Clash-A-Thon 2026

The Friction We Forget

Building Solutions for Problems We've Stopped Noticing

Kick-off	Submission Deadline	Final Presentation
Feb 24, 2026 | 09:30 AM	**Feb 26, 2026 | 4:00 PM**	**Feb 27, 2026 | On-site**

TEAM & PROJECT INFORMATION

Team Name	The Friction Fixers
Project Title	MeroPaalo
Project Category	SaaS-based System/Digital Public Service Innovation

TEAM MEMBERS

#	Full Name	Role / Responsibility	GitHub / Contact
1	Grace Gautam	Business Strategist	9844001111
2	Susant Lamsal	Backend Developer	Lamsalbaje45
3	Prajwal Koirala	Frontend Developer	Prajwal-k0ira1a
4	Sujal Shrestha	Frontend Developer	SujalShrestha-sys
5	Subin Shrestha	Backend Developer	Shresthasubin

SUBMISSION LINKS

GitHub Repository	https://github.com/Prajwal-k0ira1a/ClashAThon-TheFrictionFixers-MeroPaalo
Deployment URL	https://meropaalo-queue-frontend.vercel.app/
Tech Stack	MERN

Declaration

We declare that this project is our original work developed during Clash-A-Thon 2026. All external resources and libraries have been appropriately credited. We understand that plagiarism or misrepresentation will result in disqualification.

Abstract

Traditional queue management systems used in banks, hospitals and government offices often rely on manual token processes, leading to overcrowding, inefficient service handling and limited transparency in waiting times. To address these challenges, this project presents MeroPaalo, a web-based Smart Queue Management System designed to digitize and streamline queue operations through real-time monitoring and automated token management.

This system allows customers to join queues using QR-based digital token generation and track their status in real time. Administrators and staff are provided with a centralized dashboard to manage departments, control daily queue sessions, assign counters, and monitor service performance metrics. The backend is developed using Node.js and Express.js with MongoDB as the database, while the frontend is built using React (Vite). Real-time synchronization between users and administrators is achieved through Socket.IO.

By integrating structured queue control, role-based authentication, and live updates, MeroPaalo improves operational efficiency, reduces physical congestion, and enhances customer experience. The system is scalable and deployment-ready, making it suitable for adoption across various service-oriented institutions.

Acknowledgement

We would like to sincerely express our gratitude to everyone who supported us throughout the development of this project during Clash-A-Thon. Their encouragement and guidance played a significant role in helping us successfully complete this work on time.

We would like to extend our heartfelt thanks to Mr. Romi Khatri and Mr. Binay Koirala for their valuable guidance, motivation, and continuous support. Their insights and constructive feedback greatly contributed to improving our ideas and strengthening our project.

We are also deeply thankful to the Innovation Lab for providing us with the platform, resources, and supportive environment that enabled us to turn our concept into a practical solution during Clash-A-Thon.

Table of Contents

Table of Tables

Table of Figures

1 Introduction and Background

Figure 1 MeroPaalo Logo

“Because Waiting Shouldn’t Be Wasting.”

In alignment with the theme of Clash-A-Thon 2026, “The Friction We Forget,” our project MeroPaalo addresses one of Nepal’s most normalized yet overlooked inefficiencies of long, uncertain physical queues in hospitals, banks, colleges and government offices. Every day, citizens silently lose 30–90 minutes waiting without clarity, structure or estimated service time, leading to frustration, stress and reduced productivity.

We chose MeroPaalo because this friction is universal, recurring and solvable through simple technology.

Traditional queue systems lack real-time visibility, structured tracking and estimated waiting time, resulting in wasted time and operational inefficiency. Concluding, MeroPaalo is a real-time, QR-based digital queue management system that generates tokens, tracks live queue progress, displays current serving numbers and provides estimated wait times transforming passive waiting into intelligent waiting while improving institutional efficiency and user experience.

2 Technical Documentation

2.1 System Architecture

System architecture refers to the conceptual model that defines the structure, behavior, and mode importantly, the interaction among various components of a system. (Tiwari, 2025) It is the high-level blueprint of a system that usually defines system's components, responsibilities, communication, data flow, technology choices, and scalability and development strategy. The system architecture of our system is basically divided into three layers, i.e. Interface layer, Logic layer, and Storage layer. In our architecture, technologies used in different layer are shown and also core responsibilities performed in each layer are demonstrated.

