SOFTWARE REQUIREMENTS SPECIFICATION for

RESERVEASE :
Infrastructure Reservation
Version 1.0 approved

Prepared by
ABREENA K BIJY [**FIT21CSD004**]
ANNA DEJO [**FIT21CSD015**]
FATHIMA ISMAIL [**FIT21CSD027**]
MARIA NIMITHA [**FIT21CSD036**]

April 21, 2024

Contents

Revision History

Name	Date	Reason For Changes	Version
21	22	23	24
31	32	33	34

1 Introduction

1.1 Purpose

The purpose of this document is to outline the requirements for developing an Infrastructure Reservation System. This system aims to provide the organization with a centralized platform for efficiently managing resource reservations and allocations within its infrastructure. By clearly defining the functionalities and features required, this document will guide the development process to ensure that the final system effectively meets the organization's needs.

1.2 Document Conventions

This SRS follows standard conventions for document formatting and organization. Priorities for requirements are indicated where relevant, with higher-level priorities assumed to be inherited by detailed requirements. Each requirement statement is assigned its own priority based on its importance to the overall system.

1.3 Intended Audience and Reading Suggestions

This document is intended for various stakeholders involved in the development and use of the Infrastructure Reservation System, including developers, project managers, marketing staff, users, testers, and documentation writers. It is organized to facilitate easy navigation, beginning with overview sections and proceeding through more detailed sections relevant to each reader type.

- **Developers.**: Those responsible for implementing the infrastructure reservation system.
- **Project Managers.**: Individuals overseeing the development and deployment of the system.
- **Administrators.**: Staff members who will manage the reservation system and its resources.
- **Users.**: Employees of the company who will utilize the reservation system to book resources.

1.4 Project Scope

The Infrastructure Reservation System is designed to streamline the process of managing resource reservations within the organization's infrastructure. Its primary objectives include improving resource utilization, enhancing operational efficiency, and facilitating better coordination among different teams. By aligning with corporate goals and business strategies, this system will contribute to overall organizational success.

1.5 References

- **W3Schools.** https://www.w3schools.com/
- **Flask Tutorials.** https://www.youtube.com/playlist?list=PLzMcBGfZo4-n4vJJybUVV3Un_NFS5EOgX
- **Connection Establishment of Flask and MongoDB.** https://www.youtube.com/playlist?list=PLXmMXHVSvS-Db9KK1LA7lifcyZm4c-rwj
- **Angular Documentation.** https://devdocs.io/angular/

2 Overall Description

2.1 Product Perspective

The Infrastructure Reservation System revolves around creating a standalone solution tailored to improve the efficiency and transparency of resource reservation processes within the organization, while also considering integration with existing systems and scalability for future growth.

- **New Solution.**: The ReservEase is a novel solution developed to address the challenges associated with manually managing resource reservations and meeting room spaces within the organization. It is not an extension of an existing system or a replacement for current processes but rather a standalone product tailored to fulfill the organization's specific needs.
- **Origin.**: The genesis of the ReservEase lies in the organization's recognition of the inefficiencies and complexities involved in managing resources and meeting spaces using traditional methods.
- **Self-Contained System.**: While the ReservEase may interact with other systems for user authentication or data synchronization, it is primarily designed as a self-contained system. Its core functionality, including resource availability tracking, reservation management, and user interface, operates independently.
- **Components and Interfaces.**: The project involves the development of various components such as user authentication, resource availability database, reservation management module, and user interface. External interfaces may include integration points with employee databases, calendar systems, and notification services.

2.2 Product Functions

- **User Registration and Authentication.**: Allow users to register and authenticate themselves securely to access the system.
- **Resource Availability Check.**: Enable users to check the availability of meeting rooms, equipment, or other resources within the infrastructure.
- **Resource Reservation.**: Allow users to reserve resources for specific dates and times according to their requirements.

- **Reservation Management.**: Provide functionality to manage and update reservations, including canceling or modifying existing bookings.
- **Administrative Tools.**: Offer administrative tools for managing user accounts, resource availability, and resolving conflicts in reservations.
- **Accessibility and User Experience.**: Ensure the system is accessible to all users and provides a user-friendly experience across different devices and platforms.

