

**A PROPOSED OFFERING OF AN ONLINE CLINIC APPOINTMENT
AND WALK-IN QUEUE MANAGEMENT SYSTEM FOR TONSUYA
SUPER HEALTH CENTER**

A Design Document Presented to the
Faculty of Datamex College of Saint Adeline, Inc.

In Partial Fulfillment of the Requirements for the
Degree of Bachelor of Science in Information Technology

By:

Roldan John Robert

Esternon, Danielle Joshua

Honi, Mark Kelly

Junio, Jeric

Malapitan, Junmar

INTRODUCTION

Going to a health center is something many people regularly do, whether for check-ups, consultations, or treatment. However, one of the most common issues patients encounter is the long waiting time before they can be assisted. At Tonsuya Super Health Center, patients often arrive very early in the morning just to secure a spot. Since the process is still mostly manual, where names are listed on paper and patients wait in crowded areas, it usually results in confusion, delays, and unnecessary stress. This situation is not only exhausting for patients but also challenging for healthcare workers who have to handle both the paperwork and patient care at the same time.

Because of this, there is a strong need for a solution that can make the process smoother and more organized. The proposed Online Clinic Appointment and Walk-In Queue Management System is designed to help both patients and staff. Instead of waiting in long lines, patients will now have two simple choices: book their appointment online in advance at a time that works best for them, or, if they prefer to walk in, register quickly through the system and receive a queue number. This way, they can easily monitor their turn, avoid overcrowding in waiting areas, and have a clearer idea of their waiting time.

For patients, this system means convenience and less stress. They no longer have to waste hours sitting and waiting without knowing when they will be called. They can come at the right time, spend less time in the clinic, and focus more on getting the healthcare services they need. It also gives patients the confidence that the process is fair and transparent since the queue will be managed digitally. This small change can have a big impact on how patients feel about their experience in the health center, making them more satisfied and more willing to seek medical help when needed.

On the other hand, the health center staff will also benefit from this system. Instead of spending too much time checking names, managing lists, and keeping track of who is next, the system will do all of these automatically. It can show the current status of each patient—whether they are waiting, being served, or already done making the flow more organized. It will also keep

records digitally, which reduces errors and avoids the problem of lost or incomplete paperwork. With fewer manual tasks, staff can focus more on giving quality service to the patients.

Another important part of this project is security and accuracy. Patient information, such as their appointment details and visit history, will be stored safely in the system and only authorized staff will have access to it. This ensures that the clinic complies with privacy laws while also improving the accuracy of patient data. Having digital records also makes it easier to look back on patient visits, track services, and even generate basic reports for the clinic. This way, the health center can make better decisions and prepare for busier times of the day or week.

In the long run, this system is more than just a tool to fix waiting times. It is a step toward modernizing the entire healthcare experience in Tonsuya. By using a digital system, the health center can serve more people in a faster, fairer, and more reliable way. Patients will appreciate the shorter waiting times and smoother process, while staff will find their work more manageable. The project can also serve as a foundation for future improvements, such as adding reminders, integrating telemedicine services, or expanding to other health centers in the city.

Overall, this proposal aims to bring comfort, order, and reliability to both patients and healthcare workers. By moving away from the old manual methods and embracing a modern, digital approach, Tonsuya Super Health Center can deliver a healthcare service that is not only more efficient but also more patient-friendly. This project represents a step forward in making local healthcare more accessible and trustworthy for the community it serves.

SYSTEM ARCHITECTURE

The system follows a three-tier architecture consisting of the client side, application layer, and database layer. The client side provides a web-based interface accessible to both patients and staff through mobile and desktop devices. The application layer hosts the central application server responsible for managing business logic, appointment scheduling, and queue tracking. The database layer ensures secure and reliable storage of patient records, appointments, queue numbers, and system logs.

High-Level Components and Interactions

- Patient Portal – Provides functionalities such as appointment booking, walk-in registration,
- Staff Dashboard – Enables staff to manage patient records, monitor queue progress, and update statuses accordingly.
- Admin Panel – Allows administrators to oversee system operations, manage staff accounts, and access system logs for auditing.
- Database – Stores and retrieves patient information, appointment details, and queue data to support system operations.
- Notification Module – Sends Scheduled alerts and reminders to patients a day before their scheduled appointments.

Deployment Architecture

The system adopts a cloud-based, web-based client–server model to provide accessibility and flexibility for both patients and staff. Users can access the application directly through standard web browsers on desktops, laptops, tablets, or smartphones without the need for additional installations. This ensures compatibility across multiple devices and allows the system to be available anytime and anywhere with an internet connection.

The system is hosted on a cloud infrastructure, utilizing Firebase services for backend operations such as authentication, hosting, and real-time synchronization. A NoSQL database (Firebase Realtime Database) is used to store structured and semi-structured data like patient information, appointments, queue records, and logs. This

design combines Firebase’s scalability with reliable data management, ensuring the system can accommodate more users and services as the health center grows.

Communication Protocols and Interfaces

- HTTPS – Ensures secure communication between clients and the server.
- RESTful APIs – Facilitates interaction between the client-side interface and the backend services.
- JSON – Used as the standard data format for efficient and lightweight data exchange.

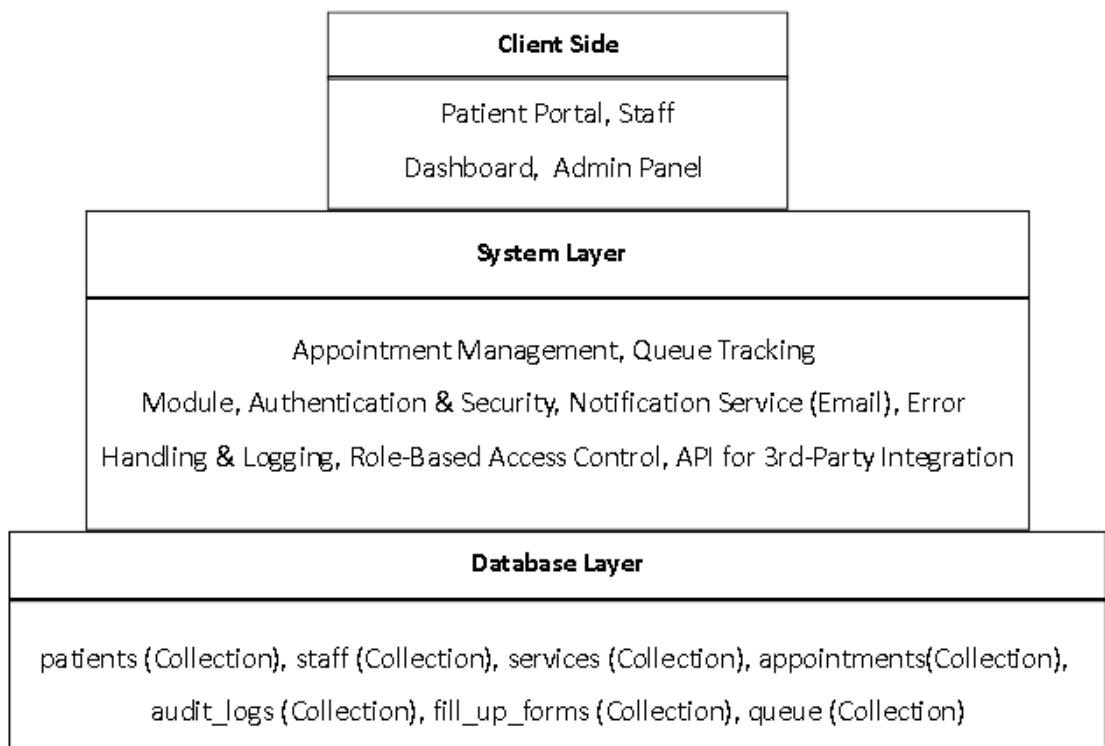


Figure 1: System Architecture

DATABASE DESIGN

Database design is the process of structuring and organizing data into a logical and efficient system that supports storage, retrieval, and management. It involves identifying entities, relationships, and constraints to ensure accuracy, consistency, and scalability. A well-designed database improves data integrity, minimizes redundancy, and enhances overall system performance.

Database Description

- **patients:** Stores patient information such as full name, email, phone number, date of birth, address, queue number, and references to services. It also includes a sub-map for booked_by details.
- **staff:** Contains staff details including full name, email, password, role, and created_at.
- **services:** Holds available services with their name, description, and duration in minutes.
- **appointments:** Connects patients, staff, and services through references. It includes appointment date, status, and created_at.
- **audit_logs:** Records system actions performed by patients or staff, including action type, IP address, and timestamp.
- **fill_up_forms:** Captures patient form submissions with references to patients, personal details, reason for visit, appointment date, booked_by details, and consent confirmation.

