Software Requirements Specification

for

Information System with IoT-Enabled Incident Reporting and E-Services for Barangay Gumaoc East

Version 0.1

Prepared by

Group Name: DVBD

Job Matthew M. Bernardo	**2022-131840**	**jobmatthewbernardo@gmail.com**
Ghin Ivan F. Dacanay	**2022-130060**	**ghindacanay84@gmail.com**
Mar Yvan S. Dela Cruz	**2022-330055**	**biofrostyv@gmail.com**
Archie M. Verania	**2022-330066**	**archieverania1130@gmail.com**

Instructor: ***MIKEE V. GONZAGA***

Course: **Bachelor's of Science in Information Technology Major in Mobile and Internet**

Date: **September 17, 2025**

CONTENTS ..... **II**

Version	Primary Author(s)	Description of Version	Date Completed
Draft 1.0	Ghin Ivan F. Dacanay	Initial draft made by reusing and improving the content from last term's documentation.	09/11/25
Draft 2.0	Ghin Ivan F. Dacanay	Chapter 2 revised by adding the Product Overview with system context diagram, Product Functionality, and system details.	09/14/25
Draft 3.0	Ghin Ivan F. Dacanay, Job Matthew M. Bernardo	Modified Chapter 2-3, realigning it to the past term's documentation.	09/15/25
Draft 4.0	Archie M. Verania, Mar Yvan S. Dela Cruz	Improving Chapter 4, aligning it to chapters 1-3.	09/16/25
Draft 5.0	Ghin Ivan F. Dacanay, Job Matthew Bernardo	Improved and modified appendices and 3D Kiosk Interface example	09/17/25
Draft 6.0	Ghin Ivan F. Dacanay	Revised Chapter 1.5 accordingly to the document.	09/28/25

1 Introduction

This document presents the Software Requirements Specification (SRS) for the Barangay Information System with IoT-Enabled Incident Reporting and E-Services for Barangay Gumaoc East. The project was developed to replace the current manual, paper-based processes with a digital platform that makes services more organized, efficient, and accessible. By combining a web-based system with IoT-enabled kiosks, the system aims to improve how residents request documents, report incidents, and interact with the barangay office, while also helping barangay staff and officials manage and monitor services more effectively.

In this chapter, the reader will find a brief explanation of the purpose of the document, the scope of the product, the intended audience, and the conventions and references used. This serves as the foundation for the rest of the SRS, giving the reader the necessary background before moving into the detailed descriptions in later chapters.

1.1 Document Purpose

The purpose of this document is to describe the software requirements for the Information System with IoT-Enabled Incident Reporting and E-Services for Barangay Gumaoc East, Revision 1.0. The system is designed to replace the manual, paper-based processes in the barangay with a digital platform. It will include resident profiling, document requests, incident reporting, and a queuing system through IoT-enabled kiosks. A web-based platform will serve as the main access point for the residents, barangay staff, and officials.

The scope of this SRS covers the web system and IoT kiosks only. It does not include future features such as mobile applications, financial modules, or integration with other LGU systems. This means the system is focused only on Barangay Gumaoc East and not across multiple barangays or higher-level government platforms. This version of the document serves as a guide for the development team and stakeholders to make sure the system addresses the needs of Barangay Gumaoc East.

1.2 Product Scope

The Information System with IoT-Enabled Incident Reporting and E-Services for Barangay Gumaoc East is a web-based platform that will bring important barangay services online. It will have IoT-enabled kiosks for walk-in residents with RFID access, internet connection, real-time data updates, and queue number printing. For online users, the same services can be accessed through the web platform, where they can request documents and report incidents without using RFID or printed queue numbers.

The purpose of this system is to replace the manual, paper-based processes with a digital platform that is more accurate, accessible, and convenient. Its benefits include reduced paperwork, shorter waiting times, fewer errors, and improved use of real-time data to help staff manage services more efficiently. The main objectives are to improve efficiency, organize processes better, and provide residents with a faster and easier way to access services. The system’s overall goal is to make the services easier and more reliable for residents, while also helping the barangay itself improve and adapt to modern digital tools.

1.3 Intended Audience and Document Overview

This SRS is for the client, which is the Barangay Gumaoc East administration, as well as the professor, project adviser, project consultant, and panelists who will evaluate and guide the project. It is also meant for the project team, like developers and testers, since it explains the system requirements that need to be followed. Future developers, researchers, and students may also use this document as a reference if they want to study or improve the system. For the client, it shows how the system will improve barangay services and what features will be included. For the professor and panelists, it shows that the requirements are clearly written and organized. For the advisers, it serves as a guide to check if the project stays on track with its goals.

The rest of this SRS is divided into sections that describe the system. It starts with the introduction, which explains the purpose and scope of the project. Then it moves to the overall description and specific requirements, which explain the system's features, functions, and limits. The later parts include non-functional requirements, diagrams, and other references that support the design. The suggested way to read this document is to begin with the introduction and product scope, continue with the system features and requirements, and then review the technical sections and appendices for supporting information.

1.4 Definitions, Acronyms and Abbreviations

- **API** – Application Programming Interface; allows different parts of the system to communicate.
- **BIS** – Barangay Information System; the main system being developed in this project.
- **COMET** – Collaborative Object Modeling and Architectural Design Method; the design method required for this project.
- **DB** – Database; where records of residents, requests, and incidents are stored.
- **HTML** – Hypertext Markup Language; used for structuring content on the web.
- **IoT** – Internet of Things; refers to the kiosks with RFID readers and queue printers that connect to the web system.
- **JS** – JavaScript; a programming language used for front-end system functions.
- **LGU** – Local Government Unit; refers to Barangay Gumaoc East in this project.
- **MySQL** – A relational database management system used for storing data.
- **NPC** – National Privacy Commission; government body that enforces the Data Privacy Act in the Philippines.
- **PHP** – Hypertext Preprocessor; the backend programming language used in this project.
- **RFID** – Radio Frequency Identification; technology used in kiosks for resident login and authentication.
- **SRS** – Software Requirements Specification; this document, which describes the system requirements.
- **UML** – Unified Modeling Language; the standard diagramming language used to model the system.