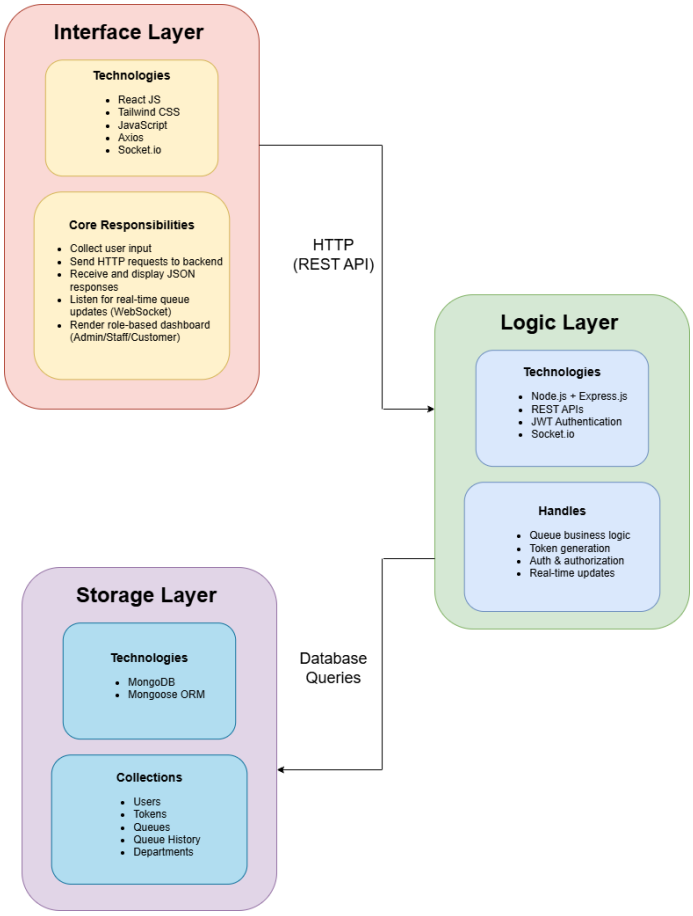

Figure 2 System Architecture

2.2 Component Breakdown

This section summarizes the major modules or components that comprise the system. The modules are the individual functional units, which perform certain tasks or services in the application. Knowing these key elements assists in imagining the system structure, their roles and their interaction to bring about the overall functionality.

Major components of our system:

1. **Authentication & User Management Module**

The Authentication and User Management module is concerned with all user identity and access issues in the system. It handles user registration, user login, user verification, and user identity. This module is based on JWT authentication so that only authorized personal can access the system. Users are assigned a role during registration where they can be an Admin, Staff, or Customer and this role is enforced based on the type of role they are assigned.

The communication between the components works as follows:

- The frontend communicates the login or registration requests to backend APIs.
- The credentials are verified and user data is stored in the MongoDB.
- JWT tokens are subsequently produced and sent back to the front facing system, allowing safe access to the application.

2. **Customer Queue Module**

Customer Queue module permits the customers to seek a token number, monitor their status in the queue, and monitor the progress of the queue in real time. At the point, when a customer orders a token, the system will automatically create a queue number and it will be stored in the database.

The communication between components work as follows:

- React frontend interact with the backend via a token request, which is then handled by the backend to form an entry in a queue in MongoDB.
- The updated queue information is also relayed to the frontend and information about the queue is given out dynamically to the customers, with their true and real-time positions in the queue.

3. Staff Queue Management Module

The Staff Queue Management module allows the staff members to operate the queue. Staffs will be able to look at queues that are active, call the next customer and check the customer marked as served in a serial manner. The staff manually controls the queue flow, which provides orderly service.

Communication flow:

- Every action of a staff leads to API calls to the backend that change the queue status in MongoDB.
- This update is subsequently displayed on staff and customer dashboards to ensure all the system has coordinated and real-time queue information.

4. Admin Module

Admin module offers system level control and management. Admins are in charge of user checks and verifying users to only allow authorized and verified users to work in the system. The backend gets informed of the actions done by an Admin, e.g. approving the user or role management through API requests. User records within MongoDB are updated by the backend and the updates are disseminated all over the system ensuring secure and consistent data within all modules.

5. Component Communication

A well-organized flow of communication is used amongst all components of the system. The React frontend makes a few HTTP requests to the backend of the Node.js and Express server that executes business logic and performs authentication. Its backend handles the MongoDB storage and retrieval of data, where it responds in the form of JSON to the frontend. Based on these responses the frontend dynamically changes the user interface. The communication model guarantees a safe flow of data, role-based access control and effective queue management across the system.

