SANTA CRUZ CONNECT: A WEB-BASED PLATFORM FOR STRENGTHENING CITIZEN-LGU IN SANTA CRUZ

A Final Project

Presented to the

Faculty of College of Computer Studies

Laguna State Polytechnic University

Santa Cruz Campus

In partial Fulfilment of the Requirements for the course

CLIENT SERVER TECHNOLOGIES

Submitted by:

Luis Mario, F. Acovera

Jeffrey J. Almojela

Carl Jeremie F. Eusebio

Nico Adrielle N. Montorio

December 2025

CHAPTER I

INTRODUCTION

In the municipality of Santa Cruz, there is a noticeable lack of digital platforms that connect residents directly with the Local Government Unit. Many community issues and concerns are often posted on social media, particularly Facebook, where they tend to be overlooked or not properly addressed by the authorities.

This practice leads to delays in responses and miscommunication between residents and the LGU. To address this problem, the idea of developing a dedicated website for Santa Cruz was conceived a platform where citizens can directly submit complaints, requests, or assistance inquiries to the local government.

Through this website, communication can become faster, more organized, and more transparent. Ultimately, this project aims to promote better community engagement and encourage the proper use of digital technology in addressing local issues.

Background of the Study

In today’s digital age, technology plays a crucial role in how communities interact and access services. Localized digital systems are needed to connect different areas efficiently. For example, Santa Cruz can benefit from adopting digital platforms like those in Naga City, improving communication, access to information, and overall service delivery. This ensures the municipality keeps pace with modern developments and meets the needs of its residents.

Research Problem

The municipality of Santa Cruz currently lacks a dedicated digital platform for residents to communicate directly with the Local Government Unit (LGU), resulting in community concerns being posted on social media where they are often overlooked or delayed. This leads to miscommunication, slow response times, and limited citizen engagement. This study aims to address how a dedicated website can improve communication efficiency, provide an organized system for submitting complaints or requests, and enhance transparency and responsiveness in local governance.

Project Objectives

To develop a dedicated digital platform that enhances communication between the residents of Santa Cruz and the Local Government Unit (LGU) for faster, more transparent, and organized public service delivery.

Specifically, it aims to:

1. To design and develop a user-friendly website where residents can submit complaints, requests, or assistance inquiries directly to the LGU.
2. To create a centralized system that allows the LGU to efficiently track, manage, and respond to citizen concerns.
3. To promote transparency and accountability in local governance through accessible and organized digital communication.
4. To encourage community participation and engagement by utilizing digital technology for local issue reporting and feedback.

Scope and Limitation

This project focuses on developing a web-based platform that enables residents of Santa Cruz to communicate directly with the Local Government Unit (LGU) by submitting complaints, requests, and inquiries. The system aims to improve communication, transparency, and service efficiency within the community. However, the project is limited to a website version and requires internet access, which may affect accessibility for some users. It will not include additional e-governance services such as payments or permits, and its effectiveness will rely on the LGU’s responsiveness and proper system management.

Significance of the Study

This study is important as it introduces a digital platform to improve communication between the residents of Santa Cruz and the Local Government Unit (LGU). It provides citizens with a faster and more organized way to submit concerns while helping the LGU manage and respond efficiently. The project also promotes transparency, accountability, and community engagement through the use of modern digital technology.

1. Faster Response to Concerns

Residents can submit issues instantly, and the LGU receives them in real time. This reduces delays and allows quicker action compared to traditional walk-in or paper-based reporting.

2. More Organized Management of Complaints

The platform centralizes all reports, making it easier for the LGU to track, categorize, prioritize, and assign tasks to the appropriate departments.

3. Improved Transparency and Accountability

Residents can monitor the status of their concerns, and the LGU can show exactly how and when issues are being addressed—building trust between the public and the government.

4. Better Community Engagement

Citizens become more involved because reporting is convenient and accessible, and they feel heard. This leads to stronger collaboration between the community and the LGU.

5. Reduced Workload and Errors

Automated logs, notifications, and digital records reduce manual paperwork for the LGU and minimize human errors, resulting in more accurate record-keeping and smoother operations.

CHAPTER II

SYSTEM ANALYSIS AND DESIGN

Overview of Client-Server Model

The Santa Cruz Connect system uses a client-server architecture, which separates the user interface (client) from the processing and data management. This architecture is ideal for a citizen–LGU communication platform because it allows users to submit concerns, view community updates, and interact with staff, while the server securely handles data storage, report processing, and system logic. This separation ensures efficient operation, real-time communication, and organized service management for the municipality of Santa Cruz.

System Requirements

Client-Side Software:

Browser: Any modern browser (Chrome, Firefox, Edge, Safari), Operating System: Android 7+, iOS 11+, Windows 7+, macOS Sierra+ Recommended**:** Latest Chrome or Firefox**,** Latest Android, iOS, Windows, or macOS version**,** JavaScript enabled.

Server-Side Software

Since the system uses PHP + Apache + Firebase, the following are required:

- **PHP Version:** 7.4, **Web Server:** Apache 2.4 **Database:** Firebase Realtime

Database or Firestore **PHP Extensions:** cURL, JSON, OpenSSL, mbstring

- **Libraries / APIs:** Firebase PHP SDK, Free Weather API, ToolKit API, PSGC API, PHP Mailer
- Gemini Chatbot API **Recommended: PHP Version:** 8.1+ (better performance and security) **Web Server:** Apache 2.4+ with HTTPS enabled **Database:** Firebase Firestore (faster, scalable) Composer for managing PHP libraries SSL Certificate (HTTPS) for secure data transfer

Networking Requirements

Client-Side Internet Connection

- Minimum**:** 2 Mbps connection (loading pages, sending reports)
- Recommended: 5–10 Mbps for faster loading, community feed, and chat features

Server-Side Network Requirements

- Minimum: Stable hosting with basic bandwidth**.** Supports outgoing HTTPS API calls
- Recommended: Dedicated or cloud hosting with low latency Supports high-frequency API requests (Firebase, Free Weather, Gemini, PSGC)

Network Protocols

- Minimum: HTTP/1.1**.** REST API communication
- Recommended: HTTPS (TLS 1.2+) for encrypted data**,** HTTP/2 support for faster loading

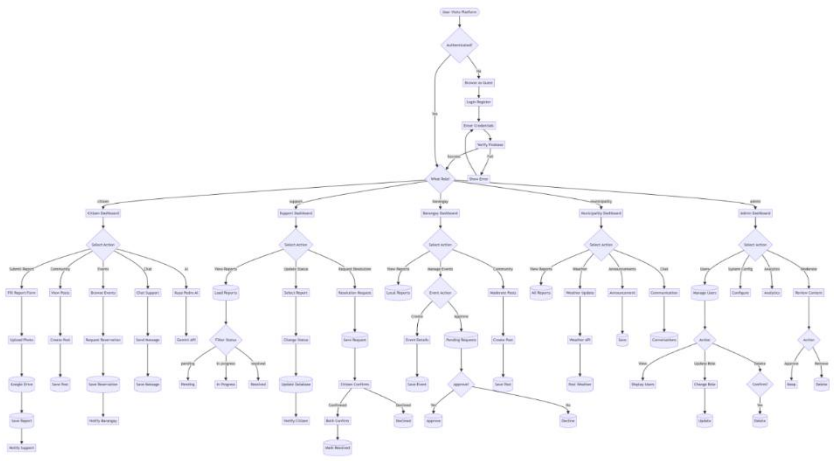


Figure 1: System Design

This diagram shows the overall workflow of a multi-role community management platform. Users begin by accessing the system, where they can log in, register, or browse as guests before being routed to a dashboard based on their role. Each role such as citizen, support staff, barangay officials, municipality personnel, and administrators has its own set of actions, ranging from submitting reports and managing events to moderating content and configuring system settings. The flowchart outlines how these actions connect, including submitting and processing reports, updating statuses, posting community content, and managing users. Overall, it maps out how the entire platform operates and how different user groups interact within it.

