Software Requirements Specification
for
Library Reservation System

Version 1.0

Prepared by Cem Akçiçek (122200044), Ece Doğa Gül (121200123), Berkay Özdelice (119200057), Muhammed Ali Akdoğan (122202038) and Emirhan İnan (122202085)

CODING HELPFUL WEBSITES

ABCDE

13.11.2024

Table of Contents

5. Other Nonfunctional Requirements

5.1 Performance Requirements
5.2 Reliability Requirements
5.3 Security Requirements
5.4 Maintainability Requirements
5.5 Portability Requirements

6. Other Requirements

6.1 Glossary Appendix
6.2 Analysis Models Appendix
6.2.1 Data Flow Diagram (DFD)
6.2.2 Class Diagram
6.2.3 State-Transition Diagram
6.3 Issues List Appendix

1. Introduction

1.1 Purpose

This SRS describes the software functional and nonfunctional requirements for the draft 1 version 1.0 of the Library Reservation System (LRS). The system is meant to help college library users to reserve study areas. It also allows library staff to check reservations.

1.2 Document Conventions

In this document:

Bold is used for important terms and system parts. Italics is used to point out key features or details. Requirements are organized in numbered sections.

1.3 Intended Audience and Reading Suggestions

This document is intended for developers, project managers, users and testers.

1.4 Project Scope

The Library Reservation System is software that allows users to reserve study areas and meeting rooms in the library. Its goal is to make the reservation process easier and more efficient for both users and library staff. This system helps the library improve its services by offering a convenient way for users to book spaces online.

1.5 References

The following documents were used to help create this SRS:

-Library Reservation System Vision Document, Version 1.1, August 2024.

-IEEE Standard for SRS, IEEE 830-1998.

-Library User Interface Guidelines, Version 2.0, September 2024.

2. Overall Description

2.1 Product Perspective

This software is a self-contained product. University students can make library reservations with this system. It interacts with users through a user interface(UI), reservation management and mail domain check feature. This system operates without dependence on external systems.

2.2 Product Features

User Interface: A simple and user-friendly interface allows users to input the details about their reservations such as time and library location selection.

Reservation Management: User can select a time and library location from the list for the reservation. System checks for capacity and approves or rejects reservation accordingly.

University Email Validation: The system ensures that only university students can make reservations through the mail domain check feature.

Capacity Check: System tracks the available capacity for each library based on predetermined capacity limits.

Confirmation and Rejection: System confirms or rejects reservation request based on available capacity in libraries and notifies the outcome to the users.

2.3 User Classes and Characteristics

This sysem is designed for university students.

User: The users are university students who use the system for daily reservation operations. Users can only make selections for library reservations and waits for approval from the system.

Database Administrator: Manages and maintains the system database. They can control capacity management.

System Administrator: Manages and maintains the system software, handling performance improvements and managing general configurations.

2.4 Operating Environment

Library Reservation System works on desktop, tablet and mobil devices.

System works on Chrome, Firefox, Safari and Edge.

2.5 User Documentation

A PDF document will ve provided, explaining how to use the system step by step. It will include instructions on how to make reservations and interact with the user interface.

2.6 Assumptions and Dependencies

The system requires an internet connection. Users must have a university email address. The system is compatible with mobile and desktop devices.

3. System Features

3.1 Reserve Study Space

3.1.1 Description and Priority

University students with appropriate edu domained university email addresses can reserve seats in various libraries on campus.

3.1.2 Stimulus/Response Sequences

Stimulus: Student requests to reserve a study space for a specific date and time.
Response: System checks availability for the selected date, time, and location, then displays confirmation or rejection accordingly.

3.1.3 Functional Requirements

Reserve.Place: The system shall allow a student using a domain check feature to reserve a study space for a specific date and time.

Reserve.EmailCheck: The system shall check that the user's email address is an appropriate edu domained address before allowing access to reservation features.

