct notification message.					

Table 3.4.8: Test case: Notification System

3.4.9 Test case: Audit Log System

Test case- 09	**Test case Name: Audit Log System**
System: Admin / Audit	**Subsystem:**
Designed by: Md. Sabbir Ahmed Shanto	**Designed date:** 05/10/25
Executed by:	**Executed Date:**
Short description: Verify that critical user actions are logged with correct user, timestamp, and activity details.	
Pre-condition: User 'student@email.com' submits complaint C-1001. User 'admin@email.com' logs in.	

Step	**Action Performed**	**User**	**Expected Log Entry (in Admin Panel)**	**Response**	**Pass/Fail**
1	Login	student@email.com	USER_LOGIN: User 'student@email.com' logged in.	Log entry found and is accurate.	Pass
2	Submit Complaint	student@email.com	COMPLAINT_SUBMIT: User 'student@email.com' created [C-1001].	Log entry found and is accurate.	Pass
3	Meeting Scheduling	co@email.com	MEETING_SCHEDULED: User 'co@email.com' SCHEDULED [C-1001].	Log entry found and is accurate.	Pass
4	Access Denied	student@email.com	AUTH_FAIL: User 'student@email.com' failed to access /admin.	Log entry found and is accurate.	Pass
Post-condition: The Admin can view the accurate log entry for the action.					

Table 3.4.9: Test case: Audit Log System

3.4.10 Test case: System Performance & Load

Test case- 10	**Test case Name: System Performance & Load**
System: Performance	**Subsystem:**
Designed by: Md. Sabbir Ahmed Shanto	**Designed date:** 05/10/25
Executed by:	**Executed Date:**
Short description: Test system response time for major workflows and page loads.	
Pre-condition: System is deployed in a staging/test environment.	

Step	Test Scenario	Load (Concurrent Users)	Expected Response Time	Response	Pass/Fail
1	Load Login Page	1	< 2 seconds	Page loads in [Actual Time].	Pass
2	Submit Complaint (Full)	1	< 3 seconds	Complaint submitted in [Actual Time].	Pass
3	Load Proctor Dashboard	1	< 4 seconds	Dashboard (with data) loads in [Actual Time].	Pass
4	Submit Complaint (Full)	50	< 10 seconds	System stable. 0% error rate.	Pass
5	Load Proctor Dashboard	50	< 12 seconds	System stable. 0% error rate.	Pass

Post-condition: The system remains responsive and does not crash under load.

Table 3.4.10: Test case: System Performance & Load

Chapter 4: Deployment and Maintenance

4.1 Introduction

This chapter presents the deployment process of the Proctor Repository System into a functional environment and subsequent regular maintenance toward its continued operation. It involves deployment on the server, database, and application to ensure accessibility of the system by all user roles: students, Coordination Officers, Proctors, and administrators. Deployment also includes final testing, setting up version control, configuration of security, and launching the system for real use at the university.

The maintenance phase also assists in ensuring the reliability and functionality of the system by monitoring the health of the system, fixing bugs, improving different features of the system based on appropriate feedback from the concerned stakeholders. Other routine maintenance tasks involve database backups, log monitoring, and performance optimizations. These processes put together mean the system is stable, secure, and responsive to changing complaint-handling needs of the university.

4.2 Deployment Following the Software Release Life Cycle (SRLC)

The deployment process of the Proctor Repository System involves the structured stages of the Software Release Life Cycle, which helps ensure the safe and stable launch of the system in the university setting. The Software Release Life Cycle offers a systematic way of preparing, releasing, and developing software in a series of stages.

4.2.1 Pre-Alpha Phase

In the pre-alpha level, the base modules, like the process of user registration, complaint submission, and dashboard loading, were initially created and tested in house. The core functions were also verified, and the key bugs were removed before proceeding to the next level.

4.2.2 Alpha Phase

The alpha version of the system contained all key functionalities but was in a preliminary state. The alpha version was used in the internal testing process, analyzing functional, workflow, and linkages between modules of the system, as in the complaint submission process → explanation upload → case file generation. The alpha version was accessible only to developers and some members of the faculty.

4.2.3 Beta Release

The beta version was a more stable form, which was also made available to a limited set of actual users like students and Coordination Officers. This stage was mainly involved in usability, interface, performance, and issue spotting. The information collected in the beta version was necessary for refining the usability as well as eliminating the existing defects in the software.

4.2.4 Release Candidate (RC)

After the stabilization of key modules, the creation of the Release Candidate was started based on the feedback received from the beta testers. The Release Candidate was close to the final version, having the most optimal workflow, improved UI, latest security settings, and the best role-based access control. The Release Candidate was examined by supervisors for approval.

4.2.5 Production Release (Stable Version)

The system was then validated, after which it was rolled out as the stable production level in the university setting. This involved the set-up of the live server, the link to the production database, the use of live user accounts, as well as the establishment of security measures. The system was open to the use of all the target users, which include students, Coordination Officers, Proctors, Deputy Proctors, Assistant Proctors, as well as Administrators.

4.2.6 Post-Release Maintenance and Updates

After the deployment, there are ongoing updates that are done to make the software work properly. This includes patching, adding functionality, updates, and other optimizations based on the input received. These updates are done as patches when they are small, while when they are big, they are done as version updates after the SRLC cycle again.

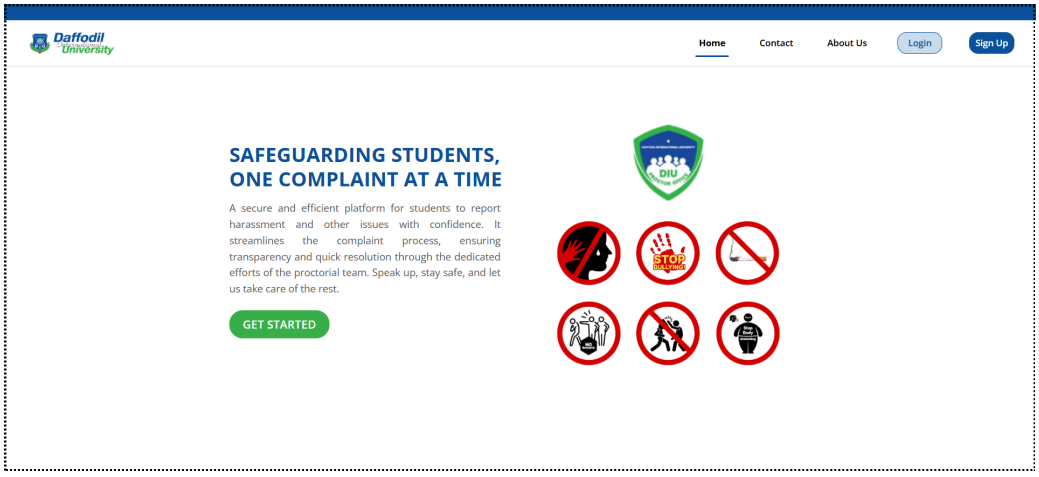
Chapter 5: User Manual

5.1 Introduction

This chapter offers a straightforward way of using the Proctor Repository System. The details describe the manner in which each level of users, ranging from Students, Coordination Officers, Proctors, Assistant Proctors, Deputy Proctors, and Admins, can use the site in order to accomplish the necessary tasks related to the users' role. This also involves explanations related to the submissions, cases, evidence, preparing the case file, meetings, as well as the dashboard management.

