Practical 10

Team Members:

21BCE166 - Nikunj Jayvin Mahida
21BCE168 - Niti Chirag Patel
21BCE237 - Prisha Tushar Shah
21BCE253 - Richa Anilkumar Yadav

AIM: Product deployment and Acceptance Testing. Include Deployment Strategies, Acceptance Test Plan considered for each sprint.

Sprint-1:

I. Deployment Strategies:

1. Agile Development: Employ an iterative development process, focusing on delivering a potentially shippable increment at the end of the sprint.

2. Pair Programming: Implement pair programming for critical features to improve code quality and share knowledge among team members.

3.Test-Driven Development (TDD): Start with writing tests before coding functionalities to ensure every piece of code is tested and to minimize bugs.

4.Code Review: Establish a peer review process to maintain code quality and consistency across the project.

5.Continuous Integration (CI): Integrate a CI pipeline to automate builds and tests, which helps in identifying integration issues early.

6. Responsive Design: Ensure the application is usable across different devices and screen sizes, enhancing accessibility and user experience.

II. Acceptance Test Plan:

1.As a driver, I want a selection page...

- Test Case: Ensure the selection page clearly labels options for "Driver" and "Police Officer".

- Acceptance Criteria: The user can visually distinguish and select either option without confusion.

2.As a driver, I want to be directed to the registration page...

- Test Case: Click on "Driver" to check redirection to the registration page.

- Acceptance Criteria: The redirection should be immediate and correct.

3. As a driver, I want to be directed to the login page after I have registered...

- Test Case: Register a new driver account and verify redirection to the login page.

- Acceptance Criteria: Upon successful registration, the user should be redirected to the login page.

4. As a driver, I want to receive an error message upon entering incorrect login credentials...

- Test Case: Attempt to log in with incorrect credentials.

- Acceptance Criteria: An appropriate error message is displayed.

5. As a driver, I want a "change password" button...

- Test Case: Verify the presence and functionality of the "change password" button.

- Acceptance Criteria: The button should lead to a secure password change form.

6. As a driver, I want to have a table with violations and their respective fines...

- Test Case: Check the table for completeness and accuracy against predefined data.

- Acceptance Criteria: The table displays all relevant information correctly.

7. As a driver, I want a “Fines” Button where I can view my pending fines...

- Test Case: Click on the “Fines” button and verify the display of pending fines.

- Acceptance Criteria: The page lists all pending fines associated with the user’s account.

8. As a driver, I want a “Pay fine” button...

- Test Case: Use the “Pay fine” button to initiate a payment.

- Acceptance Criteria: The payment process starts, with options to pay securely.

9. As a driver, I want a “View Receipt” button...

- Test Case: Click to view receipts after payments.

- Acceptance Criteria: Receipts for all completed payments should be accessible and accurate.

10. As a driver, I want an “Account” button where I can view my account details...

- Test Case: Access the account details section via the “Account” button.

- Acceptance Criteria: User should see all relevant personal and account details that can be edited.

Sprint-2:

I. Deployment Strategies:

1. Modular Development: Build the portal in modular components to isolate functionalities (like registration, fine generation, etc.), making the application easier to manage and scale.

2. API-First Approach: Design and implement APIs for all backend functionalities before developing the frontend. This ensures that the frontend and backend integration is seamless and that each component can be tested independently.

3. Security Measures: Given the sensitivity of police work, implement strong security protocols including data encryption, secure authentication (like OAuth), and regular security audits.

4. Responsive Design: Ensure the application interface is responsive and accessible across various devices, particularly mobile devices which may be frequently used in the field.

5. User-Centric Design: Focus on creating an intuitive and user-friendly interface, considering that the users will be police officers who need quick and easy access to functionalities.

6. Continuous Integration and Deployment (CI/CD): Use CI/CD pipelines to automate testing and deployment, ensuring that changes are reliably and continuously integrated into the production environment without downtime.

II. Acceptance Test Plan:

1. Registration and Login

- Test: Ensure that clicking "Police" redirects to the registration page, and after registration, redirects to the login page.

- Procedure:

  - Click "Police" on the selection page.

  - Fill out and submit the registration form.

  - Check for redirection to the login page.

  - Attempt to log in with both correct and incorrect credentials to validate error handling.

- Expected Result: Police officers are redirected appropriately and receive correct feedback on their actions.

2. Navigation and Usability

- Test: Confirm that all navigation buttons (Home, Generate Fines, Receipt, License Card, Account) are present and function correctly.

- Procedure:

  - Log in as a police officer.

  - Verify each button's presence and functionality.

  - Test the responsiveness of navigation by clicking on each, especially the "Home" button and icon.

- Expected Result: Each button leads to the respective page or functionality without errors or delays.

3. Functional Buttons

- Test: Verify the functionalities specific to police officers like generating fines, viewing receipts, and accessing license cards.

- Procedure:

  - Use the "Generate Fines" button to create a new fine and verify if it is saved and displayed correctly.

  - Click the "Receipt" button to view all receipts and check their accuracy.

  - Access the "License Card" button to view all license cards and ensure details are correct and complete.

- Expected Result: All functions perform as intended with all data presented accurately and securely.

4. Account Management

- Test: Ensure that the account details can be viewed and edited correctly.

- Procedure:

  - Navigate to the "Account" section using both the button and icon.

  - Verify that personal details are correct and can be updated.

  - Check for the proper application of changes.

- Expected Result: Account details are accurate, editable, and updates are reflected immediately.

Sprint-3:

I. Development Strategy

1. Agile Development: Utilize Scrum methodology to manage and execute these features within the sprint cycle. This approach supports iterative development and regular feedback.