1.5 Document Conventions

This document follows the IEEE SRS format to ensure the requirements are well-organized and easy to read. It uses Arial font in sizes 11 and 12, single spacing, and 1-inch margins. Italics are applied for comments or instructions, while bold text highlights section and subsection titles. The content is justified and properly indented for consistency.

For diagrams, the UML standard is used, including use case, class, sequence, activity, and deployment diagrams. Requirements and constraints are written in a clear, simple way to make them understandable for both technical readers (developers, testers) and non-technical readers (professors, panelists, barangay officials).

Naming conventions are also followed in the document:

- Abbreviations and acronyms are listed in Section 1.4 in alphabetical order.
- System names (e.g., Barangay Information System, IoT-enabled Kiosk) are capitalized to show that they refer to specific components.
- References follow APA 7th edition formatting to ensure proper citation of sources.

1.6 References and Acknowledgments

This SRS refers to the following documents, standards, and web sources:

- IEEE. (1998). *IEEE recommended practice for software requirements specifications (IEEE Std 830-1998).* Institute of Electrical and Electronics Engineers. https://doi.org/10.1109/IEEESTD.1998.88286

- Dacanay, G. I. F., Bernardo, J. M. M., Dela Cruz, M. Y. S., & Verania, A. M. (2025). *Information system with IoT-enabled incident reporting and e-services for Barangay Gumaoc East: Project documentation* [Unpublished manuscript]. National University – Fairview, School of Engineering and Technology.

- Barangay Gumaoc East. (n.d.). *Barangay Gumaoc East official Facebook page*. Facebook. Retrieved September 14, 2025, from https://www.facebook.com/brgygumaoc.east

- City Government of San Jose del Monte. (2023). *E-governance initiatives for barangays.* https://csjdm.gov.ph/

- Department of the Interior and Local Government (DILG). (2020). *Full disclosure policy portal*. https://fdpp.dilg.gov.ph/

Acknowledgments

The project team would like to thank the professor, project adviser, project consultant, and panelists for their guidance and feedback in preparing this document. The team also acknowledges the Barangay Gumaoc East administration for providing valuable insights and information about their current services and processes.

2 Overall Description

2.1 Product Overview

The Barangay Information System with IoT-Enabled Incident Reporting and E-Services is a new system made to replace the old manual and paper-based processes in Barangay Gumaoc East. It is not part of any product family but was created especially to make barangay services faster, more organized, and easier to access. The system works as the main hub that connects residents with barangay staff and officials. Walk-in residents can use the IoT-enabled kiosks with RFID and queue printing, while online users can access the web platform to request documents, report incidents, and check the status of their requests. On the other side, barangay staff and officials use the system to approve requests, update records, handle reports, and generate outputs. The system also keeps backups and logs to make sure the data is accurate and safe.

The context diagram below shows how the system interacts with its users and environment. Residents connect to the system through either the kiosk or the web platform, while staff and officials handle approvals, monitoring, and decision-making. This high-level view shows how the system acts as a bridge between the community and the barangay, helping services become more efficient and transparent.

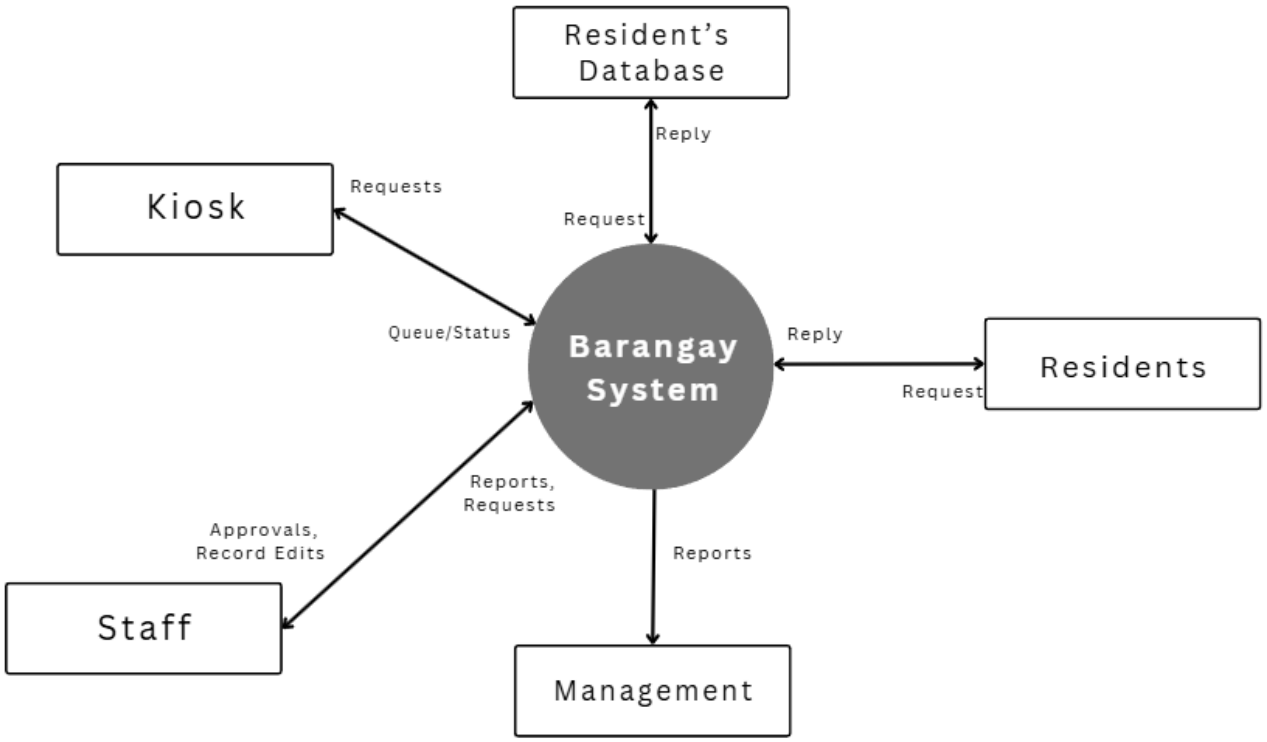

Figure 1. Context Diagram

2.2 Product Functionality

The Barangay Information System with IoT-Enabled Incident Reporting and E-Services is designed to perform the following main functions:

- **Resident Profiling** – store and update information about residents for easier processing of services.
- **Document Requests** – let residents request barangay documents through the web system or the kiosk.
- **Incident Reporting** – allow residents to file incident reports for barangay staff and officials to review.
- **Queue Management** – give queue numbers to walk-in residents using RFID at the kiosk to keep services organized.
- **Request Tracking** – let residents check the progress or status of their requests online.
- **Request Processing** – allow barangay staff and officials to review, approve, or reject requests and reports.
- **Email Notifications and Updates** – send updates through email so residents know the status of their requests and when outputs are ready.
- **Reports and Monitoring** – create reports and summaries that help staff and officials track services and make decisions.
- **Data Backup and Logs** – save backups and logs to make sure records are safe and reliable.

2.3 Design and Implementation Constraints

The Barangay Information System with IoT-Enabled Incident Reporting and E-Services has some limits that affect how it will be designed and built:

- **Design Method and Diagrams** – The project must follow the COMET method for software design and use UML diagrams (use case, class, sequence, activity, deployment) to explain the system.

- **Budget** – The total cost of developing and setting up the system should not go over ₱600,000, as set by the barangay and project sponsors.

- **Timeframe** – The whole system must be finished, tested, and to be deployed within 4 months (March to October 2025).

- **Technology Stack** – The system will be web-based only and must use PHP for backend, MySQL for database, and HTML, CSS, JavaScript with Bootstrap for the interface. This fits the team's skills and available tools.

- **IoT Kiosk** – Kiosks are limited to RFID login, queue number printing, and internet connection. They need to stay connected to the web system, but can handle short offline moments. Extra devices like cameras or biometrics are not included.

- **Internet Connection** – Since internet in some areas is unstable, the system must be lightweight, work with low bandwidth, and allow basic offline caching.

- **Hardware** – The barangay only has limited kiosks and mid-range PCs, so the system must run smoothly even on simple hardware.

- **Performance** – Normal pages should load in about 3 seconds, and kiosks must print queue numbers right away. Daily backups will be done, with data recovery limited to the most recent 24 hours.

- **Security and Privacy** – The system must follow the Philippine Data Privacy Act of 2012 and National Privacy Commission (NPC) rules. All data must be protected with HTTPS, password hashing, logs, and role-based access.

- **Maintenance** – The system should be simple enough for barangay staff with only basic computer skills to manage after turnover. Simple user guides will be provided.

- **Standards** – This SRS follows the IEEE format. System design will use COMET with UML diagrams. Source code will be handled in Git version control.

- **Interfaces** – The system will support:

    - Web interface (for online residents)
    - Kiosk interface (for walk-in residents)
    - REST API (internal use)
    - Email (for notifications)

- Out of scope: mobile apps, LGU/national system integration, SMS, payments, and advanced analytics.

2.4 Assumptions and Dependencies

The following assumptions and dependencies were identified for the Barangay Information System with IoT-Enabled Incident Reporting and E-Services. These factors may affect the project design and implementation.

Assumptions:

- The barangay will provide stable internet and electricity during system development and daily operations.

- Barangay staff will be available and willing to attend basic training to learn how to use and maintain the system.

- Residents will register and use RFID cards to securely access kiosks and system features.

- Users (residents, staff, and officials) will cooperate in adopting the digital system and moving away from manual processes.

Dependencies:

- The system depends on the availability and proper functioning of IoT kiosks, RFID card readers, and staff/admin computers.
- The online platform requires stable and secure web hosting services to remain accessible.
- The system relies on the availability of an **email service (SMTP server or provider)** to send notifications and updates to residents.
- The project relies on continued support and approval from the Barangay Gumaoc East council for funding, implementation, and policy adoption.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The main user interface for the Barangay Information System is provided through the **IoT-enabled kiosk** and the **web platform**. For the kiosk, the interface is displayed on a **touchscreen monitor**, as shown in the figure below.

Graphic (Kiosk Interface Example):

Figure 2. Kiosk Interface

Logical Characteristics:

- **Touchscreen Interaction** – Users interact with the system by tapping the on-screen menus and buttons. The touchscreen makes navigation simple and intuitive, even for residents with limited technical knowledge.
- **RFID Access** – Walk-in residents tap their RFID cards on the scanner below the screen to log in and authenticate their identity before accessing services.
- **Menu-Based Navigation** – The interface uses clear menus to guide residents in requesting documents, reporting incidents, or checking the status of requests.
- **Visual Feedback** – The system displays confirmation messages, queue numbers, and service updates directly on the screen. For walk-ins, the thermal printer provides a physical copy of queue numbers or receipts.
- **Email Notifications** – For requests made through the kiosk or web, updates are also sent to the resident's registered email address for convenience.

This design ensures that both residents and barangay staff can use the system easily, with simple menus, touch interaction, and real-time feedback, making barangay services more accessible and transparent.

3.1.2 Hardware Interfaces

Generic Windows RFID (Card) Reader
The system uses the RFID reader to scan and transmit unique tag IDs for user authentication, which is connected to the desktop via USB as an input device.

HP EliteDesk 800 G3 DM 35W Mini Desktop
This desktop serves as the main processing unit hosting the system's application, database, and services, and is physically connected to all peripheral devices including the RFID reader, touch monitor, and thermal printer.

Anmite 15.6 Touch Screen Portable Monitor
The system outputs visual content and receives touch input through this monitor, which is connected to the desktop via HDMI for display and USB for touch interaction.

PT-210 Thermal Receipt Mini Printer
The system uses this compact thermal printer to print queue numbers, connected to the desktop via USB as a dedicated ticket-printing device.

3.1.3 Software Interfaces

Mobile Companion Module
The system includes a mobile-accessible companion module that allows residents to access selected features of the web application using their smartphones. This module enables users to submit service requests, report incidents, and generate queue numbers remotely. It does not operate as a standalone mobile application; instead, it functions as a mobile-friendly interface that

relies on the main web-based system to process commands, store data, and display real-time status updates.