Reserve.TimeSelect: The system shall allow the student to select the desired reservation time, ensuring it is within the library's operating hours.

Reserve.LocationSelect: The system shall allow the student to select the desired library location from a predefined list of available libraries on campus.

Reserve.CheckAvailability: The system shall check availability for the selected date, time, and library location and display a confirmation or rejection message.

3.2 Mail Domain Check Feature

3.2.1 Description and Priority

The system will ensure that only users with university-affiliated email addresses are able to access and make reservations, preventing unauthorized access.

3.2.2 Stimulus/Response Sequences

Stimulus: A user attempts to make a reservation with an email address.
Response: The system checks if the email domain is a valid edu domain. If valid, access is granted to the reservation features; otherwise, the user is notified of the restriction.

3.2.3 Functional Requirements

DomainCheck.VerifyEmail: The system shall verify that the email address has a valid edu domain to allow access.

DomainCheck.RestrictAccess: If the email domain is not a valid edu domain, the system shall restrict access and display a notification about the requirement.

DomainCheck.Registration: The system shall enforce email domain restrictions during the reservation process, only allowing users with valid edu domained email addresses to make a reservation.

4. External Interface Requirements

4.1 User Interfaces

With the intention of allowing users to make reservations through the specified library, we designed a website that is visually appealing, useful and last but not least uncomplicated. A simple outline of this website is shown in the figure below.