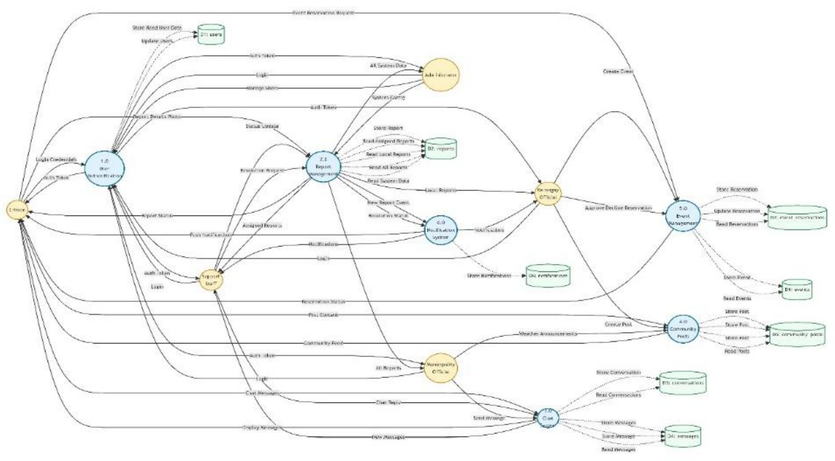


Figure 2: Data Flow Diagram

This diagram represents the data flow and interactions within a community management system involving different user roles and core system modules. It shows how citizens, support staff, barangay officials, municipality officials, and administrators exchange information with modules like user authentication, report management, event management, community posts, notifications, and chat. Each module both sends and receives specific types of data—such as reports, reservations, posts, notifications, and messages—which are stored in dedicated databases. The flow demonstrates how user actions trigger system processes, such as creating events, updating reservations, submitting reports, or posting content. Overall, it provides a clear overview of how information moves throughout the platform and how each role participates in various system functions.

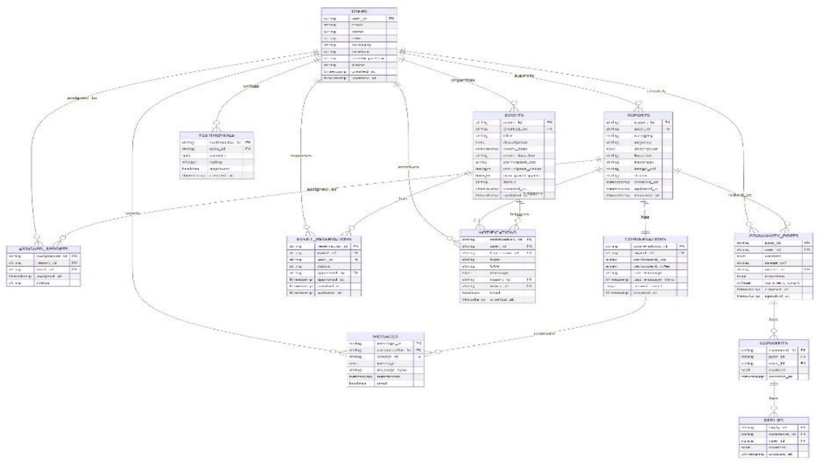


Figure 3: Entity Relationship Diagram

This diagram is an entity-relationship model that outlines how data is structured and connected within the system. At the center is the **Users** table, which links to major features such as reports, events, community posts, conversations, messages, notifications, and testimonials. Each surrounding table represents a specific function for example, **Reports**, **Events**, **Event Reservations**, **Assigned Reports**, and **Community Posts** and shows how users create, submit, organize, or interact with these records. The diagram also illustrates one-to-many and many-to-one relationships, indicating which data entities depend on or reference others. Overall, it provides a clear overview of how information is stored, related, and managed throughout the platform's database.

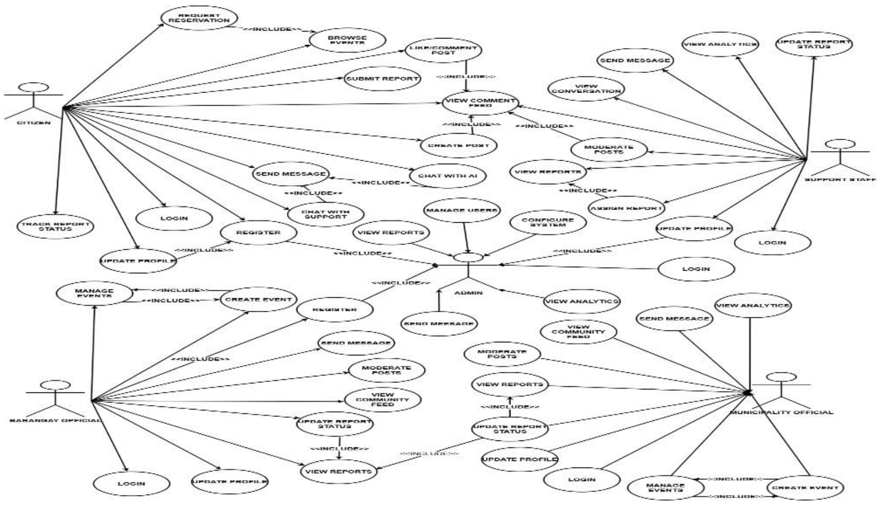


Figure 4: Use Case Diagram

The diagram is a **Use Case Diagram** that illustrates how different types of users interact with a reporting and community-management system. It shows five main actors—**Citizen, Barangay Official, Facility Official, Support Staff, and Admin**—each connected to the specific functions they can perform. Citizens can register, log in, submit and track reports, communicate through messages, view posts and events, and update their profiles. Barangay and Facility Officials have additional responsibilities such as managing and updating reports, moderating content, creating events, sending messages, and viewing analytics. Support Staff assist in communication, analytics viewing, and report updates. The Admin has the broadest privileges, including managing users, moderating reports, overseeing system settings, and viewing analytics.

API Integration

The **Santa Cruz Connect** system uses several external APIs to support essential features such as location validation, email notifications, image storage, authentication, and database management. These APIs help improve accuracy, automate processes, and ensure reliable system performance. This section explains the purpose of each API, the data exchanged, the endpoints used, and how they are integrated into the system.

2.8.1 PSGC API Integration (Barangay Data Only)

Purpose

The PSGC (Philippine Standard Geographic Code) API is used to retrieve official barangay names within the municipality of Santa Cruz. This ensures that all reports submitted by citizens contain accurate and standardized barangay information.

Functionality

Provides official barangay list for Santa Cruz, ensures accurate location tagging in reports, Removes manual entry errors (e.g., misspelled barangays)

Endpoints Used

GET /barangays?municipality_code=XXXXX (Filtered to Santa Cruz only)

Data Exchanged

- Request: HTTP GET request to fetch barangay list
- Response: JSON data containing barangay codes and names

Integration Method

The API is called when the citizen fills out the report form. The system loads barangay options dynamically, ensuring the location field is always valid.2.8.2 PHP Mailer Integration (Email Notifications)

Purpose

PHP Mailer is used for sending automated system emails such as: Password reset links, Report status updates (Accepted, In Progress, Resolved) Account verification or notifications

Functionality

Sends emails using SMTP, Supports HTML-formatted messages, Automates all communication that normally requires manual messaging

Data Exchanged

- Request: Email message details (recipient, subject, content)
- Response: Delivery success or failure

Integration Method

- The system fetches weather data when:
- A citizen reports weather-related issues (flooding, storms, etc.)
- The dashboard loads weather alerts
- The weather conditions help validate the report (e.g., confirming rainfall complaints).

- Weather alerts are displayed to users and can be pushed as notifications.

Purpose

The Weather API is used to gather real-time weather information for Santa Cruz. This helps improve the accuracy of reports related to flooding, heavy rainfall, or disaster risks. It also allows the system to display weather warnings to citizens

Functionality

- Retrieves current weather, rainfall, and temperature data
- Provides alerts for severe weather conditions
- Enhances the accuracy of environment-related reports submitted by citizens
- Supports LGU decision-making during emergency responses

Data Exchanged

- Request: File metadata + image uploaded
- Response: File ID and public link

Integration Method

- The system fetches weather data when:
- A citizen reports weather-related issues (flooding, storms, etc.)
- The dashboard loads weather alerts
- The weather conditions help validate the report (e.g., confirming rainfall complaints).

- Weather alerts are displayed to users and can be pushed as notifications.

