PES UNIVERSITY

Department of Computer Science and Engineering

UE22CS341A: Software Engineering

PES UNIVERSITY
Department of Computer Science and Engineering
UE22CS341A: Software Engineering

Project Planning Document

Software Requirements Specification

for

INTERNATIONAL AIRPORT MANAGEMENT SYSTEM

Prepared by:

Nitheesh Pugazhanthi: PES2UG22CS371
Nikhil Srivatsa: PES2UG22CS357

PES UNIVERSITY

SOFTWARE LIFECYCLE

WATERFALL MODEL

For these kinds of smaller projects, the structured nature of Waterfall ensures that dependencies between systems (e.g., flight tracking and security) are handled in a sequential manner. It allows ample time for thorough documentation

The Waterfall model works best when the requirements are well understood and fixed, allowing for a structured, sequential approach to development.

The Waterfall model provides a high level of control and predictability. Each phase (requirements, design, implementation, testing, etc.) is completed before moving to the next, ensuring that the project progresses in a linear, organized fashion. For a project like this which involves multiple functions and a lot of data dependencies, its best to use a sequential model.

The Waterfall model allows stakeholders (e.g., airport management, security agencies) to be involved at key review points (requirements gathering, design review, etc.), ensuring that the project remains aligned with their expectations throughout development.

In summary, the Waterfall model's emphasis on a clear, sequential approach aligns with the stable and well-defined requirements of this project, making it an ideal choice.

Tools Used

Tool Type	Tool name
Planning Tool	***Jira*- We can set up the project plan, Assign Project Team Roles and Permissions and track the progress of the project**
Design Tool	***ERDPlus, DrawSQL* -These tools were used to create the Entity-Relationship Diagram and the Relational Schema.**
Version Control Tool	***Git/GitHub*- Used for collaborative development and for maintaining multiple iterations/versions of the project.**
Developer Tool	***VSCode, Neovim, MySQL(cmd line and workbench)***
Deployment Tool	***Docker, Kubernetes*- Used for providing containers for software testing and deployment.**
Testing Tool	***Selenium IDE* – open-source tool used for automated web-testing and browser automation**