Entity-Relationship Model (ERM)

patients (Collection)

```
{patientId}
{
  "address": "string",
  "appointment_id": "string",
  "appointment_type": "string",
  "booking_source": "string",
  "created_at": "timestamp",
  "current_medications": "string",
```

```
"date_of_birth": "timestamp",  
"email": "string",  
"full_name": "string",  
"phone_number": "string",  
"pin_code": "string",  
"preferred_date": "timestamp",  
"priority_flag": "string",  
"service_ref": "reference",  
"sex": "string",  
"status": "string",  
"updated_at": "timestamp"  
}
```

staff (Collection)

```
{ staffId }  
{  
  "created_at": "timestamp",  
  "email": "string",  
  "full_name": "string",  
  "password": "string",  
  "role": "string"  
}
```

services (Collection)

```
{ serviceId }  
{  
  "created_at": "timestamp",  
  "description": "string",  
  "duration_minutes": "number",  
  "service_name": "string"  
}
```

appointments (Collection)

```
{ appointmentId }
```

```
{
  "appointment_date": "timestamp",
  "appointment_id": "string",
  "booked_by_name": "string",
  "booking_source": "string",
  "contact_number": "string",
  "created_at": "timestamp",
  "current_medications": "string",
  "email_address": "string",
  "patient_birthdate": "timestamp",
  "patient_full_name": "string",
  "patient_sex": "string",
  "present_checkbox": "boolean",
  "relationship_to_patient": "string",
  "service_ref": "reference"
}
```

audit_logs (Collection)

```
{logId}
{
  "action": "string",
  "email": "string",
  "role": "string",
  "staff_full_name": "string",
  "timestamp": "timestamp"
}
```

fill_up_forms (Collection)

```
{formId}
{
{
  "appointment_date": "timestamp",
  "appointment_id": "string",
  "booked_by_name": "string",
```



```
"booking_source": "string",
"contact_number": "string",
"created_at": "timestamp",
"current_medications": "string",
"email_address": "string",
"medical_history": "string",
"patient_birthdate": "timestamp",
"patient_full_name": "string",
"patient_sex": "string",
"present_checkbox": "boolean",
"relationship_to_patient": "string"
}
}
```

queue (Collection)

```
{date}{
  {queueId}
  {
    "address": "string",
    "appointment_type": "string",
    "booking_source": "string",
    "created_at": "timestamp",
    "date_of_birth": "timestamp",
    "email": "string",
    "full_name": "string",
    "phone_number": "string",
    "priority_flag": "string",
    "queue_number": "string",
    "service_ref": "reference",
    "status": "string"
```

```
}  
}
```

Data Normalization Techniques

The database design follows the principles of Third Normal Form (3NF) to ensure data consistency, reduce redundancy, and improve maintainability.

1. First Normal Form (1NF):

All attributes are atomic, meaning each field contains only a single value. For example, patient information such as `full_name`, `email`, and `phone_number` are stored in separate fields.

2. Second Normal Form (2NF):

Each non-key attribute is fully dependent on the primary key of the entity. Since every collection/document (e.g., patients, staff, services, appointments) uses a unique identifier (`patientId`, `staffId`, `serviceId`, etc.), there are no partial dependencies.

3. Third Normal

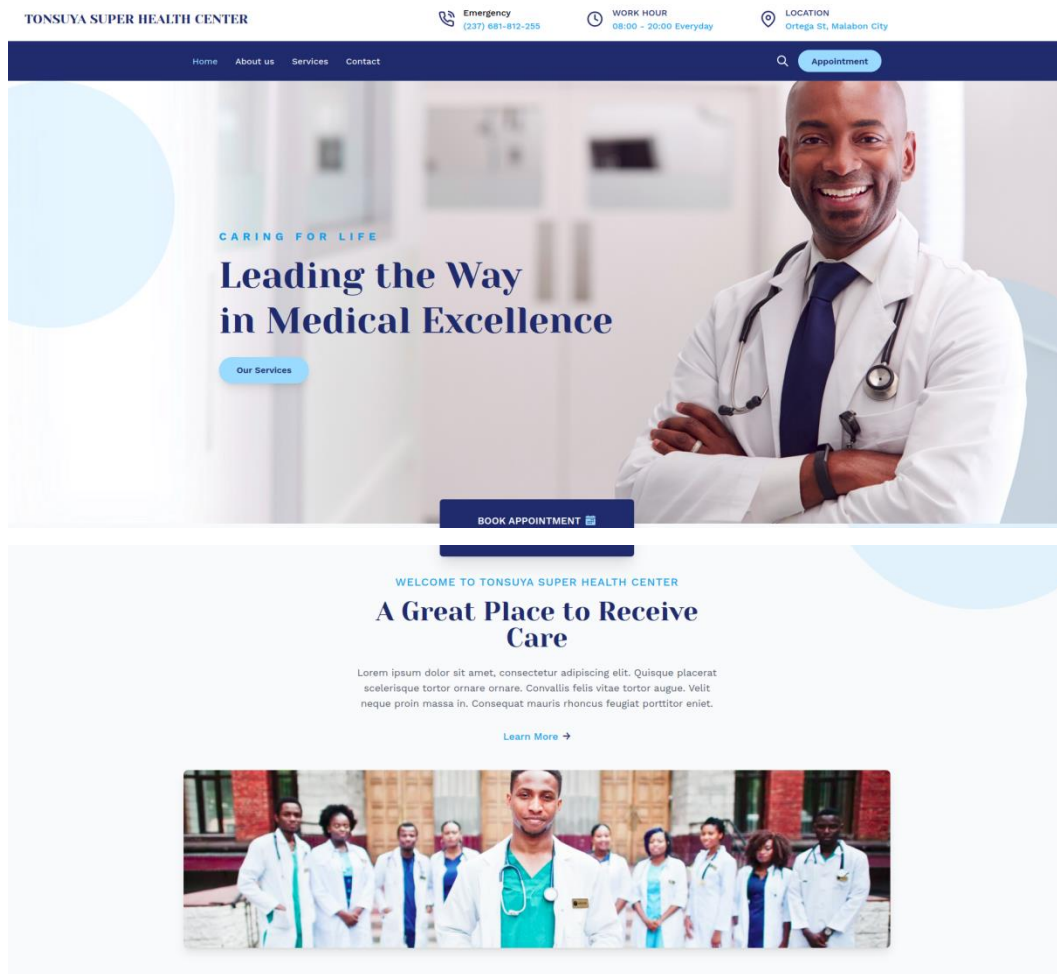
Transitive dependencies are eliminated by using references instead of duplicating data. For instance, appointment records only store references (e.g., `patient_ref`, `service_ref`, `staff_ref`) instead of repeating all patient, service, or staff details. Similarly, audit logs link to either patients or staff through `user_ref`, ensuring that updates to user details do not require changes in multiple places.

Form (3NF):

dependencies using references to avoid duplicating data. For example, an appointment record stores references to the patient, service, and staff instead of their full details.

USER INTERFACE DESIGN

The system's user interface is designed as a web-based platform that is simple, responsive, and user-friendly. It uses clearly arranged forms, buttons, and menus that work smoothly across different devices. Patients can conveniently book appointments and monitor their queue online, while staff can efficiently manage records through the web application.



CARE YOU CAN BELIEVE IN

Our Services

Medical Checkup

Maternal & Child Health

Immunization

Consultation



View All

A passion for putting patients first.

- Care for Your Little Ones
- All our best
- A Legacy of Excellence
- Care for Mothers
- Believe in Us
- Always Caring

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque placerat scelerisque tortor ornare ornare. Quisque placerat scelerisque tortor ornare ornare. Convallis felis vitae tortor augue. Velit nascetur proin massa in. Consequat faucibus porttitor enim et.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque placerat scelerisque. Convallis felis vitae tortor augue. Velit nascetur proin massa in.

ALWAYS CARING

Clinic Services

Medical Checkup

Maternal & Child Health

Immunization

Consultation

Family Planning

Senior Citizen Care

Wound Care
"Professional wound treatment"

Follow-Up Visit

Mental Health

Medical Certificates

CARE YOU CAN BELIEVE IN

Our Services

Medical Checkup

Maternal & Child Health

Immunization

Consultation



View All

A passion for putting patients first.

- Care for Your Little Ones
- All our best
- A Legacy of Excellence
- Care for Mothers
- Believe in Us
- Always Caring

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque placerat scelerisque tortor ornare ornare. Quisque placerat scelerisque tortor ornare ornare. Convallis felis vitae tortor augue. Velit nascetur proin massa in. Consequat faucibus porttitor enim et.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque placerat scelerisque. Convallis felis vitae tortor augue. Velit nascetur proin massa in.

ALWAYS CARING

Clinic Services

Figure 2.1. Landing Page

The Landing Page serves as the welcoming screen of the system. It introduces the Tonsuya Super Health Center's Online Clinic Appointment and Walk-In Queue Management System and provides quick access to the main features. From here, users can easily navigate to the appointment booking, walk-in registration, and