Purpose

Firebase serves as the main backend for the system. It handles: User authentication (citizens, staff, admin), Cloud database storage for reports, chats, feed posts, and user profiles

Functionality

Authentication: Email/password and Gmail login, Realtime Database / Firestore: Stores all system records, Real-Time Updates: Instant syncing for dashboards, chats, and status changes.

Endpoints Used

- **POST /reports** - creating new reports
- **PATCH /reports/{id}** - updating report status
- **GET /messages/{report_id}** - retrieving chat conversations

Data Exchanged

- **Request Types:** GET, POST, PATCH, PUT
- **Format:** JSON
- **Authentication:** Firebase API key and tokens

Integration Method

All system data such as report details, chat messages, user accounts, and community posts is stored and retrieved from Firebase through REST API calls. The real-time capabilities of Firebase allow dashboards and conversations to update automatically without manual refresh.

Purpose:

Toolkit API is used for secure email-based verification and password reset functionality. Since Firebase's online database requires custom password reset implementation, Toolkit API handles:

- Sending verification codes via email
- Validating email verification codes stored in Firestore
- Enabling password changes after code verification
- Reading and managing verification codes in Firestore

Functionality:

- Generates random 6-digit verification codes
- Sends codes to user email addresses
- Stores verification codes temporarily in Firestore
- Validates codes before allowing password changes
- Auto-expires codes after specified time period

Endpoints Used:

- POST /send-verification - sends email with verification code
- GET /verify-code - validates provided verification code
- POST /reset-password - changes password after successful verification

Data Exchanged:

- Request: User email, verification code (for validation), new password
- Response: Success/failure status, code expiration time

Integration Method:

When a user requests password reset, the system generates a verification code using Toolkit API, stores it in Firestore with expiration timestamp, and sends it to the user's email. The user enters the code, system validates it against Firestore, and allows password change if valid.

Data Flow and User Interface

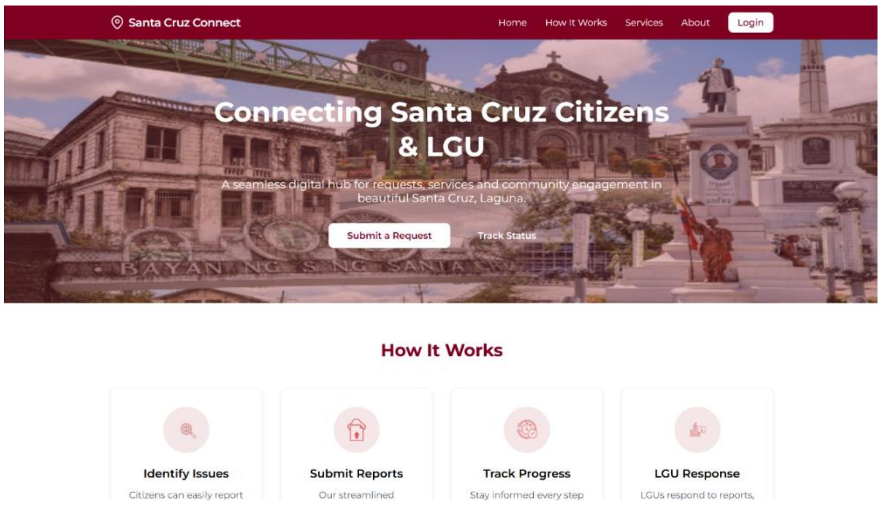


Figure 5. Landing Page Interface

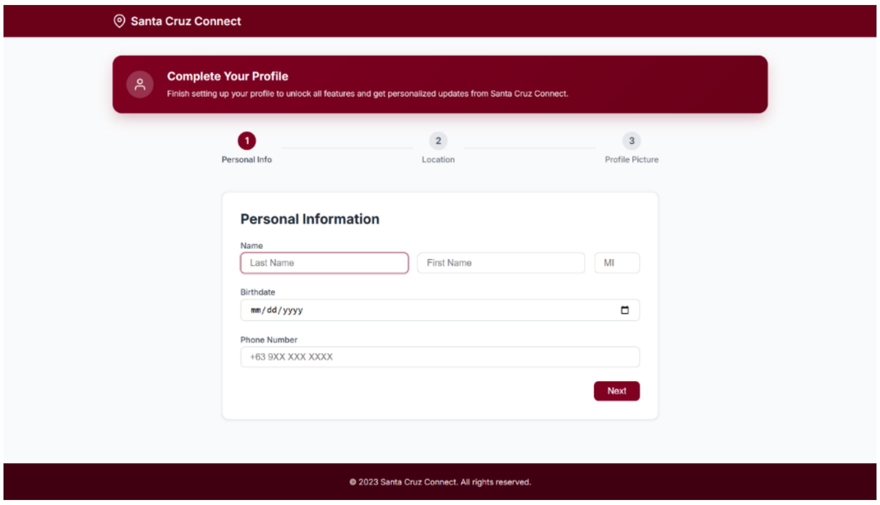
The landing page serves as the main entry point of Santa Cruz Connect. It introduces the purpose of the platform and provides users with easy access to essential features such as logging in, viewing services, learning how the system works, and navigating to report submission functions. Its design focuses on clarity and accessibility to ensure residents can use the system with ease.

User Interaction

From the landing page, users can navigate through different sections of the system. They can read about the platform’s purpose, explore available services, check how the reporting process works, or proceed to login and submit requests. This page acts as the starting point for residents, barangay staff, and LGU personnel when accessing the system.

Data Flow Description

When the user opens the landing page, the browser sends a request to the server to retrieve the required page components such as the HTML layout, styles, scripts, and images. The server responds by delivering these files, which the browser renders into the interface. If the landing page contains dynamic content such as announcements or testimonials, the browser may request additional data from Firebase through API calls. User interactions, such as clicking “Login” or “Submit a Report,” trigger navigation to other pages where further data processing occurs.


Santa Cruz Connect

Complete Your Profile

Finish setting up your profile to unlock all features and get personalized updates from Santa Cruz Connect.

1 Personal Info — 2 Location — 3 Profile Picture

Personal Information

Name: Last Name | First Name | MI

Birthdate: mm/dd/yyyy

Phone Number: +63 9XX XXX XXXX

Next

© 2023 Santa Cruz Connect. All rights reserved.

Figure 6. Complete Profile Interface

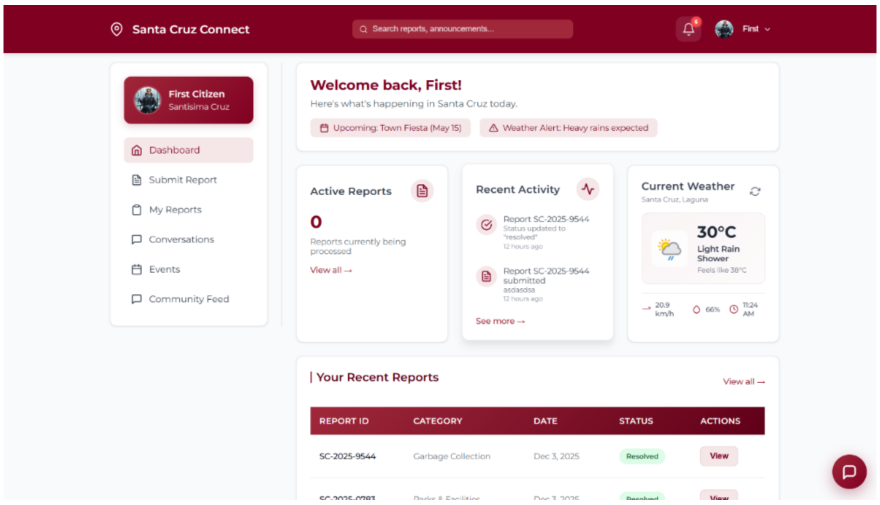
The Complete Profile page guides users through finishing their personal information, location details, and profile picture. It ensures that all necessary data is collected to provide a personalized experience and unlock full platform features. The page uses a step-by-step interface to make the process simple and clear for residents and municipal users.

User Interaction

Users move through three steps: entering personal information, selecting their location, and uploading a profile picture. Navigation buttons allow users to go back or proceed to the next step. Once completed, users gain access to all features of Santa Cruz Connect and can interact with the system smooth and efficiently.

