# Interim Report

*Grama-Link*

Grama Niladhari Services Management System

Group - CS 13
28/11/2024

# Contents

# Chapter 1

# Introduction

## 1.1 Domain Description

The Grama Niladhari (GN) administrative system plays a crucial role in delivering government services at the grassroots level in Sri Lanka. It serves as a bridge between citizens and the government by managing a wide range of activities, including citizen applications for certificates, handling complaints, conducting field visits, maintaining address records, and escalating matters to Administrative Grama Niladharis (AGNs).

Despite its significance, the current GN system is predominantly manual, relying on paper-based processes and in-person interactions. These outdated methods result in inefficiencies, delays, and a lack of transparency for citizens. Problems such as missing records, miscommunication, and the inability to track request statuses hinder the effectiveness of the system.

To address these challenges, we propose **Grama-Link**: a web-based platform designed to digitize and centralize the GN administrative framework. Grama-Link aims to streamline operations, automate workflows, and enhance citizen interaction with government services, ultimately creating a transparent, efficient, and user-friendly administrative system.

## 1.2 Current System and Limitations

**Current System:** The existing GN system is entirely manual, involving paper-based record-keeping, face-to-face communication, and fragmented processes. Key administrative tasks, such as tracking citizen applications, managing field visits, and resolving complaints, are labor-intensive and prone to errors. There is limited use of technology, leading to inefficiencies in service delivery and delays in critical operations.

**Limitations:**

- **Lack of Centralization:** The absence of a centralized database results in redundancy and missing records.
- **Manual Workload:** Manual data handling is time-consuming, increases the chances of human errors, and results in delays.
- **Poor Communication:** Citizens and government officials lack a streamlined communication channel, leading to mismanagement and frustration.

- **Limited Tracking:** Citizens cannot track the status of their requests, causing dissatisfaction and reduced trust.
- **Resource Mismanagement:** Inaccurate record-keeping hampers resource allocation and decision-making.

## 1.3 Goals and Objectives

**Goal:** To develop **Grama-Link**, an online administrative system that enhances efficiency, transparency, and accessibility in managing GN services. The platform will streamline operations for GNs, AGNs, and citizens while improving communication and accountability.

**Objectives:**

- **Automate Administrative Tasks:** Simplify field visit scheduling, application handling, and complaint resolution.
- **Enable Citizen Engagement:** Provide an easy-to-use platform for citizens to submit, track, and manage requests.
- **Enhance Communication:** Facilitate real-time notifications and interactions between GNs, AGNs, and citizens.
- **Improve Resource Management:** Centralize data storage and streamline resource allocation and decision-making.
- **Ensure Secure Data Management:** Implement robust data security protocols to protect sensitive information.
- **Provide Reporting and Analytics:** Offer tools for analyzing complaints, applications, and performance metrics to support informed decision-making.
- **Increase Transparency and Trust:** Build trust by allowing citizens to track the progress of their requests and access guidelines.

## 1.4 Assumptions

- All users, including citizens, GNs, and AGNs, have access to the internet.
- The system integrates with existing government databases to verify and manage citizen data.
- GNs and AGNs have basic computer literacy to operate the system.
- Citizens can access the platform via web and mobile applications.
- All necessary legal and administrative approvals are in place to implement the system.

# Chapter 2

# Feasibility Study

## 2.1 Technical Feasibility

The development of Grama-Link leverages modern, widely-used, and reliable technologies to ensure scalability, maintainability, and ease of use. Below is a breakdown of the technologies and tools utilized:

| **Category** | **Technologies/Tools** |
|---|---|
| **Front-end technologies** | HTML, CSS, JavaScript |
| **Back-end technologies** | PHP, MySQL |
| **Development tools** | Visual Studio Code, XAMPP |
| **Collaboration/version control** | GitHub, Git |
| **Designing/Visualization** | Figma, Canva, Draw.io |
| **Documentation tools** | Google Docs, Google Drive |
| **Communication tools** | Zoom, Microsoft Teams, WhatsApp |

Table 2.1: Technologies and Tools for Grama-Link Development

Grama-Link is feasible technically because it uses open-source and widely accessible tools, ensuring a cost-effective development process. Additionally, the project timeline allows ample time for learning and implementing these technologies effectively.

## 2.2 Operational Feasibility

Grama-Link's operational feasibility is assured by the clear requirements gathered from stakeholders, including citizens, Grama Niladharis (GNs), and Administrative Grama Niladharis (AGNs). The following factors support its viability:

- **Intuitive Design:** User-friendly interfaces with responsive designs for accessibility on both desktop and mobile devices.
- **Streamlined Processes:** Automation of manual tasks, such as complaint handling, application approvals, and scheduling, ensures efficiency.
- **Effective Communication:** Real-time notifications for request status updates and other alerts improve communication among all stakeholders.

- **User Training and Manuals:** Detailed user manuals and FAQs ensure smooth onboarding for all users.

By addressing the current system's inefficiencies, Grama-Link ensures a seamless operational process, making it both viable and user-friendly.

## 2.3 Economic Feasibility

Grama-Link is designed to be a financially sustainable project with minimal development costs. Below are the cost considerations:

1. **Development Cost:** The project is built by university students, with no external charges for development.
2. **Hosting Cost:** A free hosting service is used to minimize operational expenses.
3. **Software/Hardware Costs:** Free and open-source tools (e.g., PHP, MySQL, Visual Studio Code) are utilized. Development is done on personal devices, eliminating hardware expenses.

Given its reliance on open-source technologies and minimal operational expenses, Grama-Link is an economically feasible system.

## 2.4 Legal and Ethical Feasibility

To ensure compliance with legal and ethical guidelines, Grama-Link incorporates the following measures:

- **Data Security:** Sensitive citizen data is encrypted during storage and transfer.
- **Privacy Protection:** Access to user data is restricted to authorized personnel only.
- **Regulatory Compliance:** The system aligns with Sri Lankan regulations regarding data protection and citizen records.
- **Transparency:** User interactions and request statuses are logged for accountability.

By prioritizing privacy, security, and compliance, Grama-Link adheres to legal and ethical standards.

## 2.5 Schedule Feasibility

The project commenced in mid-2024 and follows a detailed timeline with an anticipated completion by mid-2025. Key points regarding the timeline include:

- **Development Hours:** Each team member dedicates five hours daily, totaling approximately 1,760 man-hours across the team.

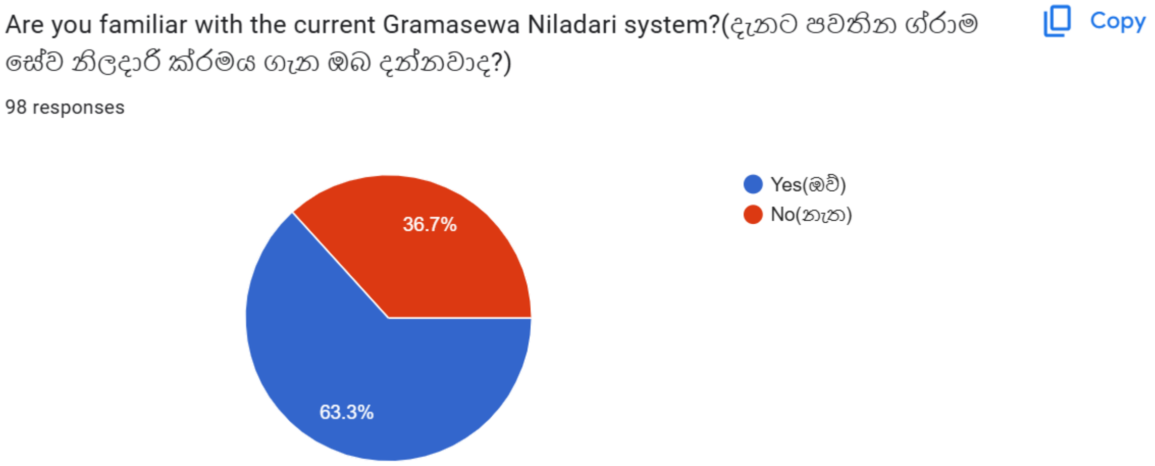
- **Development Model:** An iterative waterfall model ensures consistent progress and adaptability during development.
- **Stability of Requirements:** Requirements have been finalized, minimizing risks of major changes during the development process.

The detailed timeline and clear milestones demonstrate the schedule feasibility of the Grama-Link project.

## 2.6 Social Feasibility

Grama-Link addresses social feasibility by considering citizen needs, GN workflows, and AGN oversight. Stakeholder feedback indicates strong support for the system due to the following reasons:

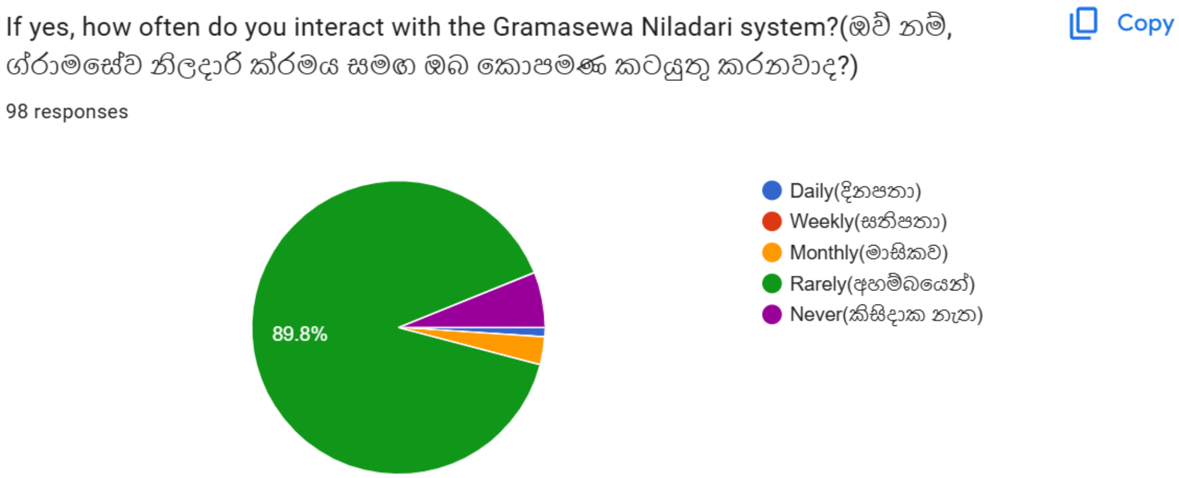
- **Streamlined Services:** Citizens find it convenient to submit applications, track complaints, and receive real-time updates.
- **Enhanced Efficiency:** GNs and AGNs appreciate the reduction in manual tasks, allowing them to focus on core responsibilities.
- **Collaboration Opportunities:** The system promotes coordination among government officers, improving decision-making and communication.
- **Survey Insights:** Informal surveys highlight significant interest among users, with most respondents supporting a digital platform for government services.