2.3 Data Design

The MeroPaalo project is developed on MongoDB and Mongoose that handles all the system data. This system will have core collections used to manage users, departments and counters,

and runtime collections used to manage queues, tokens, displays and histories. This data model guarantees scalability, modularity and precision in real-time tracking of queue activities.

Database structure of our system:

1. **User**

This is the collection of all individuals who are logging into the system whether they are administrators or not members of staff. It maintains basic identity data, credentials of authentication and role-based access data.

Entities and Datatypes:

Department = ObjectID

Name = String

Email = String

Phone = String

Password = String

Role = ENUM("admin", "staff", "customer")

resetPasswordToken = String

resetPasswordExpires = Date

2. **Departments**

Departments are functional areas like OPD, Billing or Finance.

Entities and Datatypes:

Name = String

Description = String

AvgServiceTime = Number

isActive = Boolean

3. **Counters**

Under a department, there are the physical points of service, called counters.

Entities and Datatypes:

counterName = String

Department = ObjectID

Staff = String

Status = ENUM("open", "closed")

4. **QueueDays**

This collection includes the queue session per day of a department.

Entities and Datatypes:

Department = ObjectId

Date = Date

startTime = String

endTime = String

status = ENUM("active", "paused", "closed")

5. **Tokens**

This store tokens for each customer. Each token is linked to different department.

Entities and Datatypes:

Department = ObjectId

queueDay = ObjectId

Customer = ObjectId

tokenNumber = String

status = ENUM("waiting", "called", "serving", "completed", "missed", "cancelled")

issuedAt = Date

calledAt = Date

servingAt = Date

completedAt = Date

6. **Displays**

This represents public display boards that show which token is currently being served at each counter, providing real-time queue updates to customers.

Entities and Datatypes:

Department = ObjectId

Counter = ObjectId

currentToken = ObjectId

updatedAt = Date

7. TokenHistories

This maintains an audit trail of all changes in token status, recording who made the change, when it occurred, and the previous and updated status.

Entities and Datatypes:

Token = ObjectID

Counter = ObjectId

Status = String

Note = String

changedAt = Date

2.3.1 Data Flow Overview

- The first step is creating departments.
- Staff users are assigned to departments.
- Counters are created and linked to departments, with optional staff assignment.
- A QueueDay document is created daily for each department to manage that day's queue.
- Customers issue tokens for a specific department, and each token is associated with a QueueDay.
- Staff manage tokens by updating their status (waiting, called, serving, completed, missed, cancelled).
- Token status updates trigger updates to the Display and create corresponding Token History records.

- Socket.IO broadcasts real-time updates to dashboards, displays and connected clients within the department.

2.3.2 Relationships

- A department can have many counters, but a counter belongs to only one department.
- A department can have many staff members, but a staff member belongs to only one department at a time.
- A department can have many QueueDays over time, but a QueueDay belongs to only one department.
- A QueueDay can generate many tokens in a day, but a token belongs to only one QueueDay.
- A token belongs to one department.
- A token can have many TokenHistory records, but each TokenHistory belongs to only one token.
- A display row can display only one token at a time.
- A department can have multiple display rows.

2.4 API / Interface Specifications

The MeroPaalo system provides a RESTful API developed using Node.js and Express.js to enable communication between the React-based frontend and backend services.

Base URL: *http://<host>:<port>/api*

Note: All endpoints use application/json format, except the QR endpoint which returns image/png.

Authentication is implemented using JSON Web Tokens (JWT), transmitted via HttpOnly cookies or the Authorization: Bearer <token> header. The system supports role-based access control with three roles: Customer, Staff, and Admin.

1. **Authentication:**

POST /auth/register →Register new customer

POST /auth/login → Authenticate user and issue JWT

POST /auth/logout → Terminate session

POST /auth/forgot-password → Initiate password reset

POST /auth/reset-password/:token → Reset password

2. **Administrative Interfaces:**

Administrators manage departments, counters, and users:

GET/**POST**/**PATCH**/**DELETE** /departments → Department management

GET/**POST**/**PATCH** /counters → Counter management and staff assignment

GET /users → Retrieve users

PATCH /users/:id/role → Update user roles

3. **Queue and Token Management**

Queue lifecycle operations include:

POST /queue-days/open → Open queue

PATCH /queue-days/:id/pause|resume|close → Control queue state

POST /queue-days/:id/reset → Reset queue

Token operations include:

POST /tokens/issue → Generate queue token (Public)

GET /tokens/:id/status → Check token status

POST /tokens/serve-next → Call next token (Staff/Admin)

PATCH /tokens/:id/complete|miss|cancel → Update token status

Token states include: waiting, called, serving, completed, missed, and cancelled.