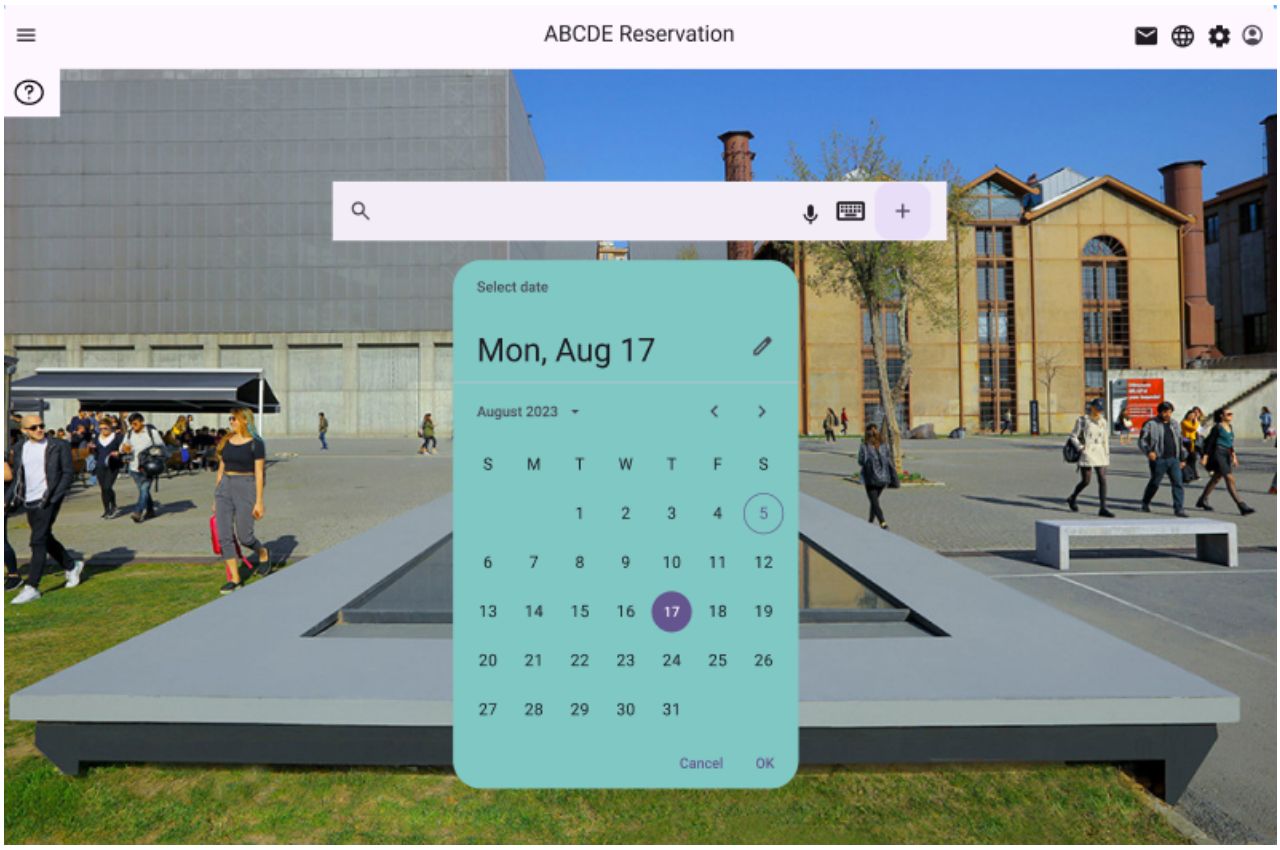

Fig.4.1. Outline of the Website

The first screen that the user will encounter when entering the website is as shown in the figure. On this screen, there are buttons where the user can log in, change the language according to their preference, and observe their own profile. In addition, a search bar is provided where users can search for resource information in the library, as well as a date box where users can select the reservation date.

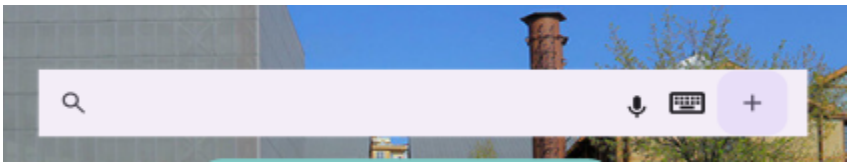

Fig 4.2. Search Bar

It is open to various uses with the bar in Fig.4.2, voice and keyboard buttons where users can search.

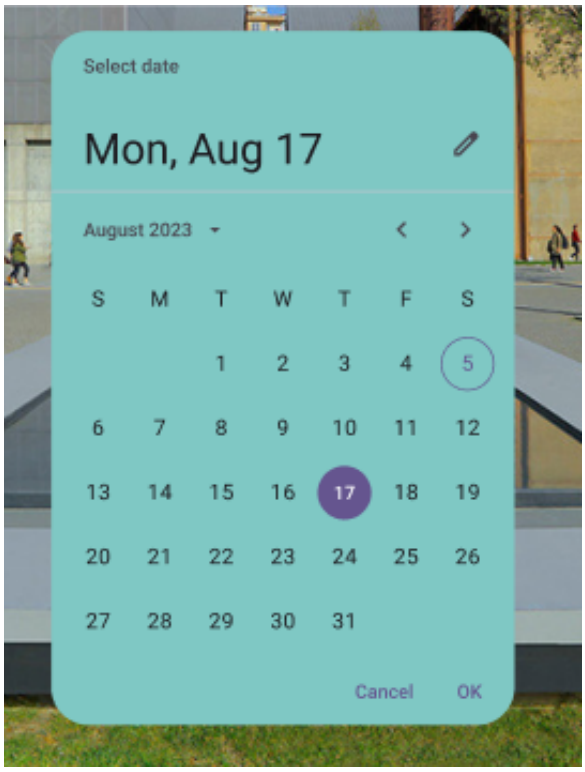

Fig 4.3. Date box

The icon where the date data will be entered by the user is shown in the Fig.4.3.

4.2 Software Interfaces

The Library Reservation System interfaces with several software components, databases, and tools to deliver a comprehensive experience for users and administrators. Below are the key software interfaces and the nature of their connections.

4.2.1 Database Interface

Database Name: LibraryDB (version 2.5)

Type: SQL-based relational database (e.g., MySQL or PostgreSQL)

Purpose: Manages and stores all data related to users, books, reservations, and library policies.

