QUEUE & APPOINTMENT OPTIMIZER

A SOFTWARE ENGINEERING AND AGILE DEVELOPMENT(6CS1016) REPORT

Submitted by

Mohd Danish Ansari (2023BCSE07AED292)

Arpit Kumar (2023BCSE07AED323)

Akshat Srivastava (2023BCSE07AED324)

Soumya Ranjan Pradhan (2023BCSE07AED600)

In partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Under the Supervision of

Dr. N Rajkumar

ALLIANCE SCHOOL OF ADVANCED COMPUTING

ALLIANCE UNIVERSITY, BENGALURU

APRIL - 2026

ALLIANCE SCHOOL OF ADVANCED COMPUTING

ALLIANCE UNIVERSITY, BENGALURU

DEPARTMENT OF CSE

CERTIFICATE

This is to certify that the Software Engineering and Agile Development work entitled "QUEUE & APPOINTMENT OPTIMIZER" is the Bonafide work done by Mohd Danish Ansari (2023BCSE07AED292), Arpit Kumar (2023BCSE07AED323), Akshat Srivastava (2023BCSE07AED324), Soumya Ranjan Pradhan (2023BCSE07AED600) submitted in partial fulfilment of the requirements for the award of the degree Bachelor of Technology in Computer Science and Engineering during the year 2026-2027.

Dr. N Rajkumar
Associate Professor
Dept. of CSE

ALLIANCE UNIVERSITY

ALLIANCE SCHOOL OF ADVANCED COMPUTING

DECLARATION

This is to declare that the report titled "QUEUE & APPOINTMENT OPTIMIZER" has been made for the partial fulfilment of the Programme - Bachelor of Technology in Computer Science and Engineering, under the guidance of Dr. N Rajkumar. We confirm that this report truly represents my work undertaken as a part of Software Engineering and Agile Development work. This work is not a replication of work done previously by any other person. I also confirm that the contents of the report and the views contained therein have been discussed and deliberated with the Mentor.

NAME	REGESTRATION NO.	SIGNATURE
MOHD DANISH ANSARI	2023BCSE07AED292	
ARPIT KUMAR	2023BCSE07AED323	
AKSHAT SRIVASTAVA	2023BCSE07AED324	
SOUMYA RANJAN PRADHAN	2023BCSE07AED600	

ALLIANCE UNIVERSITY

ALLIANCE SCHOOL OF ADVANCED COMPUTING

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of the task would be put incomplete without the mention of the people that made it possible, whose constant guidance and encouragement crown all the efforts successfully.

We are very thankful to my supervisor Dr. N Rajkumar for his sustained inspiring guidance and cooperation throughout the process of this project. Their wise counsel and valuable suggestions are invaluable.

We would like to thank Dr. S. Oswalt Manoj, Head of the Department and Dr. Vullikanti Vivek, Deputy Director, for their encouragement and cooperation at various levels of Project.

We avail this chance to express my deep sense of gratitude and hearty thanks to the Management of Alliance University, for providing world class infrastructure, congenial atmosphere, and encouragement.

We express my deep sense of gratitude and because of the teaching and non-teaching staff at our department who stood with me during the project and helped me to make it a successful venture. We place highest regards to my parents, my friends and well-wishers who helped plenty in making the report of this project.

Mohd Danish Ansari (2023BCSE07AED292)

Arpit Kumar(2023BCSE07AED323)

Akshat Srivastava(2023BCSE07AED324)

Soumya Ranjan Pradhan(2023BCSE07AED600)

1. INTRODUCTION

In many service-based environments, managing queues efficiently is a significant operational challenge. Customers often experience long and uncertain waiting periods, while service counters may remain idle due to lack of coordination. Existing solutions are either hardware-dependent or lack real-time adaptability.

This project proposes a Real-Time Queue & Appointment Optimizer that replaces manual coordination with an automated, software-driven system. By using a centralized backend, live communication channels, and automated queue progression, the system improves service flow, transparency, and overall user experience.