2.3 User Classes and Characteristics

The Infrastructure Reservation System is anticipated to serve several user classes, including regular employees requiring access to book meeting rooms and resources, administrators responsible for system maintenance and conflict resolution, managers/team leads overseeing team reservations, and external stakeholders such as clients or guests needing to book resources. Users will vary in technical expertise, frequency of use, and privilege levels, with administrators and managers typically possessing higher access levels and responsibilities. The system must cater to these diverse user needs by providing intuitive interfaces and appropriate levels of access and functionality.

2.3.1 Employee Users

:

- **Frequency of Use.**: Regularly, as they need to reserve resources for meetings, workspaces, etc.
- **Subset of Product Functions Used.**: Primarily booking resources and managing their reservations.
- **Technical Expertise.**: Varied, but generally not highly technical as they are primarily focused on their job roles.
- **Security or Privilege Levels.**: Standard user privileges to book resources and view their own reservations.
- textbfEducational Level or Experience.: Varied, but typically have basic computer literacy skills.
- **Characteristics.**: They are the primary users of the system, relying on it to efficiently reserve resources for their work needs.

2.3.2 Administrator Users:

:

- **Frequency of Use.**: Regularly, to manage the reservation system, users, and resources.

- **Subset of Product Functions Used.**:Administering user accounts, configuring resources, generating reports, etc.

- **Technical Expertise.**: Moderate to high technical expertise, as they need to understand system configuration and administration..

- **Security or Privilege Levels.**: Elevated privileges to access and modify system settings..

- textbfEducational Level or Experience.: VGenerally have a higher level of education or experience in IT or system administration.

- **Characteristics.**: Responsible for maintaining and managing the reservation system, ensuring its smooth operation, and resolving any issues that arise.

2.4 Operating Environment

The Infrastructure Reservation System will operate within a standard office environment, primarily accessed through web browsers on desktop computers, laptops, and mobile devices. It will be compatible with commonly used web browsers such as Chrome, Firefox, Safari, and Edge. The system will run on a cloud-based server infrastructure, ensuring scalability and accessibility from anywhere with an internet connection. It must seamlessly integrate with existing authentication systems, email servers for notifications, and potentially with calendar applications for scheduling purposes.

2.4.1 Software Requirements:

- **Operating System**: Compatibility with Windows, macOS, and Linux distributions.

- **Web Browser.**:Support for modern web browsers such as Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge for accessing the Angular-based front end.

- **Angular Framework.**: Ensure the server hosting the Angular application supports Node.js and npm (Node Package Manager) for managing dependencies.

- **Flask Framework.**: Deployment on a server with Python support, preferably using virtual environments to manage dependencies.

- **MongoDB.**:Installation and configuration of MongoDB, along with appropriate drivers for Python (PyMongo) to interact with the database.

- **Python.**: Ensure the server has Python installed, preferably with a compatible version for Flask development.

2.4.2 Hardware Requirements:

- **Server Infrastructure**: Adequate CPU, RAM, and storage resources on the server hosting both the Angular front end and Flask back end. The specific requirements will depend on factors such as expected traffic volume and data processing needs.

- **Client Devices.**:Support for various client devices such as desktops, laptops, tablets, and smartphones accessing the Angular front end.

2.4.3 Development and Testing Environment::

- **Development Tools:**: Use of Angular CLI for Angular development, Flask CLI for Flask development, and tools like Visual Studio Code or PyCharm as the IDE for coding.

- **Testing Tools.**:S Integration testing can be performed using tools like Protractor for Angular and Flask-Testing for Flask.

2.4.4 Network Requirements.

- **Internet Connectivity**: Continuous internet connectivity is required for users to access the Angular front end and communicate with the Flask back end.

- **Network Security:**:Implementation of appropriate security measures such as firewalls, encryption, and HTTPS to protect data transmission between client devices and the server.

2.5 Design and Implementation Constraints

Creating the Infrastructure Reservation System comes with certain limits and rules that affect how it's built. For instance, we have to follow company rules about keeping data safe and private, as well as any laws about privacy that apply to us. We also need to make sure our computers have enough power and space to run the system smoothly. When we connect our system with other programs we use, like email or login systems, we have to make sure they work well together and don't mess up our data. The tools and technology we choose to build the system will depend on things like cost, how well they can grow with us, and how easy they are to use. Security is a big deal, so we'll need to make sure only the right people can access the system and that our data stays safe from hackers. And when we're writing the code for the system, we'll follow some rules and guidelines to make sure it's easy to understand and work with in the future.

