Loyola-ICAM College of Engineering and Technology, Nungambakkam

SOFTWARE REQUIREMENTS SPECIFICATION

For

MVD TO-DO LIST APPLICATION

Prepared by:-

Daniel Thomas J 311120205020
David Ajay 311120205021
Mohamed Rameez N 311120205038
Vishal M 311120205061

Academic Year: *2023-2024*

1. Introduction

1.1 Purpose

The primary purpose of this document is to outline the requirements for the development of the To-Do List Application. This document provides a comprehensive description of both the functional and non-functional requirements specified for the project. The project aims to create a user-friendly digital platform for managing tasks and to-do lists, enhancing productivity and organization for users.

1.2 Document Conventions

- The entire document should be justified.
- Convention for Main title:
  Font face: Times New Roman
  Font style: Bold
  Font Size: 14
- Convention for Subtitle:
  Font face: Times New Roman
  Font style: Bold
  Font Size: 12
- Convention for Body:
  Font face: Times New Roman
  Font Size: 12

1.3 Scope of Development Project

The To-Do List Application aims to transition the traditional manual task tracking process into a modern, web-based system. It caters to individuals and teams looking to manage tasks efficiently, prioritize work, and keep track of their progress.

The project targets both task creators and users. It provides a comprehensive user interface for task management, including creating, editing, and tracking tasks. The application's features are flexible and can be customized to suit different requirements, making it suitable for various domains.

The project offers scalability and adaptability, allowing for the addition of new features and modules as needed. The technology stack chosen for this project includes React for the front-end and Spring Boot for the back-end, ensuring performance, cross-platform compatibility, and ease of development.

1.4 Definitions, Acronyms and Abbreviations

React -> JavaScript library for building user interfaces
Spring Boot -> Framework for building Java-based web applications
API -> Application Programming Interface
UI -> User Interface

CRUD -> Create, Read, Update, Delete (basic operations in database systems)
REST -> Representational State Transfer (architectural style for web services)
SRS -> Software Requirement Specification

1.5 References

 Books
  - "Learning React" by Alex Banks and Eve Porcello
  - "Spring Boot in Action" by Craig Walls
  - "RESTful Web Services" by Leonard Richardson and Sam Ruby
 Websites
  - React documentation: https://reactjs.org/
  - Spring Boot documentation: https://spring.io/projects/spring-boot
  - React Router documentation: https://reactrouter.com/