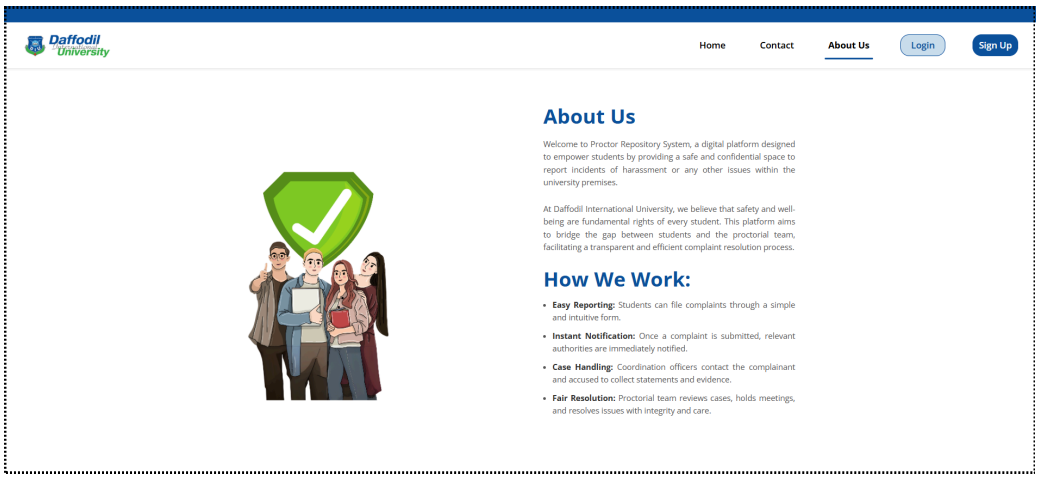
5.2 Project Functionalities

5.2.1 Homepage



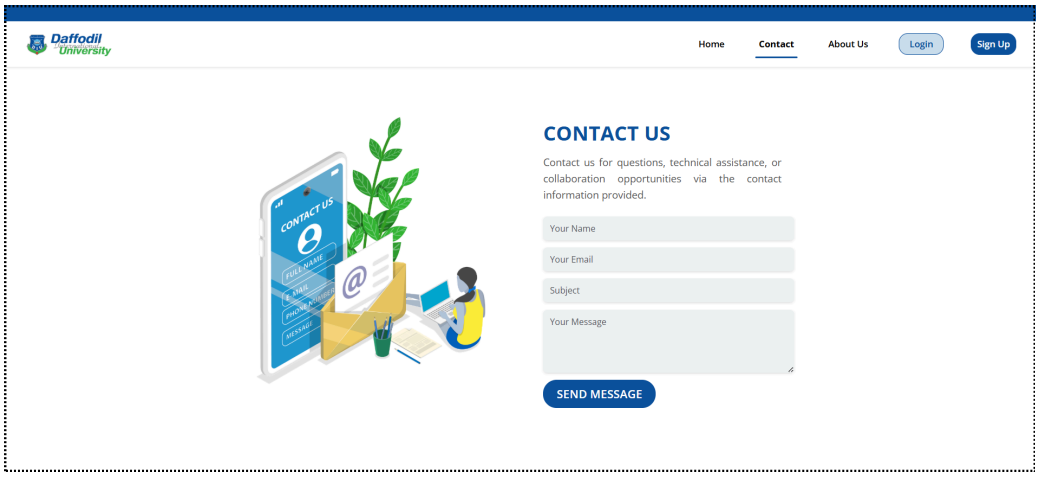
Appendix 5.2.1: Homepage

5.2.2 About Us



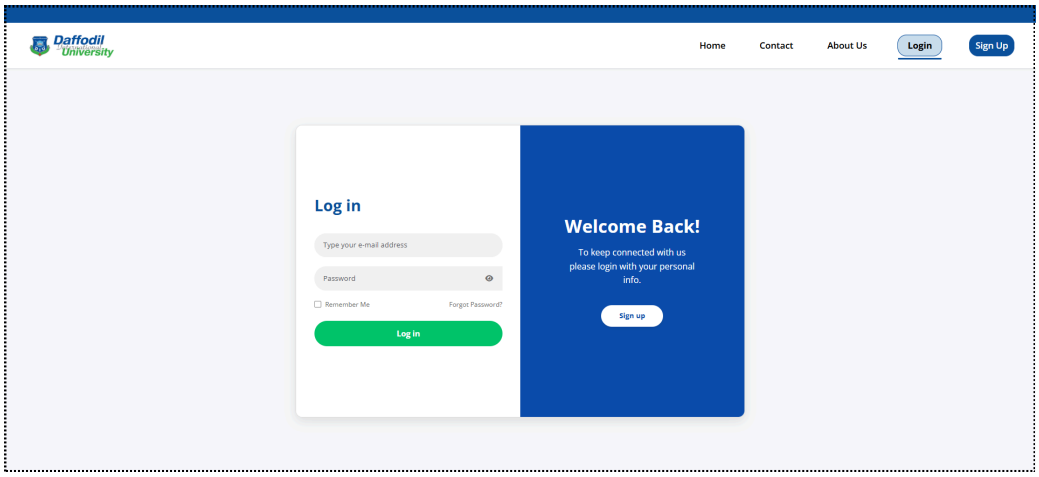
Appendix 5.2.2: About Us

5.2.3 Contact Us



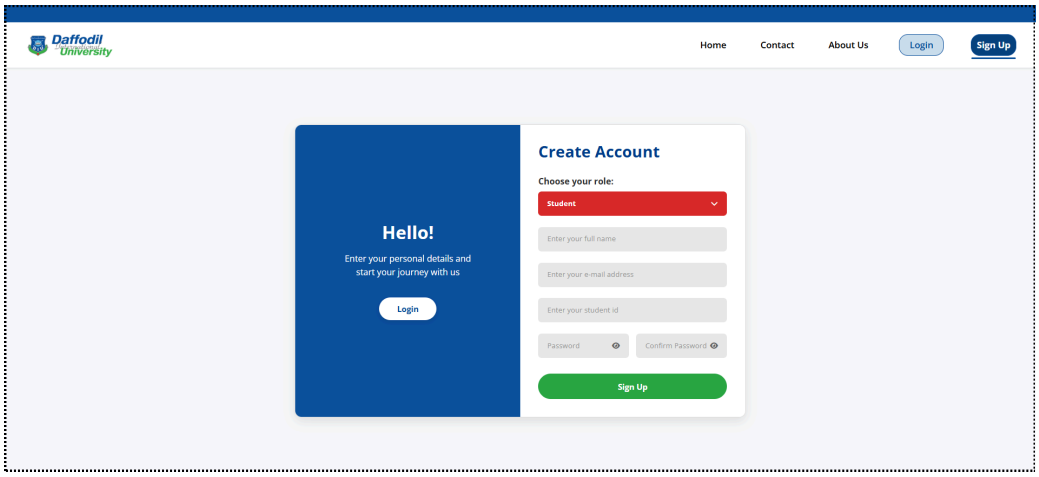
Appendix 5.2.3: Contact Us

5.2.4 Login



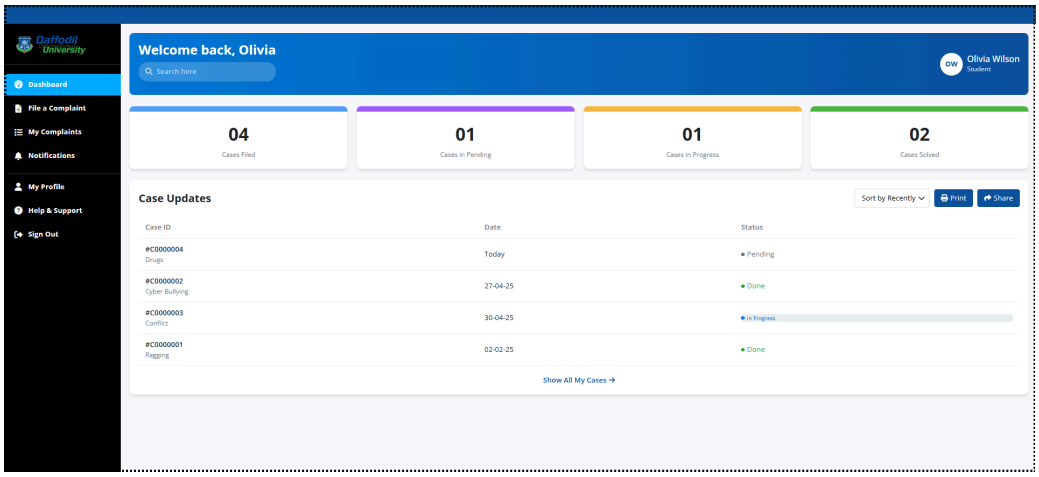
Appendix 5.2.4: Login

5.2.5 Sign Up



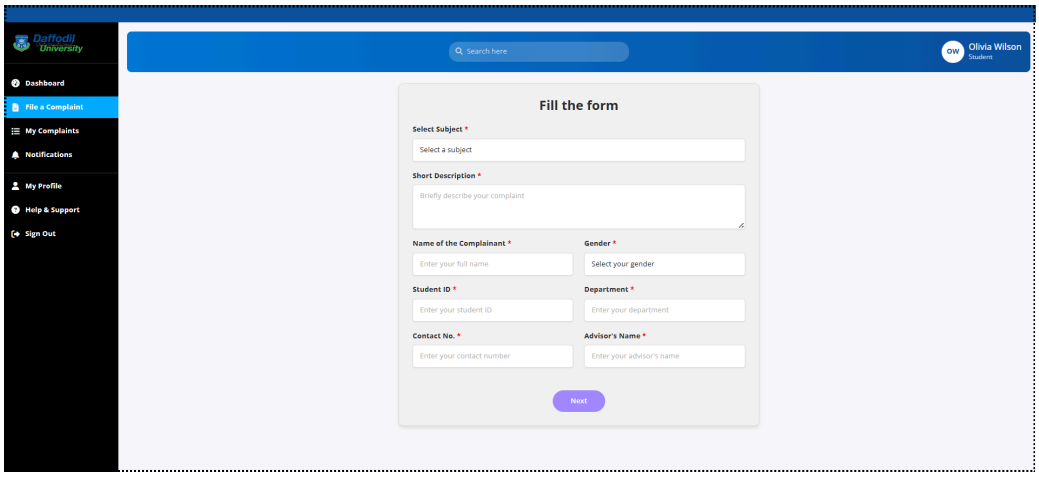