Data Flow Description

When a user fills out each step, the browser sends the form data to the server for processing and storage. The server validates and saves the information, often updating the database or a service like Firebase. After submission, the user may be redirected to their dashboard. Any dynamic updates, such as loading barangay options, are fetched from the server or external APIs to populate the form fields.



Welcome back, First!
Here's what's happening in Santa Cruz today.
Upcoming: Town Fiesta (May 15)
Weather Alert: Heavy rains expected

Active Reports
0
Reports currently being processed

Recent Activity
Report SC-2025-9544
Status updated to "resolved"
12 hours ago
Report SC-2025-9544 submitted
asdasdsa
12 hours ago

Current Weather
Santa Cruz, Laguna
30°C
Light Rain Shower
Feels like 38°C
20.9 km/h
66%
11:24 AM

Your Recent Reports

REPORT ID	CATEGORY	DATE	STATUS	ACTIONS
SC-2025-9544	Garbage Collection	Dec 3, 2025	Resolved	View
SC-2025-0783	Parks & Facilities	Dec 3, 2025	Resolved	View

Figure 7. Citizen Dashboard Interface

The citizen dashboard serves as the central hub for users on Santa Cruz Connect. It provides residents with access to essential features such as viewing their reports, participating in community discussions, checking upcoming events, chatting with the LGU, and managing their account—including changing their password. The interface is designed to be intuitive and responsive, ensuring that users can navigate seamlessly between sections.

User Interaction

From the dashboard, users can:

- Access their **chat conversations** with LGU staff and mark reports as resolved.
- View their **submitted reports** and check their status.
- Browse **community events** filtered by type (upcoming, past, or today).
- Participate in the **community feed** by posting messages, photos, or comments.
- **Change their password** through the account settings, provided the old password is validated.
- Interact with modals for confirming actions, viewing posts, submitting testimonials, archiving posts, or deleting comments.

Each feature is reachable through clearly labeled tabs or buttons, providing a consistent experience across the platform.

Data Flow Description

1. **Initial Load:**

- When the user opens the dashboard, the browser sends a request to the server for HTML, CSS, JavaScript, and images.
- The server responds with the necessary files, which the browser renders into the dashboard interface.

2. **Dynamic Content Requests:**

- Sections like **My Reports**, **Community Feed**, and **Community Events** fetch data asynchronously via API calls to the server or database (e.g., Firebase or MySQL).
- Chat messages, comments, and post engagement are loaded dynamically without refreshing the page.

3. **User Actions and Interaction:**

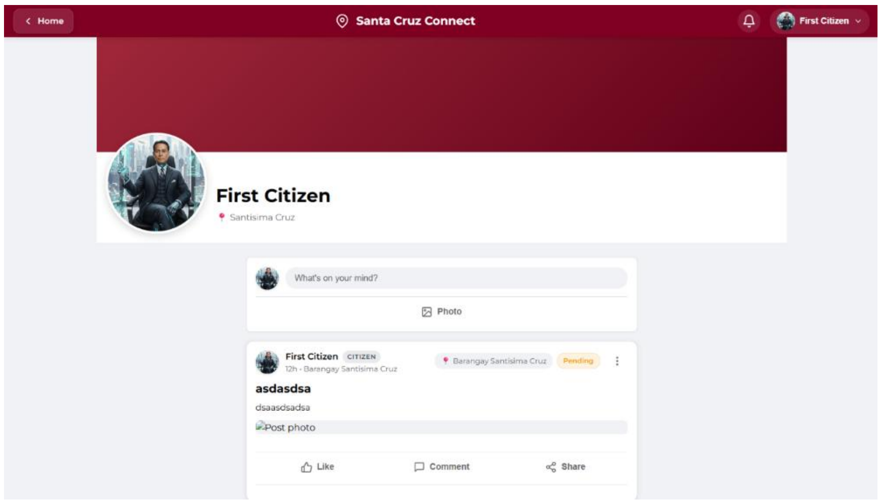
- Clicking **Send Message** in a chat triggers a request to store the message on the server and update the conversation in real time.
- Marking a report as resolved opens a modal for confirmation; once confirmed, the server updates the report's status and prevents further chat.
- Creating a post, adding photos, or submitting a testimonial triggers file uploads and database updates, which are reflected immediately in the community feed.
- Changing the password validates the current password before sending the update to the server. If the password is valid, the server saves the new password securely, and the user receives a confirmation.

4. **Modal Interactions:**

- All modals (e.g., resolution confirmation, delete comment, post viewer) act as overlays that interact with the server only when necessary.
- Closing a modal without confirming an action does not trigger data changes, ensuring safe user interactions.

5. **Security Considerations:**

- Password changes and chat messages are processed securely to prevent unauthorized access.
- All sensitive actions are confirmed via modals to prevent accidental updates



asdasdsa

dsaasdsadsa

Post photo

Figure 8. Profile Page Interface

The profile page allows users to view and manage their personal information, profile picture, cover photo, and posts on Santa Cruz Connect. It also provides functionalities like editing the profile, posting updates, managing photos, messaging other users, and securely logging out. The interface emphasizes clarity, personalization, and easy access to all user-related actions, ensuring a seamless experience for citizens, support staff, municipality personnel, and administrators.

User Interaction

From the profile page, users can:

- **View their own profile** or check other users' profiles depending on permissions.

- **Change profile and cover photos** using overlay buttons and drag-and-drop positioning.
- **Edit personal information** by navigating to the “Edit Profile” page.
- **Create new posts** including text, images, and optional location tagging.
- **View posts** in a scrollable feed, with modal previews for images and post details.
- **Comment on posts** and delete comments if they are the author.
- **Send messages** to other users through the messaging button (if viewing someone else’s profile).
- **Navigate back to their dashboard** or other sections of the platform using the header navigation.

Interactive elements such as modals, dropdown menus, and overlay buttons guide users to perform actions intuitively while ensuring no accidental changes are made.

Data Flow Description

1. Initial Load:

- When the profile page loads, the browser requests the HTML, CSS, JavaScript, and images from the server.
- The server delivers the page content, including dynamic user-specific data like profile information and posts.

2. Profile Data Fetching:

- User details such as profile picture, display name, first name, and barangay are retrieved from session data ($_SESSION) or Firebase.

- If viewing another user’s profile, their user ID is used to fetch relevant data.

3. Dynamic Content & Modals:

- Posts, comments, and engagement data are loaded asynchronously through JavaScript modules (citizens.js and profile_page.js).
- Clicking on posts opens **post viewer modals** with image navigation, zoom controls, and comment management.
- Creating a post opens a modal where users can upload photos, enter text, and optionally select a location.

4. Profile Photo & Cover Photo Updates:

- Updating profile or cover photos triggers file upload requests.
- For cover photos, users can drag to position images, with the updated coordinates sent to the server for storage.

5. User Actions & Security:

- Dropdown menus allow editing the profile, accessing the dashboard, or logging out.
- Logging out submits a hidden form with CSRF protection to prevent unauthorized access.
- Deleting comments prompts a confirmation modal to prevent accidental deletions.

6. Real-Time Feedback:

- All dynamic actions such as creating posts, updating profile photos, and commenting provide immediate UI feedback while interacting with the server asynchronously.