4. **Public and Display Interfaces**

GET /public/queue/:departmentId/info → View queue details

GET /display → Live queue display feed

GET /qr?department=<id> → Generate QR code for queue access

5. **Response Format**

All responses follow a standardized structure:

{

"success": true,

"data": {

…

}

}

Error responses include appropriate HTTP status codes (**200**, **400**, **401**, **403**, **404**) and descriptive messages.

2.5 Setup and Deployment

1. System Requirements

To run the MeroPaalo Queue Management System, the following software and tools are required:

Node.js (v16 or later)

npm (Node Package Manager)

MongoDB (Local installation or MongoDB Atlas cloud database)

Modern web browser (Chrome, Edge, Firefox)

Cloud hosting service such as Render, Vercel.

2. Installation and Local Setup

- **Clone the Repository**

Command:

→ **git clone** <repository-url>

→ **cd** MeroPaalo

- **Backend Setup**

Command:

→ **cd** backend

→ **npm install**

Create a .env file with the following environment variables:

```
PORT=5000
MONGO_URI=<your-mongodb-connection-string>
JWT_SECRET=<your-secret-key>
CLIENT_URL=http://localhost:5173
```

Start the backend server:

→ **npm run dev**

❖ **Frontend Setup**

→ **cd** frontend

→ **npm install**

→ **npm run dev**

The frontend will typically run on:

http://localhost:5173

3. Deployment

For production deployment:

- Configure environment variables securely on the hosting platform.
- Build the frontend using:

npm run build

- Deploy the backend server to a Node.js-compatible hosting service.
- Use a cloud-hosted MongoDB instance (e.g., MongoDB Atlas).
- Enable HTTPS and secure cookies for authentication.

The system follows a client–server architecture and can be deployed independently for frontend and backend services, enabling scalable and modular production deployment.

3 Business Model Canvas

The Business Model Canvas below outlines the nine key components of MeroPaalo's business strategy, demonstrating how the system creates, delivers and captures value as a queue management SaaS platform targeting Nepalese institutions.

Key Partners	Key Activities	Value Propositions	Customer Relationships	Customer Segments
• Internet service providers • Cloud hosting (Vercel, Render) • SMS/notification gateways (While scaling) • Hospital & bank IT depts • Government digital initiatives • QR code technology providers	• Platform development • Real-time queue engine • Client onboarding & training • Customer support & feedback • System monitoring & updates	• Eliminate physical waiting • Real-time queue visibility • SMS/app turn notifications • Reduce congestion & stress • Admin insights on visitor flow • Zero hardware requirement	• Dedicated onboarding support • Self-service admin dashboard • In-app help & documentation • Feedback collection via app • Regular feature updates	• Banks & financial institutions • Hospitals & clinics • Government offices • College admin blocks • Telecom service centers • Urban institutions in KTM, Dharan, Pokhara

Table 1 Business Model Canvas I

Key Resources	Channels	Cost Structure	Revenue Streams	Pricing Tiers
• MERN stack codebase • Cloud infrastructure • Development team (5 members) • Domain & brand identity • User & queue data	• Direct institutional sales • Web platform (SaaS) • QR code at institution entry • Social media marketing • IT vendor partnerships	• Cloud hosting & server costs • SMS/notification fees • Development & maintenance • Marketing & sales outreach • Customer support operations	• Monthly SaaS subscription • Tiered pricing (Basic/Pro) • Premium analytics dashboard • SMS notification packages • Multi-branch add-on • Priority queue feature	• Basic: Single queue, real-time tracking, admin panel • Pro: Analytics, ETA, advanced reporting, priority queue • Enterprise: Multi-branch, API access, custom integrations

Table 2 Business Model Canvas II

Brief Revenue Model Summary: MeroPaalo follows a B2B SaaS subscription model where institutions pay a monthly fee based on active service counters. End-users (visitors) access the system for free via QR code. The basic plan covers single-queue real-time tracking with an admin dashboard, while the Pro plan adds daily analytics, estimated wait time predictions and advanced reporting capabilities.