Appendix 5.2.5: Sign Up

5.2.6 Student Dashboard



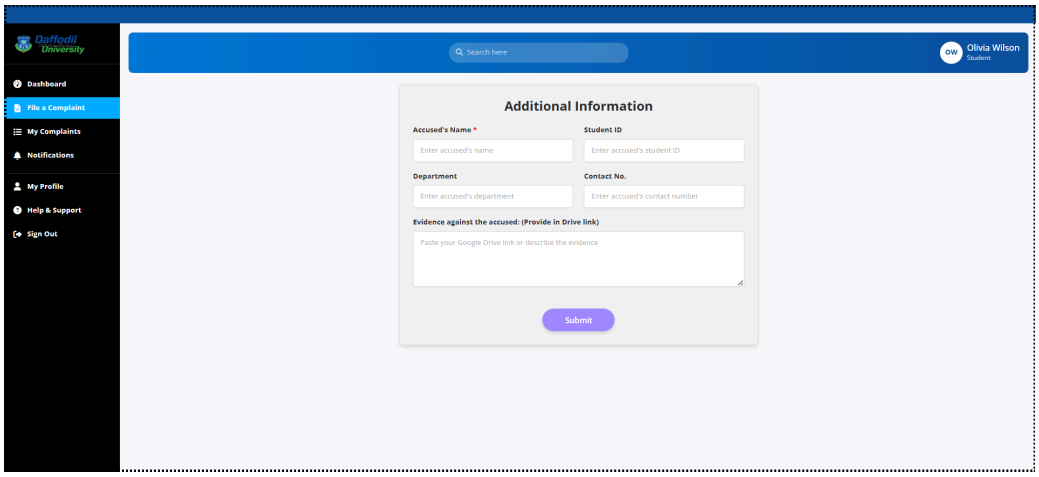
Appendix 5.2.6: Student Dashboard

5.2.7 File a Complaint



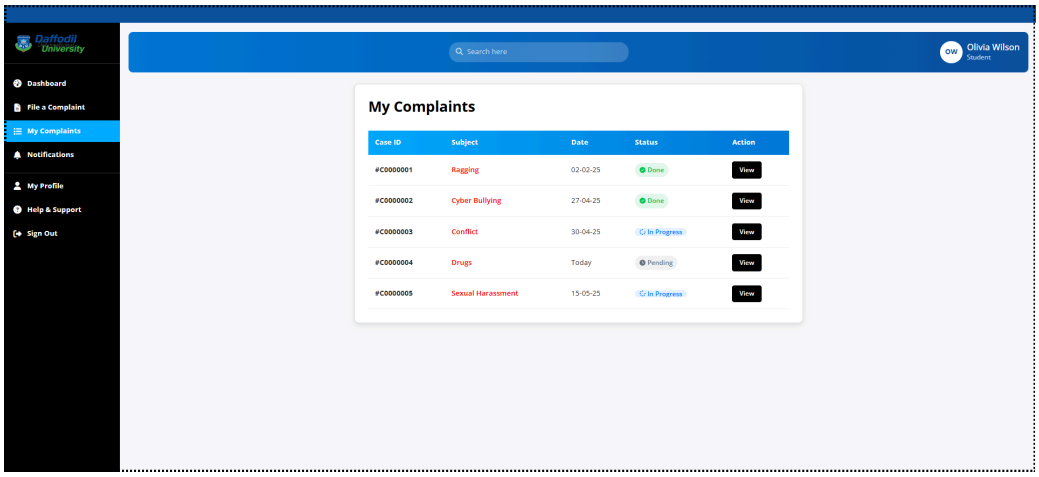
Appendix 5.2.7: File a Complaint

5.2.8 Next Form



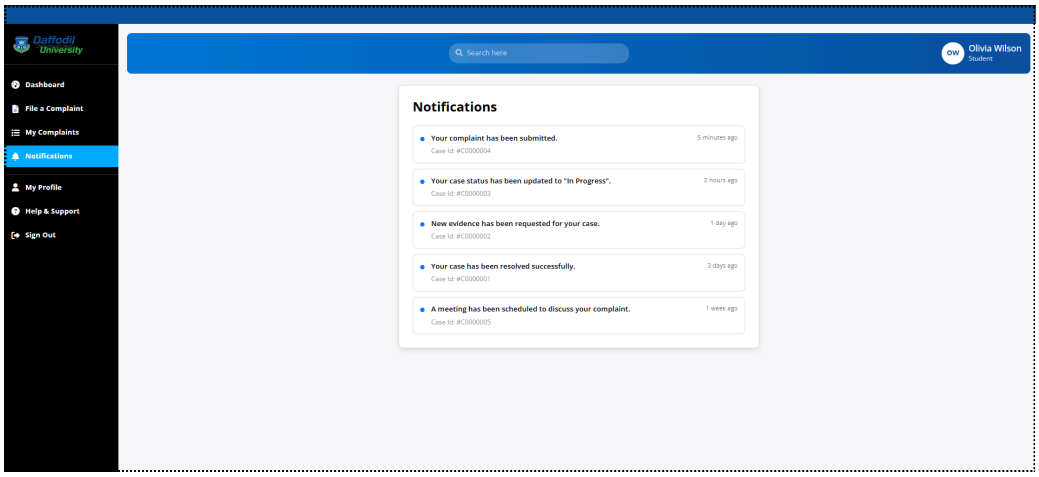
Appendix 5.2.8: Next Form

5.2.9 My Complaints



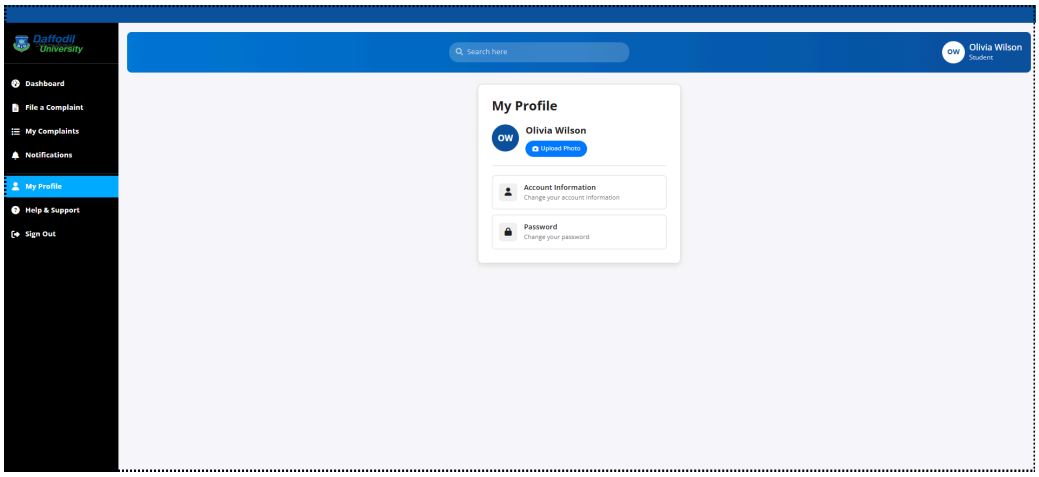
Appendix 5.2.9: My Complaints

5.2.10 Notifications