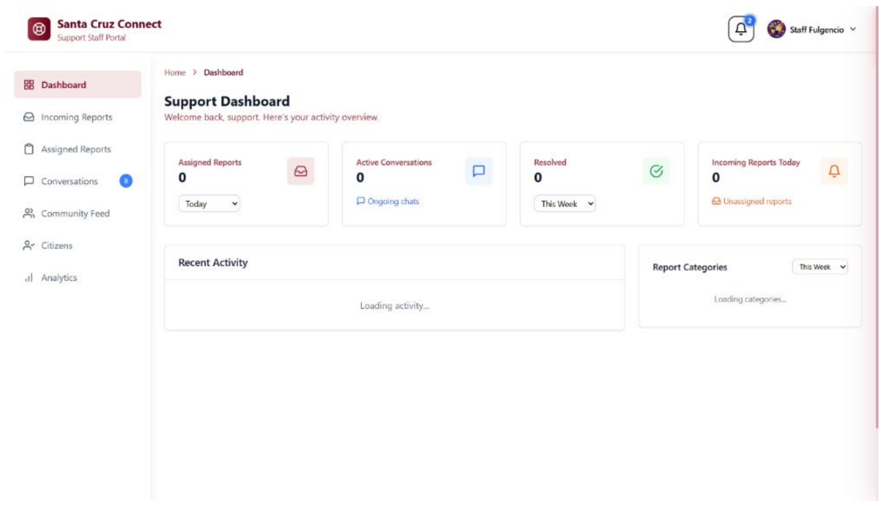


Figure 9. Support / Staff Interface

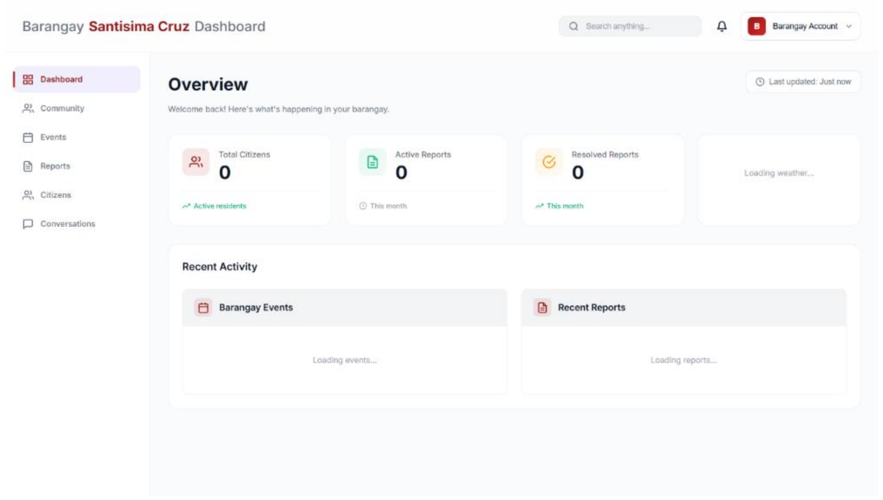
The Support Staff Portal provides staff with a centralized interface to monitor citizen reports, view resolution times, and track community engagement. It presents key information through interactive analytics cards, modals, and dynamic dashboards, allowing staff to efficiently manage reports, testimonials, and community posts. The page is designed for clarity and quick access, ensuring staff can respond to citizen concerns effectively.

User Interaction

Staff interact with the portal by viewing report categories, resolution statistics, and testimonial ratings through visual cards and progress bars. Modals allow staff to request report resolutions, submit or review testimonials, view post details, and create community posts. Navigation buttons and interactive elements guide staff through each workflow, enabling seamless management of reports and communications.

Data Flow Description

When staff perform an action, such as submitting a testimonial or creating a post, the browser sends data to the server or Firebase for processing and storage. Firestore and server-side databases validate and save the information, updating relevant analytics and user interfaces in real time. Dynamic updates, such as loading profile pictures, post content, or report details, are fetched from the server or APIs to ensure the portal displays the latest data to staff.



Welcome back! Here's what's happening in your barangay.

Figure 10. Barangay Interface

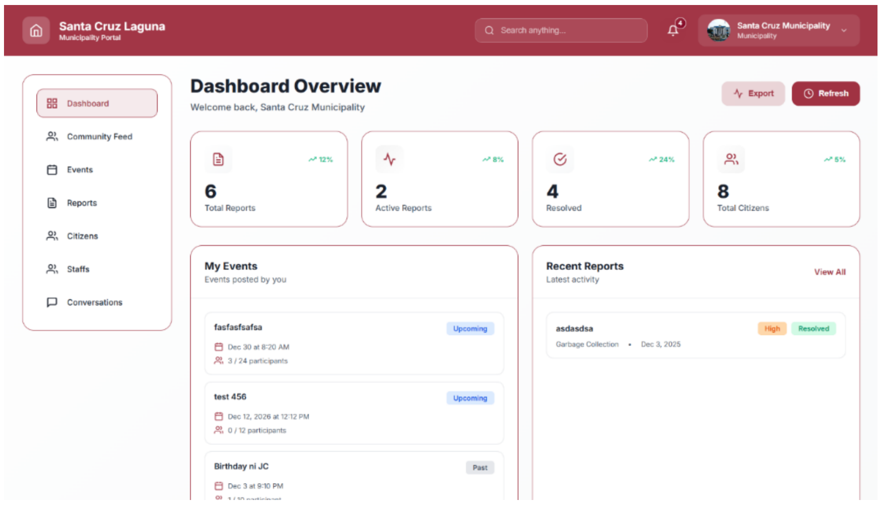
The Barangay Staff Portal provides a centralized interface for barangay personnel to manage citizen reports, events, conversations, and community posts. It displays key information through tables, chat areas, modals, and interactive dashboards, enabling staff to efficiently monitor reports, respond to citizens, manage event participants, and share community updates. The layout is designed for clarity and quick access, ensuring that staff can perform administrative tasks and engage with the community effectively.

User Interaction

Staff interact with the portal by searching and viewing citizens, accessing report details, and monitoring event participation through tables and interactive cards. Modals allow staff to create or edit events, view detailed reports, read and respond to conversations, and manage posts shared on the community feed. Navigation elements, search bars, and action buttons guide staff through workflows, allowing seamless communication and management of community activities.

Data Flow Description

When staff perform actions—such as sending a message, creating a post, or adding an event—the browser sends the relevant data to the server or database for processing. Backend systems validate, store, and update records in real time. Dynamic content, including citizen details, chat messages, event participants, post media, and report statuses, is fetched and rendered dynamically, ensuring that the interface reflects the most up-to-date information.



Dashboard Overview
Welcome back, Santa Cruz Municipality

12%	8%	24%	5%
6	2	4	8
Total Reports	Active Reports	Resolved	Total Citizens

My Events
Events posted by you

fasfasfsafsa — Upcoming
Dec 30 at 8:20 AM
3 / 24 participants

test 456 — Upcoming
Dec 12, 2026 at 12:12 PM
0 / 12 participants

Birthday ni JC — Past
Dec 3 at 9:10 PM

Recent Reports
Latest activity

asdasdsa — High — Resolved
Garbage Collection • Dec 3, 2025

Figure 11. Municipality Dashboard Interface

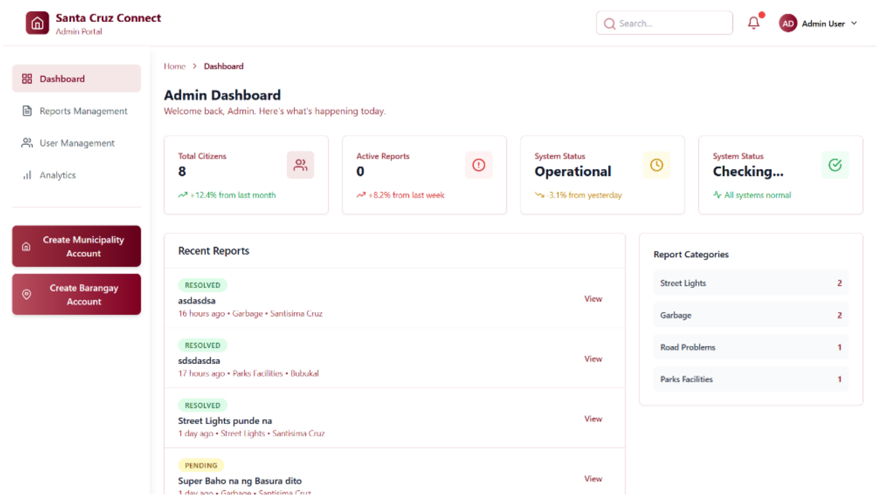
The Municipality Dashboard provides local government staff with a centralized, interactive platform to manage community engagement, events, citizen reports, and internal staff operations. The dashboard consolidates essential information through dynamic dashboards, tables, cards, and modals, allowing staff to efficiently monitor reports, coordinate events, engage with citizens, and manage tasks. Its layout emphasizes clarity, accessibility, and real-time responsiveness, ensuring officials can respond promptly to community needs and maintain effective municipal operations.