4 Unique Selling Proposition

What makes MeroPaalo different?

Unlike existing queue management solutions in Nepal that rely on expensive hardware kiosks, physical token dispensers and wired counter displays, MeroPaalo is a fully software-based, mobile-first solution that requires zero specialized hardware. Visitors simply scan a QR code with their smartphone to join a queue and track their position in real time.

The one thing we do better than anyone else:

MeroPaalo provides real-time remote queue visibility. Users do not need to be physically present in a waiting area. They can monitor their queue position from anywhere, a nearby café, their vehicle or even from home and receive a notification when their turn approaches. No existing solution in Nepal offers this level of freedom to visitors.

Why should institutions choose MeroPaalo?

The system eliminates capital expenditure on hardware kiosks and display units. It can be deployed within hours through a simple web setup, making it accessible even to small institutions with limited IT budgets. The administrative dashboard provides actionable insights into visitor flow, peak hours and average service times data that hardware-only systems cannot provide.

Competitive Advantage:

MeroPaalo’s edge lies in its low deployment cost, instant setup, mobile-first design and real-time WebSocket-powered updates. While competitors require physical installation and maintenance contracts, MeroPaalo operates entirely in the cloud with a lightweight SaaS model.

4.1 Competitor Comparison Table

Feature	MeroPaalo	Raj Trading QMS	PCQMS (Real Time Solutions)	Signage Plus
QR-Based Digital Token (No Hardware)	✓	✗	✗	✗
Real-Time Remote Queue Tracking	✓	✗	✗	~
SMS / App Turn Notification	✓	✗	✗	~
Zero Hardware / Installation Cost	✓	✗	✗	✗
Admin Analytics Dashboard	✓	~	~	✓
Cloud-Based SaaS Model	✓	✗	✗	✓
Mobile-First User Interface	✓	✗	✗	~
Estimated Wait Time Display	✓	✗	✗	✗

Table 3 Competitor Comparison Table

Index:

✓	Fully Support
~	Partially Support
✗	Not Supported

Table 4 Index of Competitor Comparison Table

5 Market Analysis

5.1 Target Market

MeroPaalo targets institutions in Nepal that serve walk-in visitors and experience regular queue congestion. The primary customer segments include banks and financial institutions (with over 6,500 branches nationwide) (Chaudhary, 2025), hospitals and healthcare facilities (over 125 hospitals and 14,000+ registered health facilities) (Public Health Update, 2023), government service offices (CDO offices, tax offices, transport offices, passport offices across 77 districts) (Wikipedia, 2026), and educational institutions (college and university administration blocks). The first phase focus on urban centers of Kathmandu Valley, Pokhara, Dharan, Itahari and Birgunj where visitor density and queue friction are highest.

5.2 Market Size

Market Tier	Estimated Size	Description
TAM	20,000+ institutions	All banks, hospitals, government offices, colleges, and service centers across Nepal handling walk-in visitors
SAM	3,000–5,000 institutions	Urban institutions in major cities with internet connectivity and moderate-to-high visitor footfall
SOM (Year 1)	50–100 institutions	Early adopters in Kathmandu Valley like banks, hospitals, and government offices willing to pilot digital queue solutions

Table 5 Market Size

5.3 Competition

The current queue management landscape in Nepal is dominated by hardware-based vendors. Raj Trading Concern is the largest QMS hardware supplier in Nepal, offering push-button and touchscreen kiosk systems primarily deployed in banks and government offices. (Raj Trading Concern, 2020) Real Time Solutions Pvt. Ltd. provides the PCQMS hardware platform used in most banks and government utility offices including NEA, NTC, and Ncell. Signage Plus is a newer entrant offering hybrid digital signage with queue integration, but focuses primarily on display screens rather than visitor-facing mobile solutions. (Nepalibazar, 2026) All existing competitors require significant upfront hardware investment, making them inaccessible to

smaller institutions. None offer real-time remote queue tracking or smartphone-based token management.

5.4 Market Opportunity

Several trends make this the right time for MeroPaalo. Nepal's digital banking adoption is accelerating rapidly, with over 26.7 million mobile banking users as of mid-2025 and increasing smartphone penetration across urban populations. (Aktar, 2025) The government's push for digital governance and e-service delivery is creating institutional readiness for digital queue solutions. Post-pandemic awareness of crowd management and social distancing has made both institutions and visitors more receptive to virtual queuing. The absence of any affordable, software-only queue management solution in the Nepalese market represents a clear gap that MeroPaalo is positioned to fill.