Data Flow: The system writes new reservations, updates availability statuses, and retrieves information on books and users.

Services Needed:

Insert, Update, and Query for book availability.

User account management, including registration, login, and activity logs.

Reservation details, including creation, update, and cancellation.

Data Sharing Mechanism: Access is managed through secure API calls, and SQL queries are handled via a dedicated service layer in the system.

Implementation Constraints: To ensure consistency, a transactional approach is used for database operations involving multiple tables (e.g., reservation and book status updates).

4.2.2 Operating System

Operating System: Windows Server 2022 or Linux-based servers.

Purpose: Hosts the Library Reservation System's backend and ensures scalability and security.

Services Needed: File handling, process management, and network connectivity.

Nature of Communication: System services and components interact with the OS through standard libraries and OS-provided APIs.

Implementation Constraints: The system must support both Windows and Linux environments, with configurations optimized for each.

4.2.3 Application Server

Server Type: Apache Tomcat or Microsoft IIS.

Purpose: Handles HTTP requests and processes API calls for the Library Reservation System.

Data Flow:

Incoming: Requests from the user interface for book searches, reservation creation, and user logins.

Outgoing: Responses with book details, reservation confirmations, and user account information.

Data Sharing Mechanism: The application server connects with the database and other components using RESTful APIs and XML or JSON as the data format.

Implementation Constraints: The system must use SSL/TLS for secure communication, ensuring data integrity and confidentiality during transmission.

4.2.4 User Interface (UI) Framework

Framework Name: React.js (version 18) or Angular (version 15).

Purpose: Provides a responsive, interactive interface for library patrons and administrators.

Data Flow:

Incoming: User requests (e.g., search, reserve, or cancel reservation).

Outgoing: Display results for searches, confirmations for reservations, and error messages.

Data Sharing Mechanism: The UI communicates with the backend via RESTful API endpoints, processing JSON data to render it dynamically on the frontend.

Implementation Constraints: The UI must be compatible with modern web browsers and responsive to accommodate various device sizes.

4.2.5 Payment Gateway API (Optional)

Provider Name: PayPal API (version 2.0) or Stripe API (version 3.1).

Purpose: Processes payments for reservation fees or overdue fines.

Data Flow:

Incoming: User payment information and transaction requests.

Outgoing: Payment confirmations and transaction details.

Data Sharing Mechanism: Payment information is transmitted securely using HTTPS with the gateway's API.

Implementation Constraints: PCI compliance is required for all data related to financial transactions, and the API must follow secure coding standards.

4.2.6 Notifications Service

Tool: Email/SMS service, such as SendGrid API (version 1.0) or Twilio API.

Purpose: Sends reservation confirmations, reminders for upcoming reservations, and overdue notifications to users.

Data Flow:

Outgoing: Messages include user information, reservation details, and alert types.

Data Sharing Mechanism: Uses HTTPS with JSON or XML payloads to integrate seamlessly with the notification services.

Implementation Constraints: The service must comply with data privacy laws, ensuring user contact information is securely managed and stored.

4.2.7 Logging and Monitoring Tool

Tool: ELK Stack (Elasticsearch, Logstash, Kibana) or Prometheus.

Purpose: Monitors application performance, logs system activity, and assists with troubleshooting.

Data Flow:

Outgoing: Logs for user actions, API responses, and system errors.

Data Sharing Mechanism: The tool aggregates and analyzes logs in real-time, providing insights for maintenance.

Implementation Constraints: Sensitive information within logs must be obfuscated to maintain user privacy.

Each of these components connects through secure APIs, maintaining data consistency and operational integrity across the Library Reservation System. Detailed API documentation outlines the protocols, data formats, and security requirements for each service, ensuring robust interaction between components and compliance with organizational and legal standards.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Response Time: The website should load within 10 seconds under normal conditions and within 25 seconds during peak loads.