3.2 Functional Requirements

3.2.1 F1: Login (Web/Mobile)

The system shall authenticate residents using a username and password to access resident features.

3.2.2 F2: RFID Login (Kiosk)

The system shall authenticate kiosk users when a valid RFID card is tapped and start a kiosk session.

3.2.3 F3: Apply for Residency

The system shall allow an applicant to submit a residency application (kiosk or mobile web) and allow an Admin to approve or reject it; on approval, the system shall create a resident profile and optionally bind an RFID card.

3.2.4 F4: Submit Document Request (Online)

The system shall let a logged-in resident choose a document type, fill required fields, attach files, and submit a request saved with status Submitted.

3.2.5 F5: Submit Document Request (Walk-in)

The system shall let a kiosk user, after RFID login, submit a document request that is added to the processing queue.

3.2.6 F6: Generate Queue Number

The system shall generate the next queue number for a selected service and persist it to prevent reuse or skipping.

3.2.7 F7: Print Queue Ticket

The system shall send the generated queue number to the thermal printer and print a ticket showing only the queue number and service/window (no personal data).

3.2.8 F8: Reprint Last Ticket

The system shall allow reprinting of the last issued ticket when a print error occurs and shall record the reprint in the audit log.

3.2.9 F9: Track Request Status

The system shall let residents view the current status and timeline of their own document requests.

3.2.10 F10: Report Incident

The system shall let residents file an incident report with category, description, location, and optional photos.

3.2.11 F11: View Incident Status

The system shall let residents view status updates and the final resolution of incidents they submitted.

3.2.12 F12: Manage Resident Profiles

The system shall let Admins create, view, update, activate/deactivate resident profiles and bind or unbind RFID cards (e.g., lost-card replacement).

3.2.13 F13: Process Document Requests

The system shall let Admins review, approve, reject, mark as released, and generate/print requested barangay documents.

3.2.14 F14: Manage Incidents

The system shall let Admins assign incidents, add notes/updates, change status, and close incident reports.

3.2.15 F15: Dashboards and Reports

The system shall display Admin dashboards with counts and trends for requests, incidents, and queues, and shall allow CSV export.

3.2.16 F16: Role-Based Access Control

The system shall enforce roles (Resident, Admin) so only authorized users can access administrative functions and sensitive data.

3.2.17 F17: Audit Logging

The system shall record who, what, when, and where for sign-in/out, queue issuance/reprint, and create/update/delete actions on residents, requests, and incidents.

3.2.18 F18: Backup and Restore

The system shall run nightly backups and provide an Admin action to restore from the latest successful backup.

3.2.19 F19: Consent Capture

The system shall display a short data-privacy consent on forms and store the user’s consent timestamp and version upon submission.

3.2.20 F20: Configuration

The system shall let Admins manage reference data (document types, fees if any, and service windows) without requiring database schema changes.