User Interaction

Users navigate the dashboard through multiple tabs, including Dashboard, Community Feed, Events, Reports, Citizens, Staffs, and Conversations. Interactive elements such as search bars, filter dropdowns, action buttons, and chat boxes allow staff to post updates, create or manage events, respond to citizen reports, initiate or reply to conversations, and view participant information. Tables and cards summarize key statistics—active reports, resolved reports, total citizens, event participants—while modals provide dedicated spaces for creating posts, adding events, managing reports, and sharing weather updates. Inline controls like image uploads, comment inputs, and send buttons streamline workflows, enabling staff to perform tasks directly within the interface.

Data Flow Description

When a staff member performs an action—such as posting an update, sending a chat message, creating or editing an event, or updating a report—the client-side interface communicates with server-side scripts and databases for validation, processing, and storage. The server responds by updating dashboards, tables, cards, and modals, reflecting changes immediately across the interface. Dynamic content, including citizen details, conversation threads, event lists, report statuses, community posts, and weather updates, is fetched and rendered in real time, providing staff with a comprehensive, up-to-date view of municipal operations and citizen engagement.



Admin Dashboard

Welcome back, Admin. Here's what's happening today.

Total Citizens 8 — +12.4% from last month

Active Reports 0 — +8.2% from last week

System Status Operational — -3.1% from yesterday

System Status Checking... — All systems normal

Recent Reports

RESOLVED
asdasdsa
16 hours ago • Garbage • Santisima Cruz — View

RESOLVED
sdsdasdsa
17 hours ago • Parks Facilities • Bubukal — View

RESOLVED
Street Lights punde na
1 day ago • Street Lights • Santisima Cruz — View

PENDING
Super Baho na ng Basura dito — View

Report Categories

Street Lights	2
Garbage	2
Road Problems	1
Parks Facilities	1

Figure 12. Admin Interface

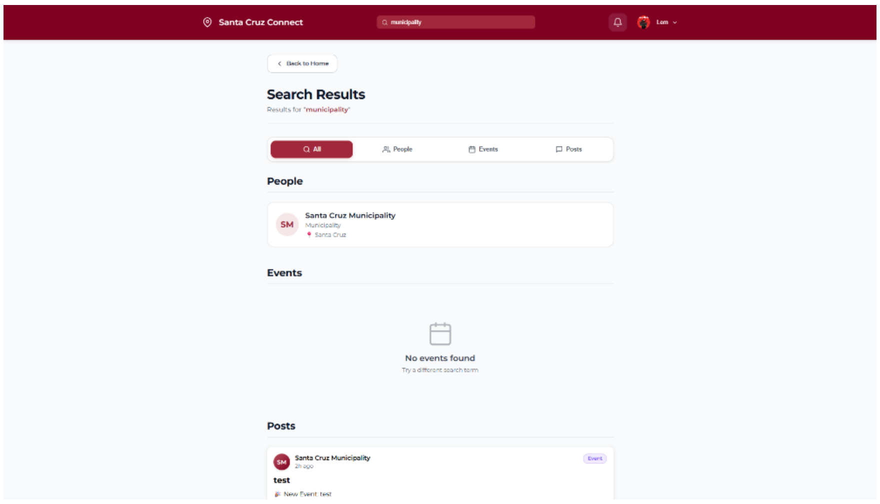
The User Management Interface provides administrators and municipal staff with a centralized platform to manage system users, roles, and accounts for both municipality and barangay levels. It consolidates user data into interactive tables, dynamic cards, and modals, allowing staff to add, edit, or delete users efficiently. The interface emphasizes clarity, accessibility, and security, ensuring user information is handled safely while enabling smooth administrative operations.

User Interaction

Administrators interact with the interface through features such as user tables, role assignment toggles, and modals for creating or deleting accounts. Staff can search for users, filter by role, assign roles such as Administrator or Staff, and view detailed user information via modals. Creating new accounts for the municipality or barangays involves filling out forms with email, password, and role selections, with real-time validation and auto-generated display names for barangay accounts. Action buttons like "Save Changes," "Delete User," and "Create Account" trigger immediate feedback through modals and notifications.

Data Flow Description

When an administrator creates, updates, or deletes a user, the client interface communicates with backend services (e.g., Firebase) to validate inputs, update records, and handle permissions. Role changes, account creation, and deletions are reflected dynamically in user tables and modals. All interactions—such as toggling role checkboxes, submitting forms, or confirming deletions—trigger backend updates, which are immediately rendered on the client-side interface. This ensures that administrators always have an accurate, real-time view of user accounts, roles, and permissions, supporting efficient and secure management of the municipal digital platform.



Search Results

Results for "municipality"

People

SM Santa Cruz Municipality
Municipality
Santa Cruz

Events

No events found
Try a different search term

Posts

SM Santa Cruz Municipality
2h ago
Event
test
New Event: test

Figure 13. Search Results Page Interface

The Search Results Page provides users with a centralized way to find people, events, and posts within Santa Cruz Connect. It features a search bar, tab filters, and dynamically loaded results that adjust based on the user’s query. The interface is designed for clarity across both desktop and mobile layouts.

User Interaction

The Search Results Page provides users with a centralized way to find people, events, and posts within Santa Cruz Connect. It features a search bar, tab filters, and dynamically loaded results that adjust based on the user’s query. The interface is designed for clarity across both desktop and mobile layouts.

Data Flow Description

When the page loads, the browser retrieves the layout, scripts, and styles from the server, then initializes Firebase to obtain user details and search results. The search query triggers dynamic Firestore lookups that update the displayed results without reloading the page. Tab switches, post views, and user actions generate additional data requests handled through JavaScript and Firebase, ensuring real-time and responsive interaction.

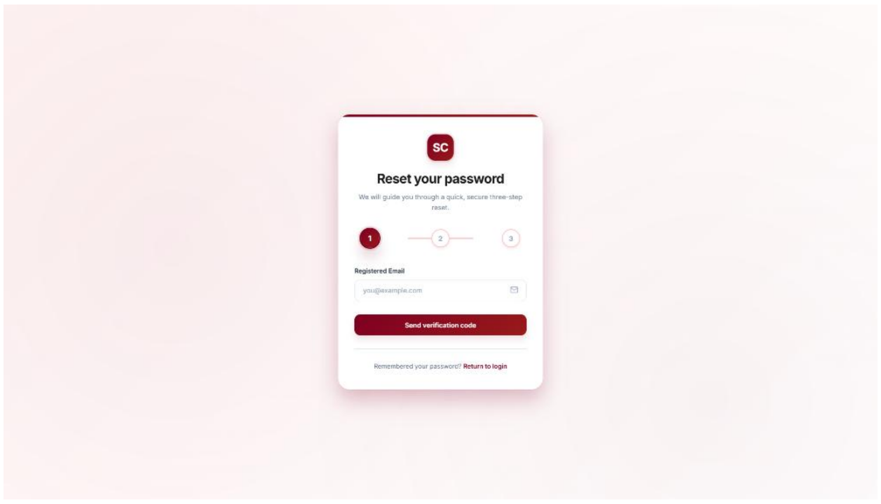

SC
Reset your password
We will guide you through a quick, secure three-step reset.
1
2
3
Registered Email
you@example.com
Send verification code
Remembered your password? Return to login

Figure 14. Forgot Password Interface

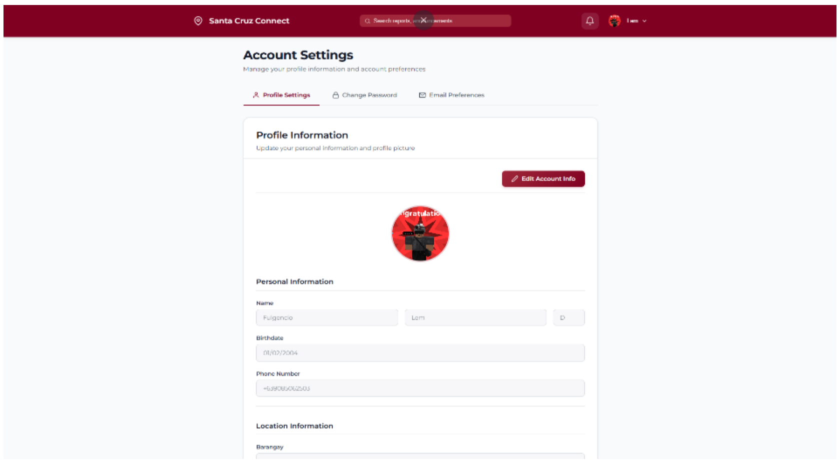
The Forgot Password Page provides a guided three-step process for users who need to securely reset their account credentials. It includes a stepper indicator, email input, verification code entry, and password reset form, all presented within a clean and focused card layout. The design ensures clarity and ease of use during the recovery process.