Deliverables

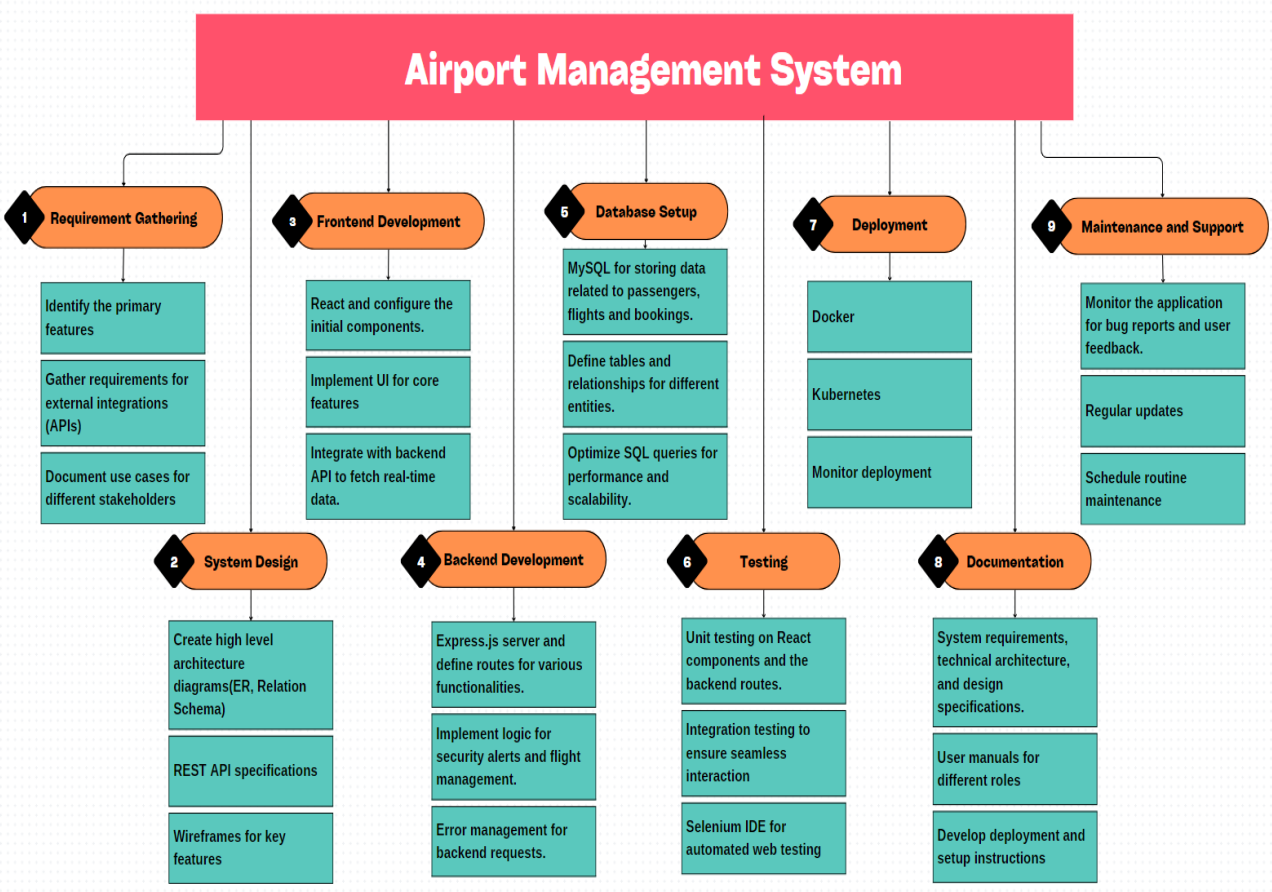
Flight Status and Management Module	**Reuse Component**	**The core functionality can be implemented using third-party APIs**
Security Alerts and Flagging System	**Build Component**	**Custom development ensures compliance with local and international aviation standards of security.**
Price Tracking and Future Planning Module	**Build Component**	**We will use MySQL queries to provide the best costs for the flights.**
Passenger Management and Check-In System	**Build Component**	**This module will be custom-built to handle passenger check-ins, baggage management, and seat assignments tailored to the specific needs of the airport**

Work-Breakdown Structure

Main Task	Sub Tasks
Requirement Gathering	• **Identify the primary features (like flight status, passenger management, check-in system).** • **Gather requirements for external integrations (APIs for flight data).** • **Document use cases for different stakeholders (passengers, airport staff, etc.).**
System Design	• **Create high level architecture diagrams(ER, Relation Schema) for database structure and details.** • **REST API specifications** • **Wireframes for key features(like flight booking, check-in, etc.).**
Frontend Development	• **Setup project using React and configure the initial components.** • **Implement UI for core features(like flight status, booking, etc.)** • **Integrate with backend API to fetch real-time data.**
Backend Development	• **Setup Express.js server and define routes for various functionalities.** • **Implement logic for security alerts and flight management.** • **Ensure data validation and handle error management for backend requests.**

Database Setup	• **Setup MySQL for storing data related to passengers, flights and bookings.** • **Define tables and relationships for different entities.** • **Optimize SQL queries and appropriate indexes for performance and scalability.**
Testing	• **Perform unit testing on React components and the backend routes.** • **Integration testing to ensure seamless interaction between components.** • **Selenium IDE for automated web testing for UI**
Deployment	• **Use Docker to containerize frontend, backend and database services.** • **Setup Kubernetes to manage containers in scalable deployment environment.** • **Monitor deployment for any performance issues and unexpected errors.**
Documentation	• **Document system requirements, technical architecture, and design specifications.** • **Create user manuals for different roles (e.g., passengers, airport staff).** • **Develop deployment and setup instructions for developers and admins.**

Maintenance and Support	• **Monitor the application for bug reports and user feedback.** • **Perform regular updates to fix issues and add new features.** • **Schedule routine maintenance to ensure system security and data integrity.**