- **1. Technology Stack Compatibility**: Constraints related to the compatibility of Angular, Flask, and MongoDB versions. Ensure that the versions of these technologies are compatible with each other to avoid potential conflicts during development and deployment.

- **2. Resource Limitations**: Constraints related to server resources such as CPU, memory, and storage. Ensure that the server infrastructure can handle the expected load and traffic volume to prevent performance issues or downtime.
- **3. Integration Challenges**: Constraints related to integrating third-party services or systems with your reservation system. Ensure that APIs or interfaces are available and well-documented for seamless integration with other systems such as authentication services or calendar applications.
- **4. Scalability Requirements**: Constraints related to scalability and future growth of the system. Ensure that the architecture and design of the system allow for scalability to accommodate increasing user demands and expanding features.
- **5. User Experience Constraints**: Constraints related to user experience design and usability. Ensure that the system is intuitive and user-friendly, considering the diverse needs and preferences of the target user base.
- **6. Maintenance and Support Constraints**: Constraints related to ongoing maintenance and support requirements. Ensure that adequate resources are allocated for maintenance activities such as bug fixes, updates, and user support.

2.6 User Documentation

We enumerate the various forms of user documentation accompanying the Infrastructure Reservation System, such as user manuals, online guides, or tutorials. Additionally, we specify any prescribed standards or formats for delivering this documentation to users.

2.6.1 User Manuals.

Detailed manuals providing step-by-step instructions on how to navigate the system, make reservations, check resource availability, manage bookings, and perform other relevant tasks.

2.6.2 Tutorials.

: Informative tutorials and walkthroughs designed to help users familiarize themselves with the system's interface, tools, and capabilities. These tutorials will cover common tasks and workflows.

2.6.3 Reservation Guides.

A step-by-step guide on how to make a reservation, covering the process from logging in to selecting available resources, specifying dates and times, and confirming the booking.

2.6.4 Account Management.

Instructions for managing user accounts, including creating new accounts, logging in, resetting passwords, and updating personal information.

2.6.5 Resource Availability.

: Information on how users can check the availability of resources, including meeting rooms, workspaces, equipment, etc., and how to view available time slots

2.6.6 Reservation Policies.

Clear explanations of any policies related to reservations, including cancellation policies, maximum booking durations, rules for reserving multiple resources simultaneously, etc.

2.6.7 Notifications.

Guidance on how users will receive notifications about their reservations, such as confirmation emails, reminders, and updates on any changes or cancellations.

2.7 Assumptions and Dependencies

In infrastructure reservation, assumptions and dependencies play a critical role in ensuring the smooth development and operation of the system. We assume the availability and compatibility of third-party components, access to necessary development tools and environments, and the stability of the operating environment. Additionally, we depend on external systems for functionalities like authentication and payment processing, and compliance with regulatory standards. Documentation and knowledge sharing among team members are also crucial dependencies. Lastly, resource availability, including human resources and budget allocations, must be considered to manage project timelines and scope effectively. Addressing these assumptions and dependencies will help mitigate risks and ensure the project's success.

1. **Third-party Services.**:

   - **Assumption**:We assume the availability and compatibility of third-party components, such as libraries, APIs, or services, required for features like user authentication, database management, and calendar integration.
   - **Dependency**:Any changes or disruptions in their availability or compatibility could impact the functionality and performance of the system.

2. **Development Environment Stability.**:

   - **Assumption**:The development environment, including development tools, frameworks, and platforms, will remain stable and supported throughout the project lifecycle.

- **Dependency**:Changes or instability in the development environment could require adjustments to development processes, tooling, or even the chosen technologies, potentially affecting project timelines and deliverables.

3. Regulatory Compliance.:

- **Assumption**:The system design and implementation will comply with relevant regulatory requirements and industry standards.
- **Dependency**: Failure to meet regulatory compliance could result in legal issues or reputational damage, highlighting the importance of thorough compliance checks and adherence to standards throughout the ReservEase lifecycle.