3.3 Use Case Model

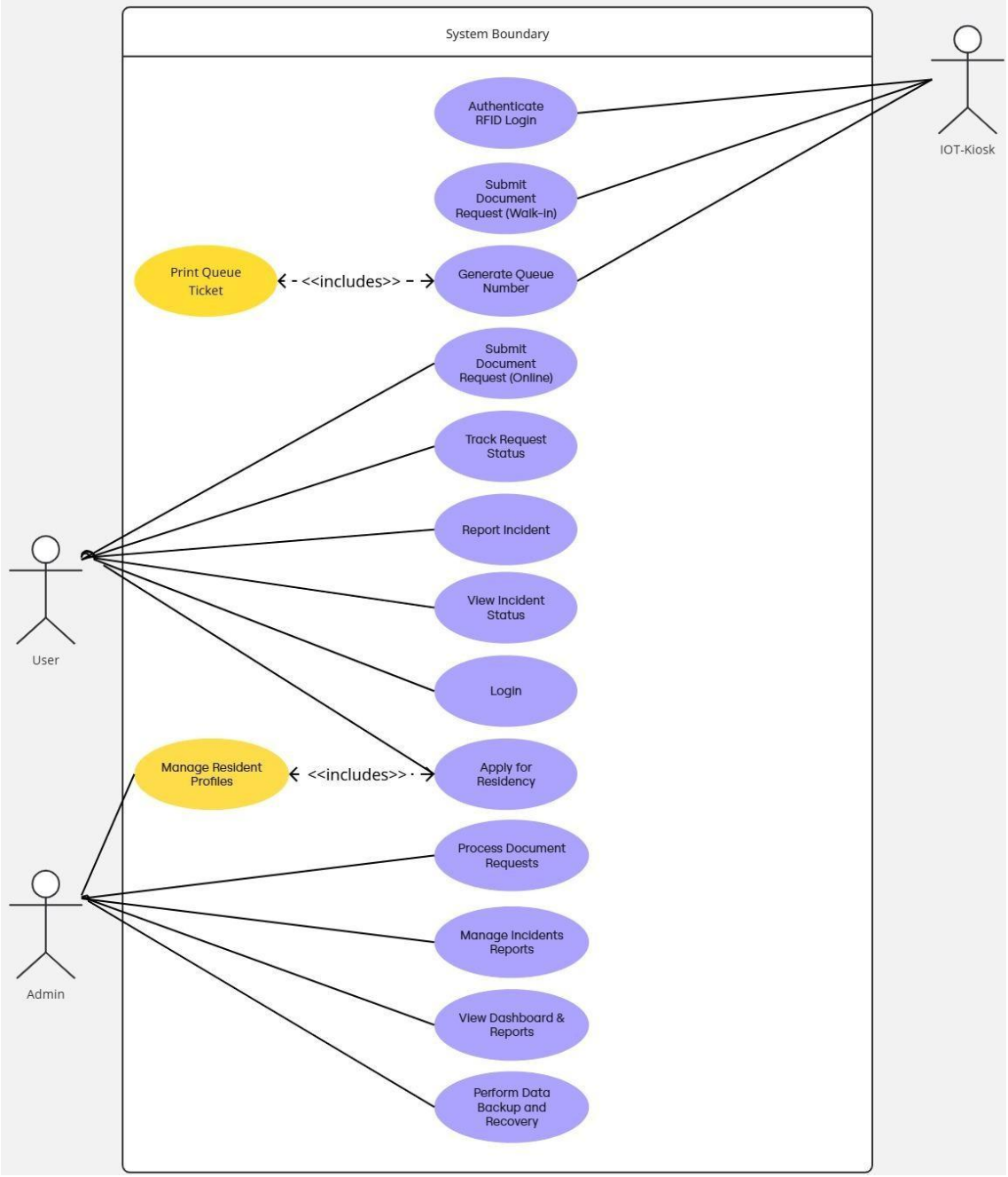

Figure 3. Use Case Model

3.3.1 Use Case #1 (Barangay Information System (Web App + Database + API) – Main Use Case Diagram)

Author: DVBD
Purpose: Describe all actor–system interactions for online and walk-in services, queueing, and

administration.

Scope: Entire BIS within a single system boundary.

Actors (outside boundary)

- **Resident** — uses web (mobile/desktop) for online services.
- **IoT Kiosk** — walk-in channel with RFID login and queue ticket print.
- **Admin** — barangay staff performing back-office tasks.

Use cases (inside boundary)

- **Resident:**
    1. Login
    2. Apply for Residency
    3. Submit Document Request (Online)
    4. Track Request Status
    5. Report Incident
    6. View Incident Status
- **IoT Kiosk (Walk-in):**
    7) Authenticate (RFID Login)
    8) Submit Document Request (Walk-in)
    9) Generate Queue Number
    10) Print Queue Ticket
- **Admin:**
    11) Manage Resident Profiles
    12) Process Document Requests (Review/Approve/Issue)
    13) Manage Incidents (Assign/Update/Close)
    14) View Dashboards & Reports
    15) Perform Data Backup and Recovery

Relationships shown on the diagram

- **«include»**:

  - **Generate Queue Number includes Print Queue Ticket**.

  - **Apply for Residency includes Manage Resident Profiles**.

- **Associations** (solid lines) exactly as drawn:

  - **Resident** → Login; Apply for Residency; Submit Document Request (Online); Track Request Status; Report Incident; View Incident Status.

  - **IoT Kiosk** → Authenticate (RFID Login); Submit Document Request (Walk-in); Generate Queue Number.

  - **Admin** → Manage Resident Profiles; Process Document Requests; Manage Incidents; View Dashboards & Reports; Perform Data Backup and Recovery.

Diagram-level preconditions

- System online; database reachable.
- At least one Admin account exists.
- IoT Kiosk peripherals (RFID reader, thermal printer) connected and configured.

Diagram-level postconditions

- Requests, incidents, queue numbers, and profile changes are persisted.
- Audit logs recorded for security-relevant actions.

High-level flows represented

- **Online channel (Resident):** Login → (Submit Document Request / Report Incident / Track/View status).
- **Walk-in channel (IoT Kiosk):** RFID Authenticate → (Submit Document Request (Walk-in) / Generate Queue Number → «include» Print Queue Ticket).
- **Administration (Admin):** Manage Resident Profiles; Process Document Requests; Manage Incidents; View Dashboards & Reports; Perform Backups.

Assumptions/notes

- “Login” (online) and “Authenticate (RFID Login)” (kiosk) are separate because mechanisms differ.
- “Print Queue Ticket” occurs only for walk-in queueing via the kiosk.
- “Apply for Residency” triggers Admin profiling via the «include» relation.

4 Other Non-functional Requirements

4.1 Performance Requirements

PR-01 Page response (online web):
The system shall render any authenticated page in ≤ **2.0 s** (P95) over a **5–10 Mbps** connection; P99 ≤ **3.0 s**. Test: synthetic web tests.

PR-02 Kiosk screen response:
Kiosk UI shall respond to a tap or button press in ≤ **300 ms** (P95). Test: UI latency measurement.

PR-03 RFID read & login:
After a valid RFID tap, the kiosk shall authenticate and show the next screen in ≤ **1.0 s** (P95). Test: 30–50 consecutive reads.

PR-04 Queue number issuance:
Upon request, the system shall generate a queue number in ≤ **500 ms** and print the ticket in ≤ **3.0 s** total. Test: end-to-end stopwatch, 20 samples.

PR-05 Document request submission:
Creating a request shall complete (DB write + confirmation screen) in ≤ **1.5 s** (P95). Test: API timing + UI timing.

PR-06 Dashboard freshness:
Staff dashboards shall reflect new requests/incidents within **≤ 15 s** of creation. Test: event→dashboard delta.

PR-07 Concurrency:
The system shall support **15 concurrent sessions** (combined kiosk + web) with no P95 latency breaches in PR-01/05. Test: load test.

PR-08 Data volume:
The system shall handle **≤ 6,000** resident records (design headroom **10,000**), **≤ 150,000** audit log entries, and **≤ 15,000** requests/incidents without reindexing. Test: dataset load + query timings.

PR-09 Availability:
Service uptime shall be **≥ 99.0%** monthly. Test: external uptime monitor.

PR-10 Backup & restore windows:
Nightly backup shall complete within **≤ 30 min**; **RPO ≤ 1 h**, **RTO ≤ 4 h**. Test: timed backup/restore drill.

PR-11 Print failure retry:
If the printer is unavailable, the system shall **queue and retry** printing every **30 s for 5 minutes** and provide a **reprint** action. Test: disconnect printer during issuance.

PR-12 API throughput:
Public/internal APIs shall sustain **≥ 20 req/min** with error rate **< 1%** under load. Test: load test + error budget.

4.2 Safety and Security Requirements

SEC-01 Transport security (mobile & web):
All traffic shall use **TLS 1.2+**; HTTP shall redirect to HTTPS. Test: SSL Labs A; HTTP blocked.

SEC-02 Authentication & RBAC:
Residents authenticate with credentials; kiosk users via **RFID**; admins via credentials + **MFA (recommended)**. Role-based access shall restrict data by role. Test: role matrix + access tests.