Are you familiar with the current Gramasewa Niladari system?(දැනට පවතින ග්රාම සේව නිලදාරී ක්රමය ගැන ඔබ දන්නවාද?)

98 responses

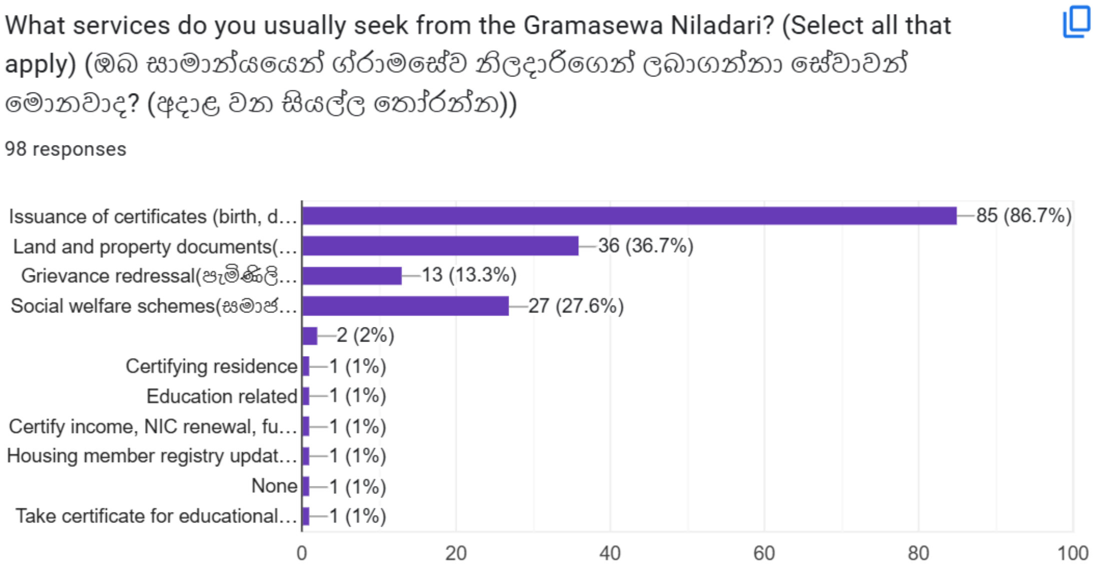
Figure 2.1: Social Feasibility Study Results - Survey Responses



If yes, how often do you interact with the Gramasewa Niladari system?(ඔව් නම්, ග්‍රාමසේව නිලදාරි ක්‍රමය සමඟ ඔබ කොපමණ කටයුතු කරනවාද?)

98 responses

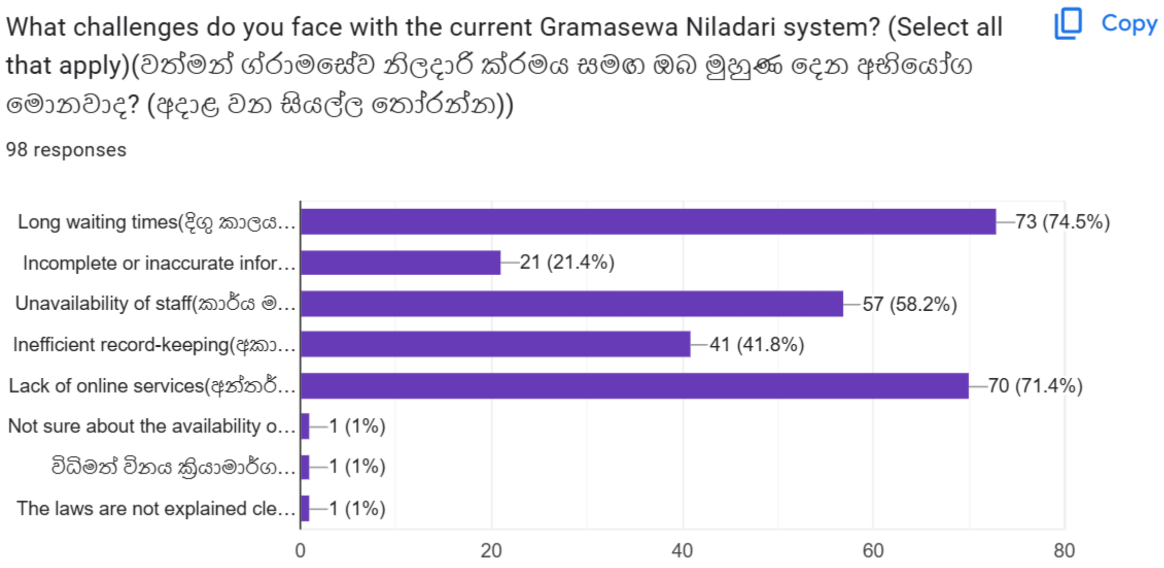
Figure 2.2: Social Feasibility Study Results - User Feedback



What services do you usually seek from the Gramasewa Niladari? (Select all that apply) (ඔබ සාමාන්‍යයෙන් ග්‍රාමසේව නිලදාරිගෙන් ලබාගන්නා සේවාවන් මොනවාද? (අදාළ වන සියල්ල තෝරන්න))

98 responses

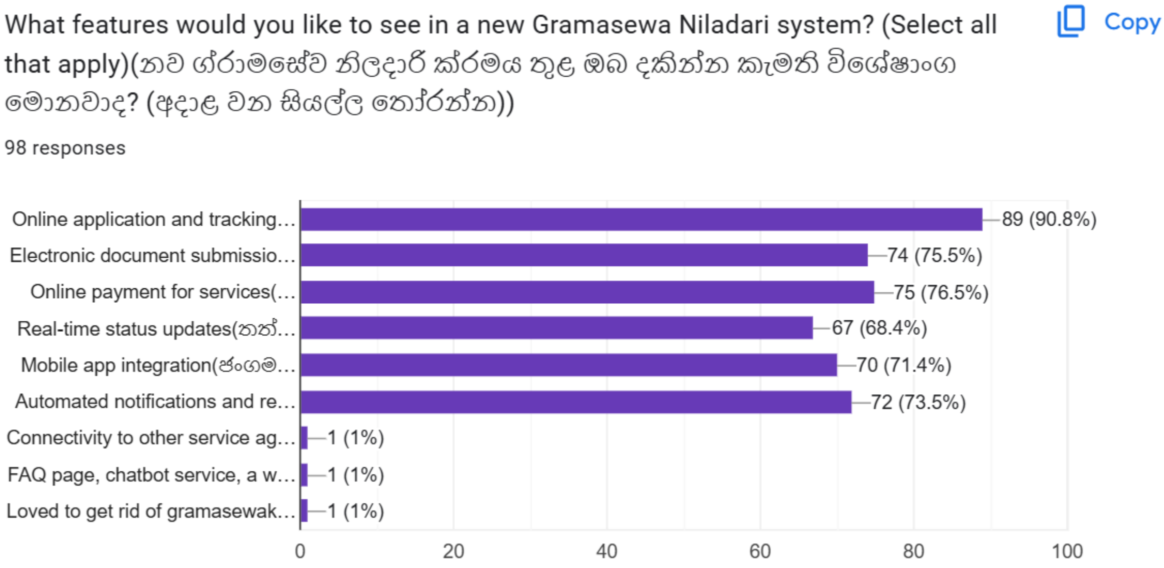
Figure 2.3: Social Feasibility Study Results - User Feedback



What challenges do you face with the current Gramasewa Niladari system? (Select all that apply)(වත්මන් ග්රාමසේව නිලදාරී ක්රමය සමඟ ඔබ මුහුණ දෙන අභියෝග මොනවාද? (අදාළ වන සියල්ල තෝරන්න))

98 responses

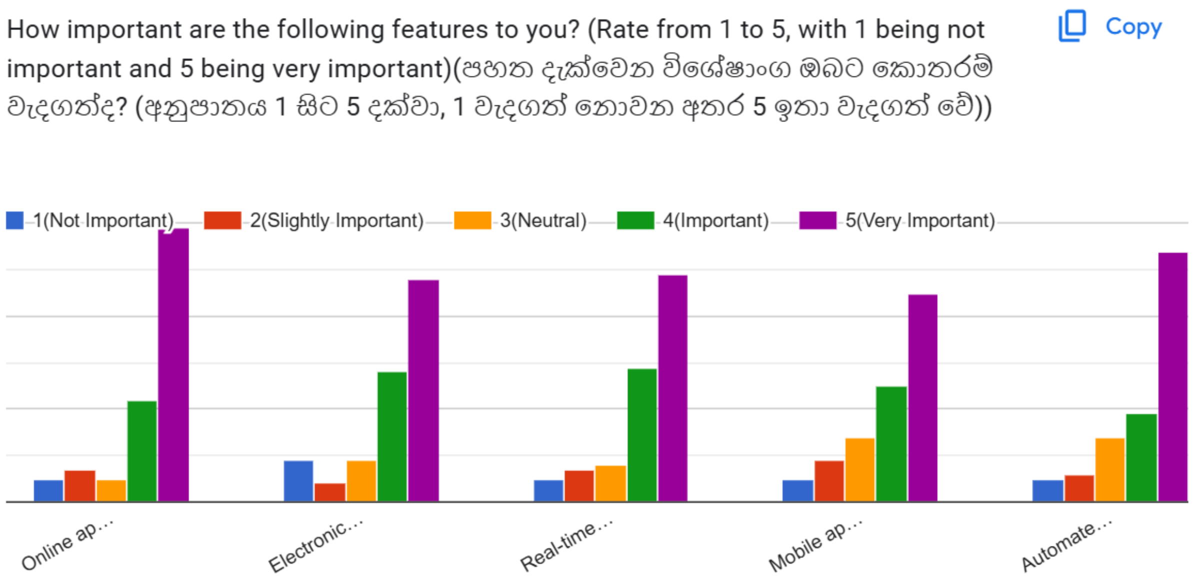
Figure 2.4: Social Feasibility Study Results - User Feedback



