**A PROPOSED OFFERING OF ENHANCING BARANGGAY SERVICES IN NBBS PROPER: THE DEVELOPMENT OF A REQUEST SYSTEM VALENZUELA BRANCH**

Prepared by:

Espinosa Angelica Pore, kim Tingson, Emerald Canada, Jessevier.

AUGUST 2025

# DESIGN DOCUMENT

## INTRODUCTION

This chapter introduces the Barangay Request System, a project developed to modernize how NBBS Proper Navotas manages resident requests and barangay services. Currently, most transactions are handled manually using paper-based records and handwritten logs, which often result in delays, misplaced documents, and difficulties in tracking the status of requests. These manual processes present significant challenges such as inefficiency, lack of transparency, and increased chances of human error. With this system, these challenges will be addressed by transforming manual workflows into a digital platform, improving the speed, accuracy, and accessibility of barangay services for both officials and residents.

This document provides a complete guide to the Barangay Request System, a web-based application created to assist barangay officials with their daily tasks. The system streamlines essential functions such as managing resident records, processing official requests, generating barangay certificates, monitoring request statuses, and producing administrative reports. By shifting from a paper-based system to a computerized one, the barangay can serve residents more efficiently, reduce errors, and ensure proper documentation of all transactions.

The Barangay Request System is developed using PHP as the programming language and MySQL (via XAMPP) as the database server. This ensures that all resident and requested information is stored in a secure and well-organized manner. The system incorporates user authentication, with roles such as Main Admin and Admin, to safeguard sensitive data and restrict access to authorized personnel only. Residents can easily submit their requests, while officials can process, track, and update them through a centralized platform.

This guide explains the different parts of the system, including how it is designed, how information is stored, and how security is implemented. It also covers the system’s architecture, database design, user interface, and communication protocols. The document is intended to serve as a reference for barangay officials, administrators, and technical staff who may need to operate, maintain, or enhance the system in the future.

**SYSTEM ARCHITECTURE**

The Barangay Request System follows a 4-layer client-server architecture. It is built using PHP for backend logic, MySQL for the database, and deployed through XAMPP. The system ensures secure, efficient communication between users, the application, and the data storage.

##### Presentation Layer

* This is the interface that residents and barangay staff interact with. It includes: Login and Registration Pages
* Request Submission Forms Request Tracking Dashboard Administrator Panel
* Its main role is to display information clearly and capture user input effectively.

##### Application Layer

* This layer processes all user interactions. It interprets inputs from the presentation layer and passes them to the business logic. Examples include:
* Handling form submissions Validating login credentials
* Managing navigation between system modules

##### Business Logic Layer

* This is the "brain" of the system. It applies the rules and workflows of the barangay request process, such as:
* Checking request completeness before approval Assigning requests to barangay staff
* Managing priority levels for urgent issues Generating reports for barangay officials
* This layer is responsible for all communication with the MySQL database. It ensures that data is securely stored, retrieved, and updated. Features include:
* Managing backups and recovery processes. Ensuring data integrity and consistency.
* High-Level Components and Their Interactions. User Management (Residents, Staff, Admins). Request Submission & Workflow.
* Approval and Escalation. Service Record Archiving. Reports & Analytics.
* These modules interact seamlessly across the four layers to provide a complete and reliable request management workflow.

##### Deployment Architecture

The system is deployed on a local server using XAMPP at the barangay office. Residents can access it using web browsers over the barangay’s LAN or intranet. All data is stored in the local MySQL database, safeguarded by firewalls and role-based access restrictions.

•Communication Protocols and Interfaces HTTP/HTTPS: Client-to-server browser communication MySQL Protocol: For database operations

•Role-Based Access Control (RBAC): Different access levels for Residents, Staff, and Admins Password Hashing & Secure Sessions: To protect login credentials and active users Parameterized SQL Queries: To defend against SQL injection attac

**DATABASE ENGINE**

The **Barangay Request System** is supported by a **relational database** designed to securely store and manage all data related to residents, service requests, staff actions, and reports. The structure ensures that information flows clearly and logically, from an initial request submission by a resident to its review, approval, and final resolution by barangay officials.

#### Core Entities

**Users**

* This table stores account information for all residents and barangay staff. It includes: Unique User IDs
* Full Name and Contact Information
* Username and Password (hashed for security)
* User Role (Resident, Staff, or Admin) to control access and permissions

#### ServiceRequests

* This is the central log of all service requests submitted to the barangay. It records: Request ID (unique identifier)
* Request Type (e.g., Clearance, Permit, Complaint, Assistance) Resident Details (foreign key linked to Users table)
* Request Description and Date Submitted
* Current Status (Pending, Approved, Completed, or Rejected)

Nullable foreign key linking to RequestActions for tracking progress

#### RequestActions

This table manages the processing workflow of requests. It stores: Action ID (unique identifier)

* Request ID (foreign key linked to ServiceRequest) Assigned Staff ID (foreign key linked to Users)
* Action Taken (Reviewed, Approved, Forwarded, Resolved) Action Date and Remarks

#### Attachments

This table stores files related to a service request, such as scanned documents, IDs, or proofs. It contains:

**Reports**

#### Attachment ID (unique identifier)

#### Request ID (foreign key linked to ServiceRequests) File Path or File Name

#### File Type (PDF, Image, etc.) Upload Date

This table generates summary data for barangay officials, supporting decision-making. It includes:

Report ID (unique identifier)

Report Type (e.g., Monthly Requests, Completed Requests, Pending Requests) Generated Date

Generated By (foreign key linked to Users) Stored report file (optional)

#### Data Integrity & Constraints

Foreign key constraints ensure that every request is linked to a valid resident and every action is linked to both a request and staff member.

Transactions guarantee that when a request is approved or updated, all related actions are consistently recorded.

Role-based access ensures only authorized users can approve or close requests.