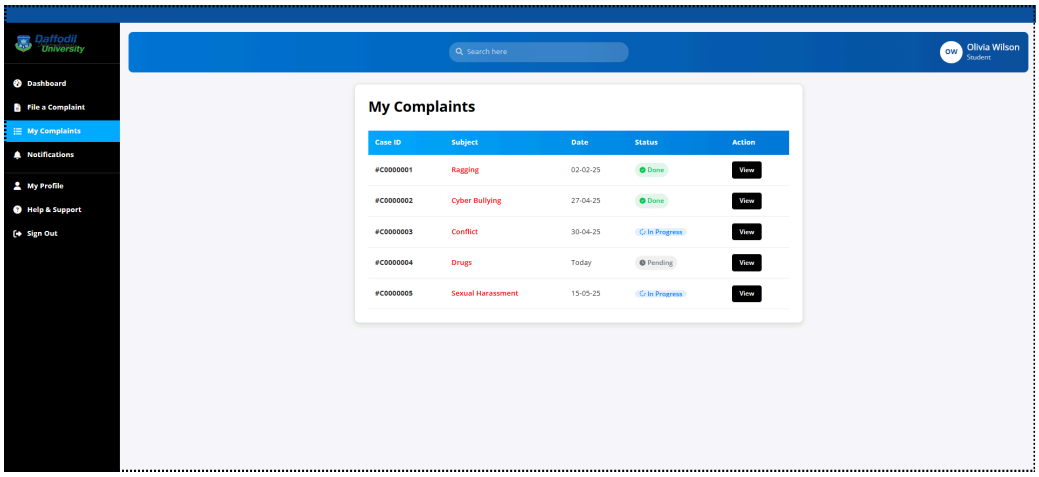
Appendix 5.2.10: Notifications

5.2.11 My Profile



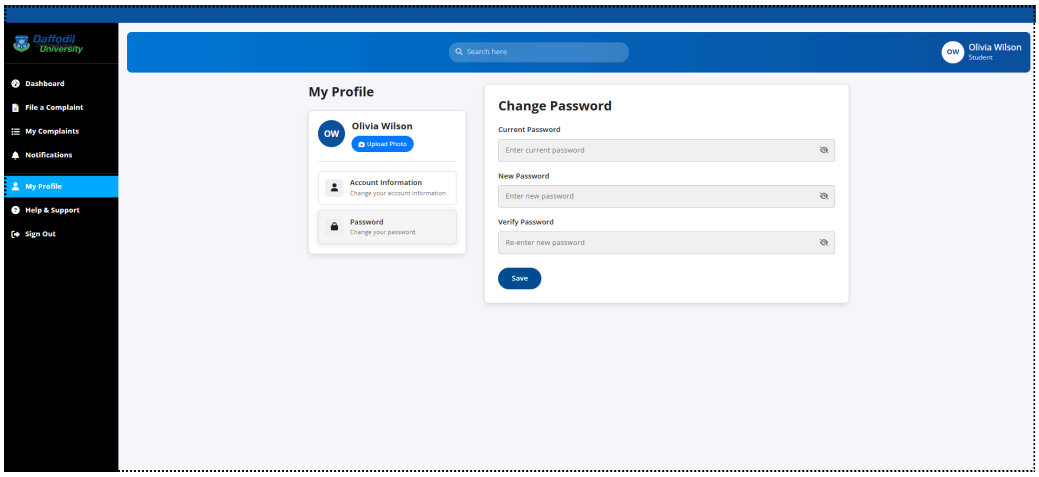
Appendix 5.2.11: My Profile

5.2.12 Account Information



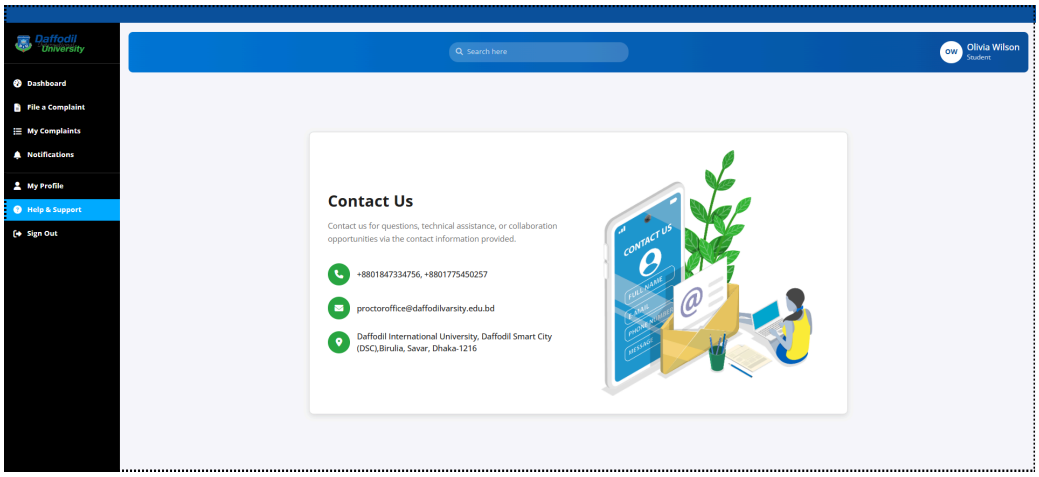
Appendix 5.2.12: Account Information

5.2.13 Change Password



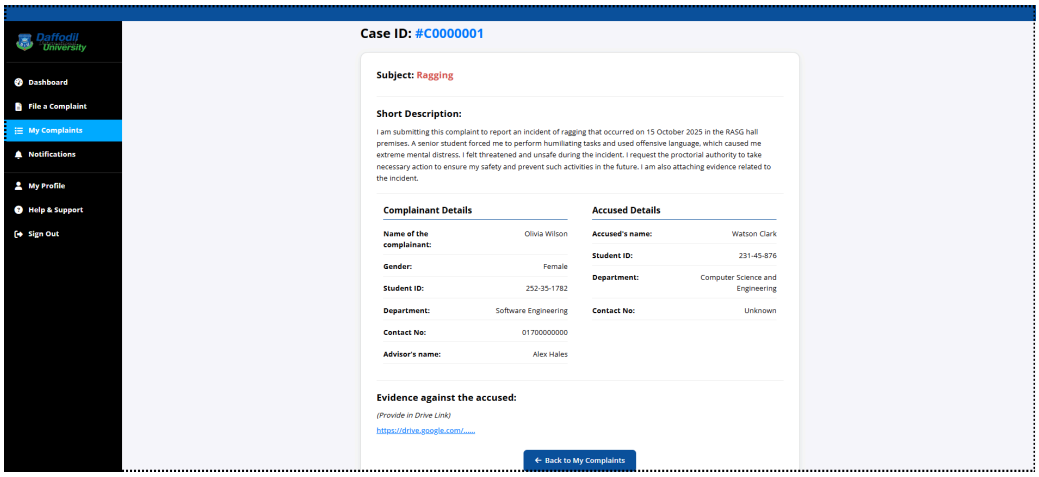
Appendix 5.2.13: Change Password

5.2.14 Help & Password



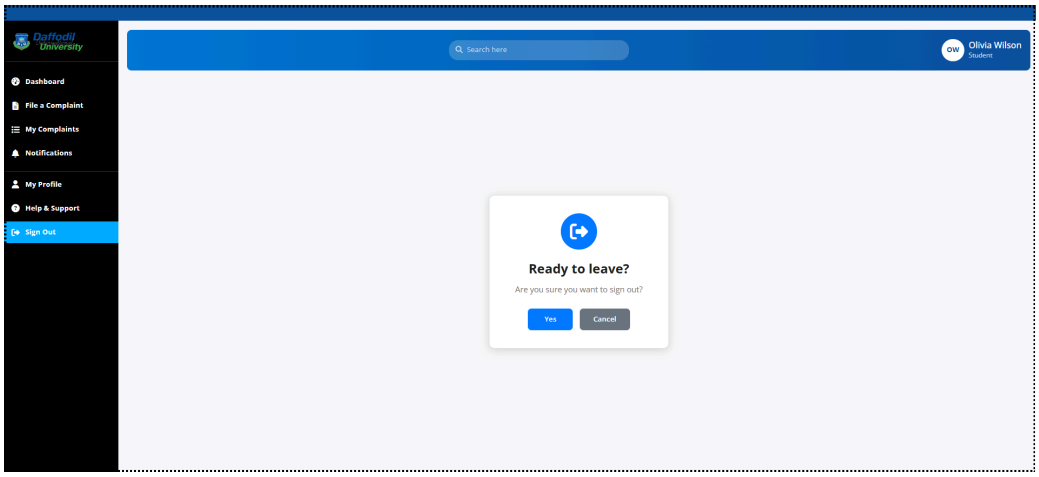