What features would you like to see in a new Gramasewa Niladari system? (Select all that apply)(නව ග්රාමසේව නිලදාරී ක්රමය තුළ ඔබ දකින්න කැමති විශේෂාංග මොනවාද? (අදාළ වන සියල්ල තෝරන්න))

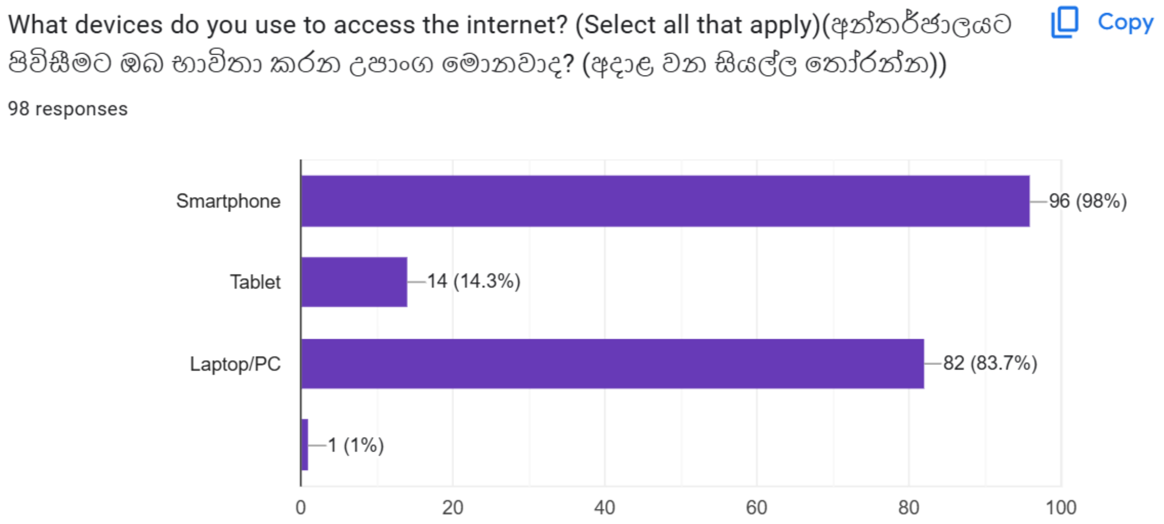
98 responses

Figure 2.5: Social Feasibility Study Results - User Feedback



How important are the following features to you? (Rate from 1 to 5, with 1 being not important and 5 being very important)(පහත දැක්වෙන විශේෂාංග ඔබට කොතරම් වැදගත්ද? (අනුපාතය 1 සිට 5 දක්වා, 1 වැදගත් නොවන අතර 5 ඉතා වැදගත් වේ))

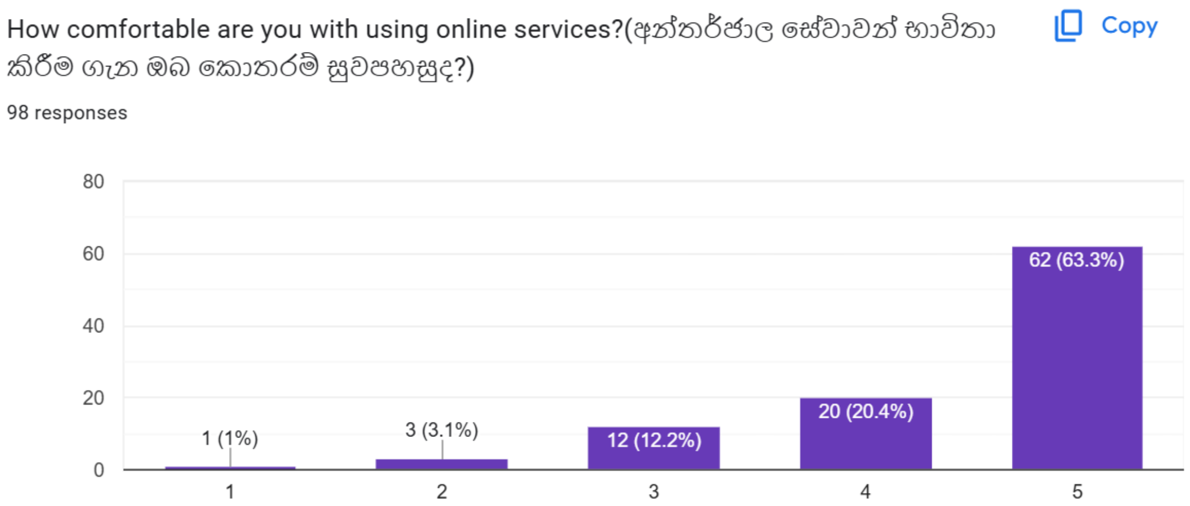
Figure 2.6: Social Feasibility Study Results - User Feedback



What devices do you use to access the internet? (Select all that apply)(අන්තර්ජාලයට පිවිසීමට ඔබ භාවිතා කරන උපාංග මොනවාද? (අදාළ වන සියල්ල තෝරන්න))

98 responses

Figure 2.7: Social Feasibility Study Results - User Feedback



How comfortable are you with using online services?(අන්තර්ජාල සේවාවන් භාවිතා කිරීම ගැන ඔබ කොතරම් සුවපහසුද?)

98 responses

Figure 2.8: Social Feasibility Study Results - User Feedback

By addressing both operational needs and social acceptance, Grama-Link ensures high adoption rates and effectiveness in transforming administrative processes.

# Chapter 3

# Requirements

## 3.1 Stakeholders / Actors

The stakeholders of the Grama-Link system include the following:

- **Citizens:** Citizens interact with the system to submit applications, file complaints, provide feedback, schedule appointments, and respond to field visit requests. They can also view notifications about their submissions and track the progress of their requests.
- **Grama Niladhari (GN):** GNs use the system to manage citizen applications and complaints, schedule field visits, and generate daily schedules. They can submit citizen requests for approval by Administrative Grama Niladharis (AGNs) and maintain detailed citizen profiles. GNs also receive notifications about complaints, field visits, and performance metrics.
- **Administrative Grama Niladhari (AGN):** AGNs oversee the operations of GNs, manage reports, handle escalated requests, and monitor compliance. They provide approvals, update citizen and GN records, and ensure that all operations align with policy standards.
- **System Administrators:** Administrators are responsible for managing user accounts, maintaining system security, and ensuring smooth operation of the platform.

## 3.2 Functional Requirements

The following functional requirements are essential to the Grama-Link system:

### 3.2.1 For Citizens

- Register, log in, and log out of the system.
- Submit, view, and track applications and complaints.
- Schedule appointments with GNs and respond to field visit requests.
- Receive real-time notifications on status updates.
- Submit feedback and view community announcements.

### 3.2.2 For Grama Niladhari (GN)

- View and manage citizen applications, complaints, and field visit schedules.
- Approve, reject, or forward applications to AGNs.
- Create and update daily schedules and field visit routes.
- Generate reports and update citizen records.
- Communicate with AGNs and receive performance metrics.

### 3.2.3 For Administrative Grama Niladhari (AGN)

- Oversee GN operations and approve forwarded requests.
- Generate and audit reports on GN performance and citizen requests.
- Handle escalations and policy updates.
- Communicate with GNs and citizens through notifications and announcements.

### 3.2.4 System-wide Features

- Secure user authentication and role-based access control.
- Real-time notifications for users regarding approvals, rejections, and updates.
- Calendar management for scheduling tasks and appointments.
- Data encryption and secure storage of user information.

## 3.3 Non-Functional Requirements

- **Usability:** The interface should be intuitive, mobile-responsive, and user-friendly for all stakeholders.
- **Performance:** The system should handle a high number of concurrent users with minimal response time.
- **Scalability:** The platform should be designed to accommodate a growing user base and increasing data.
- **Security:** User data must be encrypted, and all communications should use secure protocols.
- **Availability:** The system should be accessible 24/7 with minimal downtime.
- **Maintainability:** The system should follow modular design principles for easy updates and bug fixes.

## 3.4 In-Scope and Out-of-Scope

### 3.4.1 In-Scope

- Citizen registration, application submission, and complaint filing.
- GN and AGN management of requests and field visits.
- Notifications and updates on requests and tasks.
- Scheduling and calendar integration for appointments and events.
- Generating reports and analytics for GNs and AGNs.

### 3.4.2 Out-of-Scope

- Integration with third-party payment gateways for fees.
- Mobile application development.
- Real-time GPS tracking for field visits.
- Multi-language support.

## 3.5 Constraints and Limitations

- Each user is limited to one account per national identification number.
- Field visit schedules are restricted to pre-defined GN divisions.
- Applications and complaints require mandatory fields to be completed before submission.
- Notifications depend on users having reliable internet access.
- System functionality is limited to Sri Lankan government jurisdictions.

# Chapter 4

# Proposed System's Architecture

## 4.1 Components and Their Functionalities

The Grama-Link system is developed using the Model-View-Controller (MVC) architecture. This architecture separates the application's logic into three interconnected components, allowing for efficient development, scalability, and maintainability.

### 4.1.1 Model Layer

The **Model Layer** is responsible for handling all data-related logic. It interacts with the database, processes business rules, and provides data to the controller. Key functionalities include:

- Managing user data (citizens, GNs, AGNs) and their roles.
- Storing and retrieving applications, complaints, field visit schedules, and feedback.
- Maintaining records of notifications, appointments, and system logs.
- Managing reports and analytics for GNs and AGNs.

### 4.1.2 View Layer

The **View Layer** is responsible for presenting data to users through user interfaces (UIs). It handles the visual representation and user interactions. Key functionalities include:

- Displaying dashboards for citizens, GNs, and AGNs.
- Rendering forms for submitting applications, complaints, and feedback.
- Presenting calendars for scheduling field visits and appointments.
- Providing real-time notifications and status updates.
- Offering responsive and mobile-friendly designs for accessibility.

### 4.1.3 Controller Layer

The **Controller Layer** acts as an intermediary between the Model and View layers. It processes user inputs, updates the model, and determines which view to render. Key functionalities include:

- Handling requests from users and routing them to the appropriate modules.
- Validating user input and ensuring data integrity.
- Managing application logic, such as approving requests, scheduling tasks, and generating reports.
- Sending real-time notifications to users based on specific triggers.
- Coordinating interactions between different modules (e.g., citizen complaints and GN actions).