Cascade updates and deletes are applied carefully, preventing orphaned records (e.g., attachments cannot exist without a corresponding request).

#### Description of Database Tables, Fields, and Relationships

Users Table

#### Residents Table

ResidentID (Primary Key)

UserID (Foreign Key) Address

Birthdate Gender

HouseholdNumber

Purpose:

Contains detailed resident profiles linked to their system account in the Users Table. Ensures accurate identification for request submissions.

#### ServiceRequests Table

RequestID (Primary Key) ResidentID (Foreign Key)

RequestType (Clearance, Permit, Complaint, Assistance) Description

DateSubmitted

Status (Pending, Approved, Completed, Rejected) Purpose:

Central log of all requests submitted by residents. Each request is linked to the resident who submitted it and can be tracked until completion.

#### RequestActions Table

ActionID (Primary Key) RequestID (Foreign Key)

StaffID (Foreign Key → Users Table)

ActionTaken (Reviewed, Approved, Forwarded, Resolved) Remarks

ActionDate

#### Purpose:

Tracks the step-by-step actions taken on each request, including who handled it and what decision was made.

#### Reports Table

ReportID (Primary Key)

ReportType (Monthly, Weekly, Summary) GeneratedBy (Foreign Key → Users Table)

GeneratedDate

FilePath (optional, if exported)

#### Purpose:

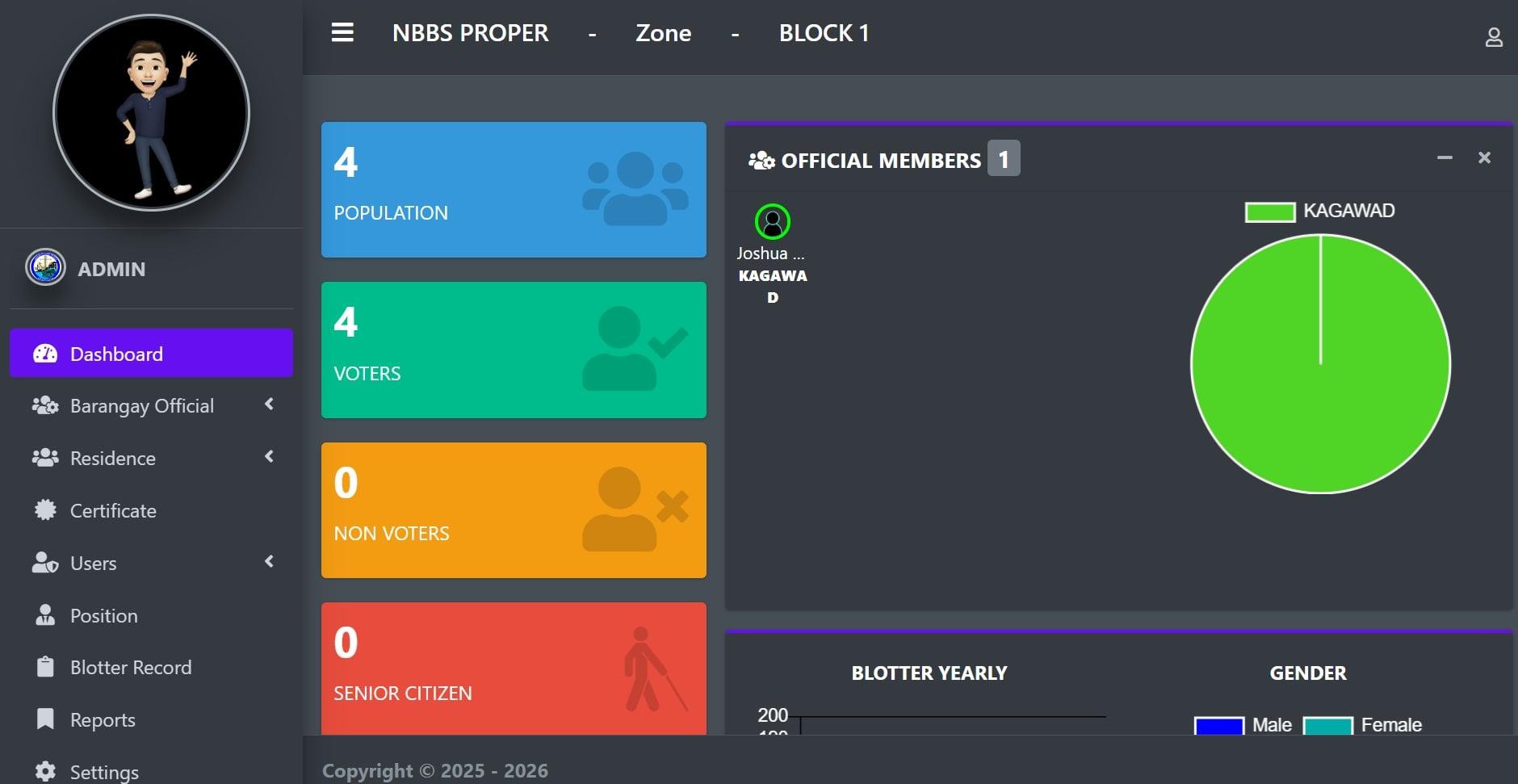
Keeps track of reports generated by admins or staff for monitoring barangay activities and decision making