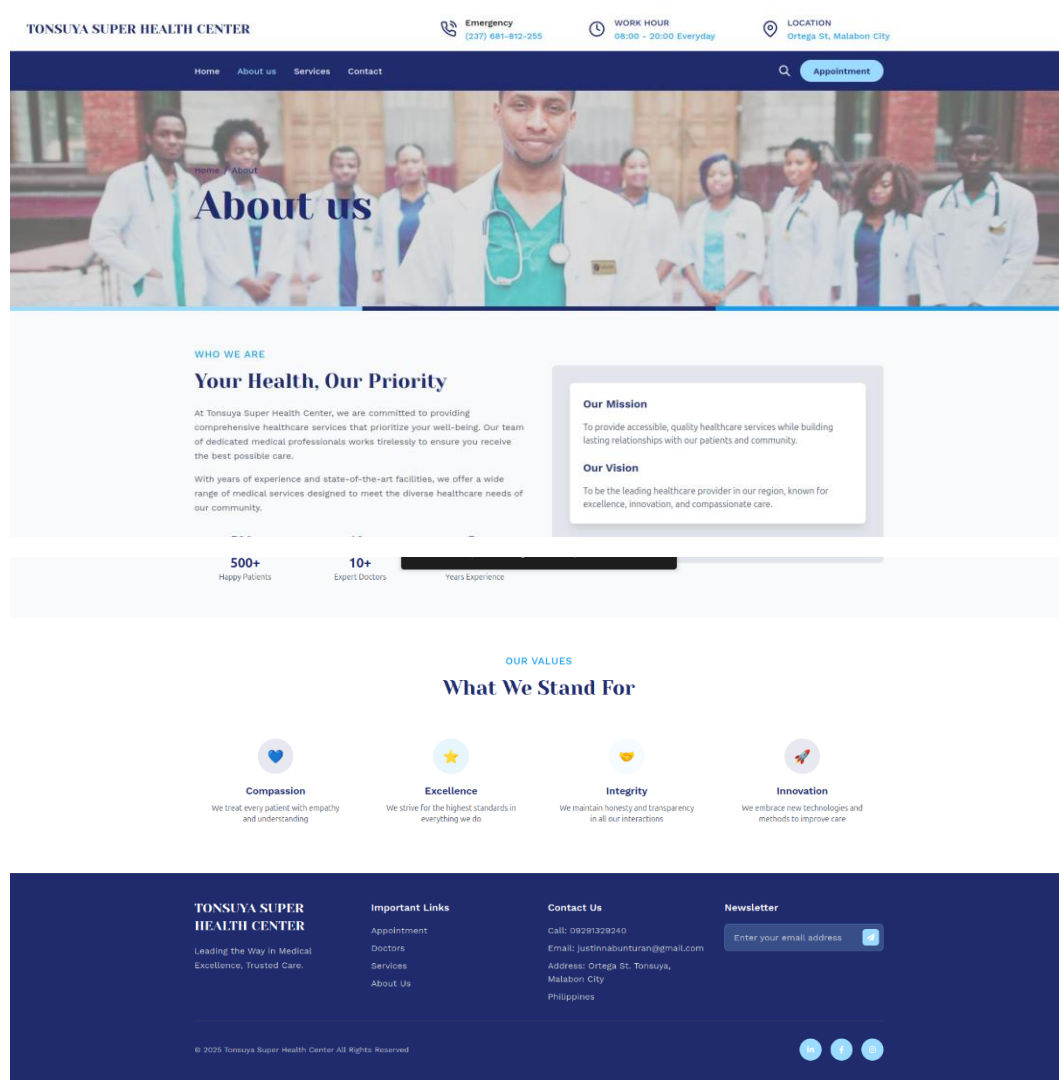


Figure 2.2 About Us Page

The About Us Page presents background information about Tonsuya Super Health Center. It highlights the mission, vision, and objectives of the health center, helping patients understand the purpose of the system and the values of the institution. This section ensures transparency and builds trust between the health center and the community it serves.

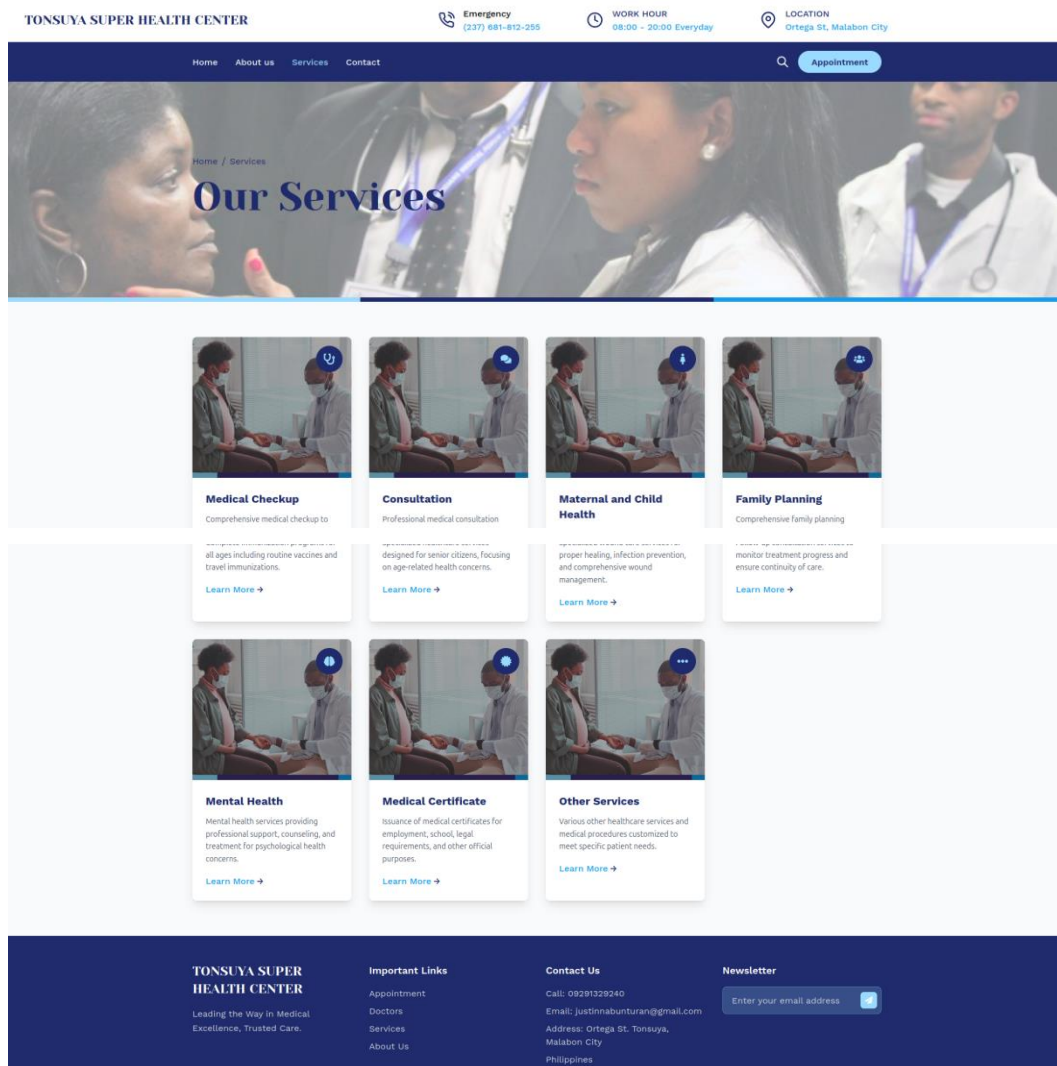


Figure 2.3 Our Service Page

The Our Service Page lists the available medical services offered by the health center. Each service includes a brief description and details such as expected duration, so patients can make informed decisions before booking an appointment or joining the walk-in queue.

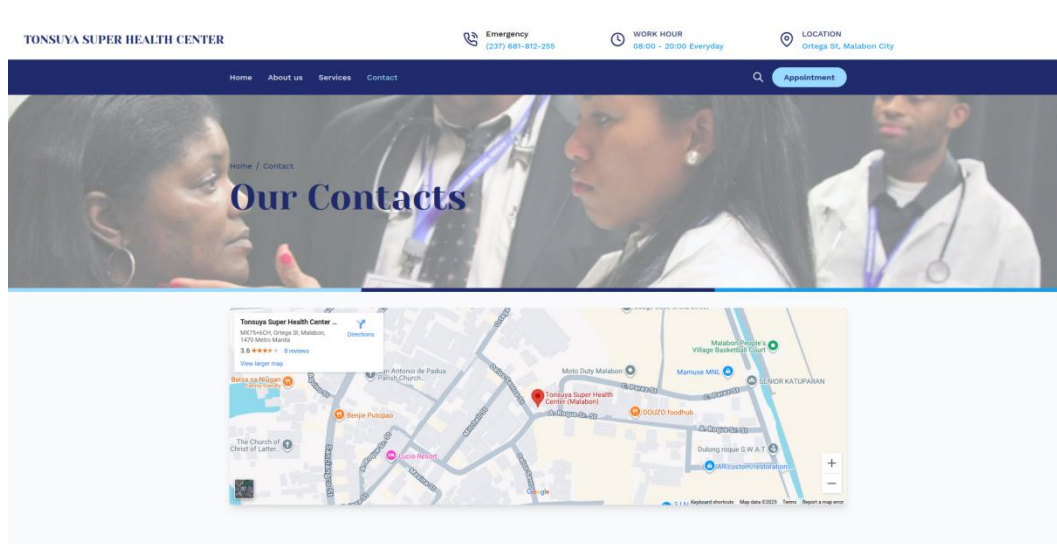


Figure 2.4 Contact Us Page

The Contact Us Page provides communication details and channels where patients can reach the health center. It typically includes phone numbers, email addresses, and a message form for inquiries. This ensures that patients can easily ask questions, request assistance, or provide feedback.

Day	Hours
Monday	09:00 AM - 07:00 PM
Tuesday	09:00 AM - 07:00 PM
Wednesday	09:00 AM - 07:00 PM
Thursday	09:00 AM - 07:00 PM
Friday	09:00 AM - 07:00 PM
Saturday	09:00 AM - 07:00 PM

Figure 2.5 Appointment Page

The Appointment Page allows patients to book their preferred schedule for consultation or treatment. It shows available slots, required details, and confirmation options. This feature minimizes waiting time and ensures that patients can plan their visits conveniently.