Appendix 5.2.14: Help & Support

5.2.15 View Case Details



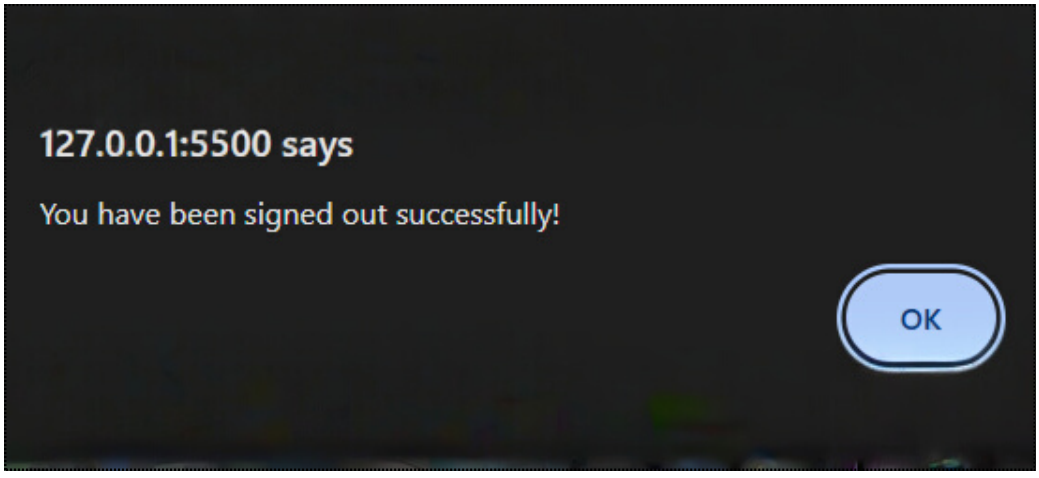
Appendix 5.2.15: View Case Details

5.2.16 Logout



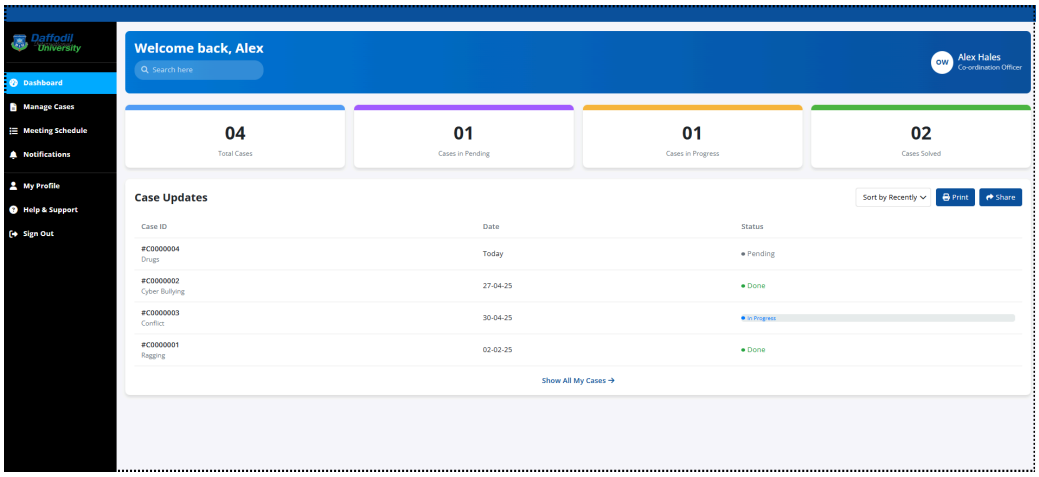
Appendix 5.2.16: Logout

5.2.17 Logout Successful



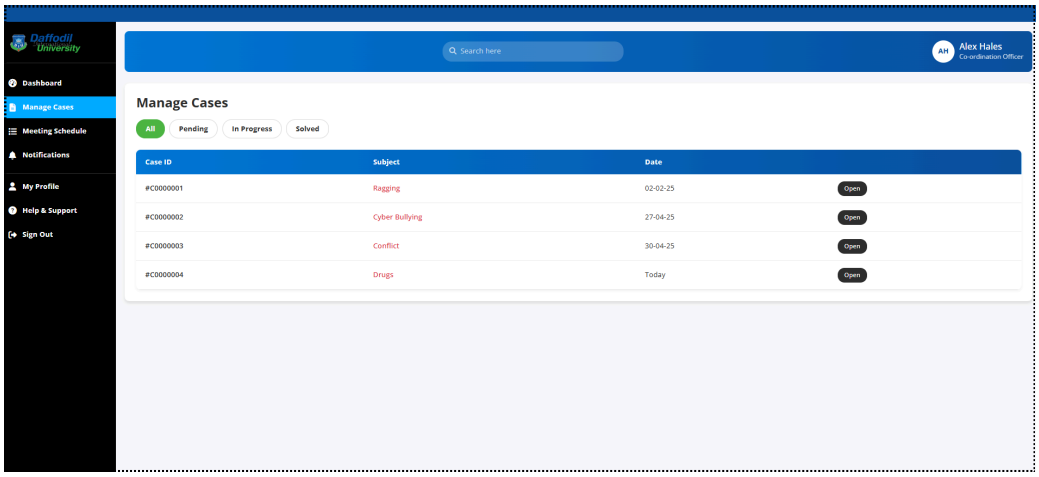
Appendix 5.2.17: Logout Successful

5.2.18 Co-ordination Officer Dashboard



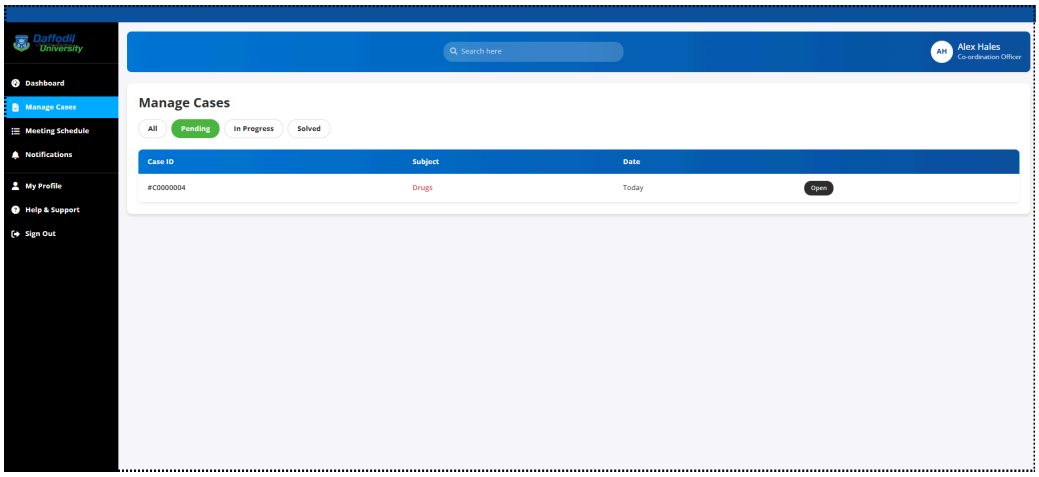
Appendix 5.2.18: Co-ordination Officer Dashboard