3 External Interface Requirements

3.1 User Interfaces

The user interface of our infrastructure reservation system will be designed with simplicity and usability in mind. It will feature intuitive screen layouts with clear navigation paths to facilitate easy access to different functionalities. Standard buttons and functions, such as search, filter, and booking options, will be prominently displayed for user convenience. Error messages will be displayed in a clear and concise manner to guide users in resolving any issues that may arise during their interactions with the system. Additionally, the interface will adhere to GUI standards and product family style guides to maintain consistency and familiarity across different pages and screens. Overall, the user interface will aim to provide a seamless and pleasant experience for users as they navigate and utilize the reservation system.

- **Login Page**: Users are presented with a login page upon visiting the website. Upon successful login, users are redirected to the homepage.
- **Signup Page.**: New users can sign up for an account by providing necessary details such as username, password, and other required information
- **Angular Framework.**: Ensure the server hosting the Angular application supports Node.js and npm (Node Package Manager) for managing dependencies.
- **Homepage.**: The homepage provides users with an overview of available resources and reservations. It displays personalized information based on the user's profile and permissions.
- **Profile Page.**:Users can access their profile page to view and manage their account details, preferences, and reservation history.
- **Reservation Page.**: The reservation page allows users to browse available resources, view details, and make reservations.
- **Space Allocation Page.**: This page allows users to book spaces such as meeting rooms, conference halls, etc. Users can select the desired space, specify the date and time of reservation, and confirm the booking.
- **Resource Allocation Page.**: Here, users can allocate resources such as projectors, laptops, etc. Similar to the space allocation page, users can select the desired resources, specify the date and time of allocation, and confirm the booking.

- **My Reservation Page.**: This page displays a list of current reservations made by the user. Users can view reservation details, edit existing reservations, or delete reservations they no longer need. Additionally, there is an option to add new reservations, which redirects users to the reservation page.
- **Logout.**: The Logout button ends the session of the user, and the user will be redirected back to the Login page.

3.2 Hardware Interfaces

Infrastructure reservation system will primarily interact with standard hardware components commonly found in office environments, such as computers, servers, and networking devices. The software will be compatible with a variety of hardware setups, including desktop computers, laptops, tablets, and mobile devices, allowing users to access the reservation system from different platforms. The nature of the data and control interactions between the software and hardware components will mainly involve data transmission over network connections, with no specific hardware dependencies or requirements beyond standard computing devices and network connectivity. Additionally, the system will support various communication protocols, ensuring compatibility with different hardware configurations and network environments.Users access the system through web browsers such as Google Chrome, Mozilla Firefox, and Microsoft Edge.

3.3 Software Interfaces

Infrastructure reservation system will interface with several software components to facilitate its functionality. These components include databases for storing reservation data, operating systems for running the application, and web servers for hosting the system. The system will communicate with the database management system (DBMS) to store and retrieve reservation information, ensuring seamless data management. Additionally, it will integrate with operating systems to ensure compatibility and efficient resource utilization. Furthermore, the system will rely on web servers to host the application, enabling users to access it via web browsers. The communication between these software components will follow standard protocols and interfaces to ensure interoperability and smooth operation of the reservation system.

- **Backend**: Flask, a micro web framework for Python, serves as the backend of the system. It handles incoming HTTP requests from clients, processes business logic, interacts with the database, and generates appropriate responses. Flask provides a lightweight and flexible environment for building web applications, allowing for easy integration with other components.
- **Frontend.**: Angular, a TypeScript-based open-source framework for building web applications, powers the frontend of the infrastructure reservation system. Angular facilitates the development of dynamic and responsive user interfaces by providing

a comprehensive set of tools and libraries. It handles user interactions, renders views, and communicates with the backend to fetch and update data.

- **Database.**: MongoDB, a NoSQL database program, is used to store and manage data for the infrastructure reservation system. MongoDB stores user profiles, reservation details, resource information, and other relevant data in a flexible and scalable manner. It offers high performance, automatic scaling, and support for complex data structures, making it well-suited for applications with evolving requirements.