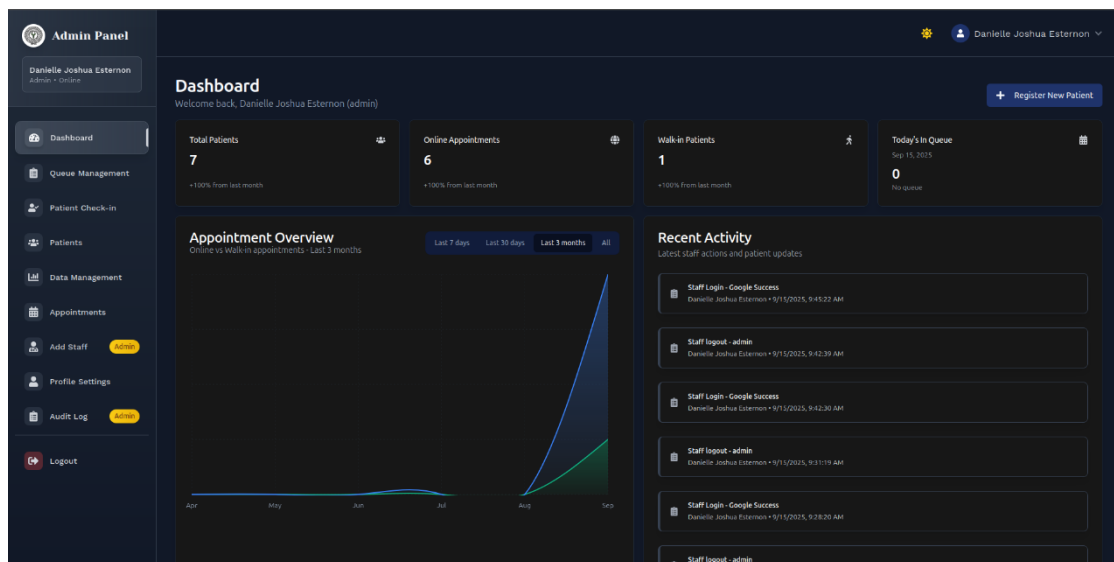


Figure 2.6 Admin/Staff Dashboard

The Dashboard Module provides administrators with a real-time overview of patient records, appointment statistics, and staff activities. It displays the total number of patients, online and walk-in appointments, and the current queue. It also includes an appointment overview graph for the last three months and a recent activity log to track staff actions such as logins and logouts.

The screenshot shows the 'Add New Staff' form in the Admin Panel. The form is titled 'Add New Staff' and includes a subtitle 'Register a new staff member for your clinic system.' The form fields are: Full Name (with a user icon), Email (with an email icon), Password (with a lock icon), Confirm Password (with a lock icon), and Role (a dropdown menu currently set to 'Admin'). A 'Send Verification PIN' button is at the bottom. The sidebar on the left shows the 'Add Staff' option highlighted. The URL at the bottom left is 'http://localhost:5173/admin/add-staff'.

Figure 2.8 Admin/ Add Staff

The Add Staff Module allows administrators to register new staff members into the system. When accessed, it displays a form on the right side where the admin can input the staff's full name, email, and password, confirm the password, and assign a role. To complete the process, the system requires sending a verification PIN to validate

the registration. This ensures that only authorized staff can be added and assigned with proper access privileges.

The screenshot shows the 'Admin Panel' on the left sidebar with the 'Profile Settings' option highlighted. The main content area is titled 'Profile Settings' with a subtitle 'Update your profile information securely.' It contains three input fields: 'Full Name' (pre-filled with 'Danielle Joshua Esternon'), 'Email' (pre-filled with 'daniellejoshua18@gmail.com'), and 'Old Password' (placeholder 'Enter your current password'). Below these is a 'New Password' field with a placeholder 'New Password (min 8 characters)'. At the bottom is a blue button labeled 'Update Profile'.

Figure 2.9 Admin/Staff Profile Settings

The Profile Settings Module allows users to securely update their account information. It provides fields to edit the full name and email, as well as options to change the password by entering the current password and setting a new one. Once updated, the changes are saved to maintain accurate and secure profile details.

The screenshot shows the 'Admin Panel' on the left sidebar with the 'Audit Log' option highlighted. The main content area is titled 'Audit Logs' with a subtitle 'Track system activities and user actions'. It includes a search bar 'Search by Staff Name' with a placeholder 'Staff full name...', an 'Action Type' dropdown set to 'All Actions', a 'Date' dropdown set to 'mm / dd / yyyy', and a 'Sort By' dropdown set to 'Newest First'. Below the filters, it says 'Showing 27 of 27 logs'. The table below lists the audit logs.

USER	ACTION	DATE & TIME
Danielle Joshua Esternon	Staff Login - Google Success	Sep 15, 2023, 09:43:22 AM
Danielle Joshua Esternon	Staff logout - admin	Sep 15, 2023, 09:42:39 AM
Danielle Joshua Esternon	Staff Login - Google Success	Sep 15, 2023, 09:42:30 AM
Danielle Joshua Esternon	Staff logout - admin	Sep 15, 2023, 09:31:19 AM
Danielle Joshua Esternon	Staff Login - Google Success	Sep 15, 2023, 09:28:20 AM
Dr. Maria Santos	Staff logout - admin	Sep 15, 2023, 09:19:27 AM
Danielle Joshua Esternon	Staff logout - admin	Sep 15, 2023, 09:14:37 AM
Danielle Joshua Esternon	Staff Login - Google Success	Sep 15, 2023, 12:37:53 AM

Figure 2.10 Admin Audit Log

The Audit Log Module records and tracks all system activities and user actions for monitoring and security purposes. It displays a list of logs containing the user's name, the action performed, and the corresponding date and time. The module also

includes filters for staff name, action type, and date, as well as sorting options to easily manage and review logs. Additionally, it provides an option to export the records in PDF format for reporting and documentation.

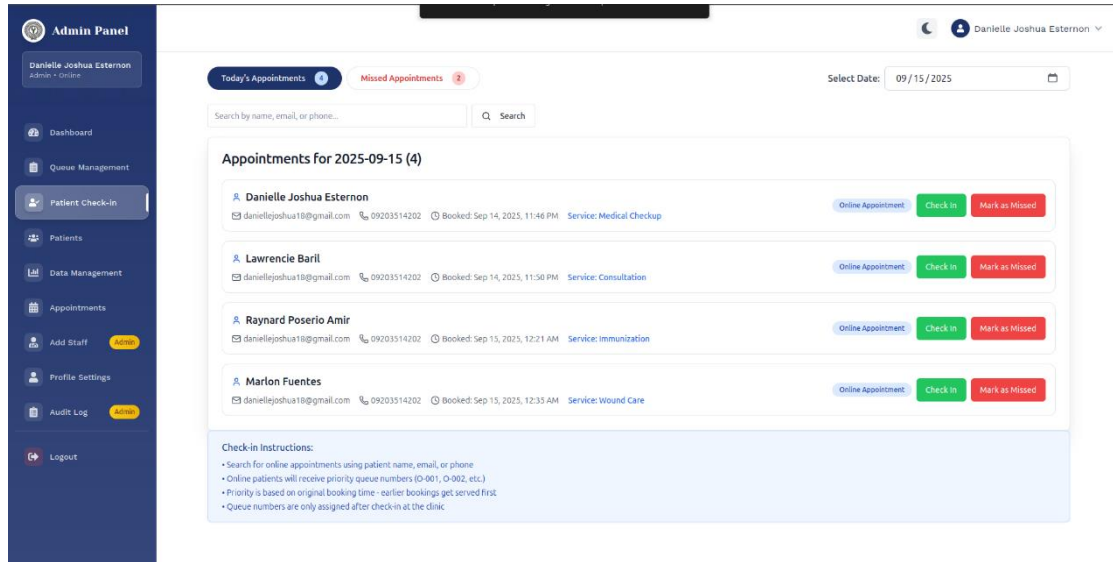


Figure 2.11 Admin/Staff Patient Check-in

The Patient Check-in page allows the admin to manage scheduled appointments by checking in patients or marking them as missed, with details such as name, email, contact number, date, time, and service displayed, along with a search bar and date selector for easy navigation. On the sidebar, the system provides access to the dashboard, queue management, patient records, data management, appointments, staff management, profile settings, audit logs, and logout options.

Admin Panel
Danielle Joshua Esterson
Admin • Online

Dashboard
Queue Management
Patient Check-in
Patients
Data Management
Appointments
Add Staff **Active**
Profile Settings
Audit Log **Active**
Logout

Appointments

Search by name, email, or phone | All Status | All Services | All Types | Sort by date: Oldest | Newest | PDF Report

PATIENT INFO	TYPE	SERVICE	DATE	STATUS	ACTIONS
M Marlon Fuentes danieljoshua18@gmail.com 09203514202	Online	Wound Care Ref: service/O_7JPHO9U9HPCPBAnd	9/15/2025, 12:35:13 AM	scheduled	View Reschedule
T Tamiea Lim danieljoshua18@gmail.com 09203514202	Online	Immunization Ref: service/O_7JPHO9U9HPCPBAnd	9/15/2025, 12:22:20 AM	missed	View
R Reynard Poserio Amir danieljoshua18@gmail.com 09203514202	Online	Immunization Ref: service/O_7JPHO9U9HPCPBAnd	9/15/2025, 12:21:15 AM	scheduled	View Reschedule
D Danielle Joshua Esterson danieljoshua18@gmail.com 09203514202	Online	Mental Health Ref: service/O_7JPHO9U9HPCPBAnd	9/14/2025, 11:55:25 PM	missed	View
L Lawrence Baril danieljoshua18@gmail.com 09203514202	Online	Consultation Ref: service/O_7JPHO9U9HPCPBAnd	9/14/2025, 11:50:12 PM	scheduled	View Reschedule
R Reynard Poserio amirposerio@gmail.com 09203514202	Walk-in	Follow-Up Visit Ref: service/O_7JPHO9U9HPCPBAnd	9/14/2025, 11:48:01 PM	In progress	View
D Danielle Joshua Esterson danieljoshua18@gmail.com 09203514202	Online	Medical Checkup Ref: service/O_7JPHO9U9HPCPBAnd	9/14/2025, 11:46:53 PM	scheduled	View Reschedule

http://localhost:5173/admin/appointments

Figure 2.12 Admin/Staff Appointments

The Appointments page displays a list of all patient appointments with details such as name, email, contact number, appointment type, service, date, and status, while also providing options to view or reschedule each entry. It includes filters for status, service, and type, along with a search bar and sorting feature for easy tracking, and allows generating a PDF report of the records.