### USER INTERFACE DESIGN

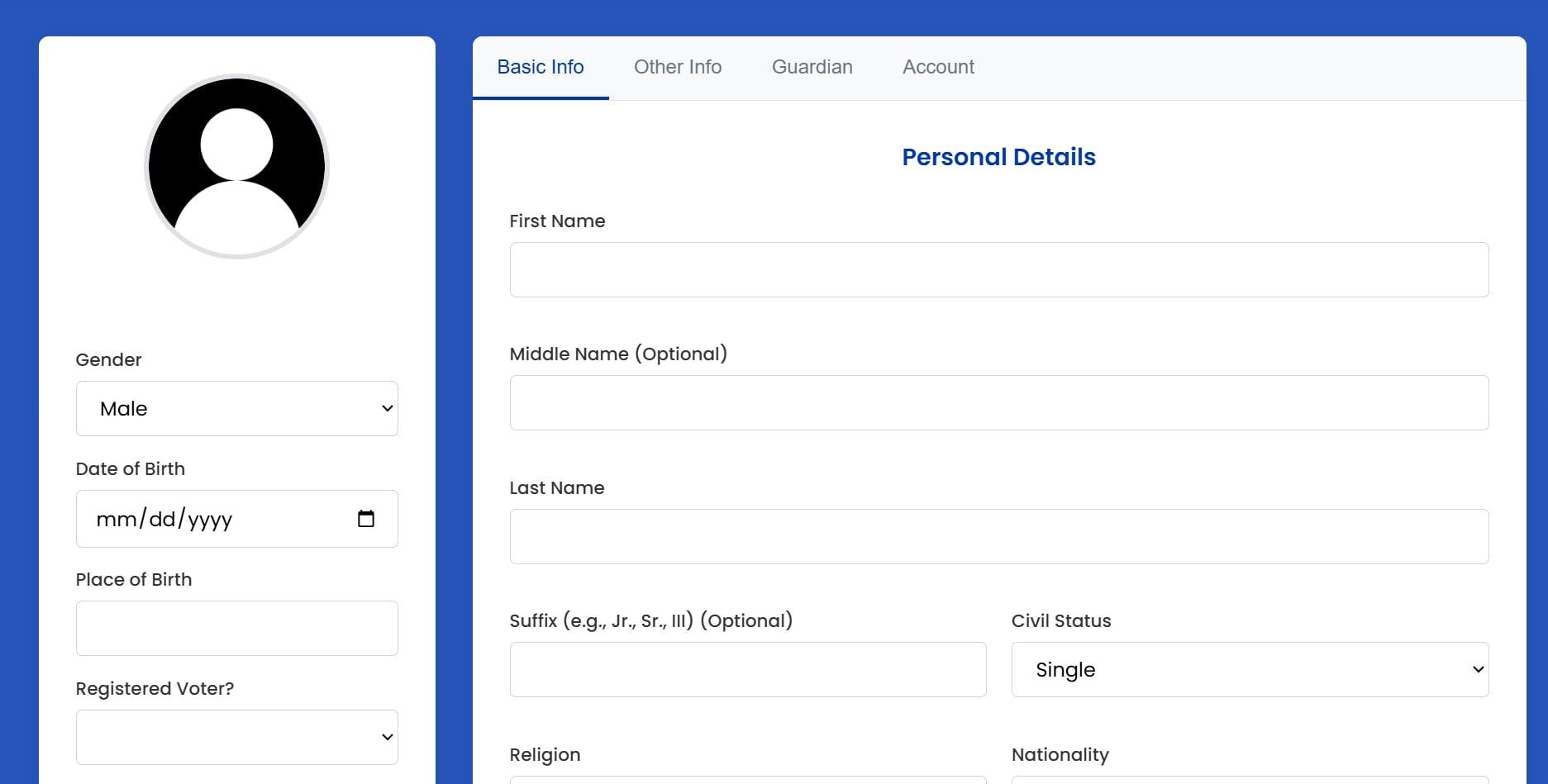
**WIREFRAMES**

****

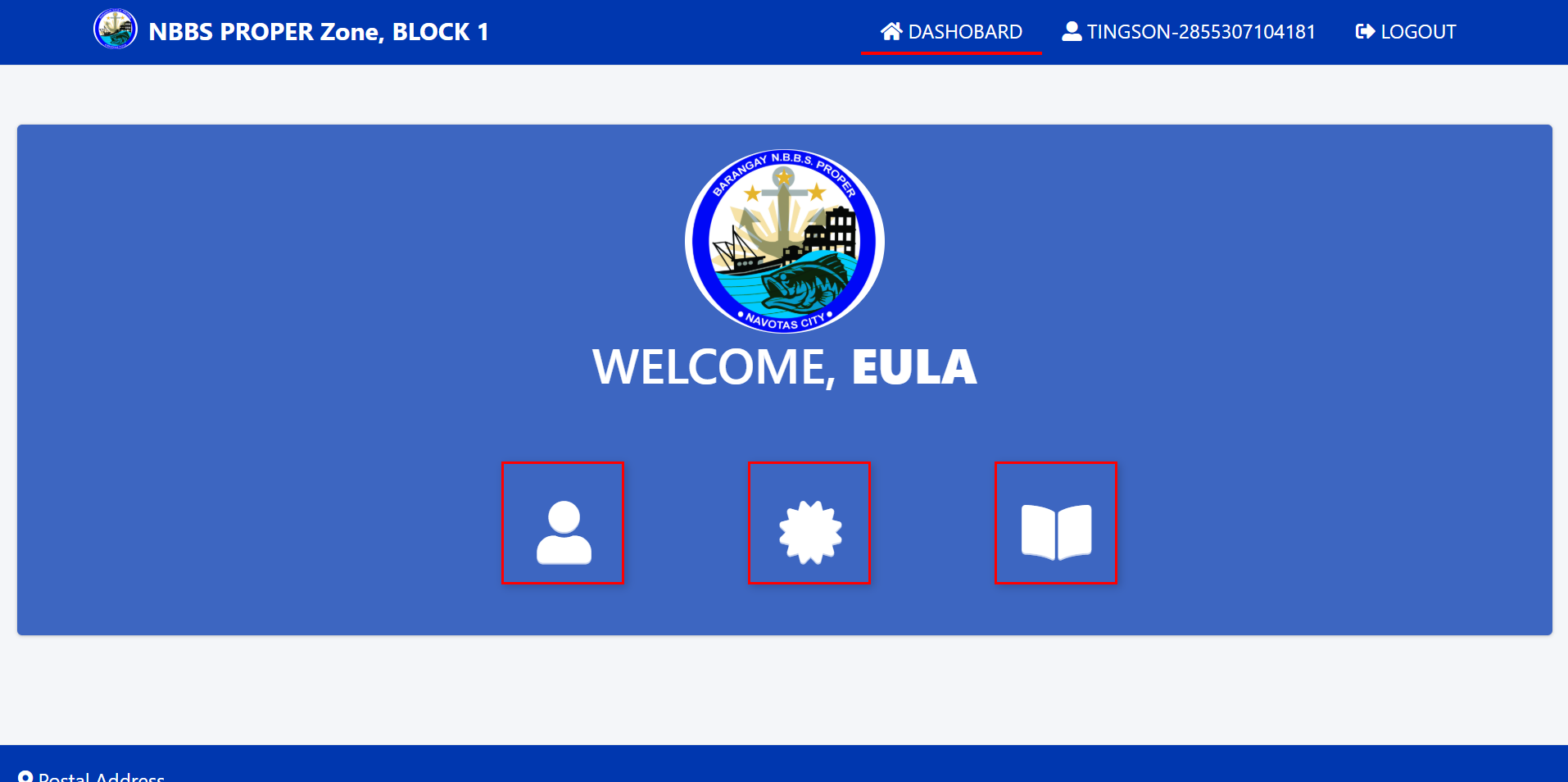
*Image 1 : Landing Page*

**

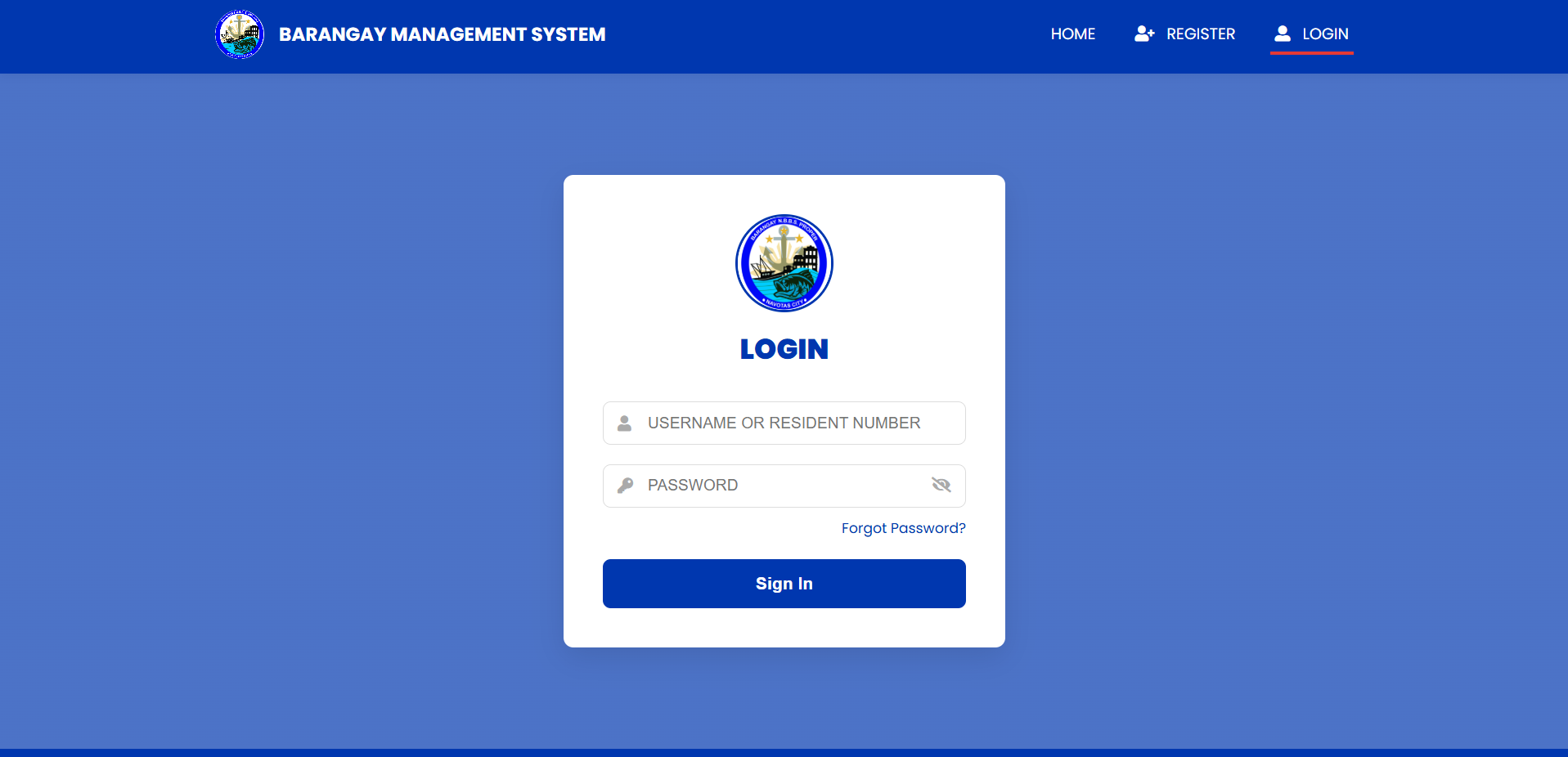
*Image 2 : Dashboard*



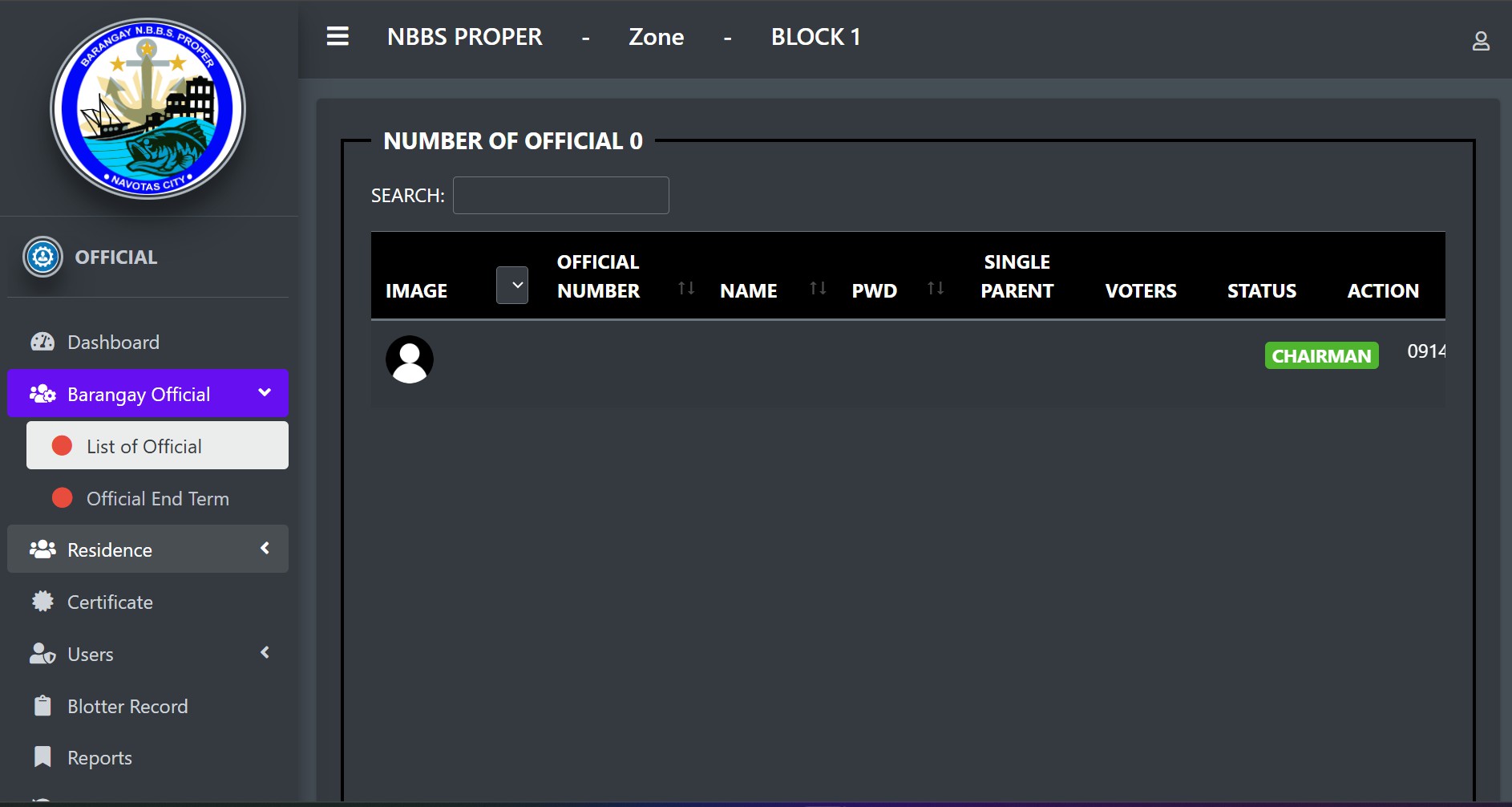
*Image 4 :Register of Residents*

**

*Image 5: Landing page of Residents*



*Image 6: Log in Page*

**

*Image 7: Barangay Officials*

## COMPONENTS DESIGN

##### Key System Components/Modules User Authentication Module

Manages secure login, registration, and logout of users. Functions:

**registerUser()** – Adds new user information to the database. **loginUser()** – Verifies username and password. **logoutUser()** – Destroys session and logs out user.

**recoverAccount()** – Allows account recovery (through security questions or reset form, not email).

##### Request Management Module

Core of the system that allows residents to submit and track service requests.