The system is designed as a web-based platform to ensure ease of access, platform independence, and scalability. It also serves as a practical demonstration of backend system development, database integration, and real-time communication mechanisms.

1.1 PURPOSE

The purpose of this Software Requirements Specification (SRS) document is to provide a detailed and structured description of the requirements for the Queue & Appointment Optimizer system. This document serves as a formal reference for understanding system functionality, constraints, and performance expectations.

The SRS acts as a baseline agreement between stakeholders, developers, and evaluators, ensuring that the system is developed according to clearly defined requirements. It also supports system validation, testing, and future enhancements by documenting both functional and non-functional aspects of the system.

1.2 DOCUMENT CONVENTION

This document follows the IEEE-830 Software Requirements Specification standard and uses the following conventions:

- Section numbering is based on IEEE-830 guidelines
- Functional requirements are labelled as FR-#
- Non-functional requirements are labelled as NFR-#
- The keyword “shall” indicates a mandatory system requirement
- The keyword “may” indicates optional functionality

- Technical terms are used consistently throughout the document

All requirements are written in a clear and unambiguous manner to avoid misinterpretation.

1.3 INTENDED AUDIENCE AND READING SUGGESTIONS

This document is intended for the following audiences:

- Project evaluators and faculty members – to assess project scope and compliance
- Software developers – to understand system behavior and implementation expectations
- Test engineers – to design and validate test cases
- Maintenance personnel – to support future system updates and enhancements

Readers are encouraged to review the sections most relevant to their role. Evaluators may focus on requirements and architecture, while developers and testers should review the functional and non-functional requirements in detail.

1.5 PROJECT SCOPE

Queue & Appointment Optimizer is a web-based application designed to manage and automate queues in walk-in service environments such as banks, hospitals, government offices, and customer service centers.

The system allows users to join queues digitally, automatically assigns customers to available service counters, and provides real-time updates to all connected clients. It eliminates manual queue handling and reduces waiting confusion by ensuring continuous and transparent service flow.

The scope of the project includes backend development, real-time communication using WebSockets, database integration using MySQL, and basic administrative monitoring. Advanced analytics and large-scale enterprise integrations are considered outside the scope of the current implementation.

1.6 REFERENCES

- IEEE Std 29148-2018 (Reaffirmed 2025), *Systems and Software Engineering — Life Cycle Processes — Requirements Engineering*, IEEE Computer Society, 2025.

- IEEE Std 1016-2022 (Active 2025–2026), *Systems and Software Engineering — Design Descriptions*, IEEE Computer Society, 2025.

- Zhang, Y., Liu, H., & Wang, M., *"Real-Time Queue Management Systems Using Event-Driven Architectures"*, IEEE Access, Vol. 13, 2025.

- Kumar, A., Sharma, R., *"WebSocket-Based Real-Time Communication for Service Optimization"*, International Journal of Computer Applications, 2025.

- Oracle Corporation, *MySQL 8.0 Reference Manual*, 2025 Edition.

- FastAPI Development Team, *FastAPI Documentation*, Version 2025.

- Fielding, R., et al., *"Architectural Styles and the Design of Network-Based Applications"*, REST Architecture Updates, 2025 Edition.

- Internet Engineering Task Force (IETF), *RFC 6455 – The WebSocket Protocol*, Updated Notes & Usage Guidelines, 2025.

- Singh, P., Verma, S., *"Automation of Queue Handling Systems in Service-Oriented Applications"*, International Journal of Advanced Computer Science and Applications (IJACSA), 2026.

- IEEE Computer Society, *Best Practices for Real-Time Distributed Systems*, Technical Report, 2026.

2. OVERALL DESCRIPTION

This section provides a high-level description of the Real-Time Queue & Appointment Optimizer, including its operational context, primary functions, user categories, operating environment, and system constraints.