User Interaction

Users begin by entering their registered email to receive a verification code. They then input the six-digit code within the timed interface before proceeding to set a new password. Password strength indicators, visibility toggles, and match validation assist users throughout the process, ensuring accuracy and security.

Data Flow Description

When the page loads, the browser retrieves the layout along with its supporting CSS and JavaScript files. Upon submission, the system sends the email to the backend to trigger a verification code, which the user must enter to continue. Subsequent steps validate the code, accept the new password, and communicate with the server to securely update the user’s credentials before completing the reset process.



Account Settings

Manage your profile information and account preferences

Profile Information

Update your personal information and profile picture

Personal Information

Name: Fulgencio | Lem | D

Birthdate: 01/02/2004

Phone Number: +639085062503

Location Information

Barangay

Figure 15. Account Setting Interface

The Account Settings Page allows citizens to manage their personal information, change their password, and update email preferences within Santa Cruz Connect. It features organized tabs for Profile Settings, Password, and Email Preferences, displayed in a clean, card-based layout for both desktop and mobile users. A top navigation bar with search, notifications, and user options ensures seamless navigation across the system.

User Interface

Users can edit their profile details, upload a profile picture, change their password through a verification process, and toggle email notification preferences. The page includes confirmation modals for editing, saving, or canceling changes, providing a guided and secure experience. Mobile users can switch between settings tabs using a bottom navigation bar.

Data Flow Description

When the page loads, the server validates the user session and retrieves profile data from PHP sessions or Firebase. JavaScript then updates page elements like profile pictures, barangay lists, and form fields based on the fetched data. User actions—such as saving profile updates, changing passwords, or modifying email preferences—trigger requests to the server and Firebase, which process the changes and return updated information to the interface.

CHAPTER III

IMPLEMENTATION

Technology Stack

Frontend Technologies:

- HTML5 - Structure and semantic markup
- CSS3 - Styling, animations, and responsive design
- JavaScript (ES6+) - Client-side interactivity and dynamic content
- jQuery - DOM manipulation and AJAX requests

Backend Technologies:

- PHP 7.4+ - Server-side scripting and business logic
- Apache 2.4+ - Web server
- Composer - Dependency management

Database & Cloud Services:

- Firebase Firestore - NoSQL cloud database
- Firebase Storage - Image and file storage
- Firebase Authentication - User authentication and authorization
- Firebase Cloud Messaging - Real-time notifications

External APIs & Libraries:

- PSGC API - Philippine barangay validation
- Gemini API - AI chatbot functionality
- Toolkit API - Email verification codes and password reset
- Weather API - Real-time weather data
- PHPMailer 6.x - Email notifications

Development Tools:

- Git - Version control
- Visual Studio Code - Code editor
- Chrome DevTools - Debugging and testing
- Postman - API testing

System Modules

1. Authentication Module

Responsibilities:

- User registration with email verification
- Login/logout functionality via Firebase Authentication
- Password reset using email verification codes (Toolkit API)
- Verification code generation, storage, and validation in Firestore
- Session management and role-based access control

Key Functions: registerUser(), loginUser(), sendVerificationCode(), validateCode(), resetPasswordWithCode(), checkUserRole()

2. Report Management Module

Responsibilities:

- Report submission with image upload
- Report assignment to support staff
- Status tracking and updates
- Report lifecycle management (10 stages)

Key Functions: submitReport(), assignReport(), updateReportStatus(), getReportsByStatus()

3. Communication Module

Responsibilities:

- Real-time chat between citizens and support staff

- Message storage and retrieval
- Conversation threading
- Read/unread status tracking

Key Functions: sendMessage(), fetchMessages(), createConversation(), markAsRead()

4. Community Feed Module

Responsibilities:

- Post creation and display
- Comment and reply functionality
- Reaction system (like, love, etc.)
- Content moderation

Key Functions: createPost(), addComment(), addReaction(), deletePost()

5. Event Management Module

Responsibilities:

- Event creation by Barangay officials
- Event browsing and filtering
- Reservation requests by citizens
- Approval/decline workflow
- Participant tracking

Key Functions: createEvent(), requestReservation(), approveReservation(), updateParticipantCount()

6. AI Chatbot Module (Kuya Pedro)

Responsibilities:

- Process citizen queries

- Provide instant assistance
- Context-aware responses using Gemini API

Key Functions: processChatbotQuery(), sendToGeminiAPI(), formatResponse()

7. Notification Module

Responsibilities:

- Send real-time notifications for all events
- Email notifications via PHPMailer
- In-app notification display
- Notification history

Key Functions: sendNotification(), createEmailNotification(), markNotificationRead()

8. User Management Module (Admin)

Responsibilities:

- View all users
- Change user roles
- Delete user accounts
- Manage testimonials

Key Functions: getAllUsers(), updateUserRole(), deleteUser(), approveTestimonial()

9. Analytics Module

Responsibilities:

- Generate platform statistics
- Report analytics (count by status, category, barangay)
- User activity tracking

- System monitoring

Key Functions: getReportStats(), getUserActivity(), generateAnalytics()

10. Weather Module

Responsibilities:

- Fetch real-time weather data for Santa Cruz
- Display weather information on all dashboards
- Allow Municipality to post weather updates
- Send weather alerts and warnings

Key Functions: fetchWeatherData(), displayWeather(), postWeatherUpdate(), sendWeatherAlert()

Testing and Debugging

1. Unit Testing

- Individual function testing for all modules
- Validated data input/output for each API endpoint
- Tested Firebase CRUD operations
- Verified authentication flows

2. Integration Testing

- Tested interactions between modules
- Validated API integrations (Toolkit, Weather, PSGC, Gemini, Firebase)
- Tested end-to-end user workflows
- Verified real-time data synchronization

3. User Acceptance Testing (UAT)

- Role-based testing with sample users for each user type
- Report submission and tracking workflow

- Event reservation process
- Chat functionality
- Community feed interactions

4. Browser Compatibility Testing

Tested on:

- Google Chrome (latest version)
- Mozilla Firefox (latest version)
- Microsoft Edge (latest version)
- Safari (macOS and iOS)

5. Responsive Design Testing

Tested on various screen sizes:

- Desktop (1920x1080, 1366x768)
- Tablet (iPad, Android tablets)
- Mobile (iPhone, Android phones)

Debugging Tools Used:

- Chrome DevTools - Inspect network requests, console errors, and performance
- Firefox Developer Tools - Debug JavaScript and CSS
- PHP error logging - Server-side error tracking
- Firebase Console - Monitor database operations and authentication
- Postman - Test API endpoints and responses

Sample Test Cases:

Test Case 1: Report Submission

- Input: Citizen submits report with category, description, image, barangay

- Expected Output: Report saved to Firebase, image stored in Firebase Storage, notification sent to support staff
- Result: Passed

Test Case 2: Event Reservation Approval

- Input: Barangay approves pending reservation
- Expected Output: Status changes to approved, participant_count increments, citizen receives notification
- Result: Passed

Test Case 3: AI Chatbot Response

- Input: Citizen asks "How do I submit a report?"
- Expected Output: Gemini API returns step-by-step guidance
- Result: Passed

Challenges and Solutions

Challenge 1: Real-time Data Synchronization

- Problem: Initial implementation had delays in updating dashboards when reports changed status
- Solution: Implemented Firebase real-time listeners instead of polling. Used onSnapshot() method to listen for document changes, ensuring instant updates across all connected clients.

Challenge 2: Secure Password Reset Without Firebase Default UI

- Problem: Firebase provides a default password reset page, but the system required custom email verification for better user experience and control
- Solution: Implemented Toolkit API to generate and send verification codes via email. Codes are stored temporarily in

Firestore with expiration timestamps. Users enter the code in the custom interface, system validates it, and allows password change only if valid. This provides full control over the password reset flow.