##### Functions:

**createRequest()** – Save new requests to the database. **viewRequestStatus()** – Display current status of submitted requests. **updateRequest()** – Modify or cancel request (if pending).

**assignRequest()** – Barangay staff assigns the request to appropriate personnel.

##### Admin/Barangay Staff Module

Enables barangay officials to manage requests and user accounts.

Functions:

* approveRequest() – Approve, reject, or mark requests as completed. **generateReport() –** View and export request summaries by type, status, or date. **manageUsers()** – Add, update, or deactivate user accounts.
* **Database Management Module** Description: Stores all system data securely. **Database Tables:**

users (user\_id, name, username, password, role, status)

requests (request\_id, user\_id, type, description, status, date\_submitted) logs (log\_id, user\_id, action, timestamp)

##### Interface Specifications User Authentication

* registerUser(name, username, password, role) boolean (success/failure) loginUser(username, password) session token
* logoutUser() → boolean (success/failure) recoverAccount(user\_id, security\_answer) Boolean
* **Request Management**

createRequest(user\_id, type, description) request\_id

viewRequestStatus(request\_id) status updateRequest(request\_id, new\_description) boolean assignRequest(request\_id, staff\_id) boolean **Admin/Barangay Staff**

approveRequest(request\_id, decision) boolean generateReport(criteria, date\_range) array of requests manageUsers(action, user\_id, details) boolean **Database Management**

Provides CRUD operations (Create, Read, Update, Delete) for all modules through PHP and MySQL queries.