5.2.19 Manage Cases



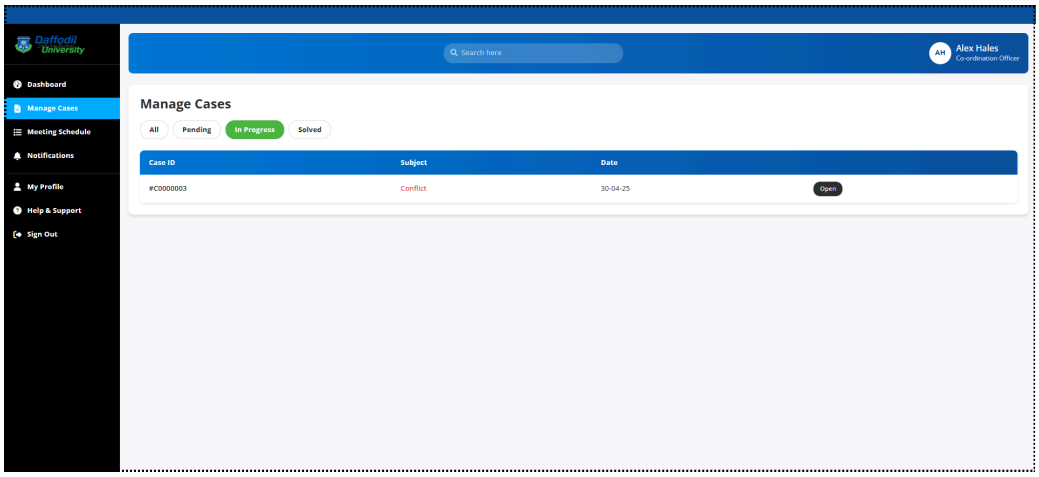
Appendix 5.2.19: Manage Cases

5.2.20 Pending Cases



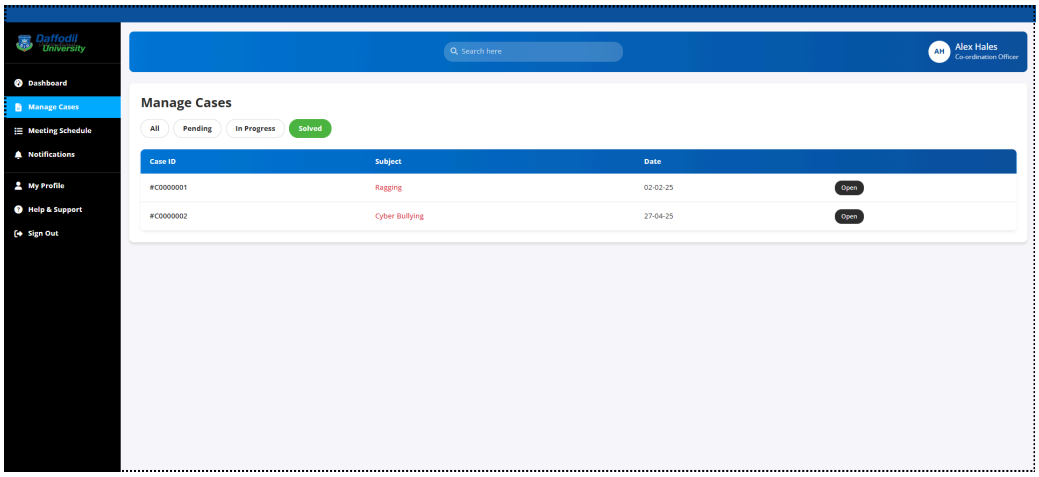
Appendix 5.2.20: Pending Cases

5.2.21 In progress Cases



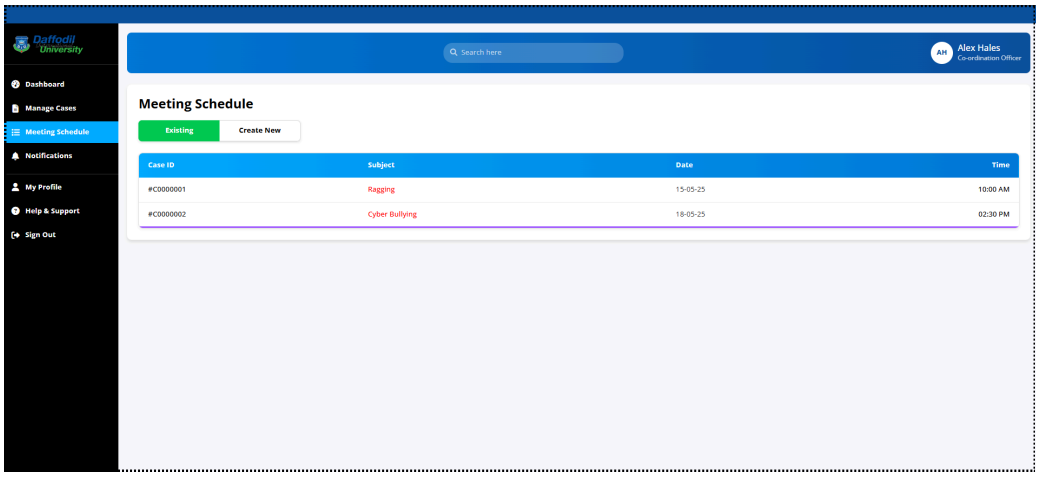
Appendix 5.2.21: In progress Cases

5.2.22 Solved Cases



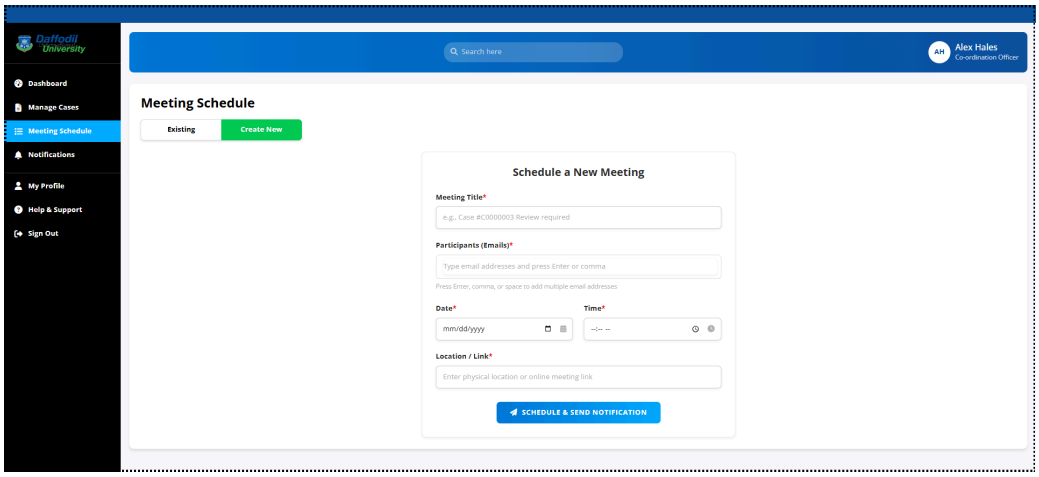
Appendix 5.2.22: Solved Cases

5.2.23 Meeting Schedule



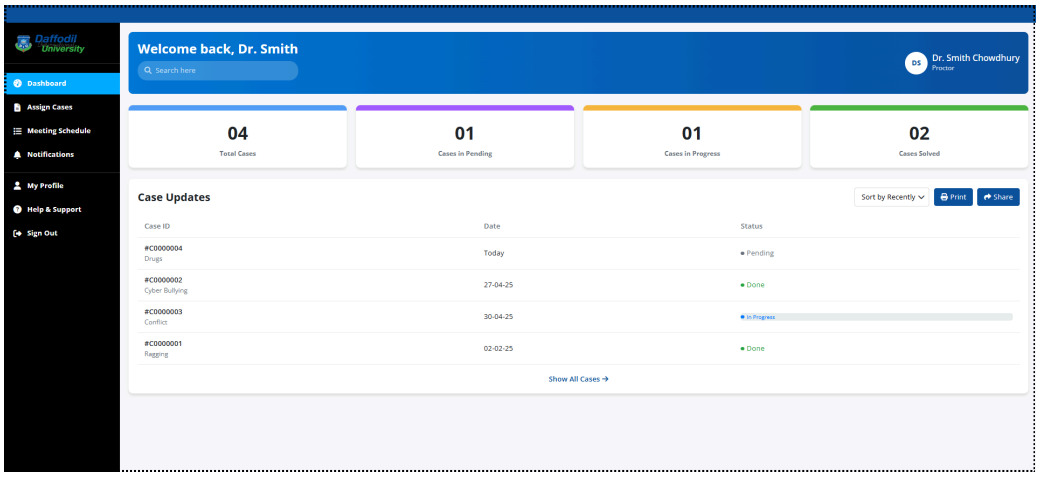
Appendix 5.2.23: Meeting Schedule

5.2.24 Create New Meeting



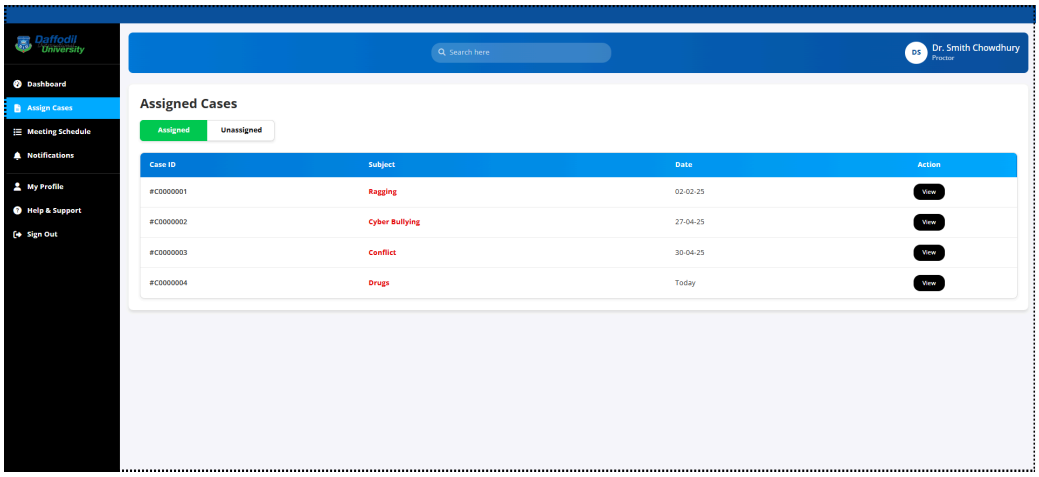
Appendix 5.2.24: Create New Meeting

5.2.25 Proctor Dashboard



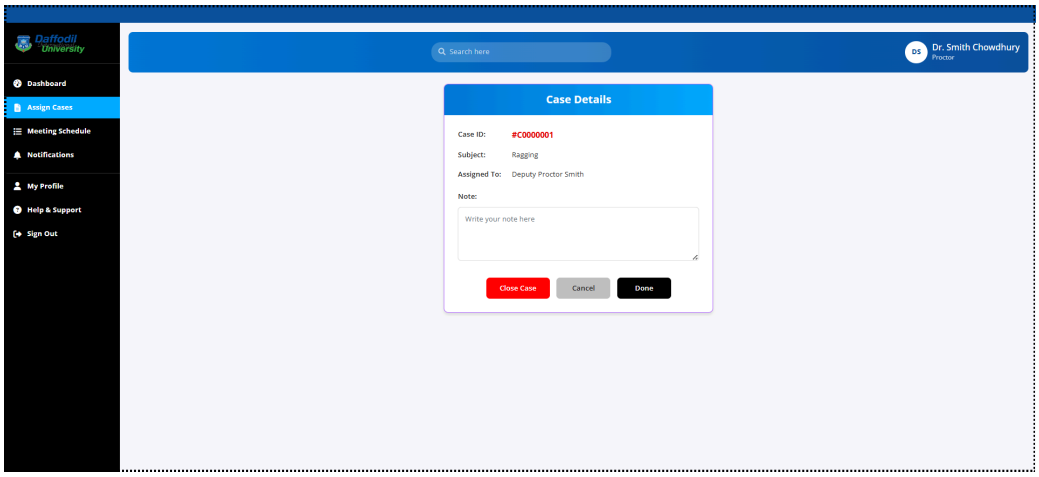
Appendix 5.2.25: Proctor Dashboard

5.2.26 Assign Cases



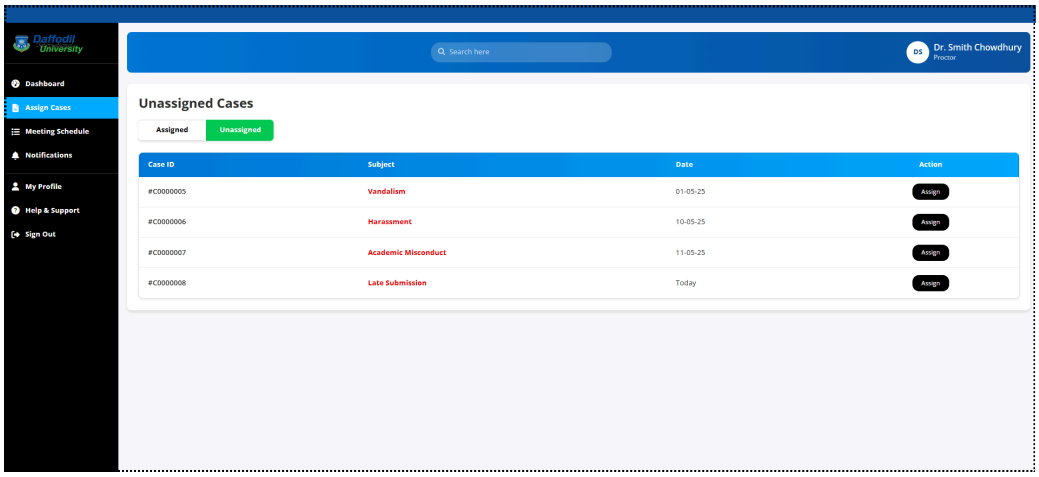
Appendix 5.2.26: Assign Cases

5.2.27 Case Details



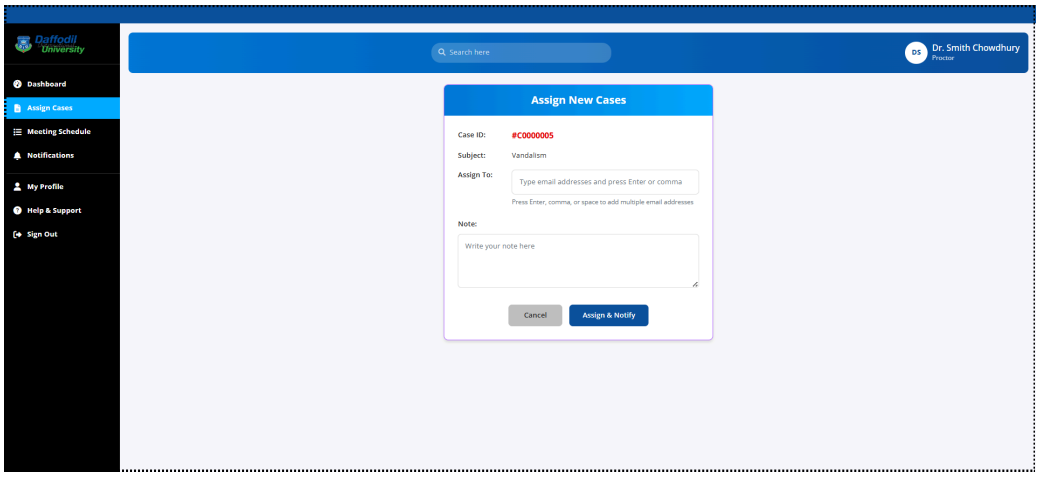
Appendix 5.2.27: Case Details

5.2.28 Unassigned Cases



Appendix 5.2.29: Unassigned Cases

5.2.29 Assign New Case



Appendix 5.2.29: Assign New Case

CHAPTER 6: PROJECT SUMMARY

6.1 Introduction

The Proctor Repository System is an integrated online platform aimed at improving the complaint management and disciplinary process at Daffodil International University. This system deals with the major issues that existed beforehand, including handling complaints manually through paperwork, a lack of transparency, the delayed process of complaint resolution, and the lack of communication between students and the proctorial staff. Through the development of an automated system, there will be increased efficiency and user friendliness throughout the proctoring process.

The system that has been developed with the latest technology stack that comprises the Front End with Angular, the Back End with ASP.NET Core, Database Management with PostgreSQL, and Security with ASP.NET Identity featuring JWT/OAuth, ensures that the system offers the highest performance characteristics as well as the highest levels of security with the assurance of scalability. The development of the system utilizes other critical development tools like Visual Studio Code.

The system comprises major functional components including user registration, login system, complaint submission system, evidence uploading system, complaint forwarding system, complaint allocation system, case file building system, explanation submission system, meeting scheduling system, and notification system. Additionally, there are facilities provided specifically for each group of users that may include students, proctors, officers, or administrators. The system's UI pages, diagrams, and user manual are designed with usability at heart.