2.1 PRODUCT PERSPECTIVE

The Real-Time Queue & Appointment Optimizer is a web-based, client–server application designed to manage walk-in service queues efficiently. The system consists of a centralized backend server and multiple client interfaces that communicate through REST APIs and real-time WebSocket connections.
The application operates as an independent system and does not rely on external enterprise platforms.

2.2 PRODUCT FUNCTION

The major functions of the system include:

- Digital queue registration
- Real-time queue status updates
- Automatic queue progression
- Service counter registration and monitoring
- Counter availability tracking
- Service history storage

2.3 USER CLASS AND CHARACTERISTICS

- Admin — monitors system status and queue activity.
- Counter Staff — serves customers and updates the counter state.
- Display Client — shows live queue information.
- Customer — joins the queue and views position.

2.4 OPERATION ENVIRONMENT

- Web browser-based user interface
- Backend server developed using Python (FastAPI)
- MySQL relational database
- Local or cloud-hosted deployment

2.5 DESIGN AND IMPLEMENTATION CONSTRAINTS

- Must support real-time communication
- Must use MySQL for data storage
- Must operate on standard computing hardware
- Must follow modular backend design principles

2.6 USER DOCUMENTATION

- System usage guide
- API documentation
- Installation and configuration guide

2.7 ASSUMPTIONS AND DEPENDENCIES

- Availability of stable internet or local network connectivity
- Availability of MySQL database service
- Browser support for WebSocket communication

3. SYSTEM FEATURES (FUNCTIONAL REQUIREMENTS)

FR-1 Queue Registration

- The system shall allow users to join a service queue digitally.
- The system shall generate a unique token for each user joining the queue.

FR-2 Real-Time Queue Updates

- The system shall update queue status in real time.
- All connected clients shall receive queue updates without page refresh.

FR-3 Automatic Queue Progression

- The system shall automatically assign the next token when a counter becomes available.
- The system shall prevent manual skipping of queue order.

FR-4 Counter Management

- The system shall allow service counters to register with the system.
- The system shall track counter status as **FREE**, **BUSY**, or **OFFLINE**.

FR-5 Counter Heartbeat Monitoring

- The system shall receive periodic heartbeat signals from active counters.
- The system shall mark counters inactive if heartbeats are not received within a defined time interval.

FR-6 Data Persistence

- The system shall store queue data and service history in a relational database.
- The system shall preserve queue records for analysis and reference.

FR-7 System Monitoring

- The system shall provide real-time visibility of queue length and counter availability.
- The administrator shall be able to view system status at any time.

4. EXTERNAL INTERFACE REQUIREMENTS

This section specifies the interfaces through which the system interacts with users, hardware, and other software components.

4.1 USER INTERFACES

- Web-based graphical user interface accessible through standard web browsers.
- Dedicated display interface for showing live queue status.

4.2 HARDWARE INTERFACES

- Standard computing devices such as desktops and laptops.
- Mobile devices for accessing queue status and counter interfaces.

4.3 SOFTWARE INTERFACES

- REST APIs for backend operations such as queue registration and counter management.
- WebSocket interface for real-time queue updates and system synchronization.

4.4 COMMUNICATION INTERFACES

- HTTP/HTTPS protocol for client–server communication.
- WebSocket protocol for real-time data exchange

5. NON-FUNCTIONAL REQUIREMENTS

NFR-1 Performance

- The system shall update queue changes within one second under normal operating conditions.
- The system shall support multiple concurrent users without noticeable delay.

NFR-2 Security

- The system shall use HTTP/HTTPS protocols for secure communication.
- Access to administrative functions shall be restricted to authorized users.

NFR-3 Reliability

- The system shall maintain queue consistency during normal operation.
- The system shall recover queue data after a restart using stored database records.

NFR-4 Usability

- The user interface shall be simple and easy to understand.
- Queue information shall be clearly visible to users and staff.

NFR-5 Scalability

- The system shall support multiple service counters.
- The system shall handle an increase in users by adjusting backend resources.