##### Dependency Management and Interaction

User Authentication Module

* Must run first; all users must log in before using other modules.
* ​
* Request Management Module
* Depends on Authentication (to identify user) and Database (to store/retrieve requests).
* ​
* Admin/Barangay Staff Module
* Depends on Request Management (to process requests) and Database (to update records).
* ​

Database Module

* The backbone of the system, every module interacts with it for data storage and retrieval.

##### Flow of System

* Resident logs in via Authentication Module.
* Resident submits a request
* Request Management stores it in MySQL.
* Barangay staff reviews the request
* Admin Module updates status.
* Status change is visible to residents on their dashboard (no email/SMS needed).

#### 

#### DATA FLOW DIAGRAM

#### Open photo

#### *FIGURE 1 “DATA FLOW”*

*The Barangay Request System is a program that helps the barangay run faster and easier. The Admin is like the main manager who controls the system and keeps records of officials, residents, and documents. The Secretary is the helper who organizes files, reports, and logs. The Residents are the people in the community who can update their information, request certificates, or file complaints. This system makes sure everyone can do their tasks in a simple and organized way.*

##### SECURITY DESIGN

The security of the Barangay Request System is a top priority to protect sensitive resident information, request records, and official barangay documents. The system is designed to ensure the confidentiality, integrity, and availability of data, while preventing unauthorized access or modifications.

Key considerations include safeguarding administrator and secretary credentials, enforcing strict access. control based on user roles, protecting request data stored in the database, and ensuring that all activities are properly logged and auditable. The security policies are structured to meet the standards required for handling local government records.

##### Authentication and Authorization Mechanisms

This section explains the measures applied to control access within the system. Authentication ensures that users are who they claim to be through valid usernames and passwords. Once authenticated, authorization defines what actions a user can perform based on their assigned role. These layered protocols are critical to protecting sensitive request data and maintaining the integrity of barangay records.

##### Role-Based Access Control (RBAC)

The Barangay Request System implements RBAC to strictly control access according to user roles:

Administrator: Has full access to all system features, including managing resident accounts, approving or rejecting requests, updating records, and generating reports.

Secretary: Can create, update, and process resident requests, but cannot modify system configurations or user accounts.

Resident/User: Can submit requests, track their status, and view approved documents, but cannot access administrative functions.

This structure ensures that users only have access to the features and data relevant to their responsibilities

##### Session Management

The system manages user sessions using secure session variables that store authentication details, including the user’s role. Sessions are initialized upon successful login and destroyed upon logout, ensuring no residual data is carried over. This prevents unauthorized reuse of credentials and strengthens the protection of sensitive data.

##### Data Storage Protection

All resident and request data is stored securely within a MySQL database hosted on XAMPP. Access to the database is protected by server-level security mechanisms and parameterized queries, which defend against SQL injection attacks and ensure data integrity.

##### Secure Database Connections

The system operates within a trusted local network environment, and all connections to the MySQL database are secured. By applying authentication and database access privileges, the system prevents unauthorized database access and avoids storing plain credentials in client-side configuration files.

##### Audit Trails and Logging

The Barangay Request System incorporates automatic logging mechanisms to provide a complete audit trail of all important actions taken within the system. These audit features ensure accountability, transparency, and traceability of all user activities.

RequestCreationTimestamp in the Requests table automatically records the exact date and time when a resident submits a new request.

LastModifiedTimestamp in the Requests table is updated whenever an administrator or secretary makes changes to a request or record.

CreatedBy and UpdatedBy fields track which user (resident, secretary, or administrator) created or modified a These audit trails prevent tampering, provide a verifiable record of transactions, and strengthen the system’s reliability in handling barangay requests and services specific record.

### PERFORMANCE DESIGN

The request system for Barangay NBBS Proper must be fast, reliable, and easy to use. Since many residents and staff will use the system, it should work smoothly without delays. The goal is to make sure the system responds quickly, can handle more users in the future, and stays stable all the time. The system should load in just a few seconds, work even when many people are using it, and not waste computer resources.If many people are using the system at the same time, load balancing can be added to share the work across different servers. Tasks like sending emails or making reports will be done in the background so they do not slow down the main system.

##### Strategies for Optimizing System Performance

To make sure the request of the residents works fast and reliably, different strategies will be used. One important method is caching, which saves common data so it can load quickly without always checking the database. Another method is database optimization, which makes searching and saving information faster. If many people use the method at the same time, load balancing can be applied to share the work across servers. The method will also use asynchronous processing to handle background tasks like sending emails or reports, so the main method will not slow down. Clean coding practices will be followed to make the method run smoothly and use fewer resources. Lastly, monitoring tools will be used to check performance and fix problems quickly.

##### Performance Testing Plan

The requesting system is used by residents and barangay staff, it must go through performance testing. This testing will check if the system is fast, stable, and able to handle many users at the same time. One test is load testing, which will simulate many people using the system to see if it still works smoothly. Another is stress testing, which will push the system beyond normal use to find its limits and see how it recovers.The system will also have scalability testing to check if it can still perform well when the number of users grows. End-to-end testing will make sure that all parts of the system, such as the website, server, and database, work well together. Even after the system is launched, it will continue to be monitored to quickly find and solve any performance issues, Through these tests, the request system can be trusted to provide a smooth and reliable service for the community.

# ERROR HANDLING MECHANISMS AND LOGGING

This section describes the mechanisms implemented to handle system errors and maintain reliable logs. Error handling ensures that unexpected issues such as invalid user input, failed file uploads, or database connection errors are managed gracefully, preventing application crashes and maintaining smooth operation. Logging mechanisms are also in place to capture important system events for auditing and troubleshooting.

##### Error Handling Mechanisms and Strategies

Effective error handling is essential for maintaining the stability and reliability of the Barangay Request System. The application implements structured error management techniques, particularly around database interactions and file transactions.