Challenge 3: Weather Data Integration and Display

- Problem: Needed real-time weather updates for Santa Cruz that all users could see
- Solution: Integrated free Weather API to fetch current conditions and forecasts. Municipality officials can post weather updates to community feed, and weather widget displays on all dashboards. Severe weather alerts trigger automatic notifications.

Challenge 4: Role-Based Access Control

- Problem: Ensuring users could only access features appropriate to their role
- Solution: Implemented middleware checks on both client and server side. Created a centralized role validation function that checks Firebase Authentication custom claims before rendering any page or processing requests.

Challenge 5: Gemini API Rate Limiting

- Problem: During peak usage, the AI chatbot occasionally hit API rate limits
- Solution: Implemented request queuing and caching for common queries. Added retry logic with exponential backoff for failed requests.

Challenge 6: Event Participant Tracking

- Problem: Race condition when multiple users tried to reserve the same event simultaneously

- Solution: Used Firebase transactions to atomically check and update participant_count, preventing over-booking of events.

Challenge 7: Email Notification Delivery

- Problem: Some SMTP servers rejected automated emails as spam
- Solution: Configured PHPMailer with proper DKIM and SPF records, used reputable SMTP provider, and added email templates with clear branding to avoid spam filters.

Challenge 8: Barangay Data Validation

- Problem: Users entering incorrect or non-existent barangay names
- Solution: Integrated PSGC API to fetch official barangay list. Changed input from text field to dropdown, ensuring only valid barangays can be selected.

Challenge 9: Verification Code Expiration and Security

- Problem: Need to ensure verification codes expire and can't be reused.
- Solution: Store codes in Firestore with timestamp, implement automatic expiration check (codes valid for 10 minutes), delete code after successful password reset, and validate code hasn't been used before.

Challenge 10: Real-time Chat Scalability

- Problem: Chat message retrieval slowed down with large conversation histories

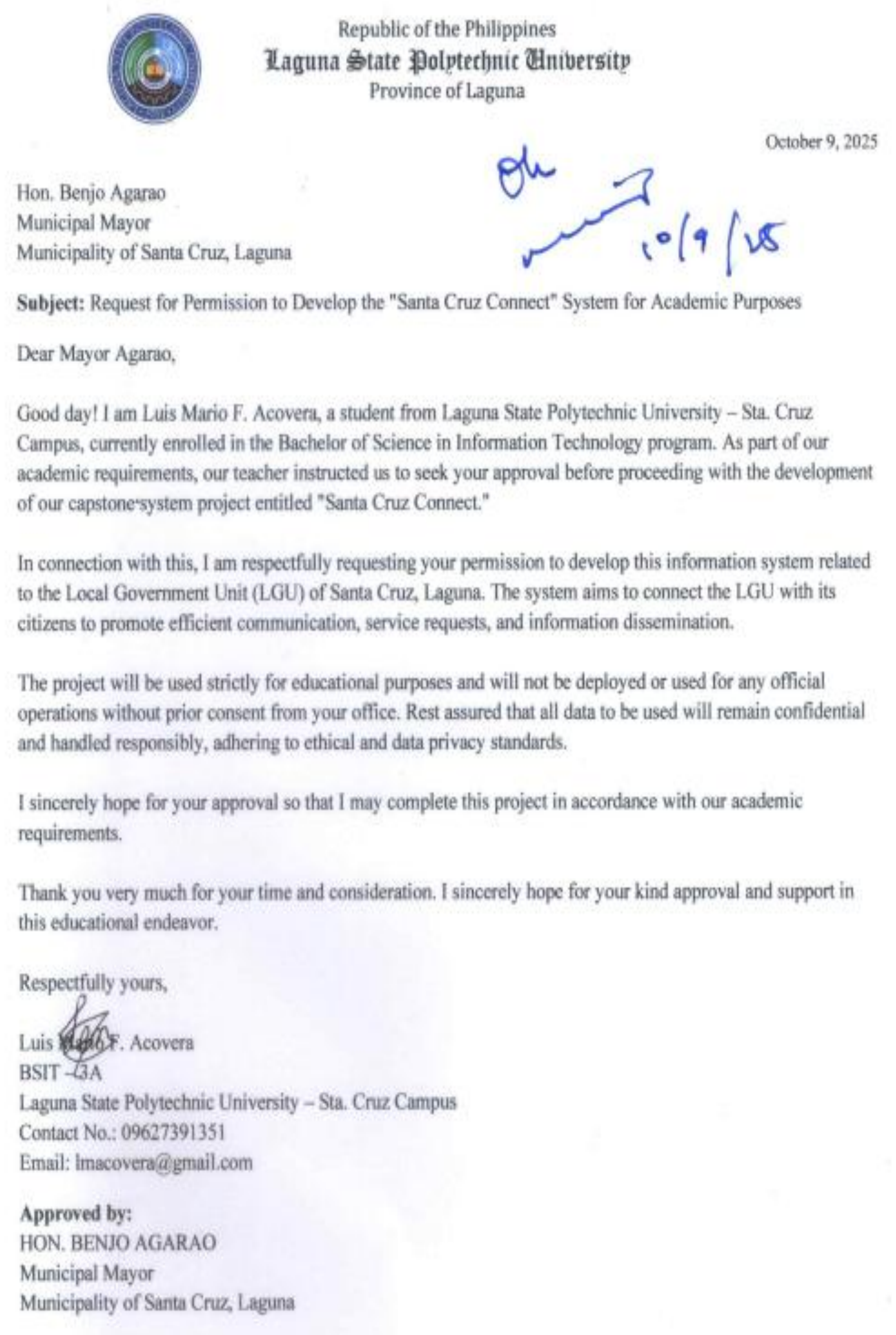
- Solution: Implemented pagination for message loading (load 50 most recent, then lazy load older messages on scroll). Used Firebase query limits and ordering to optimize performance

The Santa Cruz Connect platform successfully addresses the communication gap between citizens and the Local Government Unit through a comprehensive web-based solution. The implementation leverages modern cloud technologies, real-time data synchronization, AI assistance, and weather integration to create an efficient, transparent, and user-friendly system for community engagement.

The modular architecture ensures maintainability and scalability, while the integration of multiple APIs (Firebase, Toolkit, Weather, PSGC, Gemini) provides robust functionality. The custom password reset system using email verification codes offers enhanced security and user control, while real-time weather updates keep the community informed. Through careful testing and iterative problem-solving, the platform delivers on its objectives of faster response times, organized complaint management, improved transparency, and enhanced community participation

APPENDICES

Appendix A
Approved Letter



Republic of the Philippines
Laguna State Polytechnic University
Province of Laguna

October 9, 2025

ok [illegible] 10/9/25

Hon. Benjo Agarao
Municipal Mayor
Municipality of Santa Cruz, Laguna

Subject: Request for Permission to Develop the "Santa Cruz Connect" System for Academic Purposes

Dear Mayor Agarao,

Good day! I am Luis Mario F. Acovera, a student from Laguna State Polytechnic University – Sta. Cruz Campus, currently enrolled in the Bachelor of Science in Information Technology program. As part of our academic requirements, our teacher instructed us to seek your approval before proceeding with the development of our capstone-system project entitled "Santa Cruz Connect."

In connection with this, I am respectfully requesting your permission to develop this information system related to the Local Government Unit (LGU) of Santa Cruz, Laguna. The system aims to connect the LGU with its citizens to promote efficient communication, service requests, and information dissemination.

The project will be used strictly for educational purposes and will not be deployed or used for any official operations without prior consent from your office. Rest assured that all data to be used will remain confidential and handled responsibly, adhering to ethical and data privacy standards.

I sincerely hope for your approval so that I may complete this project in accordance with our academic requirements.

Thank you very much for your time and consideration. I sincerely hope for your kind approval and support in this educational endeavor.

Respectfully yours,

Luis Mario F. Acovera
BSIT – 3A
Laguna State Polytechnic University – Sta. Cruz Campus
Contact No.: 09627391351
Email: lmacovera@gmail.com

Approved by:
HON. BENJO AGARAO
Municipal Mayor
Municipality of Santa Cruz, Laguna

Appendix B
Photo Documentation

Action Photo

Formal Photo