Overall, the Proctor Repository System has a significant effect on the efficiency and proper management of disciplinary case handling at the university. The system consolidates the information about the disciplinary case instances at one place and also increases the efficiency of the case management procedure through the automated performance of repeating tasks. This particular project fulfills the educational requirements and also acts as a feasible solution that can generate a significant impact at the university.

6.2 Project Limitation

Even though the Proctor Repository System provides an efficient and organized method of case management in the role of the proctor, the system also has the following weaknesses:

- **Limited Offline Support:** This system requires an active internet connection and cannot run offline. This could pose an issue if there's an internet problem.

- **Limited File Upload Capacity:** The uploading of evidence may be restricted by the server space available and the file size limit settings. This may result in the failure of uploading bigger media files.

- **Role-Based Access Boundaries:** User roles are enforced strictly by the system. Any improper configuration of roles may limit access to certain functionalities

- **Scalability Constraints:** Though scalable, performance could degrade based on the growth of users, grievances, and uploaded contents should server resources not be upgraded.

- **Dependency on Third-Party Services:** Sending notifications depends on third-party email/SMS service providers. Failure of these would lead to delay in notification(s).

- **Browser Compatibility Issues:** Since it may be difficult to accommodate compatibility with older browsers and/or less powerful hardware specifications, the system might not work perfectly.

- **No Mobile Application:** It only allows access via a web interface, and there isn't any mobile application available. This will create an issue of convenience to its users.

6.3 Scope

The Proctor Repository System encompasses all major processes entailed in student complaints and disciplinary matters in Daffodil International University. All these processes are transparently and accurately maintained by the Proctor Repository System through a centralized automated workflow. Major processes of the Proctor Repository System include:

- **Centralized Complaint Management:** The system enables all complaints from students to be collected within a single platform through a unified interface. The process ensures all complaints are stored and processed from this single source.

- **Role-Based Access for Admins:** While managing data, different privileges have been granted to various levels of authorities like Proctor, Deputy Proctor, Assistant Proctor, and Coordination Officer.

- **Complaint Tracking & Evidence Handling:** Complaints accompanied by their evidence can be submitted to the officers by students. The officers would be able to check and monitor these submission.