2. Microservices Architecture: Implement services for generating, sending, receiving, and managing fines. This allows for better scalability and maintenance of the system.

3. API Development: Develop RESTful APIs for each service, ensuring that they can be consumed by the frontend and also integrated with external systems if needed.

4. Security and Compliance: Ensure all data handling complies with relevant data protection regulations (such as GDPR, HIPAA, depending on locale) and use secure transmission protocols.

5. Automated Testing: Implement continuous integration (CI) processes with automated testing (unit, integration, and E2E tests) to ensure that all features work as expected and to catch bugs early.

6. User-Centric Design: The UI/UX should be designed with the end-user in mind, ensuring it is intuitive and accessible for police officers, often working in high-stress environments.

II. Acceptance Test Plan

1. Generate Fine Slips

- Test Case: Validate the functionality to create fine slips with all required details.

- Procedure:

  - Log into the portal as a police officer.

  - Navigate to the "Generate Fine" section.

  - Enter all necessary details and submit the form.

  - Verify that the fine slip is generated and stored correctly.

- Expected Result: Fine slips are created with accurate details and stored in the database.

2. Send Fine Slips

- Test Case: Ensure that generated fine slips can be sent to the offender.

- Procedure:

  - After generating a fine, use the option to send it via email or a printed slip.

  - Confirm that the system logs the sent action.

- Expected Result: Fine slips are sent successfully, and recipients receive them.

3. Receive Fine Slips

- Test Case: Simulate the reception of fine slips by a police officer.

- Procedure:

  - Generate and send a fine slip to a police officer.

  - Check the officer’s account for the receipt of the fine.

- Expected Result: The officer receives the fine slip accurately.

4. List All Fines

- Test Case: Check the functionality to view a list of all fines generated.

- Procedure:

  - Log into the officer’s portal.

  - Navigate to the "View Fines" section.

  - Verify that all generated fines are listed with basic details.

- Expected Result: All fines are correctly listed in a comprehensive manner.

5. Open Each Fine in the List

- Test Case: Ensure each listed fine can be opened to view full details.

- Procedure:

  - From the list of fines, select a fine and open it.

  - Verify that full details are displayed.

- Expected Result: Full details of fines are accessible and correctly displayed upon selection.

6. Download Details of Each Fine

- Test Case: Test the ability to download the details of each fine.

- Procedure:

  - Open a fine from the list.

  - Use the download option to save the details.

  - Verify the downloaded document for accuracy.

- Expected Result: Details are downloadable in a correct format, containing all necessary information.

Sprint-4:

I. Development Strategy:

1. System Design

- Database Schema: Design a database schema that supports storing fine slips, user details, and transaction records. This includes relationships between fines and drivers.

- API Development: Develop APIs to handle the creation, retrieval, sending, and downloading of fine slips.

- Frontend Implementation: Design a user-friendly interface that allows drivers to interact with their fines easily. This should include views for listing fines, viewing detailed information, and options to send or download fine details.

2. Technology Stack

- Backend: Use a robust backend framework such as Node.js with Express or Spring Boot, which can efficiently handle API requests.

- Frontend: React or Angular to build a responsive and interactive interface.

- Database: A relational database like PostgreSQL or MySQL, which can handle complex queries and relationships.

- Authentication: Implement secure authentication (JWT or OAuth) to ensure that drivers can only access their fine details.

- File Handling: Integration of a PDF generation library for downloading fine details.

3. Security Measures

- Ensure data encryption in transit (SSL/TLS) and at rest.

- Implement role-based access controls to restrict access to sensitive data.

4. Testing and Deployment

- Unit Testing: Write unit tests for backend logic and frontend components.

- Integration Testing: Ensure APIs and frontend components work together as expected.

- End-to-End Testing: Simulate user scenarios to test the complete flow of actions.

- Continuous Integration/Continuous Deployment (CI/CD): Set up pipelines to automate testing and deployment processes.

II. Acceptance Test Plan

1. Send Fine Slips

- Objective: Ensure drivers can send fine slips to themselves or relevant parties.

- Test Case: Drivers use the application to send a fine slip.

- Procedure:

- Log into the application.

- Navigate to the "My Fines" section.

- Select a fine and use the "Send" feature.

- Verify the fine slip is sent to the specified email or system.

- Expected Result: The fine slip is sent successfully, and confirmation is received.

2. Receive Fine Slips

- Objective: Drivers should be able to receive and view fine slips.

- Test Case: Simulate receiving a fine slip.

- Procedure:

  - Log into the application.

  - Check the inbox or dashboard for new fines.

- Expected Result: New fines appear correctly in the dashboard or inbox.

3. List All Fines

- Objective: Ensure drivers can view a list of all fines they have incurred.

- Test Case: Display all fines.

- Procedure:

  - Log into the application.

  - Navigate to the "My Fines" section.

  - Verify all fines are listed with essential details (date, amount, status).

- Expected Result: The fines are listed accurately and are easily navigable.

4. Open Each Fine

- Objective: Drivers should be able to view full details of each fine.

- Test Case: Open and review details of a fine.

- Procedure:

  - From the list of fines, click on a fine to view more details.

  - Verify all detailed information is displayed (violation, date, amount, etc.).

- Expected Result: Full details are accessible and correctly displayed upon selection.

5. Download Fine Details

- Objective: Drivers should be able to download the details of each fine.

- Test Case: Download fine details.

- Procedure:

  - Open a fine from the list.

  - Use the download option to save the details.

  - Open the downloaded file and verify its content.

- Expected Result: Details are downloadable in a correct format and contain all necessary information.