5.5 Go-to-Market Strategy: First 100 Users

MeroPaalo's strategy to acquire the first 100 institutional users follows a phased approach:

- **Phase 1 – Pilot (Month 1–2):** Deploy free pilots at 5–10 institutions in Kathmandu Valley (2–3 banks, 2–3 hospitals, 2–3 government offices) to build case studies and testimonials.
- **Phase 2 – Local Expansion (Month 3–4):** Leverage pilot success stories for direct outreach to branch managers. Offer introductory pricing for early adopters.
- **Phase 3 – Digital Marketing (Month 4–6):** Run targeted social media campaigns on Facebook showcasing before/after queue experiences. Partner with IT associations and banking forums.
- **Phase 4 – Referral Growth (Month 5–6):** Implement referral discounts where existing clients refer peer institutions for mutual subscription benefits.

6 Sustainability & Future Scope

6.1 Financial Sustainability

MeroPaalo sustains itself through a recurring SaaS subscription model. Institutions pay a monthly fee based on the number of active queues or service counters. Because the system is fully cloud-hosted with minimal infrastructure costs (Vercel for frontend, Render for backend, MongoDB Atlas for database), operational expenditure remains low. As the user base scales, the marginal cost of adding new institutions decreases significantly, creating favorable unit economics. Additional revenue streams include premium analytics packages, SMS notification bundles, and multi-branch management add-ons. The freemium approach for end-users (visitors) ensures rapid adoption without payment barriers, while institutions who benefit directly from reduced congestion and improved efficiency are the paying customers.

6.2 Next Features to Build

- **AI-Powered Wait Time Prediction:** Machine learning models trained on historical queue data for more accurate estimated wait times for that specific organization.
- **WhatsApp Integration:** Notification delivery through WhatsApp, which has higher engagement rates than SMS in Nepal.
- **Dedicated Mobile Application:** Native Android and iOS apps for enhanced push notifications and offline token caching.
- **Multi-Branch Dashboard:** Centralized management for institutions with multiple branches.

6.3 6 Month and 1 Year Vision

Timeline	Goals
6 Month Vision	✓ Onboard 30–50 institutions across Kathmandu Valley. ✓ Launch the Pro plan with analytics and reporting. ✓ Establish a customer support team. ✓ Achieve break-even on operational costs. ✓ Begin collecting queue data to train predictive wait-time models.
1 Year Vision	✓ Expand to 100+ institutions across five major urban centers (Kathmandu, Pokhara, Dharan, Itahari, Birgunj). ✓ Launch Android and Apple app. ✓ Introduce AI-based wait time predictions. ✓ Secure partnerships with at least 2 banking groups for enterprise deployment. ✓ Achieve positive MRR growth.

Table 6 6 Month and 1 Year Vision

6.4 Key Partnerships & Resources for Scaling

- **Banking Sector Partnerships:** Collaborating with Nepal Bankers' Association or individual bank headquarters for multi-branch rollouts.
- **Government Digital Nepal Initiative:** Aligning with Nepal's e-governance goals to deploy MeroPaalo in government service delivery offices.
- **Healthcare IT Integrators:** Partnering with hospital management software providers to integrate MeroPaalo as a queue module.
- **Startup Ecosystem Support:** Applying for incubation programs and seed funding from local angel investors or innovation grants.
- **SMS/Communication Providers:** Negotiating bulk SMS rates with Nepali telecom operators (Ncell, NTC) for cost-effective notification delivery at scale.

7 Conclusion

MeroPaalo addresses a deeply normalized inefficiency in Nepal's service ecosystem by transforming unmanaged physical queues into structured, data-driven digital workflows. Through a scalable MERN-based architecture, real-time synchronization and a zero-hardware SaaS model, the system delivers measurable operational efficiency for institutions while restoring time and predictability to citizens. Its competitive advantage lies in affordability, rapid deployment, and remote queue visibility features absent in traditional hardware-based solutions. With a clear monetization strategy, defined market entry plan and future roadmap incorporating analytics and AI-driven predictions, MeroPaalo demonstrates both technical feasibility and commercial viability as a sustainable digital infrastructure solution for Nepal.