SEC-03 Password policy:
Min **8 chars**, complexity; lock account after **5** failed attempts for **15 min**; secure reset with time-bound token/OTP. Test: policy validation.

SEC-04 Session security:
Idle timeout **15 min** (resident) / **30 min** (admin); tokens/cookies **HttpOnly + Secure** with **SameSite**; CSRF tokens on state-changing requests. Test: header + CSRF tests.

SEC-05 Data at rest:
Sensitive fields (PII, incident details) encrypted at rest (e.g., **AES-256** on DB/volumes). Test: config review.

SEC-06 Audit logging:
Log sign-in/out, CRUD on residents/requests/incidents, queue issuance, and admin actions (who/what/when/where). Retain **≥ 1 year** and make logs tamper-evident. Test: generate actions; verify logs.

SEC-07 Input & upload validation:
Validate server-side; parameterized queries; output encoding; limit uploads to approved types and **≤ 10 MB**; scan for malware. Test: security scan + fuzz tests.

SEC-08 Least privilege (infrastructure):
Scoped DB users per service; no shared admin accounts; restrict printer/USB to kiosk service; disable unused ports. Test: IAM/OS review.

SEC-09 Privacy & consent:
Show consent text; store consent timestamp/version; support resident data export/delete on authorized request. Test: functional checks.

SEC-10 Availability protection:
Basic rate-limit on login/APIs and graceful degradation for dashboards under load. Test: stress test + 429 behavior.

SEC-11 Physical safety (kiosk & printer):
“Hot surface” label; **Disable Printing** control in admin UI during jams/errors; posted safe jam-clear steps. Test: UI controls + label present.

SEC-12 Secure mobile access (mobile connection):
Mobile web uses **HTTPS + HSTS**; sessions in **HttpOnly+Secure** cookies with **SameSite**; **CSRF token** required; **CORS** allow-list only; **CSP** + frame-ancestors 'none'; private pages send **Cache-Control: no-store**; idle logout **15/30 min** and **Logout-All-Devices**. Test: header checks + functional revoke.

.

4.3 Software Quality Attributes

QA-01 Reliability:
*Target: MTBF ≥ **30 days**; MTTR ≤ **2 h**; ≥ **99%** success for submit/queue/print.*
How: validation, idempotent APIs, print-retry queue, health checks.
Verify: ops metrics, error logs, replay tests.

QA-02 Maintainability:
*Target: unit coverage **≥ 60%**; ≤ **10%** lint issues; critical fix **≤ 5 days**.*
How: CI tests/lint, code review, small PRs.
Verify: CI reports; issue tracker SLA.

QA-03 Scalability:
*Target: sustain **30 concurrent users** (≈2× PR-07); key queries P95 **≤ 300 ms** at **6k** residents.*
How: indexes, pagination, light APIs, simple caching.
Verify: load test + query plans.

QA-04 Portability/Compatibility:
*Target: run on Windows 10/11 or Ubuntu LTS (HP EliteDesk 800 G3); browsers: latest Chrome/Edge/Safari (last 2); mobile ≥ **360×640**.*
How: pure web stack, responsive UI, USB printer on kiosk.
Verify: compatibility checklist per release.

QA-05 Usability:
*Target: kiosk ticket in **≤ 3 taps** and **≤ 10 s**; online request **≤ 5 steps**; basic **WCAG AA** (contrast/focus).*
How: simple flows, big buttons, PH/EN labels, clear errors.
Verify: 5-user hallway test; step/time counts; a11y check.

QA-06 Security (quality attribute):
*Target: no open **High/Critical** findings before release; Medium resolved within **30 days**.*
How: periodic SAST/DAST scans; code reviews.
Verify: scan reports.

QA-07 Robustness / Fault tolerance:
*Target: printer down → retry every **30 s** for **5 min** + **Reprint**; offline kiosk still prints and syncs later; uploads validated **≤ 10 MB**.*
How: local job queue/backoff; offline banner; validation.
Verify: unplug printer/network tests; boundary uploads.

QA-08 Testability:
*Target: full CI (unit+API+smoke) **≤ 10 min**; seed data + mocks.*
How: fixtures, mock printer/RFID.
Verify: CI timing; green pipeline.

QA-09 Observability:
Target: expose /healthz, /readyz; JSON logs; metrics (latency, errors, queue issued) with alerts on SLO breach.
How: lightweight monitoring/alerts.
Verify: dashboard screenshots; alert test.

QA-10 Reusability/Modularity:

Target: shared UI components used on ***≥ 3*** *screens; printer behind* ***PrinterAdapter****; business logic in services.*

How: component library, service layer, adapter pattern.

Verify: code import map; demo printer swap in ≤ ***1 day****.*

Appendix A – Data Dictionary/ERD

admin_logs

Column	Type	Null	Default	Comments
id *(Primary)*	int(11)	No		
admin_id	varchar(100)	Yes	system	
action_type	varchar(50)	No		
target_type	varchar(50)	No		
target_id	int(11)	Yes	*NULL*	
description	text	No		
details	longtext	Yes	*NULL*	
ip_address	varchar(45)	Yes	*NULL*	
user_agent	text	Yes	*NULL*	
created_at	timestamp	No	current_timestamp()	

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	34	A	No	
idx_admin_id	BTREE	No	No	admin_id	2	A	Yes	
idx_action_type	BTREE	No	No	action_type	17	A	No	
idx_target_type	BTREE	No	No	target_type	11	A	No	
idx_created_at	BTREE	No	No	created_at	34	A	No	

admin_users

Column	Type	Null	Default	Comments
id *(Primary)*	int(11)	No		
username	varchar(50)	No		
password	varchar(255)	No		
full_name	varchar(100)	No		
email	varchar(100)	No		
role	enum('super_admin', 'admin')	No	admin	
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	0	A	No	
username	BTREE	Yes	No	username	0	A	No	
email	BTREE	Yes	No	email	0	A	No	

business_applications

Column	Type	Null	Default	Comments
id *(Primary)*	int(11)	No		
user_id	int(11)	Yes	*NULL*	
reference_no	varchar(50)	Yes	*NULL*	