Try-Catch Blocks: All PHP database operations (using MySQL) ) are wrapped in try-catch blocks. This ensures that errors such as connection timeouts, invalid queries, or constraint violations are caught gracefully without crashing the system. Clear error messages are shown to the user while detailed logs are stored for administrators.

User Input Validation: The system validates all inputs at both the frontend and backend. Ensures that required fields are not left blank.

Checks for duplicate entries to prevent redundancy.

Validates data formats such as email addresses, contact numbers, and dates. This reduces the risk of corrupted or invalid data being saved in the database.

User-Friendly Error Messages: When an error occurs, the system provides clear and non-technical messages to residents or barangay staff. For example, instead of displaying raw SQL errors, the system shows: “Request submission failed. Please try again later.” Technical details are logged internally for debugging by system administrators.

Finally Blocks for Cleanup: Database connections and file streams are always properly closed after execution, even when an error occurs. This prevents memory leaks and ensures consistent system performance.

Transaction Rollbacks: For critical operations such as creating or updating multiple records (e.g., resident request entries linked with logs), the system uses transactions. If one step fails, the entire operation is rolled back, ensuring that the database is never left in an inconsistent state.

##### Logging Mechanisms

* Logging provides administrators with a trace of system events and errors for auditing and troubleshooting. All failed login attempts are recorded with timestamps and IP addresses.
* Request creation, modification, and approval activities are logged under the user’s account. Error logs are stored in a dedicated system\_logs table for monitoring by administrators.
* These mechanisms ensure accountability, prevent data corruption, and maintain the overall reliability of the Barangay Request System.

##### Logging Requirements and Specifications

The Barangay Request System integrates a built-in logging mechanism to ensure accountability, traceability, and proper auditing of resident requests and system activities. All critical operations are automatically recorded in the database, making it easier for administrators to track system usage and detect issues.

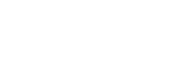
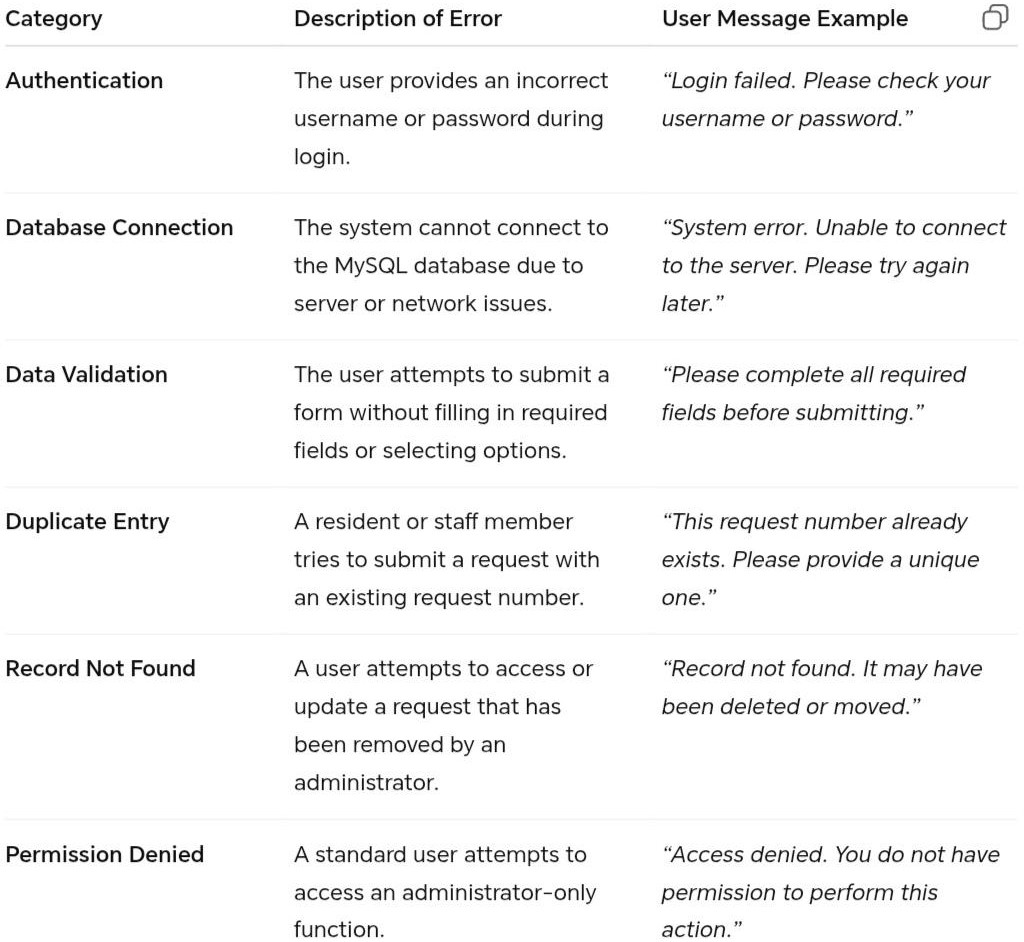
What is Logged: All resident request submissions, updates, and approvals are automatically stored in the database.

User Actions: Each request log records which user (resident or staff) performed the action, ensuring accountability.

Request Status Trail: Changes in request status (e.g., Pending, In Progress, Completed) are logged to provide a clear history of actions taken.

##### ERROR CODES AND ENABLING

The system manages errors by showing clear and straightforward messages to users . This method allows users to quickly understand the issue and know the next step to take, without having to deal with complicated error codes.



# THIRD PARTY INTEGRATIONS

The system is designed as an online request management platform for Barangay NBBS Proper, developed using PHP with XAMPP as the local development server and MySQL as the database. While deployed online, the system remains simple and excludes third-party messaging or notification services such as email or SMS gateways.

##### MySQL Database (via XAMPP)

Purpose: Stores user accounts, barangay requests, and administrative records.