- **Web Server.**: NGINX, a high-performance web server and reverse proxy server, may be used to host and serve the infrastructure reservation system to clients. NGINX handles incoming HTTP requests, forwards them to the appropriate backend components (e.g., Flask application servers), and delivers responses back to clients. It also provides features such as load balancing, caching, and SSL/TLS termination to improve performance, security, and scalability.

3.4 Communications Interfaces

In infrastructure reservation system, communication interfaces play a crucial role in enabling users to interact with the system and ensuring smooth data exchange. These interfaces include web browser interactions, allowing users to access the system's interface conveniently. Additionally, the system will utilize network server communication protocols to facilitate data transmission between clients and servers. To maintain consistency and clarity in communication, specific standards for message formatting will be defined. Furthermore, the system will adhere to established communication protocols such as FTP or HTTP for data transfer. Security measures, including encryption mechanisms, will be implemented to protect sensitive information during transmission. Moreover, the system will specify data transfer rates and synchronization mechanisms to optimize communication efficiency and reliability.

- **HTTP Protocol**: The system communicates with clients via the HTTP (Hypertext Transfer Protocol) protocol over the web. HTTP is used for transmitting requests and responses between the frontend and backend components of the system. It allows for the exchange of data, including user input, reservation details, and system updates, over the internet.

- **Web Sockets.**: Web sockets may be utilized for real-time communication between the server and clients. Web sockets enable bidirectional communication channels over a single, long-lived connection, allowing for instant updates and notifications to be sent to users. This is particularly useful for features such as live availability updates and real-time notifications for reservation status changes.

- **Authentication and Authorization Protocols**: Communication interfaces incorporate authentication and authorization protocols to verify the identity of users and enforce access control policies. This ensures that only authenticated users with the appropriate permissions can access restricted features and perform actions such as making reservations or modifying existing bookings.

4 System Features

In infrastructure reservation system, several key features are essential for providing efficient resource management and user interaction. These features include:

4.1 Functional Requirements

4.1.1 Homepage Display

Description and Priority

Users are greeted with a homepage displaying the current date and time.
Priority: Medium.

- Provide options for both sign-up and sign-in.
- Include input fields for username and password for sign-in.
- Include input fields for username, password, email, and any other required information for sign-up.
- Display error messages for invalid sign-in attempts, such as incorrect username or password.
- Upon successful sign-in, redirect the user to the profile page.

4.1.2 Profile Page

Description and Priority

The Profile Page serves as a central hub for users to view and manage their account information within the infrastructure reservation system. It displays essential user details such as username, email, and possibly other profile-related information.
Priority: Medium.

- Display the user's profile information, such as username, email, etc.
- Provide options to view, edit, or delete the user's profile.
- Include links or buttons to navigate to other sections of the application, such as reservation management, reservation view, etc.

4.1.3 Resource Allocation Page

Description and Priority

Users can allocate resources for specific reservations.
Priority: High.

- The system shall provide functionality for users to allocate resources to their reservations.
- The system shall allow users to select available resources and assign them to their reservations.
- Upon resource allocation, the system shall update the reservation information in the database.
- The system shall display a success message upon successful creation of the reservation.

4.1.4 Reservation Management Page

Description and Priority

Users can view and manage reservations for infrastructure resources.
Priority: High.

- The system shall allow users to view their existing reservations on a dedicated page.
- The system shall allow users to cancel or modify existing reservations.
- Upon cancellation or modification of a reservation, the system shall update the reservation information in the database.

4.1.5 Logout Page

Description and Priority

Users can log out of the system.
Priority: Medium.

- The system shall provide a logout button on all pages.
- Upon clicking the logout button, the system shall end the current session and redirect the user to the login page.

4.1.6 Authentication

Description and Priority

Users will be authenticated using a username and password.
Priority: High.

- The system shall provide a login page with fields for entering a username and password.
- Upon submission of login credentials, the system shall verify the existence of the username in the database.
- If the username exists, the system shall compare the entered password with the stored password.
- If the passwords match, the system shall grant access and authenticate the user.
- If the passwords do not match, the system shall display an error message indicating ”Incorrect password”..
- If the username does not exist, the system shall display an error message indicating ”User not found”.