NFR-6 Maintainability

- The backend shall follow a modular code structure.
- The system shall allow easy modification and feature extension.

NFR-7 Portability

- The system shall be deployable on standard operating systems.
- The system shall support local and cloud-based deployment environments.

6. SYSTEM ARCHITECTURE OVERVIEW

The system follows a layered architecture that separates user interaction, application logic, and data management. This approach improves maintainability, scalability, and clarity of system design**.**

6.1 PRESENTATION LAYER

The Presentation Layer provides the user interface for interacting with the system. It includes web-based interfaces for customers, counter staff, administrators, and display screens.
This layer is responsible only for displaying information and sending user actions to the backend.

6.2 BUSINESS LOGIC LAYER

The Business Logic Layer contains the core functionality of the system. It manages queue operations, automatic token assignment, counter state handling, and real-time event processing.
This layer ensures that queue rules are enforced consistently and that system behavior remains correct under different conditions.

6.3 DATA LAYER

The Data Layer is responsible for persistent storage of system data. It stores queue tokens, counter information, and service history using a relational database.
This layer ensures data consistency, recovery after system restarts, and availability of historical records.

6.4 INTEGRATION LAYER

The Integration and Communication Layer enables interaction between clients and the backend system. It uses REST APIs for standard operations and WebSocket communication for real-time queue updates.
This layer ensures instant synchronization of queue state across all connected clients**.**