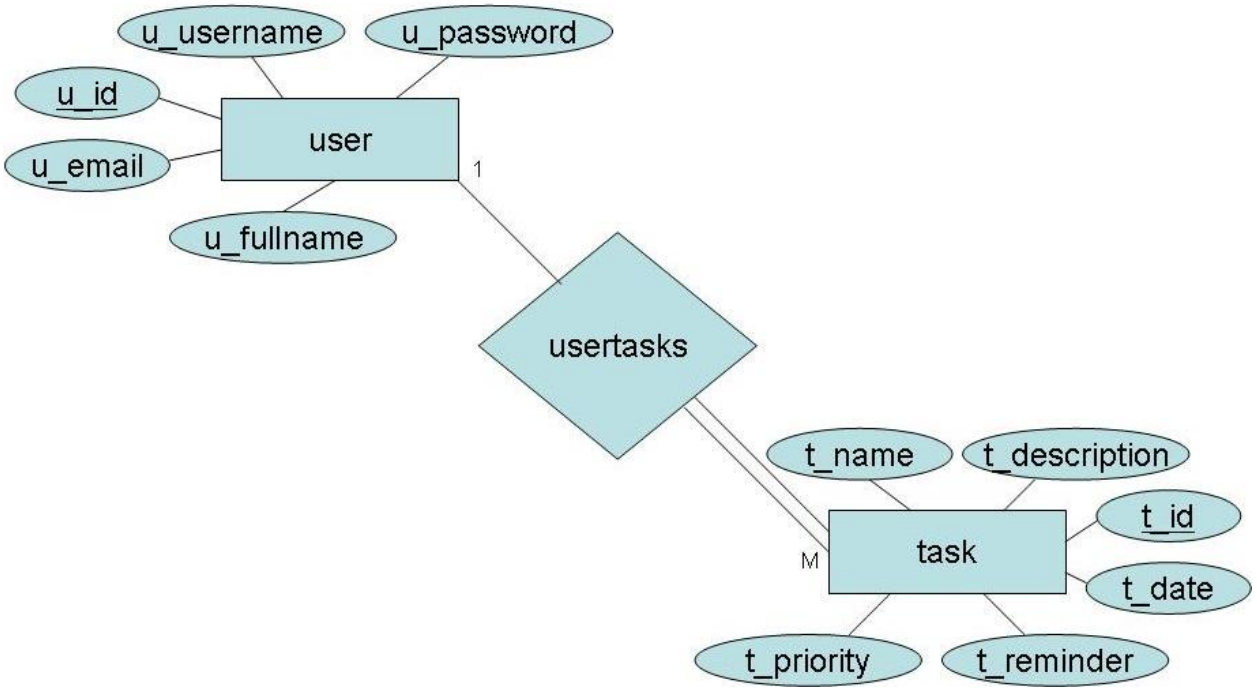
2. Overall Descriptions

2.1 Product Perspective

The To-Do List Application is designed as a web-based task management system that facilitates task creation, organization, and tracking. It is a user-friendly platform that helps individuals and teams manage their daily tasks efficiently. This system is intended to replace traditional pen-and-paper or digital to-do lists with a modern, digital solution.

2.2 Product Function

Entity Relationship Diagram of To-Do List Application



The To-Do List Application provides a set of core functionalities to enhance task management. Key features include:

- Task Creation: Users can create new tasks with titles, descriptions, due dates, and priorities.
- Task Organization: Tasks can be categorized and tagged for better organization.
- Task Editing and Deletion: Users can edit or delete tasks as needed.
- Task Prioritization: Users can assign priorities to tasks.
- Search and Filter: Users can search for tasks and filter them based on various criteria.
- User Authentication: User registration and authentication ensure secure access.
- Notifications: Users can receive notifications for important task deadlines.
- Database Management: The application stores tasks and user data securely.
- User Profile: Users can manage their profiles, including account details.

2.3 User Classes and Characteristics

The To-Do List Application caters to two main user classes:

1. Registered Users: These users are individuals who create accounts to access the application's full range of features. Registered users can create, edit, and manage tasks. They can also customize their profiles and receive task-related notifications.

2. Guest Users: While not explicitly mentioned in the diagram, guest users may access a limited set of features without creating accounts. This could include viewing public tasks or browsing available tasks without the ability to create or modify them.

The characteristics of each user class are as follows:

- Registered Users:
  - Can create and manage tasks.
  - Have access to user-specific features.
  - Can personalize their profiles.
  - Receive notifications.

- Guest Users:
  - Limited access to some features.
  - May explore available tasks without account creation.

2.4 Operating Environment

The To-Do List Application operates in a web-based environment and is accessible through modern web browsers such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari. It is designed to be cross-browser compatible to ensure a consistent user experience.

The minimum recommended system requirements for accessing the application are as follows:

- Internet Connection: Broadband or equivalent for optimal performance.
- Web Browsers: Google Chrome, Mozilla Firefox, Microsoft Edge, or Safari (latest versions).
- Screen Resolution: Recommended minimum resolution of 1280x800 pixels.
- Input Devices: Standard keyboard and mouse.

2.5 Assumptions and Dependencies

Assumptions:
- The application's code will be thoroughly tested and free from critical errors.
- User-friendly interface design ensures ease of use.
- User data and task details will be securely stored in a database accessible to the application.
- The application is available 24/7 for users.
- Users must have valid usernames and passwords to access their accounts and perform actions.

Dependencies:
- The application relies on specific hardware and software configurations for proper operation.
- Development and deployment depend on defined requirements and specifications.
- End-users, especially administrators, should have a clear understanding of the application's functionality.
- The system should have a general report storage capability.
- Database updates for task management must be recorded accurately.

2.6 Requirements

Software Configuration:
- Front-end: React (JavaScript library)
- Back-end: Spring Boot (Java-based framework)
- Database: [Specify the database management system (DBMS) used]
- Operating System: Windows, Linux, macOS
- Integrated Development Environment (IDE): [Specify the IDE used]

2.7 Data Requirement

The application takes user queries as input, including task creation, modification, and retrieval requests. Output includes solutions to queries and task-related information displayed to users.