Admin Panel
Danielle Joshua Esterson
Admin • Online

Dashboard
Queue Management
Patient Check-in
Patients
Data Management
Appointments
Add Staff **Active**
Profile Settings
Audit Log **Active**
Logout

Total Patients: 7

Search patients by name, email, or phone... | PDF Report

PATIENT INFO	PHONE NUMBER	SERVICE (REFERENCED)	APPOINTMENTS	ACTIONS
D Danielle Joshua Esterson danieljoshua18@gmail.com	09203514202	Medical Checkup Ref: service/O_7JPHO9U9HPCPBAnd	1 appointment(s) Latest with: Unassigned	
R Reynard Poserio amirposerio@gmail.com	09203514202	Follow-Up Visit Ref: service/O_7JPHO9U9HPCPBAnd	1 appointment(s) Latest with: Unassigned	
L Lawrence Baril danieljoshua18@gmail.com	09203514202	Consultation Ref: service/O_7JPHO9U9HPCPBAnd	1 appointment(s) Latest with: Unassigned	
D Danielle Joshua Esterson danieljoshua18@gmail.com	09203514202	Mental Health Ref: service/O_7JPHO9U9HPCPBAnd	1 appointment(s) Latest with: Unassigned	
R Reynard Poserio Amir danieljoshua18@gmail.com	09203514202	Immunization Ref: service/O_7JPHO9U9HPCPBAnd	1 appointment(s) Latest with: Unassigned	
T Tamiea Lim danieljoshua18@gmail.com	09203514202	Immunization Ref: service/O_7JPHO9U9HPCPBAnd	1 appointment(s) Latest with: Unassigned	
M Marlon Fuentes danieljoshua18@gmail.com	09203514202	Wound Care Ref: service/O_7JPHO9U9HPCPBAnd	1 appointment(s) Latest with: Unassigned	

http://localhost:5173/admin/patients

Figure 2.13 Admin/Staff Patients

The Patients page provides a complete list of registered patients with details such as name, email, phone number, referenced service, and the number of appointments linked to each patient. It includes a search bar for quickly locating patient records and a PDF report option for generating summaries.

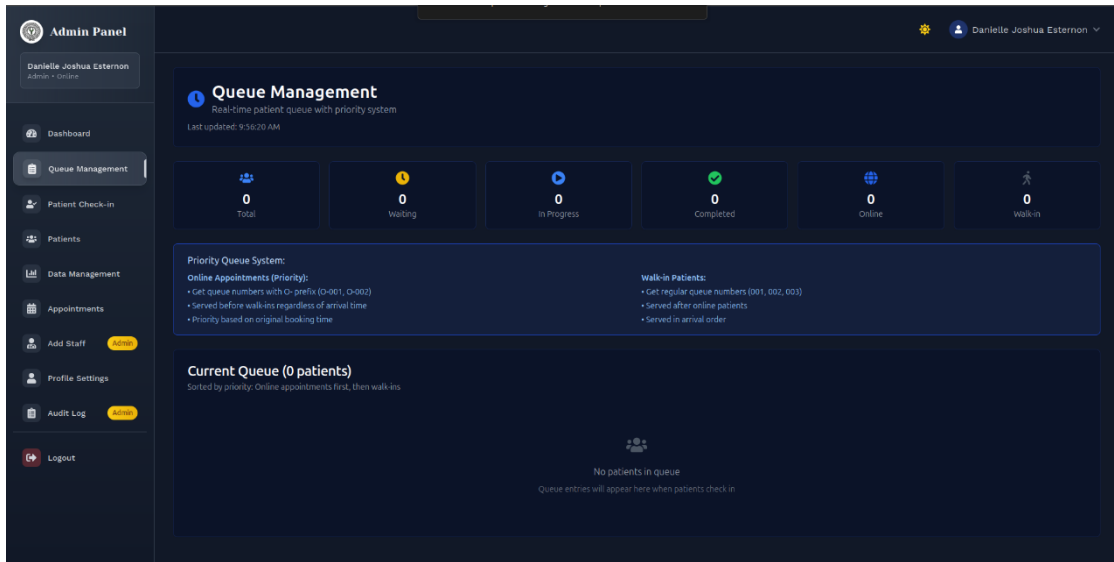


Figure 2.14 Admin/Staff Queue Management

The Queue Management page displays the real-time status of patients in the queue with categories such as total, waiting, in progress, completed, online, and walk-in. It uses a priority system where online appointments are prioritized based on booking time, while walk-in patients are queued in arrival order. The page shows the current queue list, which updates when patients check in, and provides clear rules for assigning queue numbers.

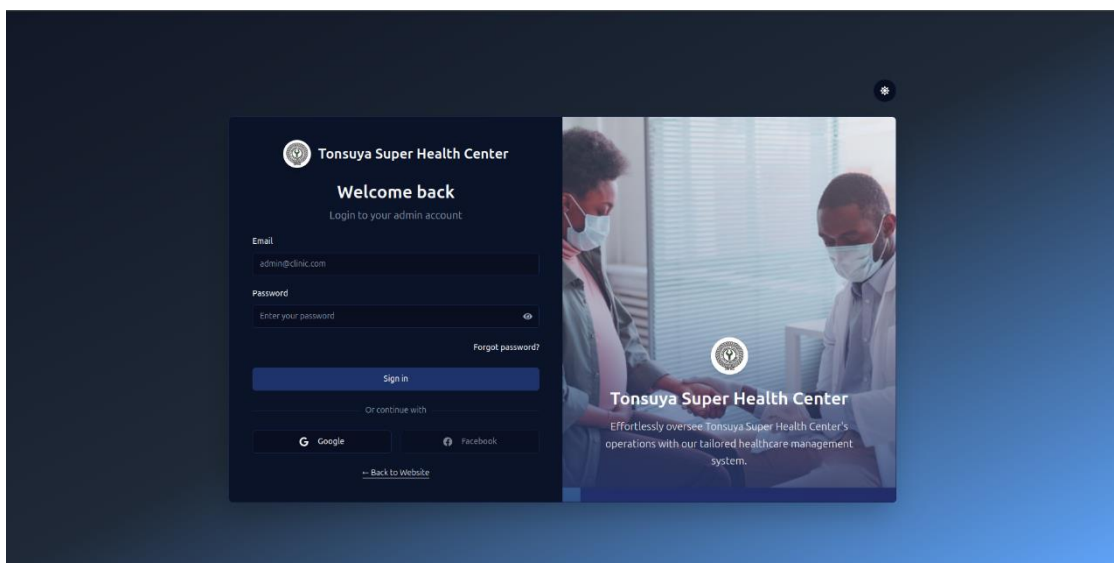


Figure 2.15 Admin/Staff Login Page

The Login page provides secure access for administrators to the Tonsuya Super Health Center management system, requiring an email and password to sign in.

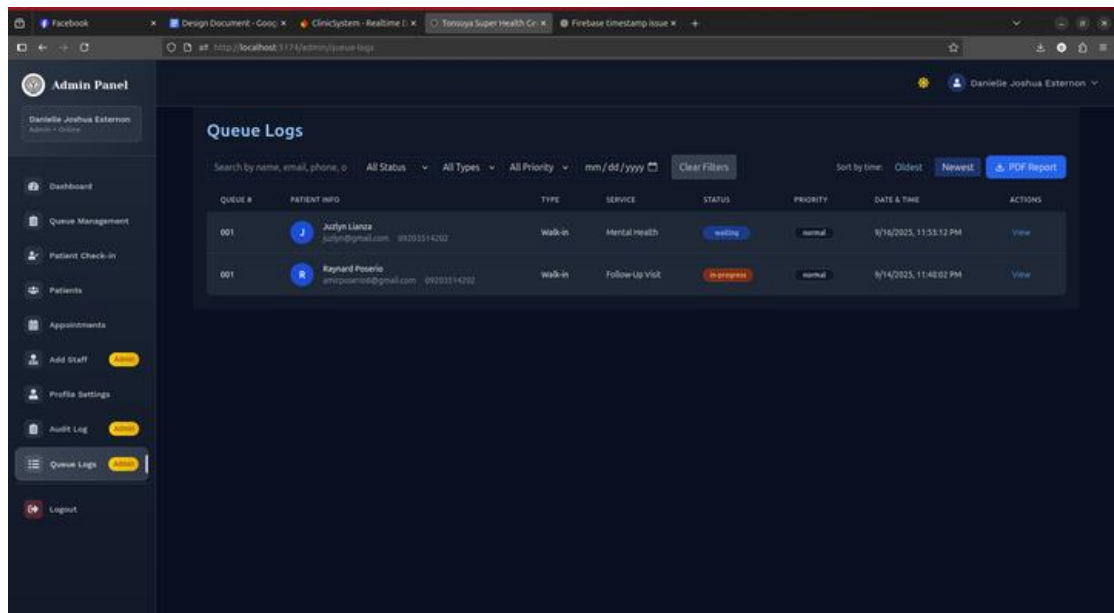


Figure 2.16 Admin Queue Logs

The Admin Queue Logs display a record of all queue-related activities within the system. It allows administrators to monitor patient queue status, including check-ins, updates, and staff actions, ensuring transparency and accountability in the management of daily operations.