Integration Point: The PHP backend connects to MySQL using mysqli or PDO for database transactions. Data Exchange Format: SQL queries (INSERT, SELECT, UPDATE, DELETE).

##### Web Server (Apache in XAMPP)

Purpose: Handles HTTP requests and serves the system’s web pages to users online.

Integration Point: The PHP scripts are executed by the Apache server, delivering dynamic content to the client browser.

##### Local File System / Export Module

Purpose: Allows administrators to generate and store barangay reports and data backups. Integration Point: PHP’s file handling functions manage report generation and export.

##### PHP Core Libraries

Purpose: Provides functionalities for form validation, session management, authentication, and password encryption.

Integration Point: Integrated directly into the backend logic of the system.

Data Exchange Format: Internal PHP data structures (arrays, JSON for structured data handling).

The system does not integrate with external notification services such as email or SMS APIs. This was a deliberate design choice to keep the system lightweight, cost-effective, and functional even without reliance on paid or internet-dependent third-party providers. Instead, the focus is on real-time online access through the web interface, where users and administrators can directly view, submit, and manage request.

### DEPLOYMENT PLAN

This section describes the deployment strategy for implementing the system into the target environment. It includes server setup, database installation, application deployment, and post-deployment verification to ensure smooth operation.

#### Overview of the Deployment Process

* Server Preparation (XAMPP & PHP Environment)
* Install and configure XAMPP on a dedicated, secure server machine.
* Enable Apache and MySQL services and configure strict security settings.
* Set strong passwords for MySQL root access and ensure the server is restricted to authorized personnel only.

#### Database Deployment

* Import the SQL database file into phpMyAdmin to create the system database and its tables
* Create the initial “Admin” user account to allow immediate system access after deployment. Set up automated MySQL backup schedules to prevent data loss.

#### Application Deployment (Client-Side)

* Upload the PHP system files into the XAMPP htdocs directory or configured server directory. Configure the database connection settings in the system’s configuration file.
* Distribute the system access link (local IP/domain) to authorized users and administrators.

Conduct User Acceptance Testing (UAT) on several client devices to ensure system stability.

Verify connectivity to the server and confirm that all modules (login, request submission, record management) function as expected.

Perform security testing to validate access restrictions and ensure that only authorized users can access admin features.

#### Hardware and Software Requirements for Deployment Client Systems (Officer/Admin Workstations)

* OS: Windows 10 or later
* Browser: Google Chrome / Mozilla Firefox (latest version) RAM: Minimum 4GB
* CPU: Dual-core or higher

#### Server System

* OS: Windows 10 / Windows Server 2016 or later Server Stack: XAMPP (Apache, PHP, MySQL) Database: MySQL (latest version)
* Storage: SSD recommended, with a redundant backup solution

# MAINTENANCE AND SUPPORT

This section defines the processes for the ongoing maintenance and support of the Barangay Request System. It outlines scheduled activities such as database backups, software updates, and bug fixes to ensure the system remains reliable, secure, and efficient over its operational lifetime.

##### Guidelines for System Maintenance and Support

**Database Backups:** Schedule regular, automated backups of the BarangayRequest database using XAMPP’s MySQL tools. Full backups should be performed daily, with incremental backups more frequently. Backup integrity must be verified on a regular basis.

**Database Health Checks:** Perform routine maintenance tasks such as rebuilding indexes and optimizing queries weekly or monthly to ensure fast and stable system performance.

**System Monitoring:** Monitor PHP error logs, Apache server logs, and MySQL logs for unusual activity, recurring errors, or performance bottlenecks.

**User Training:** Provide training sessions for barangay staff and administrators on how to properly use the system. A user manual or quick-reference guide should be provided to minimize errors and improve efficiency.

##### Procedures for Handling Software Updates, Patches, and Bug Fixes

**Staging Environment**: Before deploying updates, all changes must first be tested in a separate staging environment that mirrors the production setup. This prevents errors from directly affecting the live system.

**Estimation Testing:** After bug fixes or feature enhancements, run regression tests to ensure that new changes do not negatively affect existing functions. Apply urgent patches immediately when vulnerabilities are discovered.

**Version Control**: All updates will follow a clear versioning format (e.g., v1.0, v1.1, v2.0). Release notes must be provided, documenting bug fixes, updates, and new features.

**Security Patches**: In case of security vulnerabilities, emergency patches must be developed, tested, and deployed quickly to protect the system and its data.

##### ESCALATION PROCESS FOR RESOLVING ISSUES

Tier 1 Support (Admin User): The designated Barangay System Administrator will be the first point of contact. They will handle basic issues such as account resets, permission errors, and common troubleshooting.

Tier 2 Support (Technical Staff): If the issue cannot be resolved by the Admin, it will be escalated to technical staff who will handle system-level problems such as database errors, server downtime, and software malfunctions.

Tier 3 Support (Development Team): The development team will address more complex issues requiring code modifications, database schema changes, or the implementation of new features. Critical cases, such as data corruption or system-wide failures, will be immediately escalated to this level

**REVISION HISTORY**

|  |  |
| --- | --- |
| **Date** | **Description of Changes** |
| Aug 8,2025 | Bullets of Some parts of the documents |
| Aug 8,2025 | Title of The System |
| Aug 8,2025 | Fonts of the text |
| Aug 8 2025 | Many words is unfamiliar |
| Aug 8,2025 | Justify of the text |

**APPENDIX**

Department of the Interior and Local Government (DILG). Barangay Information System Guidelines.

Laudon, K. C., & Laudon, J. P. (2020). Management Information Systems: Managing the Digital Firm. Pearson.

Pressman, R. S., & Maxim, B. R. (2019). Software Engineering: A Practitioner’s Approach. McGraw-Hill.

Sommerville, I. (2016). Software Engineering. 10th Edition. Pearson. Official website of the City Government of Navotas.

Related undergraduate theses and research on ICT-enabled barangay request systems.