3. External Interface Requirement

3.1 GUI

The To-Do List Application offers a user-friendly graphical interface for users and administrators to manage tasks efficiently. Key GUI features include:

- Task Management: Users can easily create, update, and view task details.
- Reports: Quick access to task-related reports, such as tasks completed or overdue.
- Search: Users can search for specific tasks by title, category, or due date.
- Customizability: The administrator can customize the user interface to fit specific needs.
- Standard Design: The user interface follows a simple and standardized design template.
- User Authentication: Secure login and registration functionality.

Login Interface:
Users can either log in with their existing accounts or register to create a new account. The login interface prompts users to enter their username and password, with error messages displayed for incorrect inputs.

Task Search:
Users can search for tasks by specifying criteria such as title, category, or due date, making it easy to locate specific tasks.

Category View:
Category view displays task categories and allows administrators to add, edit, or delete categories as needed.

Administrator Control Panel:
The control panel provides administrators with the ability to manage users, tasks, and task categories. It also includes functionality for user authentication.

4. System Features

The To-Do List Application prioritizes user account security by implementing the following features:

- User Authentication: Members must authenticate using their unique usernames and passwords.
- Administrator Monitoring: Administrators can monitor and update account statuses, enforce borrowing limits, and assign fines to members for late returns.
- Data Accountability: Members are restricted from accessing other users' accounts; only administrators have access to all member accounts.

5. Other Non-functional Requirements

5.1 Performance Requirement

The To-Do List Application serves as a primary task management system within the university, interacting with staff and students across different campuses. Performance requirements include:

- Fast and Accurate Performance: The system must deliver fast and accurate responses to user requests.
- Error Handling: Robust error handling to prevent data loss and minimize downtime.
- Scalability: Ability to handle a large volume of tasks and users without performance degradation.

5.2 Safety Requirement

Safety measures include regular database backups to prevent data loss in case of crashes, proper power backup systems (e.g., UPS/inverters) to handle power supply failures, and security protocols to protect against viruses and operating system failures.

5.3 Security Requirement

Security measures are implemented to ensure data protection and access control:

- Secured Database: The system utilizes a secure database.
- User Access Control: Normal users can read information but cannot modify data except for their personal information.
- User Authentication: Strong user authentication mechanisms are in place.
- Password Security: Ensuring that user passwords cannot be easily compromised.
- Role-Based Access: Admin and member accounts are separated, restricting non-admin access to the database.

5.4 Requirement Attributes

- Multiple Admins: Multiple administrators have the right to make changes to the system.
- Open Source: The project should be open source.
- User-Friendly: The system should prioritize user-friendliness.
- Easy Installation: Users should be able to easily download and install the system.

5.5 Business Rules

Business rules encompass project costs, discount offers, and adherence to rules and regulations. Users, both admin and members, are expected to follow established rules and protocols.

5.6 User Requirement

Users include members and administrators of the university. Members are assumed to have basic computer and internet skills, while administrators possess more in-depth knowledge of system internals. Proper user documentation, including user interfaces, user manuals, online help, and installation guides, must be provided.

Administrators offer various facilities to users, including backup and recovery, password retrieval, data migration, data replication, auto-recovery, file organization, and regular server maintenance and updates.

6. Other Requirements

6.1 Data and Category Requirement

The To-Do List Application classifies users into different categories, including teaching staff, administrators, librarians, and students. Access rights are determined based on user categories. Administrators have extensive privileges, allowing them to modify, delete, and append data. Other users, except librarians, are restricted to retrieving information from the database.

Similarly, the application deals with different categories of tasks (or to-do items). Each category must have relevant data organized in a specific format. Users should be able to access tasks categorized by their type, making data retrieval and organization essential.

6.2 Appendix

Appendix A: Contains information related to administration, abbreviations, acronyms, and assumptions.
Appendix B: Focuses on business rules and information related to books.
Appendix C: Covers information related to classes, clients, and conventions.
Appendix D: Provides details about data requirements and dependencies.
Appendix G: Contains information related to the graphical user interface (GUI).
Appendix K: Covers keys and unique identifiers.
Appendix L: Includes information about the library, librarian, and members.
Appendix M: Focuses on member-related information.

Appendix N: Addresses non-functional requirements.
Appendix O: Contains details about the operating environment.
Appendix P: Discusses performance, perspective, and purpose.
Appendix R: Contains information related to requirements and requirement attributes.
Appendix S: Focuses on safety, scope, security, and system features.
Appendix U: Includes information about users, user classes and characteristics, and user requirements.