## 4.2 Component Interactions

The components in the MVC architecture interact seamlessly to ensure smooth system operation. Below is an outline of their interactions:

1. **User Interaction with Views:** Users (citizens, GNs, AGNs) interact with the system through forms, dashboards, and notifications in the View Layer. These inputs are passed to the Controller Layer for processing.
2. **Controller Processes User Requests:** The Controller receives user inputs, validates them, and performs the required business logic. It then updates the Model or retrieves data from the Model Layer.
3. **Model Handles Data Operations:** The Model interacts with the database to fetch or store data as requested by the Controller. It ensures that all data-related operations comply with business rules.
4. **Controller Updates the View:** Once the Model processes the data, the Controller updates the View Layer with the appropriate information or results (e.g., status of a request, notifications, or reports).
5. **Dynamic Updates:** The system ensures real-time updates through AJAX requests and event-based triggers, particularly for notifications, application statuses, and calendar updates.

## 4.3 Advantages of Using MVC Architecture

- **Separation of Concerns:** The clear separation between Model, View, and Controller enhances maintainability and scalability.
- **Reusability:** Models and Views can be reused for different components, reducing redundant code.

- **Scalability:** The modular nature of MVC allows for easier system expansion in the future.
- **Parallel Development:** Developers can work on different components (Model, View, Controller) simultaneously, speeding up development.
- **Testability:** Individual components can be tested independently, ensuring reliability.

# Chapter 5

# System Design Diagrams

## 5.1 Use case diagrams

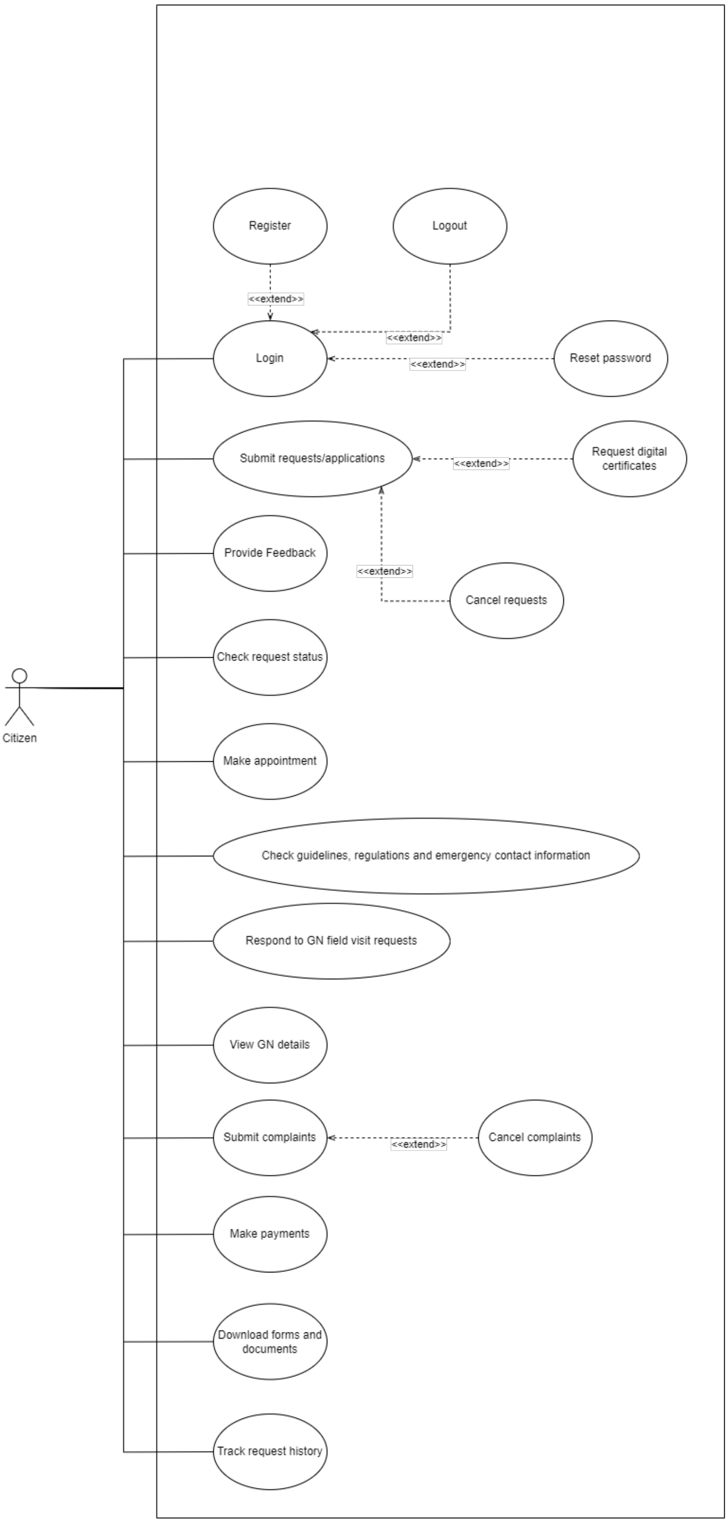


Figure 5.1: Use Case Diagram - Citizen

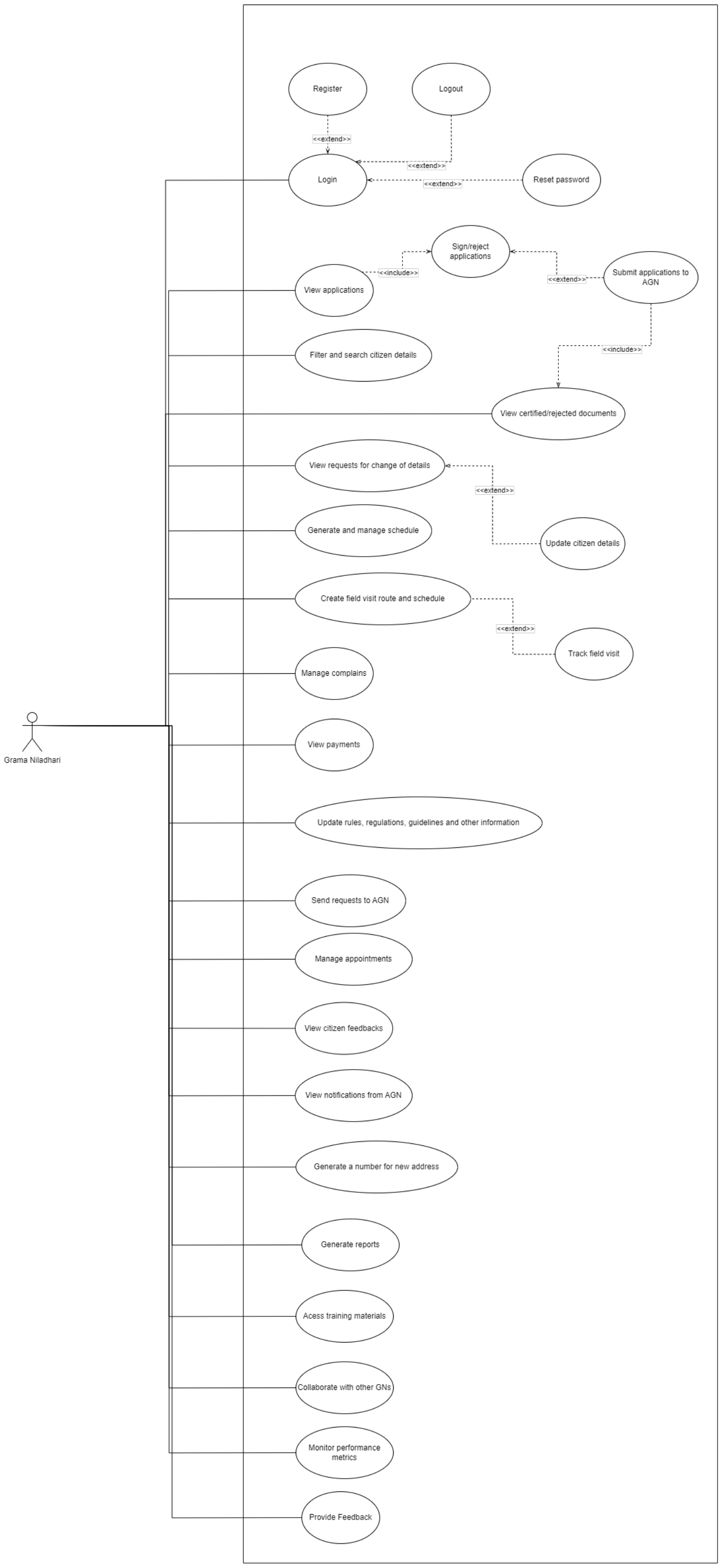

Register
Logout
<<extend>>
<<extend>>
Login
<<extend>>
Reset password
Sign/reject applications
<<include>>
View applications
<<extend>>
Submit applications to AGN
Filter and search citizen details
<<include>>
View certified/rejected documents
View requests for change of details
<<extend>>
Generate and manage schedule
Update citizen details
Create field visit route and schedule
<<extend>>
Track field visit
Manage complains
Grama Niladhari
View payments
Update rules, regulations, guidelines and other information
Send requests to AGN
Manage appointments
View citizen feedbacks
View notifications from AGN
Generate a number for new address
Generate reports
Acess training materials
Collaborate with other GNs
Monitor performance metrics
Provide Feedback