Scalability: The system should support at least 50 concurrent users without any degradation in performance.

Reservation Processing: A reservation request should be processed within 5 seconds.

5.2 Reliability Requirements

Availability: The website should maintain an uptime of at least 99.5%, excluding scheduled maintenance.

Data Consistency: Reservation data must be updated in real-time to reflect accurate availability.

5.3 Security Requirements

User Authentication: Only users with valid university email domains should be able to access the system.

5.4 Maintainability Requirements

Code Modularity: The code should be modular to allow easy updates without impacting the entire system.

Bug Tracking and Fixing: A system for tracking, reporting, and addressing bugs within a specified timeframe should be in place.

5.5 Portability Requirements

Cross-Browser Compatibility: The website should function correctly on Chrome, Firefox, Safari, and Edge.

Device Compatibility: The website should be fully responsive, working seamlessly on desktops, tablets, and mobile devices.

6. Other Requirements

A. Glossary Appendix

A - D

- **API (Application Programming Interface)**: A set of protocols and tools that allows different software components to communicate and share data.
- **Application Server**: A server that hosts applications, handling HTTP requests and executing backend operations.
- **Availability**: The percentage of time the system is operational and accessible.
- **Concurrent Users**: The number of users interacting with the system simultaneously.
- **Data Consistency**: Ensuring that all data across the system is accurate, up-to-date, and synchronized.
- **Domain Check**: A feature that verifies email domains, allowing access only to users with specific email addresses.

E - G

- **edu Domain**: Email addresses that end with ".edu," usually associated with educational institutions, which is the BILGI University here.
- **ELK Stack**: A collection of tools (Elasticsearch, Logstash, Kibana) used for monitoring and logging application performance and user activity.
- **EmailCheck**: A feature that verifies whether a user's email address is from a valid edu domain.
- **External Interface Requirements**: Specifications for how the system interacts with users, other software, or hardware.

L - M

- **LibraryDB**: The database managing library-related information, including reservations, user data, and book status.
- **Logging**: Recording system events and user actions for monitoring and troubleshooting.
- **Maintainability**: The ease with which the system can be updated or modified.
- **Modularity**: Organizing code into separate, reusable parts to facilitate maintenance and updates.

N - R

- **Notification Service**: A tool for sending emails or SMS notifications to users regarding reservations or overdue notifications.

- **Operating System (OS)**: The underlying software that supports system operations, such as file management and network connectivity.
- **PCI Compliance (Payment Card Industry)**: Security standards to protect credit card information and ensure secure transactions.
- **Portability**: The ability of the system to function across different platforms, such as browsers and devices.
- **Reservation Processing**: The handling of reservation requests, ensuring quick responses within a specified timeframe.
- **Response Time**: The time it takes for the system to process a request and return a result to the user.

S - T

- **Scalability**: The capacity of the system to handle an increasing number of users or requests without performance degradation.
- **SSL/TLS (Secure Sockets Layer / Transport Layer Security)**: Protocols for encrypting communications to ensure secure data transmission.
- **Stimulus/Response Sequences**: Events that trigger actions within the system, like a user request (stimulus) followed by a system response.
- **UI (User Interface)**: The visual and interactive elements users engage with when interacting with the system.

B. Analysis Models Appendix

B.1 Data Flow Diagram (DFD)

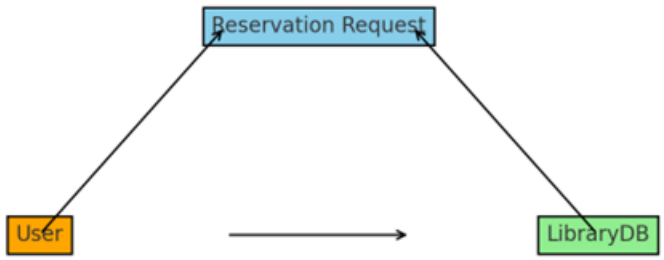

Figure 1 shows the reservation requests' flow diagram during the handling of the code.