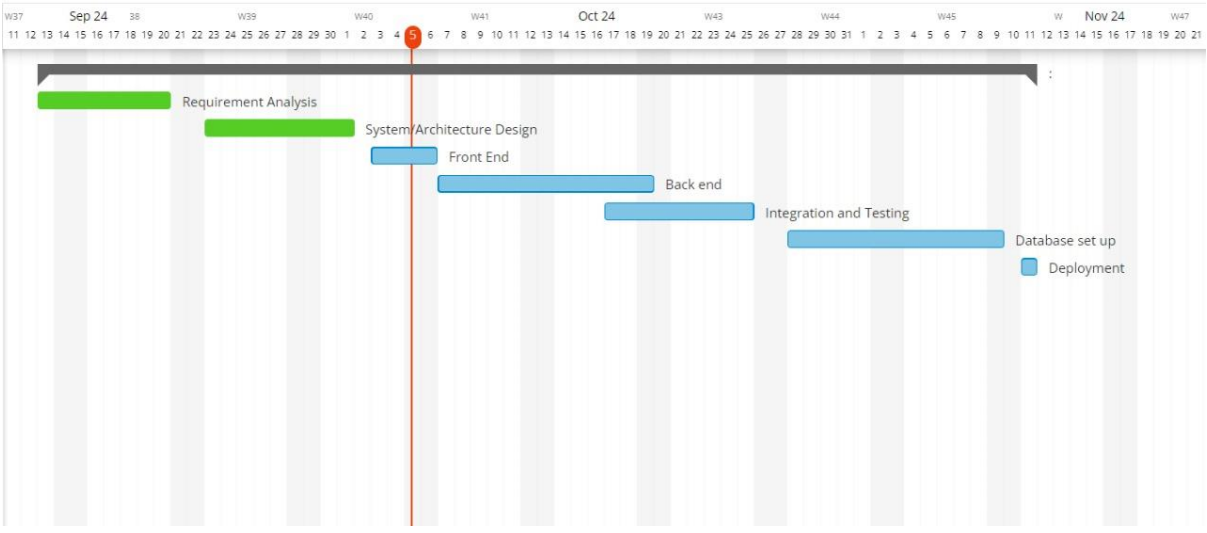
Effort Estimation

Task	Estimated Effort(Person-Months)
Requirement Gathering	0.43
System Design	0.43
Frontend Development	0.86
Backend Development	0.86
Database Setup	0.22
Testing	0.64
Deployment	0.22
Documentation	0.22

Using COCOMO model for organic projects with parameters a=2.4, b=1.05 and KLOC=1.5(approx.) we get E=3.67 Person-Months.

(where E=a x (KLOC)^b)

Gantt Chart



Coding Details

Frontend Development

- **Framework**
  React: JavaScript library for building user interfaces, creation of reusable UI components.
- **UI Components**
  Custom components for various functionalities, such as flight status displays, passenger check-in forms, security alert notifications, and ticket booking interfaces.
- **State Management**
  To manage application state effectively across various components, especially for tracking user sessions, flight data, and passenger information.
- **API Integration**
  Integrating with backend APIs built with Node.js to fetch and submit data regarding flights, passengers, and bookings. This will involve making asynchronous requests to the server and handling responses to update the UI accordingly.
- **Styling**
  Utilizing HTML and various CSS frameworks like Bootstrap or Material-UI for responsive and attractive layouts.

Backend Development

- **Framework**
  Node.js: JavaScript runtime to build scalable and high-performance web applications. It will handle server-side logic, manage API requests, and interact with the database.
- **Authentication**
  Implementing user authentication using OAuth for secure login systems for airport staff, security agencies, and passengers. This ensures that sensitive data is protected and access is restricted based on user role.

Database Connectivity

MySQL: Utilizing MySQL as the relational database management system to store data related to flights, passengers, and bookings. The backend will establish connections to MySQL, for managing database operations like CRUD (Create, Read, Update, Delete).