- **Automated Notifications & Logging:** This system will also be sending automated notifications about key steps to concerned parties regarding lodging of complaints, allocation of cases, and organizing meetings. All actions are recorded systematically for tracking and auditing purposes.

6.4 Future Work

Although this system fulfills all the criteria, some areas have also been pointed out for upgrade for further improvements:

- **Mobile Application Development:** Android and iOS applications can thus be developed. This will make it easy and accessible to students and admin users.

- **Advanced Analytics & Reporting Dashboard:** The use of data visualization for complaining trend data, time to resolve, and performance data would be useful.

- **AI-Powered Complaint Categorization:** The process of tagging or categorizing the complaints by computers using artificial intelligence would be faster.

- **Real-Time Chat or Communication Module:** Facilitating communication from both the students and police officers would be useful in avoiding delays.

- **Integrated Video Meeting Scheduling:** This involves integrating features like disciplinary meetings using tools like Zoom and Google Meeting.

- **Multi-Language Support:** Adding languages such as Bangla for ensuring greater accessibility.

- **Automated Reminder System:** Regular reminders about pending explanations, meetings, and cases.

6.5 Conclusion

The Proctor Repository System provides a complete web-based service that deals with the management of disciplinary cases at Daffodil International University in a comprehensive manner. The system not only saves time and increases accuracy by providing the ability for automated complaint submission, management of evidentiary submissions, case allocation, meeting scheduling, and notification delivery, but it also replaces paper-based systems in handling disciplinary cases. Although the Proctor Repository System meets all the existing demands, there are areas of improvement for better service delivery, including but not limited to improved reporting functions and provision of mobile services.

REFERENCES

Reference:

I learned from some websites and got help from them. I mention this webpage below.

Author or company name: Daffodil International University **Website title:** Office of the Proctor, DIU **Date of access:** 10-07-2025 **Website address(URL):** https://daffodilvarsity.edu.bd/proctor-office
Author or company name: Cornell University **Website title:** Title IX Reporting System **Date of access:** 10-07-2025 **Website address(URL):** https://titleix.cornell.edu/file-a-report/
Author or company name: ORCANOS **Website title:** Complaint management system **Date of access:** 22-07-2025 **Website address(URL):** https://www.orcanos.com/complaints-management-system
Author or company name: ComplianceQuest **Website title:** Complaints Management **Date of access:** 13-08-2025 **Website address(URL):** https://www.compliancequest.com/