COMPONENT DESIGN

The component design outlines the system's architecture by dividing it into individual modules or components. Each component specifies its purpose, inputs, outputs, and interaction with other components. This ensures that the system is organized, maintainable, and scalable. The design provides detailed descriptions of user interface components, database components, business logic, and external interfaces. By clearly defining the responsibilities of each component, the system can be developed systematically, tested independently, and integrated efficiently.

Key Components and Modules

- Appointment Booking Module – Provides patients with the ability to schedule appointments in advance, view available time slots, and receive confirmation notifications.
- Walk-in Queue Module – Handles registration of walk-in patients and automatically assigns queue numbers for efficient service flow.
- Queue Monitoring Module – Enables both patients and staff to track real-time queue progress, ensuring transparency and reducing waiting time uncertainty.
- Authentication Module – Manages secure login and role-based access control for patients, staff, and administrators.
- Reporting Module – Generates detailed reports on appointments, patient records, and queue statistics to support administrative decision-making.

Interface Specifications

The system utilizes RESTful APIs to manage communication between the client and the server. These APIs handle critical functions such as appointment scheduling and queue management, ensuring smooth and reliable operations. To secure the platform, an Authentication API is integrated to manage user login, session handling, and role-based access control. All data exchanged between components follows a standardized JSON format, which guarantees consistency and platform independence. This design ensures that the system remains scalable, secure, and easily interoperable with future modules or third-party integrations.

Dependency Management

The architecture is designed with modular components that communicate primarily through API calls, enabling clear separation of functions and promoting system flexibility. This modularity allows individual components to be updated, replaced, or scaled without disrupting the overall system. The design also supports scalability, ensuring that the system can accommodate increasing users and services

as the health center grows. A centralized database is implemented as the single source of truth, providing consistent and reliable data access across all modules. This centralized approach ensures data integrity, synchronization, and accuracy, reducing redundancy and potential errors.

DATA FLOW DIAGRAM

In this study, Data Flow Diagrams (DFDs) are used to show how data moves through the system. The diagrams help illustrate the interaction between different components, such as data sources, processing steps, and destinations. By using DFDs, we can easily understand how information is collected, processed, and stored, as well as how it flows to generate outputs. This makes it easier to identify system requirements and ensures that the system's data handling is efficient and organized

Level 0 Data Flow Diagram

The Level 0 DFD, or context diagram, shows the system as a single process interacting with external entities. Patients provide appointment requests or walk-in details, while staff and administrators manage records and queues. The system connects to the database to store and retrieve information, returning outputs such as confirmations, queue numbers, and reports.

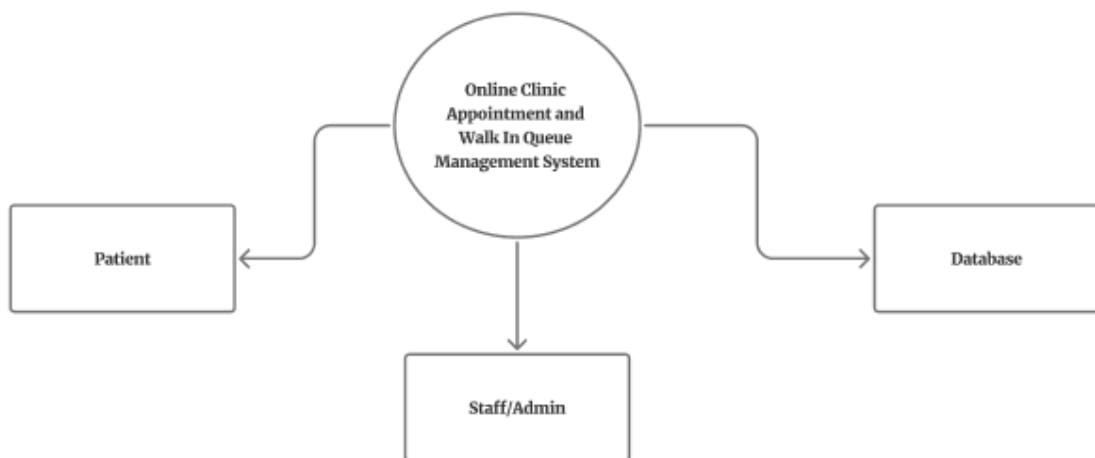
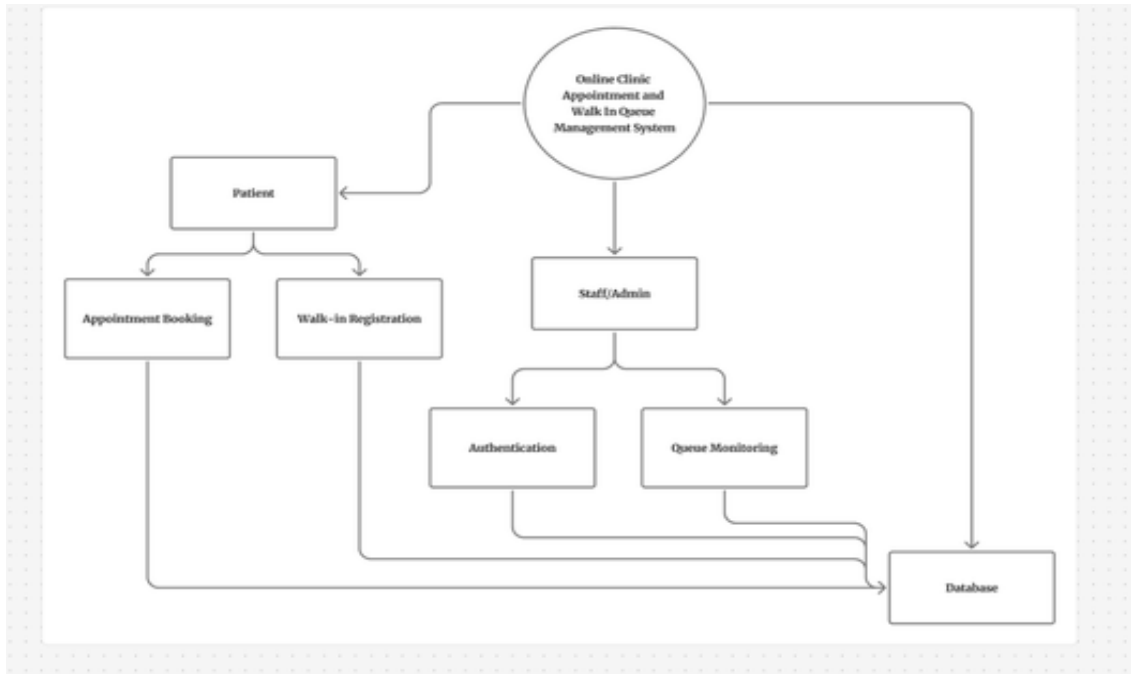


Figure 3.1. Data Flow Diagram

Level 1 Data Flow Diagram

The Level 1 DFD breaks down the main process into sub-processes. Patients can book appointments online or register as walk-ins, with data stored in the database. Staff and administrators handle authentication and queue monitoring, both of which



rely on the database to ensure accurate and real-time information flow.

Figure 3.2. Data Flow Diagram

SECURITY DESIGN

The system is designed with strict security requirements to ensure the confidentiality, integrity, and availability of all data it processes. This includes sensitive information such as patient records, appointments, queue data, and system audit logs. Security considerations are anchored on the principles of least-privilege access, accountability through audit trails, and preventive measures against unauthorized access. Additionally, compliance with relevant privacy regulations and institutional policies is observed, particularly in relation to user consent, data retention, and access rights.

Authentication and Authorization Mechanisms

Authentication is implemented using Firebase Authentication, which supports secure login through verified email and password credentials. Tokens are revoked in cases of password resets or role changes to prevent unauthorized access.

Authorization ensures that users only access data relevant to their role. Patients are limited to booking appointments. Staff members are restricted to operational data required for service delivery, while administrators hold privileges to manage users, system services, and audit logs. Access control enforcement is achieved through Firebase Security Rules in conjunction with server-side validation..

Data Encryption and Protection Measures

All data transmitted between the client and server is protected through Transport Layer Security (TLS), ensuring confidentiality and integrity in transit. Strict HTTPS enforcement and HTTP Strict Transport Security (HSTS) are also applied. At rest, data stored within Firebase Firestore, Realtime Database, and Storage is secured using Firebase's built-in encryption mechanisms.

System secrets such as API keys are securely stored in environment variables and rotated periodically to reduce the risk of exposure. Input validation and output encoding mechanisms are employed to mitigate common security threats such as injection attacks. Furthermore, error handling is structured to avoid disclosure of sensitive information. Audit logs are maintained in an append-only format with user

activity metadata, while regular data backups and restoration procedures are conducted to ensure business continuity. Finally, data retention and deletion policies are defined in accordance with institutional guidelines and privacy standards.