6.3 Glossary

The following glossary provides definitions and explanations of conventions, abbreviations, and acronyms used in this document and the project:

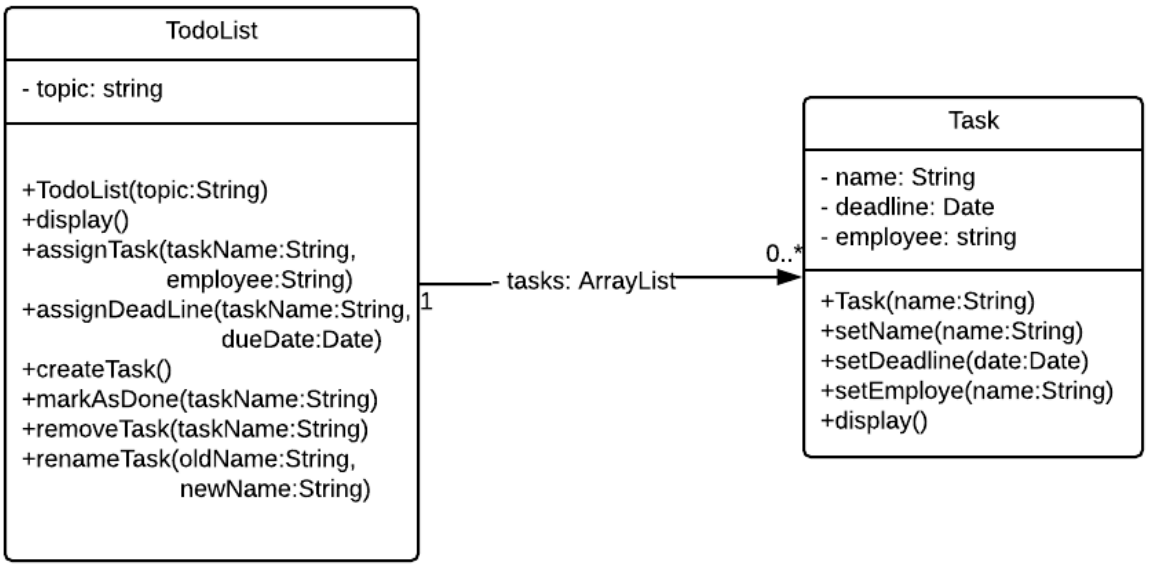
- Administrator: A login ID representing a user with user administration privileges within the software.
- User: A general login ID assigned to most users.
- Client: The intended users for the software.
- SQL: Structured Query Language; used to retrieve information from a database.
- SQL Server: A server used to store data in an organized format.
- Layer: Represents a section of the project.
- User Interface Layer: The section of the application that users interact with directly.
- Application Logic Layer: The section of the application referring to the Web Server, where computations are completed.
- Data Storage Layer: The section of the application where all data is recorded.
- Use Case: A broad-level diagram of the project showing a basic overview.
- Class Diagram: A static structure diagram that describes the system's structure, including classes, attributes, and relationships.
- Interface: A means of communication across different mediums.
- Unique Key: A unique identifier used to differentiate entries in a database.

6.4 Class Diagram

In the context of the To-Do List Application, a class diagram is used to represent the key classes and their relationships. Each class defines an abstract, user-defined data type with attributes that describe its properties and operations that can be performed on instances (objects) of that class. The class diagram captures the static model of the system, illustrating how classes are structured and how they relate to each other.

In the class diagram, relationships are depicted using role names and multiplicities to indicate the associations, aggregations, and generalizations between classes.

This class diagram represents the core structure of the To-Do List Application and serves as a blueprint for understanding how the different classes interact to fulfill the application's functionality.



The following are the main classes and their relationships in the To-Do List Application:

1. Task
   - Attributes: title, description, dueDate, priority
   - Operations: createTask(), editTask(), deleteTask(), setPriority()

2. User
   - Attributes: username, password, email, role
   - Operations: createUserAccount(), authenticateUser()

3. Administrator
   - Role: Extends the User class
   - Operations: manageUsers(), customizeUI()

4. Category
   - Attributes: name, description
   - Operations: createCategory(), editCategory(), deleteCategory()

Relationships:

- Task-User Relationship: Users (Members and Administrators) can create, edit, and manage tasks.
- Administrator-User Relationship: Administrators have extended privileges as users.
- Task-Category Relationship: Tasks can be associated with categories.
- Category-Task Relationship: Categories can have multiple associated tasks.