5 Other Nonfunctional Requirements

5.1 Performance Requirements

The system should be able to handle a large number of concurrent users without significant degradation in performance, ensuring that users can quickly and efficiently reserve resources even during peak times. Response times for user actions, such as searching for available resources and making reservations, should be kept to a minimum, ideally within 2 seconds, to provide a seamless user experience.

5.2 Safety Requirements

While the primary focus of the system may not be safety-related, it's important to ensure the security and integrity of user data. The system should implement robust data protection measures, including encryption of sensitive information and secure authentication mechanisms, to prevent unauthorized access and data breaches.

5.3 Security Requirements

Security is paramount in an infrastructure reservation system to protect sensitive information and prevent unauthorized access. User authentication should be implemented securely, with options for multi-factor authentication to enhance security. Access control mechanisms should be in place to ensure that users can only view and reserve resources for which they have the necessary permissions.

5.4 Software Quality Attributes

The system should prioritize reliability and availability to ensure that users can access the reservation platform whenever they need it. Usability is also crucial, with an intuitive and user-friendly interface that allows users to easily search for available resources, view details, and make reservations without encountering usability issues. Maintainability is essential for ensuring that the system can be easily updated and maintained over time, with clean and well-documented code that facilitates future enhancements and modifications.

5.5 Business Rules

Business rules may include constraints on resource availability, such as maximum reservation durations, minimum notice periods for reservations, and blackout periods for maintenance or other scheduled events. These rules should be enforced by the system to ensure fair allocation of resources and prevent conflicts or overbooking. Additionally, compliance with organizational policies and procedures, such as approval workflows for certain types of reservations, should be integrated into the system to ensure adherence to company guidelines.

6 Other Requirements

6.1 Database Needs

The system shall utilize a robust and scalable database management system (DBMS) to store and manage all data related to resource reservations, user accounts, and system configurations. The database should support efficient querying and transaction processing to ensure optimal performance, even during periods of high user activity. Data integrity and security features, such as encryption and access controls, should be implemented to protect sensitive information stored in the database.

- Store reservation data securely, including user information and reservation details.
- Implement regular backups of reservation data to prevent loss.
- Ensure scalability of the database to handle increased reservation volume over time.

6.2 Legal Rules

- Adhere to relevant data protection regulations such as GDPR or CCPA to safeguard user privacy.
- Clearly outline legal terms, including Terms of Service and Privacy Policy, for users accessing the reservation platform.

6.3 Code Reuse

- Develop the reservation platform with modular and reusable code components to facilitate future updates and enhancements.
- Utilize frameworks and libraries that promote code reuse across different sections of the platform

6.4 Speed and Performance

- Optimize the performance of the reservation platform to ensure fast loading times and responsive user interactions.
- Conduct performance testing to assess the platform's ability to handle concurrent users and maintain responsiveness under heavy loads.

6.5 Platform Growth

- Design the reservation platform with scalability in mind to accommodate increasing user demand and expanding infrastructure.
- Monitor platform usage and performance metrics to identify opportunities for scaling up resources and optimizing performance.

6.6 Appendix A: Glossary

- User Class: Different categories of users with varying levels of access and permissions within the system, such as administrators, employees, and guests.
- Resource Reservation: The process of booking or reserving resources, such as meeting rooms, equipment, or facilities, for specific time slots or periods.
- SRS: Software Requirements Specification - a document that specifies the functional and non-functional requirements of a software system.
- DBMS: Database Management System - software that enables users to create, manage, and manipulate databases.
- GDPR: General Data Protection Regulation - a regulation in EU law on data protection and privacy.
- WCAG: Web Content Accessibility Guidelines - a set of guidelines for making web content more accessible to people with disabilities>

6.7 Appendix B: Analysis Models

- Data Flow Diagrams: Visual representations of the flow of data within the system, showing how information moves between processes, data stores, and external entities.
- Entity-Relationship Diagrams: Diagrams that depict the relationships between entities in a database, illustrating how different entities are related to each other through attributes and keys.

6.8 Appendix C: To Be Determined List

- Clarification on specific security requirements related to user authentication and data encryption.
- Identification of external APIs or libraries to be used for internationalization support.

- Determination of scalability testing procedures and performance benchmarks.
- Performance benchmarks and thresholds for evaluating platform speed and responsiveness.
- Any additional features or enhancements to be determined based on user feedback or future requirements.