B.2 Class Diagram

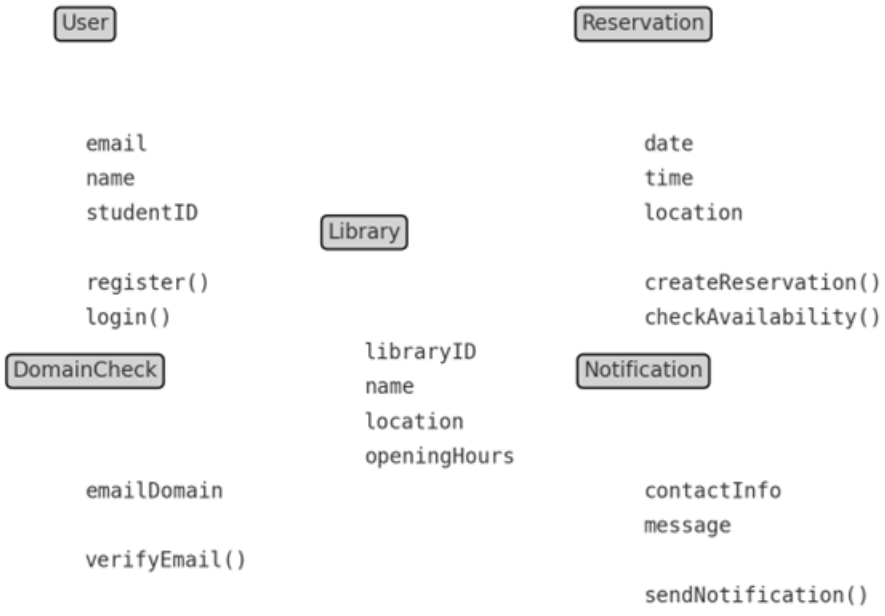

Figure 2 shows the Class Diagram which represents the structure of the system.

B.3 State-Transition Diagram

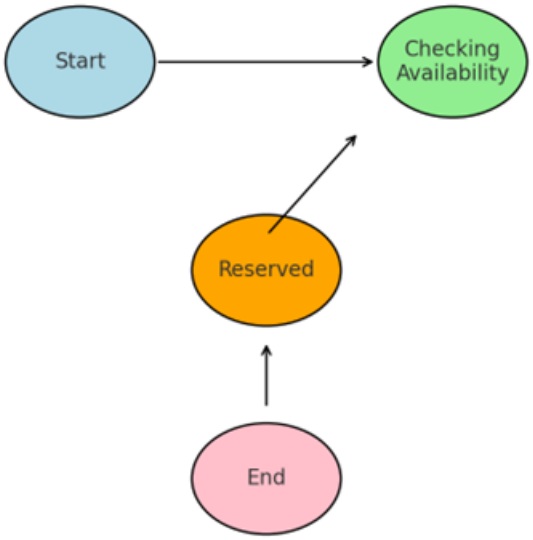

Figure 3 shows the State-Transition Diagram which illustrates the states an object can take and the transitions in-between.

C. Issues List Appendix

Issue ID	Issue Description	Status	Owner	Priority	Resolution Due Date	Comments
001	Integrating the design interface according to needs	Under Discussion	Team Members	High	TBD	Decision needed by all team members to finalize the UI
002	Confirm location estimates to finalize the task's purpose and interface components	Under Discussion	Team Members	High	TBD	Location is intricately related to the design interface and UI. All previous decisions will be relevant here
003	Determine if mobile app compatibility is needed for reservation system	Pending	Development Team	Medium	TBD	Pending uptime consideration by applicable team members

Figure 4 diagram shows the non-resolved issues that must be taken care of by the orders of High or Low. Team members are expected to take the appropriate course of action to resolve these issues in the upcoming weeks.