Figure 5.2: Use Case Diagram - GN

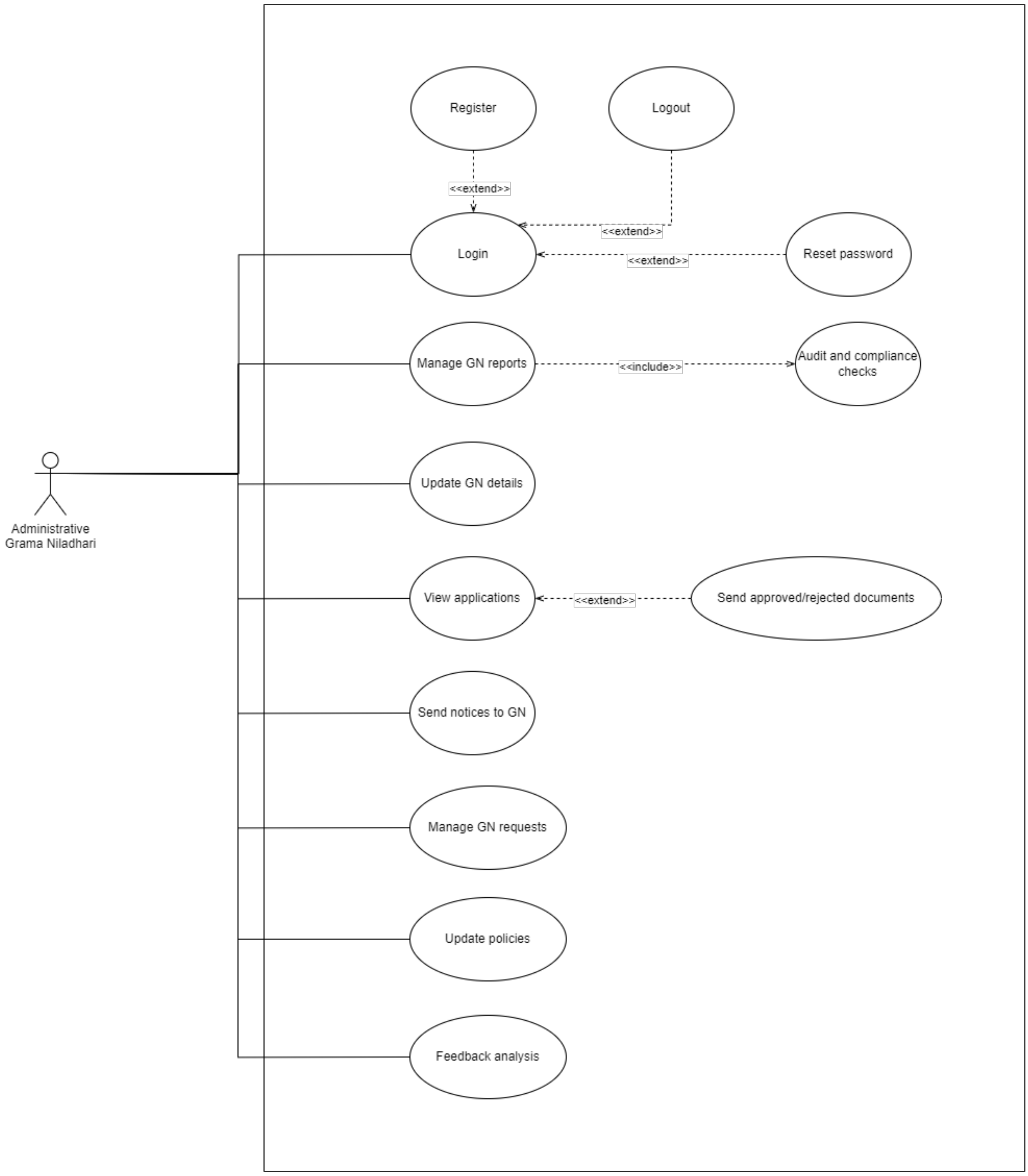


Figure 5.3: Use Case Diagram - AGN

## 5.2 Class diagrams

class.png

Figure 5.4: Class Diagram

## 5.3 ER Diagram

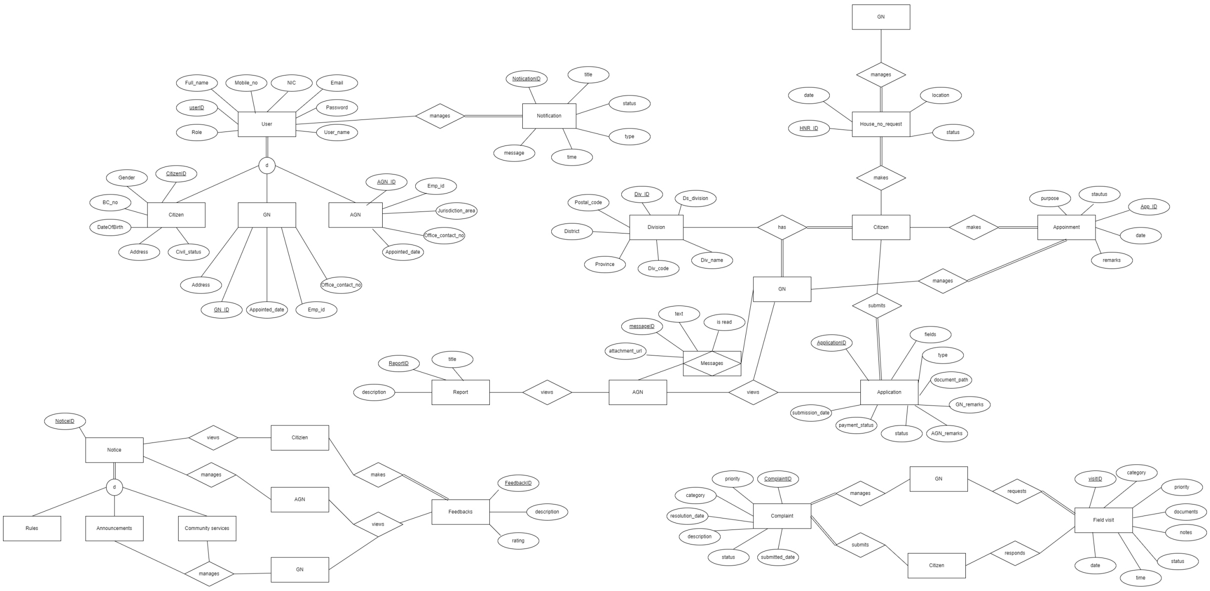


Figure 5.5: EER Diagram

## 5.4 Activity diagrams

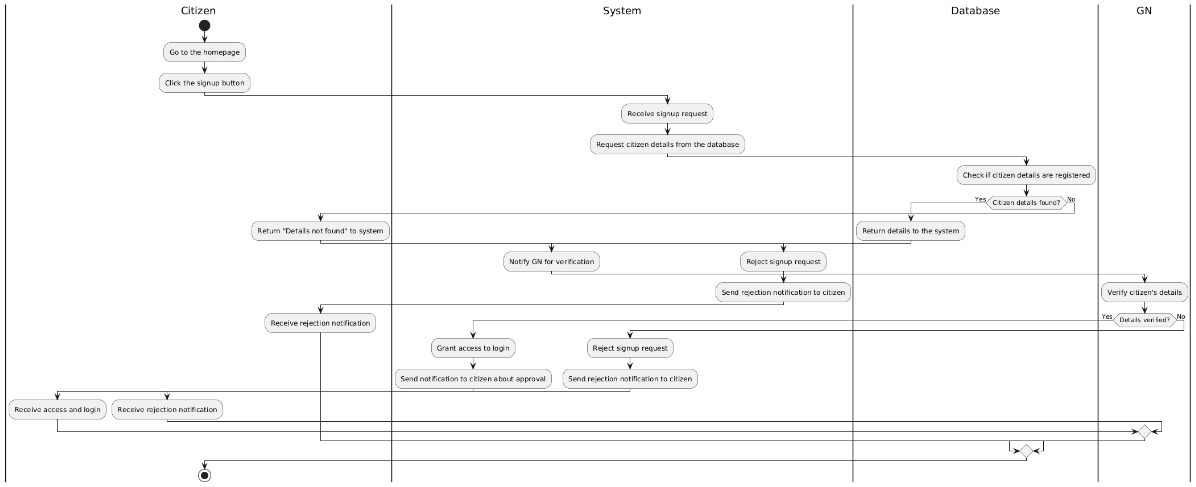

Citizen
System
Database
GN
Go to the homepage
Click the signup button
Receive signup request
Request citizen details from the database
Check if citizen details are registered
Yes
Citizen details found?
No
Return "Details not found" to system
Return details to the system
Notify GN for verification
Reject signup request
Send rejection notification to citizen
Verify citizen's details
Receive rejection notification
Yes
Details verified?
No
Grant access to login
Reject signup request
Send notification to citizen about approval
Send rejection notification to citizen
Receive access and login
Receive rejection notification