8 References

Aktar, O. (2025) *Nepal's Financial Access Expands Rapidly in 2025* [Online]. Available from: https://nepalmonitor.com/2025/06/07/nepals-financial-access-expands-rapidly-in-2025 [Accessed 26 February 2026].

Chaudhary, S. (2025) *District-wise Distribution of Bank and Financial Institution Branches in Nepal* [Online]. Available from: https://nepsetrading.com/blog/district-wise-distribution-of-bank-and-financial-institution-branches-in-nepal?utm_source=chatgpt.com [Accessed 26 February 2026].

Nepalibazar. (2026) *Nepalibazar* [Online]. Available from: https://www.nepalibazar.com/detail-m-14-p-414-n-Queue%2BManagement%2BSystem.html?utm_source=chatgpt.com [Accessed 26 February 2026].

Public Health Update. (2023) *Number of Health Facilities in Nepal* [Online]. Available from: https://publichealthupdate.com/number-of-health-facilities-in-nepal-2/?utm_source=chatgpt.com [Accessed 26 February 2026].

Raj Trading Concern. (2020) *Queue Management System* [Online]. Available from: https://rajtrading.com.np/products/queue-management-system/?utm_source=chatgpt.com [Accessed 26 February 2026].

Tiwari, V. (2025) *Understanding System Architecture: A Beginner's Guide* [Online]. Available from: https://medium.com/@vaibhavtiwari.945/understanding-system-architecture-a-beginners-guide-d2aad48f0007 [Accessed 25 February 2026].

Wikipedia. (2026) *District administration in Nepal* [Online]. Available from: https://en.wikipedia.org/wiki/District_administration_in_Nepal?utm_source=chatgpt.com [Accessed 26 February 2026].

9 Appendix

This appendix represents a structured summary of consultation conducted for feasibility validation purposes and has been documented for academic submission.

Industry Consultation of Banking Sector Validation

Institution	Everest Bank Limited
Position of Respondent	Branch Manager
Location	Silichong Rural Municipality Branch
Consultation Type	Informal Structured Discussion
Date	26th February 2026

Table 7 Industry Consultation of Banking Sector Validation

Purpose: Operational feasibility assessment of MeroPaalo digital queue system in Nepalese banking context.

Structured Questionnaire and Discussion Summary:

1. What are the major queue-related challenges faced in your branch?

The branch experiences peak congestion during pension days, remittance days and government payment cycles. Customers often wait 30–60 minutes depending on service type. The main issue is not only waiting time but lack of visibility regarding how long the wait will be.

Specific challenges include:

- Customers repeatedly asking staff about queue status
- Senior citizens standing for long periods
- Token misplacement or manual confusion
- Counter load imbalance
- Crowd formation near teller counters
- The branch currently relies on traditional physical queue systems.

2. How does queue congestion impact operational efficiency?

Queue congestion directly impacts:

- Staff productivity (frequent interruptions)
- Customer satisfaction scores
- Branch environment and discipline
- Service perception even if actual service time is efficient

The manager noted that customer frustration increases when there is uncertainty rather than delay itself.

3. How feasible would a QR-based digital queue system be in your branch?

The manager stated that feasibility depends on customer demographic.

In semi-urban areas:

- Smartphone penetration is increasing.
- Most customers use mobile banking applications.
- Younger customers would adapt quickly.

However, rural customers and elderly clients may require assistance.

He suggested a hybrid model (digital + manual support counter).

4. What constraints might arise during implementation?

Identified Constraints:

- Internet Stability: Rural branches may face occasional network interruptions.
- Change Resistance: Staff adaptation period required.
- Head Office Approval: Any system integration requires clearance from central IT department.
- Data Security & Compliance: Must align with banking regulations and customer privacy laws.
- Budget Allocation: Branch-level purchasing power is limited; decision likely from head office.

The manager emphasized that integration with existing core banking systems would require technical review.

5. What recommendations would you provide for successful adoption?

Recommendations Provided:

- Start with pilot testing in selected urban branches
- Provide staff training sessions
- Offer dashboard analytics for branch performance tracking
- Keep infrastructure requirement minimal (no heavy hardware)
- Provide fallback manual system during transition phase
- Demonstrate cost-benefit analysis to head office

He concluded that if the system proves cost-effective and reduces customer complaints, adoption potential in the banking sector is strong.

In conclusion, the Branch Manager indicated that a digital queue management system like testing, regulatory compliance, and cost justification, such a solution has practical viability within A-class commercial banks in Nepal.

Figure 3 Proof of identification