Column	Type	Null	Default	Comments
application_date	date	Yes	*NULL*	
first_name	varchar(100)	Yes	*NULL*	
middle_name	varchar(100)	Yes	*NULL*	
last_name	varchar(100)	Yes	*NULL*	
business_location	text	Yes	*NULL*	
or_number	varchar(100)	Yes	*NULL*	
ctc_number	varchar(100)	Yes	*NULL*	
business_name	varchar(255)	No		
business_type	varchar(100)	No		
business_address	varchar(500)	No		
owner_name	varchar(255)	No		
owner_address	text	Yes	*NULL*	
contact_number	varchar(20)	No		
years_operation	int(11)	No		
investment_capital	decimal(15,2)	No		
submitted_at	timestamp	No	current_timestamp()	
status	enum('pending', 'reviewing', 'approved', 'rejected')	Yes	pending	

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	2	A	No	
user_id	BTREE	No	No	user_id	2	A	Yes	

certificate_requests

Column	Type	Null	Default	Comments
id *(Primary)*	int(11)	No		
full_name	varchar(255)	No		
address	varchar(500)	No		
mobile_number	varchar(20)	Yes	*NULL*	
civil_status	varchar(50)	Yes	*NULL*	
gender	varchar(20)	Yes	*NULL*	
birth_date	date	No		
birth_place	varchar(255)	No		
citizenship	varchar(100)	Yes	*NULL*	
years_of_residence	int(11)	Yes	*NULL*	
certificate_type	varchar(100)	No		
purpose	text	No		
submitted_at	timestamp	No	current_timestamp()	
status	enum('pending', 'processing', 'ready', 'released')	Yes	pending	

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	5	A	No	

family_organizations

Column	Type	Null	Default	Comments
id *(Primary)*	int(11)	No		
registration_id	int(11)	No		
name	varchar(255)	No		
organization_type	varchar(255)	No		
created_at	timestamp	No	current_timestamp()	

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	2	A	No	
registration_id	BTREE	No	No	registration_id	2	A	No	

family_members

Column	Type	Null	Default	Comments
id *(Primary)*	int(11)	No		
registration_id	int(11)	No		
full_name	varchar(255)	No		
birth_date	date	Yes	*NULL*	
age	int(11)	Yes	*NULL*	
civil_status	varchar(50)	Yes	*NULL*	
education	varchar(100)	Yes	*NULL*	
occupation	varchar(100)	Yes	*NULL*	
skills	varchar(255)	Yes	*NULL*	
monthly_income	decimal(10,2)	Yes	*NULL*	
created_at	timestamp	No	current_timestamp()	

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	3	A	No	
registration_id	BTREE	No	No	registration_id	3	A	No	

residents

Column	Type	Null	Default	Comments
id *(Primary)*	int(11)	No		
first_name	varchar(100)	No		
middle_name	varchar(100)	Yes	*NULL*	
last_name	varchar(100)	No		
email	varchar(255)	No		
phone	varchar(20)	No		
password	varchar(255)	Yes	*NULL*	
address	text	No		
birthdate	date	No		
gender	enum('Male', 'Female', 'Other')	No		
civil_status	enum('Single', 'Married', 'Widowed', 'Separated', 'Divorced')	No		
rfid_code	varchar(50)	Yes	*NULL*	
rfid	varchar(50)	Yes	*NULL*	
status	enum('active', 'inactive', 'pending')	No	active	
reset_otp	varchar(6)	Yes	*NULL*	
otp_expiry	timestamp	Yes	*NULL*	
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	0	A	No	
residents_email	BTREE	Yes	No	email	0	A	No	
residents_phone	BTREE	Yes	No	phone	0	A	No	
residents_rfid_code	BTREE	Yes	No	rfid_code	0	A	Yes	
residents_rfid	BTREE	Yes	No	rfid	0	A	Yes	
idx_residents_status	BTREE	No	No	status	0	A	No	
idx_residents_rfid_codes	BTREE	No	No	rfid_code	0	A	Yes	
				rfid	0	A	Yes	

resident_registrations

Column	Type	Null	Default	Comments
id *(Primary)*	int(11)	No		
first_name	varchar(100)	No		
middle_name	varchar(100)	Yes	*NULL*	
last_name	varchar(100)	No		
birth_date	date	No		
age	int(11)	No		
civil_status	varchar(50)	No		
gender	varchar(20)	No		
contact_number	varchar(20)	Yes	*NULL*	
house_number	varchar(20)	Yes	*NULL*	
pangkabuhayan	varchar(100)	Yes	*NULL*	
submitted_at	timestamp	No	current_timestamp()	
status	enum('pending', 'approved', 'rejected')	Yes	pending	
land_ownership	varchar(100)	Yes	*NULL*	
land_ownership_other	varchar(255)	Yes	*NULL*	

Column	Type	Null	Default	Comments
house_ownership	varchar(100)	Yes	*NULL*	
house_ownership_other	varchar(255)	Yes	*NULL*	
farmland	varchar(100)	Yes	*NULL*	
cooking_energy	varchar(100)	Yes	*NULL*	
cooking_energy_other	varchar(255)	Yes	*NULL*	
toilet_type	varchar(100)	Yes	*NULL*	
toilet_type_other	varchar(255)	Yes	*NULL*	
electricity_source	varchar(100)	Yes	*NULL*	
electricity_source_other	varchar(255)	Yes	*NULL*	
water_source	varchar(100)	Yes	*NULL*	
water_source_other	varchar(255)	Yes	*NULL*	
waste_disposal	varchar(100)	Yes	*NULL*	
waste_disposal_other	varchar(255)	Yes	*NULL*	
appliances	text	Yes	*NULL*	
transportation	text	Yes	*NULL*	
transportation_other	varchar(255)	Yes	*NULL*	
business	text	Yes	*NULL*	
business_other	varchar(255)	Yes	*NULL*	
contraceptive	text	Yes	*NULL*	
interviewer	varchar(255)	Yes	*NULL*	
interviewer_title	varchar(255)	Yes	*NULL*	
login_id	varchar(255)	No		
password	varchar(255)	No		
user_type	varchar(255)	No		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	3	A	No	

rfid_access_logs

Column	Type	Null	Default	Comments
id *(Primary)*	int(11)	No		
user_id	int(11)	Yes	*NULL*	
rfid_tag	varchar(20)	Yes	*NULL*	
full_name	varchar(255)	Yes	*NULL*	
access_time	timestamp	No	current_timestamp()	