Figure 5.6: Activity Diagram - Signup

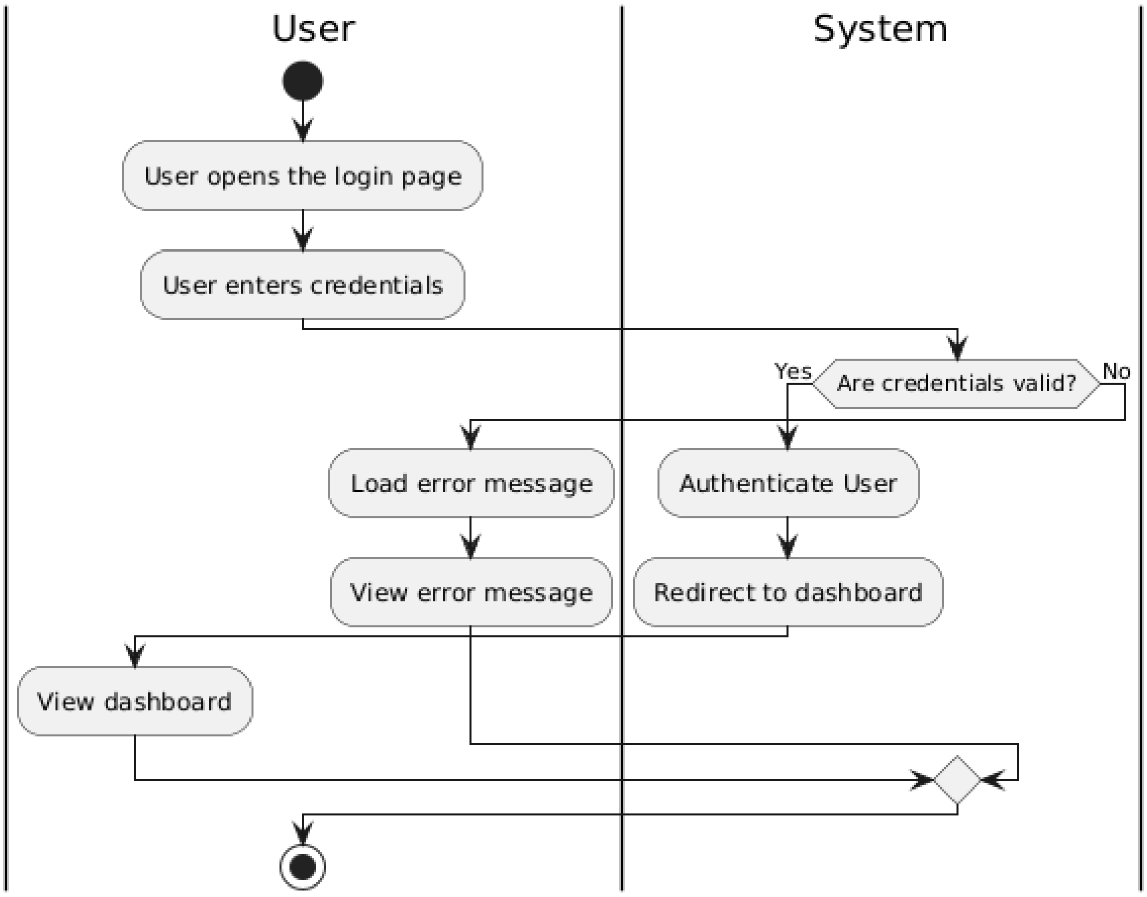


Figure 5.7: Activity Diagram - Login

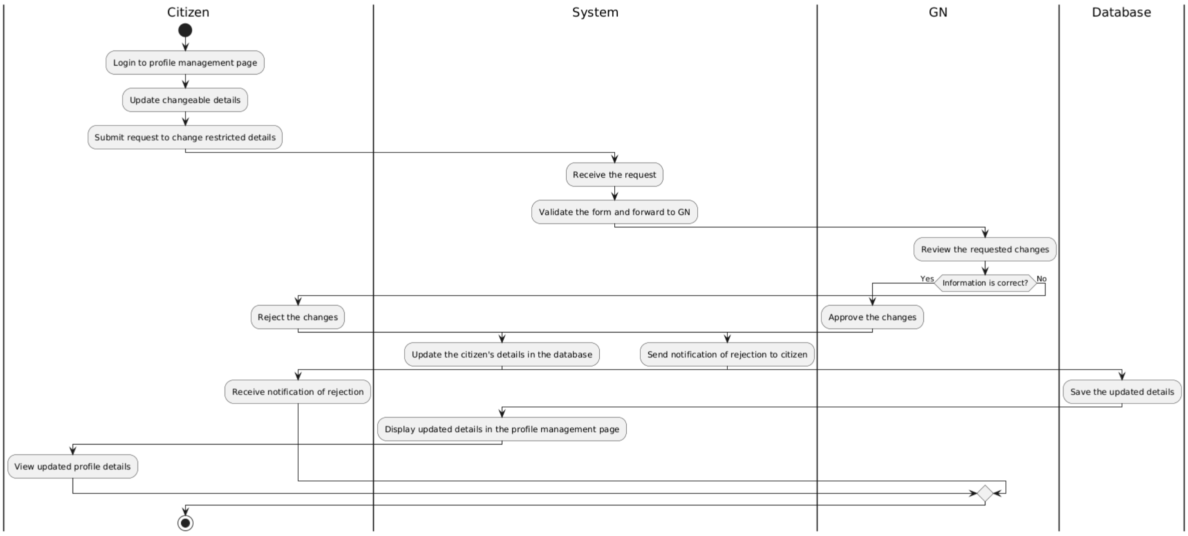

Citizen
System
GN
Database
Login to profile management page
Update changeable details
Submit request to change restricted details
Receive the request
Validate the form and forward to GN
Review the requested changes
Yes
Information is correct?
No
Reject the changes
Approve the changes
Update the citizen's details in the database
Send notification of rejection to citizen
Receive notification of rejection
Save the updated details
Display updated details in the profile management page
View updated profile details

Figure 5.8: Activity Diagram - Change Citizen Details

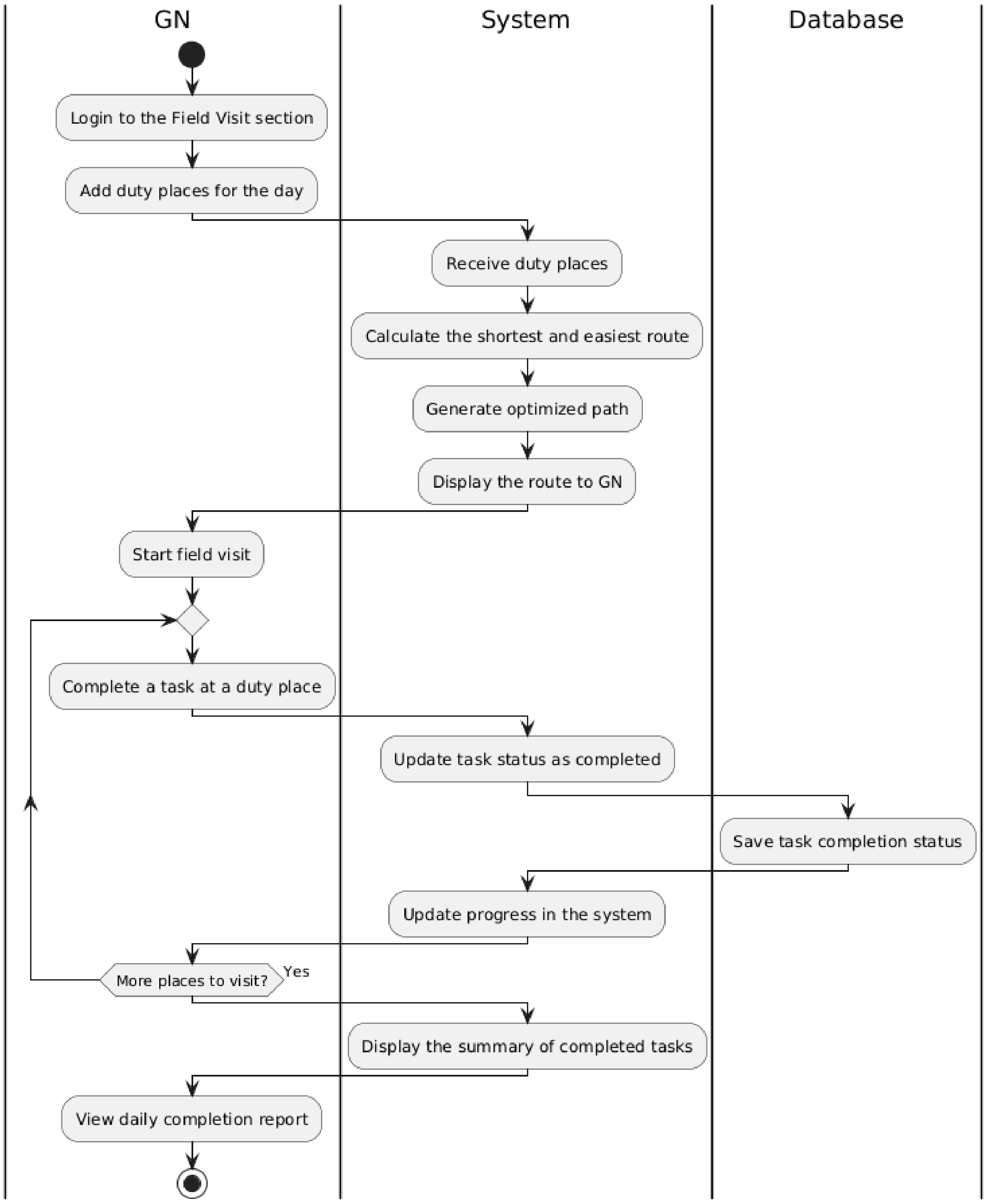

GN
System
Database
Login to the Field Visit section
Add duty places for the day
Receive duty places
Calculate the shortest and easiest route
Generate optimized path
Display the route to GN
Start field visit
Complete a task at a duty place
Update task status as completed
Save task completion status
Update progress in the system
More places to visit?
Yes
Display the summary of completed tasks
View daily completion report