PERFORMANCE DESIGN

The system must deliver high-quality performance in terms of speed, reliability, and security. It aims to reduce patient waiting time through automated appointment booking and queue management, ensuring patients spend less time waiting and more time receiving care. The system should handle both online and walk-in patients seamlessly, maintaining real-time synchronization to prevent conflicts or double bookings. Additionally, it should improve staff productivity by automating repetitive tasks, such as scheduling, queue assignment, and report generation, allowing staff to focus on critical responsibilities. Finally, the system must ensure the confidentiality and integrity of sensitive data using secure authentication, encryption, and role-based access controls.

Strategies for Optimizing System Performance

- **Caching:** Store frequently accessed data in cache memory to reduce database queries and improve response time.
- **Load Balancing:** Distribute user requests across multiple servers to prevent bottlenecks during high demand.
- **Database Optimization:** Use composite indexes, efficient query patterns, and data structuring (including denormalization when needed) to minimize reads and ensure fast performance in a NoSQL environment like Firebase.
- **Scalability:** Support both vertical and horizontal scaling to accommodate future growth.
- **Asynchronous Processing:** Handle non-critical operations like reports or notifications in the background to avoid delays in real-time interactions.
- **Resource Monitoring:** Continuously track CPU, memory, and network usage to optimize performance and address issues proactively.

Performance Testing Plan

- **Load Testing:** Simulate multiple concurrent users to measure response time, throughput, and resource utilization.
- **Stress Testing:** Test beyond expected limits to identify bottlenecks and ensure graceful failure and recovery.

- Scalability Testing: Gradually increase users and transactions to confirm the system can scale without performance degradation.
 - Reliability and Security Testing: Assess the impact of authentication, encryption, and access controls on system efficiency.
- Monitoring and Benchmarking: Track key performance metrics and compare them to benchmarks to ensure requirements are met.

ERROR HANDLING AND LOGGING

The system will implement structured error handling and robust logging practices to maintain reliability, security, and transparency. Errors will be detected, caught, and handled in a way that prevents system crashes or interruptions in core functions such as appointment booking, queue generation, and user authentication. Whenever an exception occurs, the system will log the event with detailed technical information for administrators, while showing the user a simple, clear, and non-technical message that guides them on how to resolve the issue.

All critical operations, including login attempts, appointment scheduling, and database transactions, will have built-in safeguards such as retry mechanisms, validation checks, and fallback procedures to minimize disruption. For example, if an appointment booking request fails due to a database timeout, the system will attempt a retry and log the event before notifying the user.

Logging will be tightly integrated with error handling to ensure that every error or significant system event is properly recorded. System logs will include timestamps, error codes, severity levels (Info, Warning, Error, Critical), and contextual details for troubleshooting. Logs will be securely stored in the database and periodically archived to support performance monitoring, incident response, and long-term analysis. Sensitive information, especially patient-related data, will never be exposed in logs to comply with privacy and security standards.

Together, structured error handling and secure logging create a reliable foundation for system maintenance, allowing developers and administrators to quickly identify, investigate, and resolve issues while ensuring that users experience minimal disruption.

Error Codes and Messages

- **404 – Not Found:** The requested resource, such as a webpage or API endpoint, could not be found. This usually happens due to an incorrect URL or the resource no longer existing.
- **403 – Forbidden:** The server understood the request but refuses to authorize it, often because the user lacks sufficient permissions.

- **503 – Service Unavailable:** The server is temporarily unable to handle the request, commonly due to maintenance or being overloaded.

THIRD-PARTY INTEGRATIONS

The system connects with Firebase for authentication and NoSQL database management, integrates Email APIs (Email.js) for patient notifications, and uses cloud hosting services for secure deployment and scalability.

- **Firebase Authentication** – Provides secure login and session management. It ensures that only authorized users can access specific system modules through role-based access control. It also supports password reset, two-factor authentication, and token validation.
- **Firebase Realtime Database / Firestore (NoSQL Database)** – Serves as the main storage for patient records, appointment schedules, queue numbers, and user information. The NoSQL structure allows flexible data modeling, while real-time synchronization ensures both patients and staff always see the latest updates across devices.
- **Email (Email.js)** – Handles automated notifications. Patients receive appointment reminders, queue updates, and confirmation messages. This integration helps reduce no-shows and improves patient engagement.
- **Cloud Hosting Service (Firebase Hosting or equivalent)** – Provides a secure and scalable environment for web application deployment. Hosting includes SSL certificates, automated scaling, and a global CDN for optimized performance.

Integration Points and Data Exchange Formats

- **Authentication** – The system makes use of Firebase Authentication through REST APIs to handle user access. When users log in, their credentials are verified by the authentication service, ensuring that only authorized individuals can proceed. This process strengthens system security by preventing unauthorized entry and keeping sensitive records safe.
- **Database Access** – The system communicates with a secure cloud-hosted database via HTTPS requests. Data is structured and exchanged in JSON format, allowing smooth interaction with the web-based platform. To ensure integrity and confidentiality, the database enforces security rules

and role-based permissions so that only authorized users can view or modify information.

- Notification Services – The system integrates with Email.js to send notification messages to patients. These notifications include appointment confirmations, rescheduling updates, and real-time queue progress. By sending timely updates to the patient's registered email address, the system reduces waiting uncertainty and strengthens communication between the clinic and its patients.
- Hosting and Deployment – Application code, assets, and configurations are deployed to Firebase Hosting. Continuous deployment pipelines manage version control, while auto-scaling features adjust resources based on demand. The integrated CDN accelerates content delivery for users across different regions.

DEPLOYMENT PLAN

The deployment of the system will follow a structured life cycle consisting of four primary phases: development, testing, staging, and production. During the development phase, new features and updates are implemented in a controlled environment. The testing phase ensures that all modules are validated against functional and non-functional requirements. Once verified, the application proceeds to the staging environment, which mirrors the production setup and serves as the final validation layer before live deployment. Finally, in the production phase, the application is deployed to a cloud-based environment using Firebase Hosting. To ensure reliability and efficiency, Continuous Integration and Continuous Deployment (CI/CD) pipelines will be implemented. These pipelines will automate build, test, and deployment processes, minimizing downtime and human error while ensuring consistent updates.

Hardware and Software Requirements for Deployment

Hardware:

The system will rely on cloud-based infrastructure, eliminating the need for on-premises servers. Firebase Hosting and associated cloud services will handle scalability and performance requirements.

Load balancing and auto-scaling will be managed by the hosting provider to ensure optimal performance during peak usage.

Software:

- Firebase Hosting – For web application deployment, offering SSL encryption, scalability, and CDN integration.
- Firebase Authentication – Provides secure login, role-based access, and user identity management.
- NOSQL Database (Cloud-hosted) – Centralized repository for patient data, appointment records, and queue management.
- Modern Web Browsers (Chrome, Firefox, Edge) – Required for end-user access, ensuring compatibility and responsiveness.
- CI/CD Tools (GitHub Actions) – Automates testing and deployment, ensuring stable updates to production.

Configuration Management and Version Control Procedures

- Version Control – The source code will be managed using a GitHub repository with a defined branching strategy:
 - Main Branch – Stable codebase for production deployment.
 - Develop Branch – Used for staging and integration testing.
 - Feature Branches – Isolated environments for implementing and testing new features.
- Release Management – Automated versioning will be used to track updates, with release tags applied to each stable version. This ensures that rollback procedures can be easily executed if necessary.
- Configuration Management – All configuration files will be stored securely, with environment variables used for sensitive information such as API keys, database credentials, and authentication tokens. This approach prevents exposure of critical data in the source code. Access control and permissions will also be implemented to ensure only authorized personnel can modify deployment configurations.
- Backup and Recovery – Regular database backups will be configured within the cloud environment to ensure business continuity in the event of unexpected failures or data corruption.

MAINTENANCE AND SUPPORT

The system will follow a proactive maintenance and support strategy to ensure availability, reliability, and security at all times. Regular maintenance checks will be performed to verify system performance, optimize database operations, and confirm that all modules function as intended. Preventive maintenance schedules will be established to reduce the likelihood of downtime and minimize disruptions to end-users. Additionally, system monitoring tools and logging mechanisms will be utilized to continuously track performance metrics, detect anomalies, and generate alerts for potential issues. A dedicated support process will be in place to handle technical inquiries, user-reported problems, and service requests efficiently.

Procedures for Handling Software Updates, Patches, and Bug Fixes

- Minor Updates and Patches – Deployed as needed to address security vulnerabilities, correct bugs, and maintain system stability. These updates will follow an expedited testing and deployment cycle to ensure rapid resolution of issues without compromising functionality.
- Major Updates – Scheduled on a quarterly basis and may include feature enhancements, performance improvements, and architectural upgrades. Major releases will undergo a full testing process across development, staging, and production environments before rollout.
- Update Process – All updates, whether minor or major, will adhere to the established CI/CD pipeline and version control procedures. This ensures consistency, traceability, and the ability to roll back changes if unforeseen issues arise.

Support Framework

- Helpdesk Support – A support team will be available to provide assistance to users through email or ticketing systems. Response times will be defined based on issue severity (e.g., critical errors resolved within hours, minor issues within days).
- Documentation and Training – Comprehensive user manuals and technical documentation will be maintained and updated alongside new releases. Training sessions may be provided to staff to ensure smooth adoption of new features.