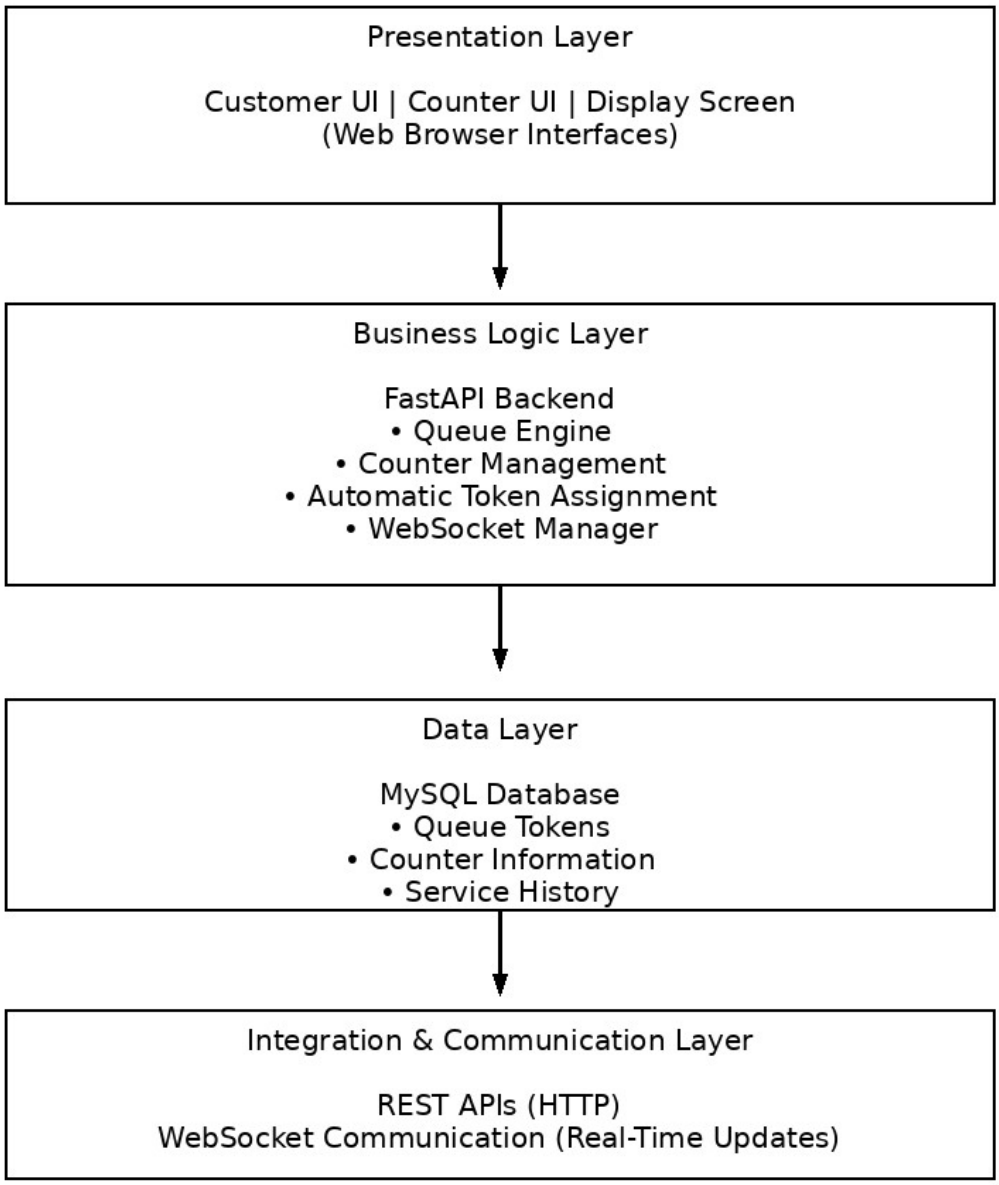

Fig 6.1 System Architecture Overview

7. SYSTEM AND DATA MODELS

This section describes the system models used to represent the structure, behavior, and data organization of the Real-Time Queue & Appointment Optimizer. These models provide a clear understanding of how different components interact and how data flows through the system during operation.The system design is represented using multiple modeling techniques to capture both functional behavior and data relationships.

7.1 SYSTEM ARCHITECTURE MODEL

The System Architecture Model illustrates the layered design of the application, including the presentation layer, business logic layer, data layer, and communication layer. It shows the interaction between user interfaces and backend services and highlights the separation of responsibilities among system components.

7.2 DATA MODEL

The Data Model defines the structure of the database used by the system. Key entities include queue tokens, service counters, and service history records. Relationships among these entities ensure that queue information is stored accurately and can be retrieved reliably during system execution.

This model supports data consistency, recovery after system restarts, and future analysis of service performance.

7.3 WORKFLOW MODEL

The Workflow Model represents the sequence of activities performed by the system. It includes user queue registration, token generation, automatic token assignment to counters, real-time queue updates, and service completion. This model helps in visualizing the flow of control and data throughout the system.

7.4 UML DIAGRAMS

Unified Modeling Language (UML) diagrams are used to represent system interactions and behavior at a conceptual level. These diagrams help in understanding system flow and user interactions without exposing implementation details.