Figure 5.9: Activity Diagram - Genarate Field Visit Route

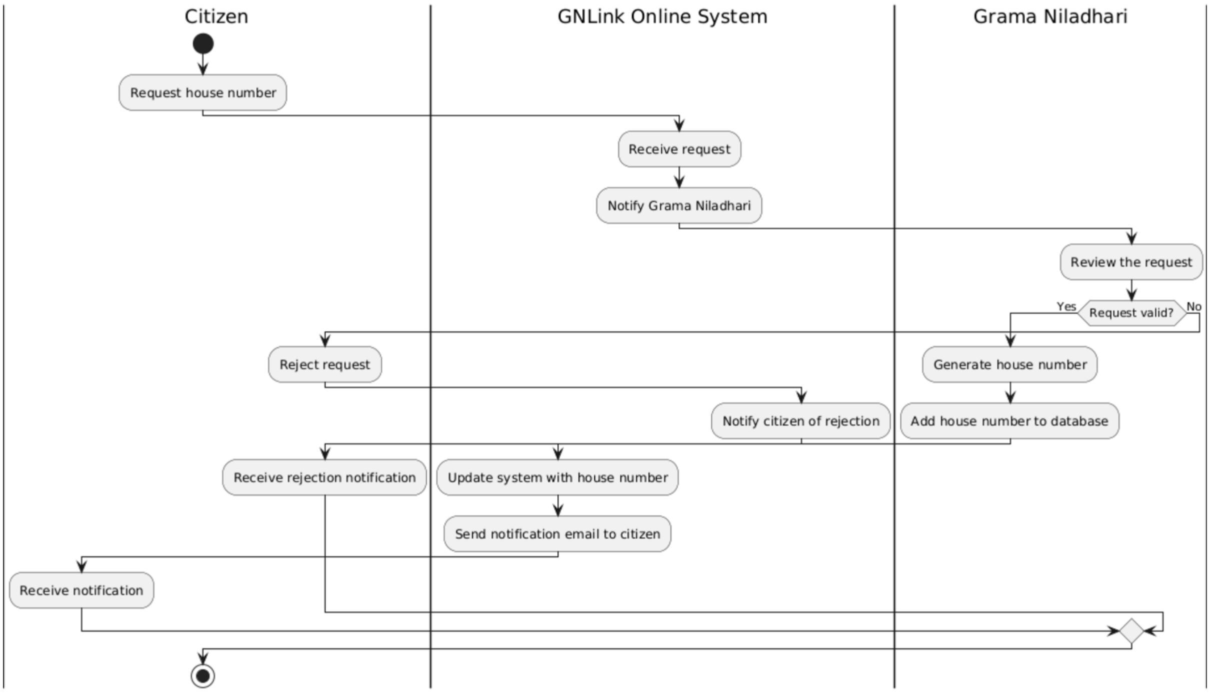


Figure 5.10: Activity Diagram - Generate House number

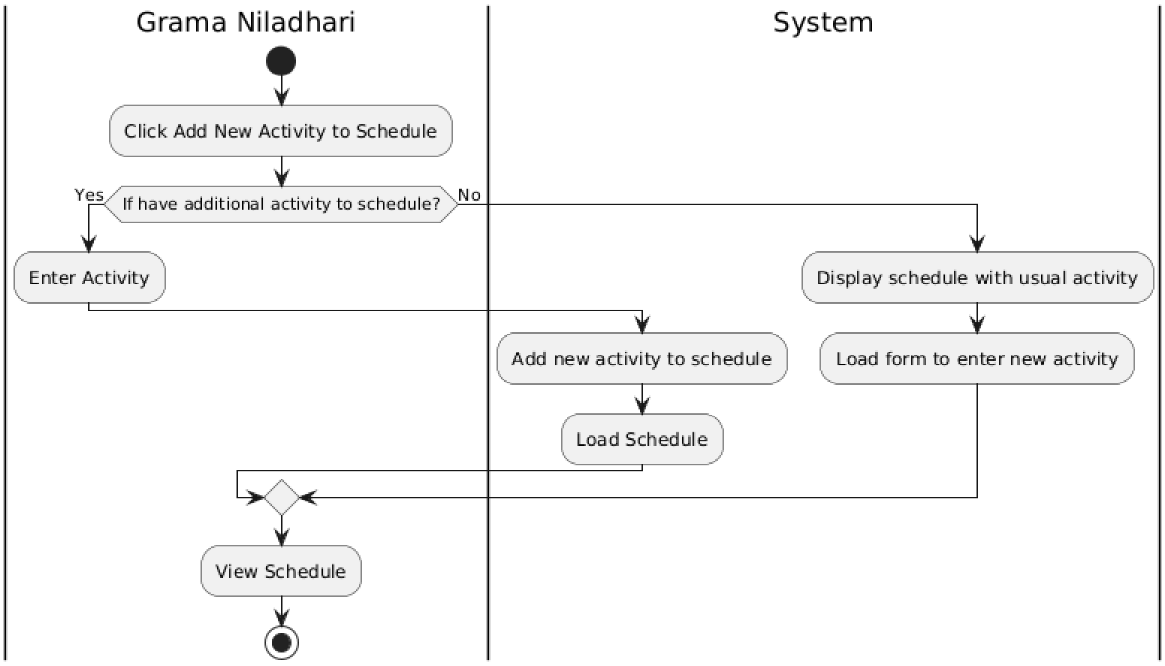

Grama Niladhari
System
Click Add New Activity to Schedule
Yes
If have additional activity to schedule?
No
Enter Activity
Display schedule with usual activity
Add new activity to schedule
Load form to enter new activity
Load Schedule
View Schedule

Figure 5.11: Activity Diagram - Generate Schedule

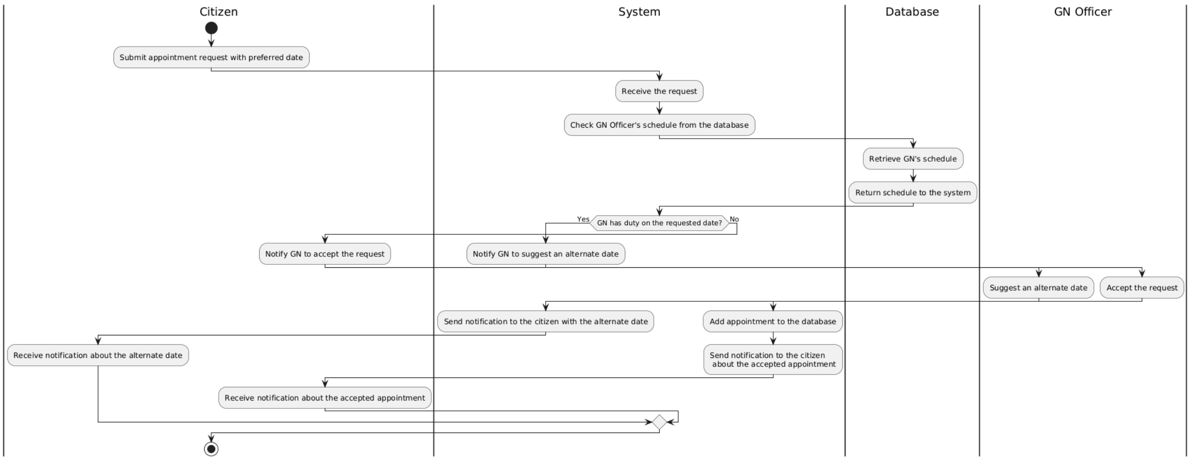

Citizen
System
Database
GN Officer
Submit appointment request with preferred date
Receive the request
Check GN Officer's schedule from the database
Retrieve GN's schedule
Return schedule to the system
Yes
GN has duty on the requested date?
No
Notify GN to accept the request
Notify GN to suggest an alternate date
Suggest an alternate date
Accept the request
Send notification to the citizen with the alternate date
Add appointment to the database
Receive notification about the alternate date
Send notification to the citizen about the accepted appointment
Receive notification about the accepted appointment

Figure 5.12: Activity Diagram - Make Appointment

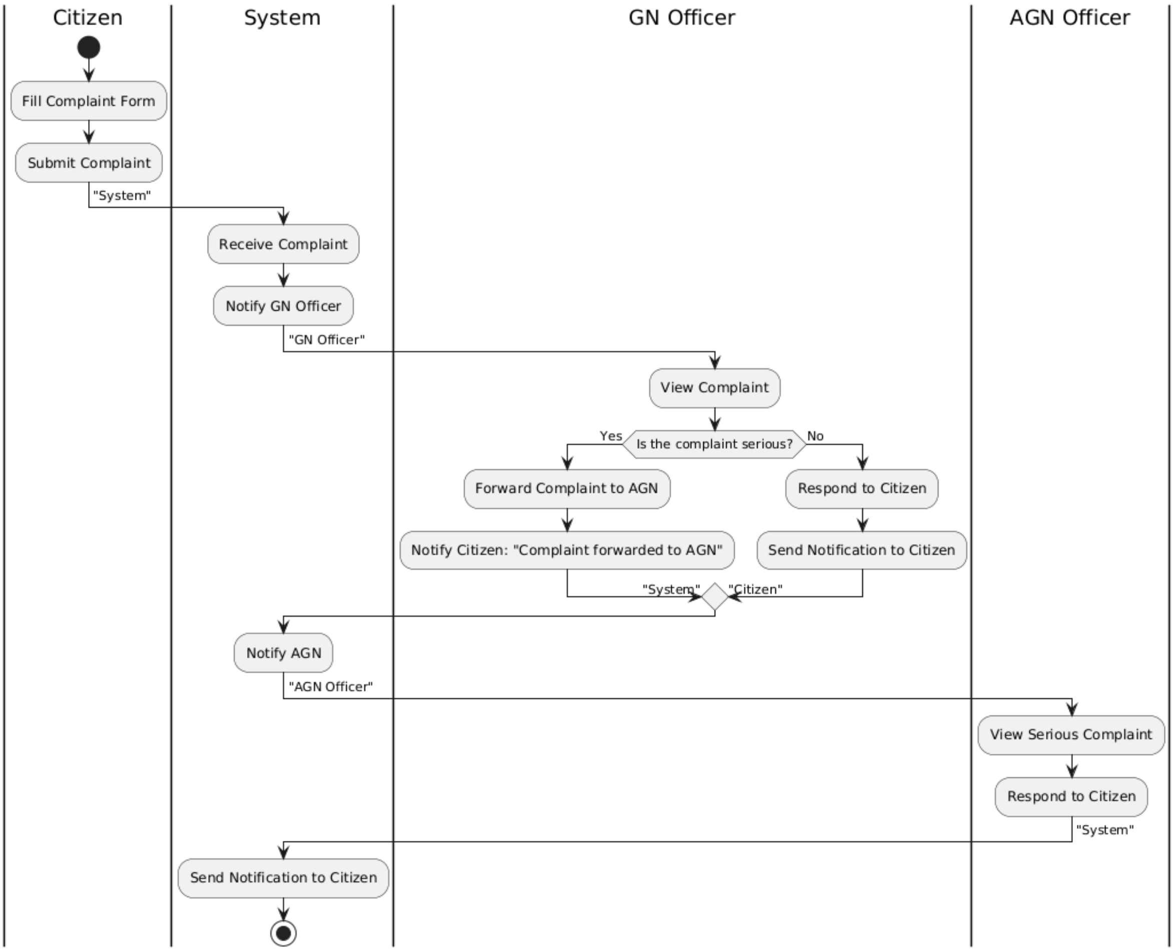


Figure 5.13: Activity Diagram - Manage Complaints

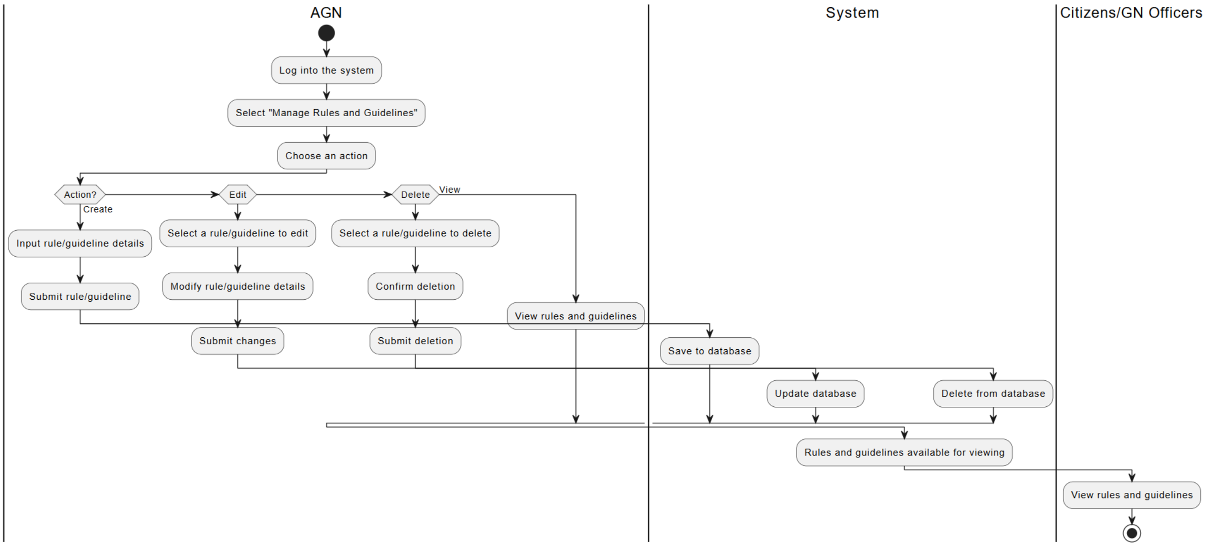

AGN
System
Citizens/GN Officers
Log into the system
Select "Manage Rules and Guidelines"
Choose an action
Action?
Create
Edit
Delete
View
Input rule/guideline details
Select a rule/guideline to edit
Select a rule/guideline to delete
Submit rule/guideline
Modify rule/guideline details
Confirm deletion
View rules and guidelines
Submit changes
Submit deletion
Save to database
Update database
Delete from database
Rules and guidelines available for viewing
View rules and guidelines