- Backup and Recovery Plan – Regular database backups will be maintained, with defined recovery procedures to restore services in the event of unexpected failures.
- Security and Compliance – Routine security audits, vulnerability assessments, and compliance checks will be performed to safeguard sensitive data and maintain regulatory compliance

REVISION HISTORY

This section documents all changes made to the design document over time. Each revision entry includes the version number, date of update, and a summary of modifications. It ensures proper tracking of progress, accountability, and clarity in the evolution of the system's design.

Version	Date	Change Made
1.0	Aug 17, 2025	Prepared the first draft of the document
1.1	Aug 18, 2025	Added the system requirements and use case details
1.2	Aug 19, 2025	Included data flow diagrams and outlined the security setup
1.3	Aug 21, 2025	Updated the database design and revised the queue process
1.4	Aug 22, 2025	Adjusted the design of the interface. A few reporting functions were included.
1.5	Aug 24, 2025	Made changes to the diagrams. Some parts of the layout were also improved.
1.6	Aug 25, 2025	Added the testing plan and corrected minor inconsistencies
1.7	Aug 26, 2025	Reviewed the content and refined the formatting
1.8	Aug 27, 2025	

Table 1: Revision History

APPENDIX

The Appendix section provides supporting materials that complement the main body of the design document. It includes additional diagrams, reference resources, and other relevant information that may not fit within the primary sections but are essential for better understanding the system. The appendix ensures completeness by presenting visual aids such as flowcharts and database schemas, as well as technical references and research sources used during the development of the system. This section improves clarity and allows readers to explore supplementary details without interrupting the flow of the main content.

Appendix A - Supporting Diagrams

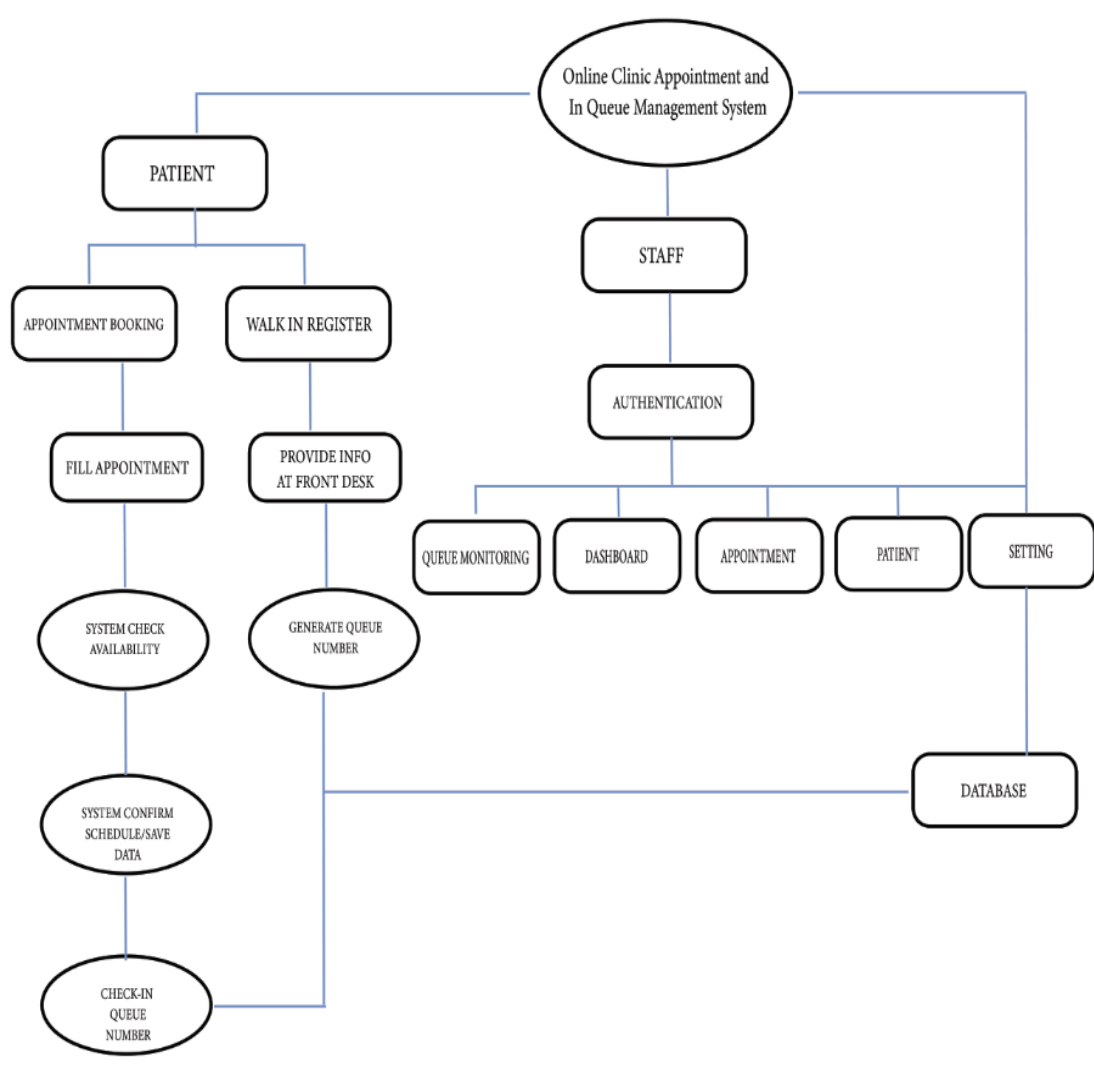


Figure A.1: Data Flow Diagram of the system

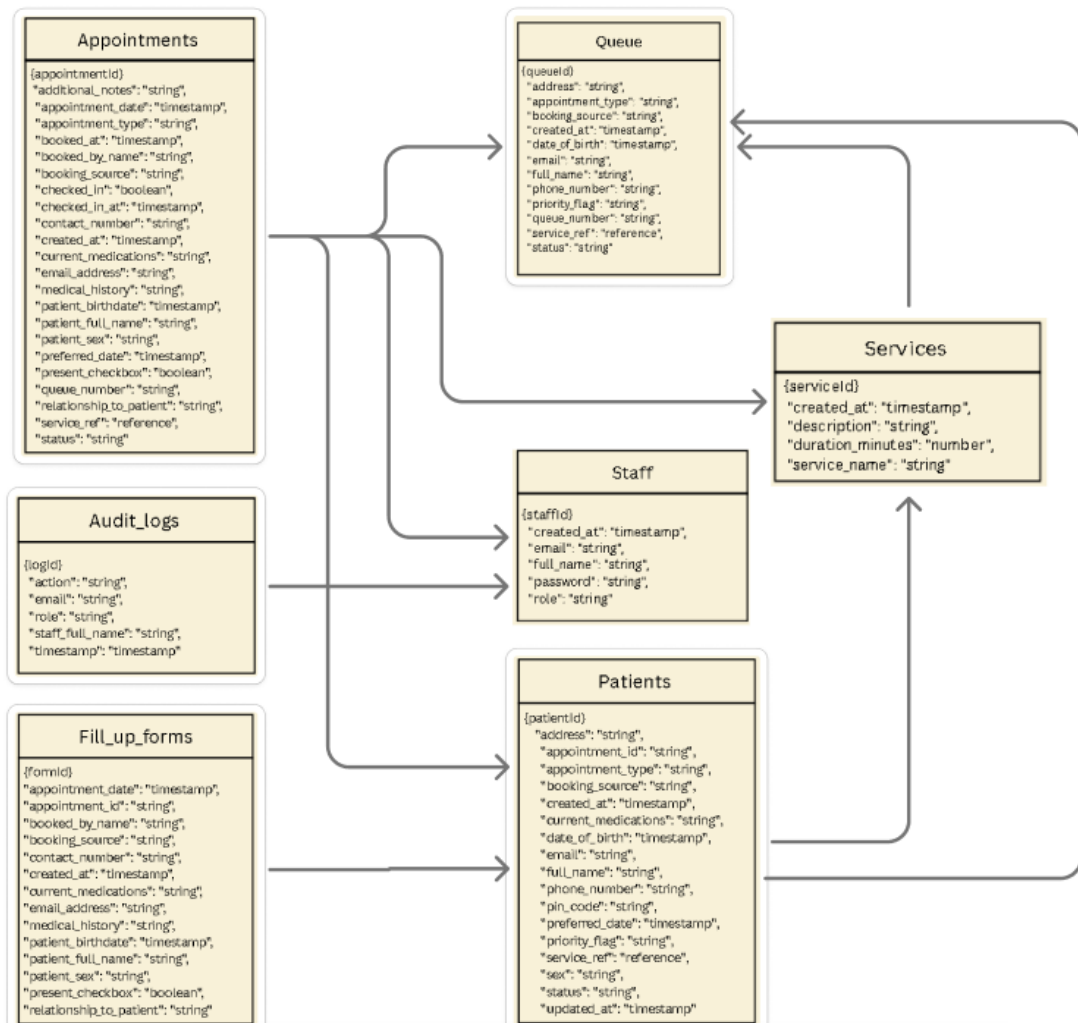


Figure A.2: Complete ERD of the database schema

Appendix B – Reference Materials

- Firebase Documentation. (2025). *Firebase Authentication, Firestore, and Hosting*. <https://firebase.google.com>
- EmailJS Documentation. (2025). *Email Sending via Client-Side Integration*. <https://www.emailjs.com/docs>
- Health IT.gov. (2022). *Security Risk Assessment Tool User Guide*. <https://www.healthit.gov>