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	0	A	No	
user_id	BTREE	No	No	user_id	0	A	Yes	

rfid_registrations

Column	Type	Null	Default	Comments
id *(Primary)*	int(11)	No		
rfid_number	varchar(50)	No		

Column	Type	Null	Default	Comments
first_name	varchar(100)	No		
middle_name	varchar(100)	Yes	*NULL*	
last_name	varchar(100)	No		
birth_date	date	No		
contact_number	varchar(20)	No		
address	text	No		
card_type	enum('resident', 'employee', 'visitor')	Yes	resident	
status	enum('pending', 'approved', 'rejected', 'active', 'blocked')	Yes	pending	
issued_date	date	Yes	*NULL*	
expires_date	date	Yes	*NULL*	
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	0	A	No	
rfid_number	BTREE	Yes	No	rfid_number	0	A	No	

rfid_users

Column	Type	Null	Default	Comments
id *(Primary)*	int(11)	No		
rfid_tag	varchar(20)	No		
full_name	varchar(255)	No		
email	varchar(255)	Yes	*NULL*	
phone	varchar(20)	Yes	*NULL*	
address	text	Yes	*NULL*	
id_type	enum('National ID', 'Drivers License', 'Passport', 'Other')	Yes	National ID	
id_number	varchar(50)	Yes	*NULL*	
status	enum('active', 'inactive', 'suspended')	Yes	active	
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	0	A	No	
rfid_tag	BTREE	Yes	No	rfid_tag	0	A	No	

services

Column	Type	Null	Default	Comments
id *(Primary)*	int(11)	No		
title	varchar(255)	No		
description	text	No		
icon	varchar(50)	No		
button_text	varchar(100)	No		
button_link	varchar(255)	No		
is_featured	tinyint(1)	Yes	0	

Column	Type	Null	Default	Comments
features	text	Yes	*NULL*	
display_order	int(11)	Yes	0	
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	4	A	No	

updates

Column	Type	Null	Default	Comments
id *(Primary)*	int(11)	No		
title	varchar(255)	No		
description	text	No		
badge_text	varchar(50)	No		
badge_type	enum('important', 'new', 'community', 'info')	Yes	info	
date	varchar(50)	No		
status	varchar(50)	No		
is_priority	tinyint(1)	Yes	0	
display_order	int(11)	Yes	0	
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	id	3	A	No	

Appendix B - Group Log

MoM #	Date (Y:M:D)	Time	Place	Attendees	Agenda
1	2025-04-05	9:00-10:00 pm	Messenger	All	Goals, scope, roles
2	2025-05-19	1:00 pm - 3:00pm	Barangay Hall	Barangay Captain, Secretary and All members	Requirements & constraints
3	2025-08-27	9:00-10:00 pm	Yvan's house	All	Hardware check (RFID, printer); kiosk UI sketch
4	2025-09-16	9:00-10:00 pm	Messenger	All	Finalize Functional Requirements; draw use case diagram
5	2025-09-16	9:00-10:00 pm	Messenger	All	ERD & Data Dictionary (entities/relations)
6	2025-09-16	9:00-10:00 pm	Messenger	All	Safety & Security (incl. mobile connection)

Appendix C – Test Plan/Test Cases

Test Plan

Scope: Resident web/mobile site, IoT kiosk (RFID + queue printing), Admin portal.
Out of scope: 3rd-party e-payments, multi-barangay federation.
Environment: HP **EliteDesk 800 G3**, **Anmite 15.6" touch**, RFID reader, PT-210 printer, 5–10 Mbps LAN/Wi-Fi, latest Chrome/Edge. Seed data: ~200 residents, ~500 requests/incidents.

Roles:

- Tester/QA (executes cases, logs defects)
- Dev (fixes, provides builds)
- Product/PM (accepts)

Entry criteria: Features code-complete; devices connected; test data loaded.
Exit criteria: ≥95% of **Priority** tests pass; **no open Critical/High** defects; Perf/Security checks pass against targets.

Main risks & mitigations:

- Printer jams → reprint flow tested; job retry queue.
- Intermittent network → offline queue issuance tested; later sync.
- Shared/public mobile devices → session timeout + logout-all tested.

ID	Title	Preconditions	Steps	Expected Result
TC-F01	Resident login (web)	Resident exists	Open site → login with valid creds	Land on resident Main page
TC-F02	Login lockout	Resident exists	Enter wrong password ×5	Account locked 5 min; message shown
TC-F03	RFID login (kiosk)	RFID bound to resident	Tap RFID card	Next screen ≤1.0 s; session started
TC-F04	Generate & print queue	Logged in at kiosk	Get Queue → choose service → confirm	Unique ticket printed ≤3.0 s; saved in DB
TC-F05	Reprint last ticket	Printer was disconnected	Generate ticket → reconnect → Reprint	Single correct ticket prints; reprint logged
TC-F06	Submit document request (online)	Logged in	Open form → fill valid data → submit	Request saved as **Submitted**; confirmation
TC-F07	Submit request (walk-in)	RFID session	Fill kiosk form → submit	Request saved; appears in Admin

				queue
TC-F08	Track request status	One request exists	Open “My Requests”	Correct status & timeline shown
TC-F09	Report incident w/ photo	Logged in	Open Incident → fill details + image → submit	Incident **Open** with attachment
TC-F10	Admin approve & release doc	Admin logged in	Open pending → approve → mark released	Status updates; printable doc available
TC-F11	Manage resident profile + RFID	Admin logged in	Create profile → bind tag → unbind	Actions succeed; audit log entries present
TC-F12	Consent capture		Submit any main form	Consent stored (time + version)