Figure 5.14: Activity Diagram - Manage Rules and Guidelines

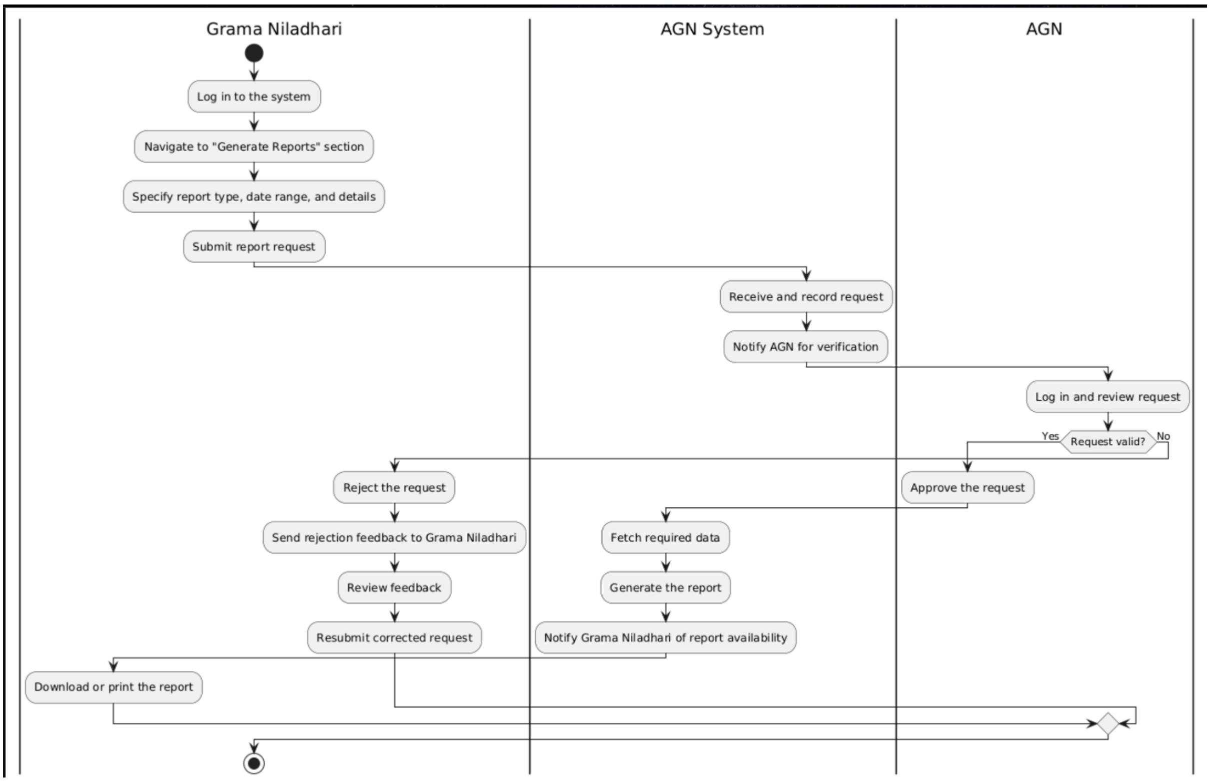

Grama Niladhari
AGN System
AGN
Log in to the system
Navigate to "Generate Reports" section
Specify report type, date range, and details
Submit report request
Receive and record request
Notify AGN for verification
Log in and review request
Yes
Request valid?
No
Reject the request
Approve the request
Send rejection feedback to Grama Niladhari
Fetch required data
Review feedback
Generate the report
Resubmit corrected request
Notify Grama Niladhari of report availability
Download or print the report

Figure 5.15: Activity Diagram - Generate Reports

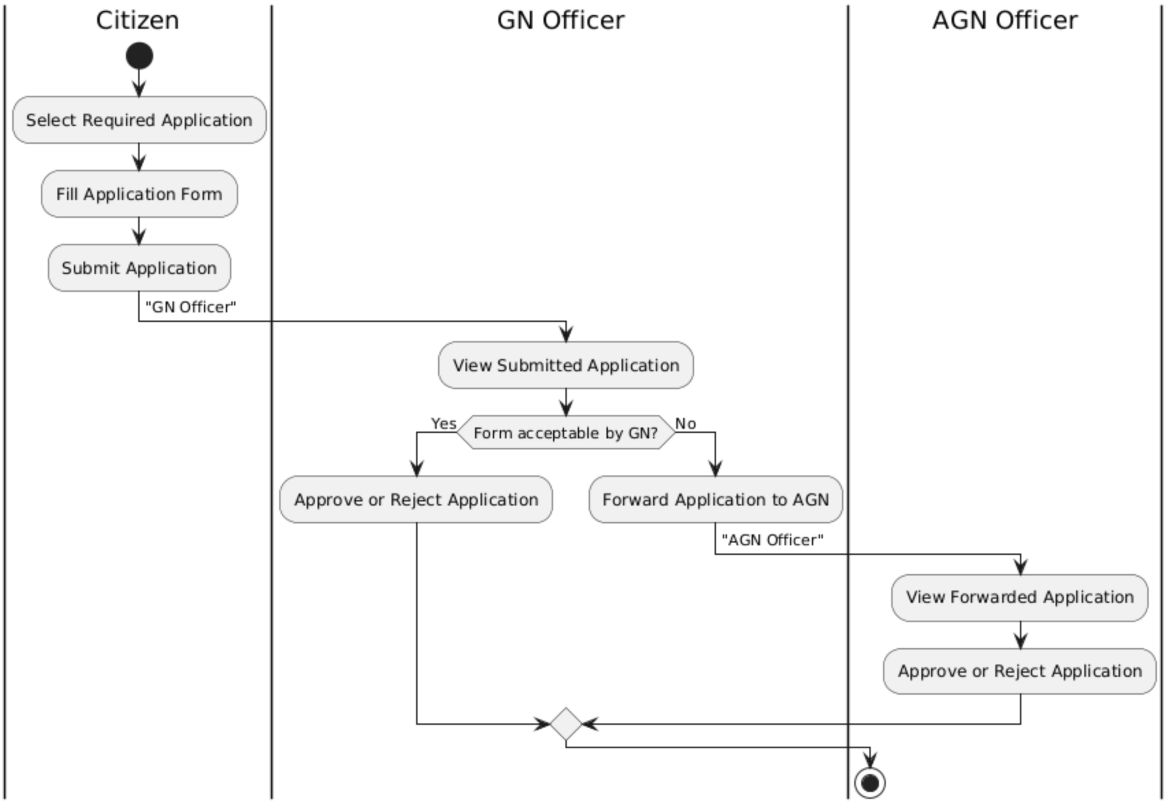

Citizen
GN Officer
AGN Officer
Select Required Application
Fill Application Form
Submit Application
"GN Officer"
View Submitted Application
Yes
Form acceptable by GN?
No
Approve or Reject Application
Forward Application to AGN
"AGN Officer"
View Forwarded Application
Approve or Reject Application

Figure 5.16: Activity Diagram - Submit Application

# Chapter 6

# Current Progress

## 6.1 System Development Progress Concerning Requirements

The development of the Grama-Link system is progressing steadily, with significant milestones achieved. Below is an overview of the progress:

### 6.1.1 1. Progress Overview

- **UI Development:** Approximately 95% complete, covering most of the system's interfaces for citizens, GNs, and AGNs.
- **Backend Development:** Approximately 30% complete, with significant progress in CRUD operations for core functionalities such as complaint handling and application management.
- **Database Development:** Initial schema design is complete, and approximately 40% of the tables have been populated with sample data.

### 6.1.2 2. Completed Work

The following work has been successfully completed:

- Developed responsive User Interfaces (UI) for:
  - Citizens (application submission, complaint management, feedback).
  - GNs (field visit schedules, complaint handling, and citizen interaction dashboards).
  - AGNs (dashboard management and GN monitoring).
- Implemented CRUD operations for:
  - Complaint Management.
  - Application Management.
  - Feedback and Notifications.
- Initial algorithms for:

  - Optimal route generation for field visits.
  - House number generation for new addresses.
- Authentication system (login, signup, and password reset) is fully functional.

### 6.1.3 3. Remaining Tasks

The remaining work is planned as follows:

- Complete the backend development for:
  - Field visit scheduling and report generation.
  - Application approval workflows.
  - Analytics and reporting for AGNs and GNs.
- Finalize the database schema and populate all necessary data.
- Integrate notification and real-time updates for all user roles.
- Refine UI elements for better usability.
- Perform system-wide testing and debugging to ensure stability.

## 6.2 System Completion Percentage

- Overall system completion: **60%**.
- UI development: **100%**.
- Backend development: **20%**.
- Database development: **20%**.

## 6.3 Each Member's Contribution

The contributions of the team members are as follows:

| **Team Member** | **Tasks Completed/Assigned** |
|---|---|
| **Amantha** | • Designed and implemented the Citizen UI.<br>• Developed the backend for complaint submission and management.<br>• Implemented CRUD for feedback handling. |
| **Dilshani** | • Created the GN dashboard UI.<br>• Implemented the notification system backend.<br>• Developed algorithms for field visit route optimization. |
| **Sujan** | • Designed and implemented the AGN interface.<br>• Built backend for application approval and monitoring.<br>• Created the database schema and populated initial data. |
| **Kavishka** | • Developed citizen-side functionalities for application tracking.<br>• Created reporting features for analytics dashboards.<br>• Implemented the house number generation algorithm. |

Table 6.1: Team Contributions