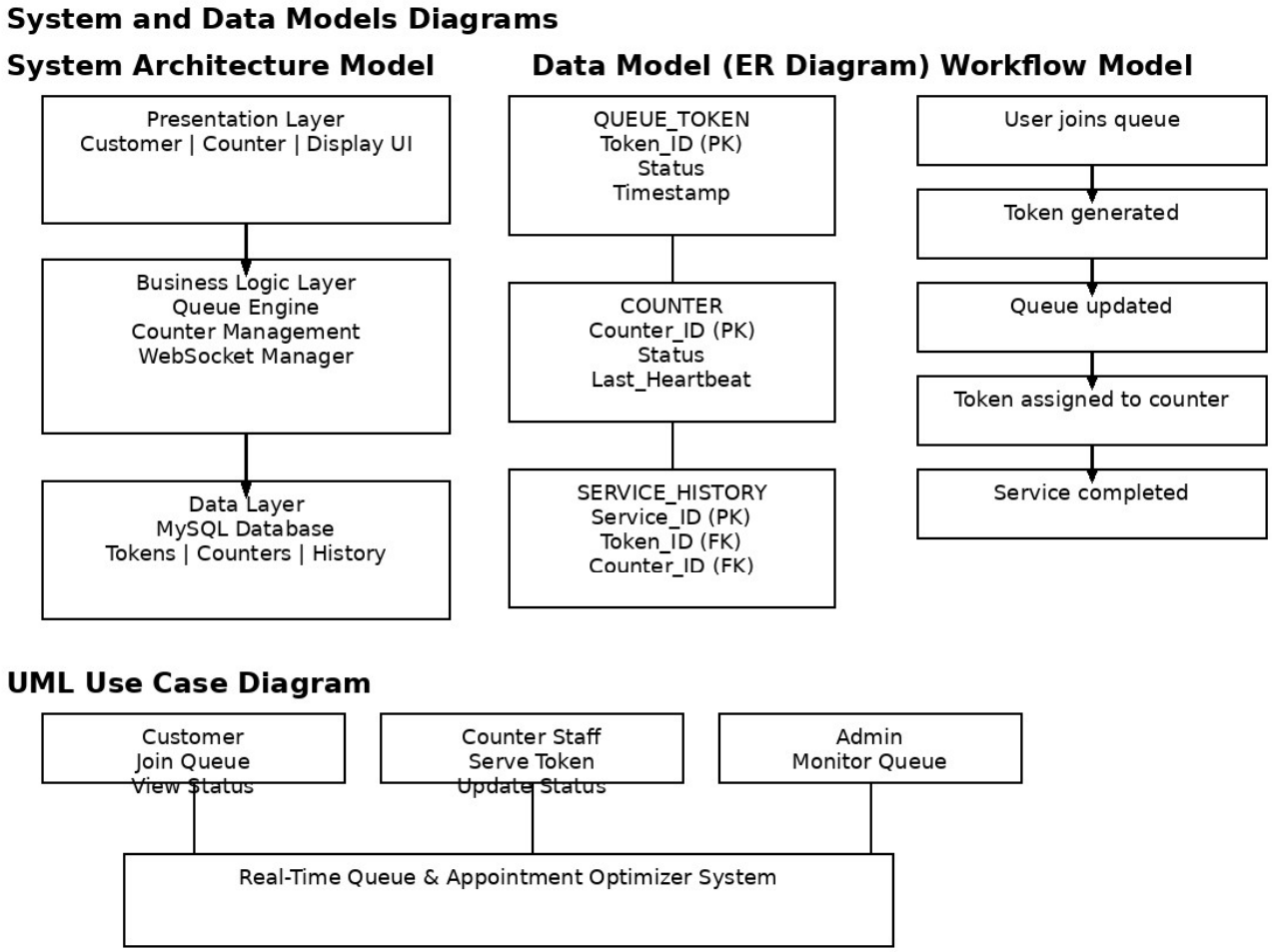

Fig 7.1 System and Data Models Diagram

8. VALIDATION AND ACCEPTANCE CRITERIA

This section defines the criteria used to verify that the Real-Time Queue & Appointment Optimizer meets its specified requirements.

- All functional requirements shall be validated using predefined test cases.
- Each system feature shall be tested to ensure correct operation under normal conditions.
- Real-time queue updates and automatic token assignment shall be verified through system demonstration.
- The system shall be accepted only after successful execution of all test cases and demonstration of expected system behavior

9. APPENDICES

Appendix-A: Glossary

- Queue Token – A unique identifier assigned to a user when joining the queue.
- Counter – A service point where users are served.
- Real-Time Update – Instant system update without page refresh.
- WebSocket – A protocol enabling real-time, two-way communication between client and server.

Appendix-B: Sample Data

Sample data includes example queue tokens, counter identifiers, and service records used during system testing and demonstration.
This data is used to verify system behavior under various operational scenarios.

Appendix-C: Compliance Checklist

- Functional requirements implemented and tested
- Real-time updates verified
- Database persistence validated
- System architecture followed as specified
- Documentation completed