SOFTWARE REQUIREMENTS SPECIFICATION

For

TO - DO - LIST

Prepared by:-

Joicy A
Jefna Besvi B
Kamali G
Tresa Niranjini S

Academic Year: 2023-2024

1.Introduction

1.1 Purpose

The To-Do List Project aims to streamline the age-old practice of creating and managing task lists, catering to both personal and professional needs. In a world where tasks are prioritized based on their importance, this project offers a user-friendly website as the solution. It allows users to efficiently plan and execute their actions, maintain a clear record of notes, action items, and ideas, and prioritize tasks effortlessly.

By embracing technology, this project seeks to replace traditional paper-based lists with a digital, accessible, and user-focused platform. What sets it apart is its simplicity; no sign-in or personal information is required. Users can easily record, mark completed tasks, and revisit their lists anytime, ensuring that important tasks are never forgotten. Whether managing chores at home or tasks in the workplace, the To-Do List Project offers a versatile tool for productivity and organization.

1.2 Document Conventions

Entire document should be justified.

Convention for Main tile

- Font face: Times New Roman
- Font style: Bold
- Font Size: 14

Convention for Sub title

- 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 Project encompasses a comprehensive scope, with the primary objective of streamlining the practice of creating and managing task lists to meet both personal and professional needs. This endeavor centers on the development of a user-friendly website as the primary solution for efficient task management. Users will benefit from robust functionality, enabling them to plan and execute tasks seamlessly, maintain organized records of notes, action items, and ideas, and effortlessly prioritize their to-dos.

Embracing modern technology, the project seeks to replace traditional paper-based lists with a digital platform designed for accessibility and ease of use. Notably, this platform stands out for its simplicity, as it requires no sign-in or personal information. Users can effortlessly record tasks, mark completed ones, and revisit their lists at their convenience. This versatility extends to both personal tasks, such as home chores, and professional tasks, facilitating workplace organization.

In summary, the To-Do List Project offers a user-friendly digital solution that simplifies task management across various domains while promoting accessibility and productivity.

1.4 Definitions, Acronyms and Abbreviations

- JAVA -> platform independence
- SQL-> Structured query Language
- ER-> Entity Relationship
- UML -> Unified Modeling Language
- IDE-> Integrated Development Environment
- SRS-> Software Requirement Specification
- ISBN -> International Standard Book Number
- IEEE ->Institute of Electrical and Electronics Engineers

1.5 References

Books

Software Requirements (Microsoft) Second EditionBy Karl E. Wiegers
Software Engineering: A Practitioner's Approach Fifth Edition By Roger S. Pressman

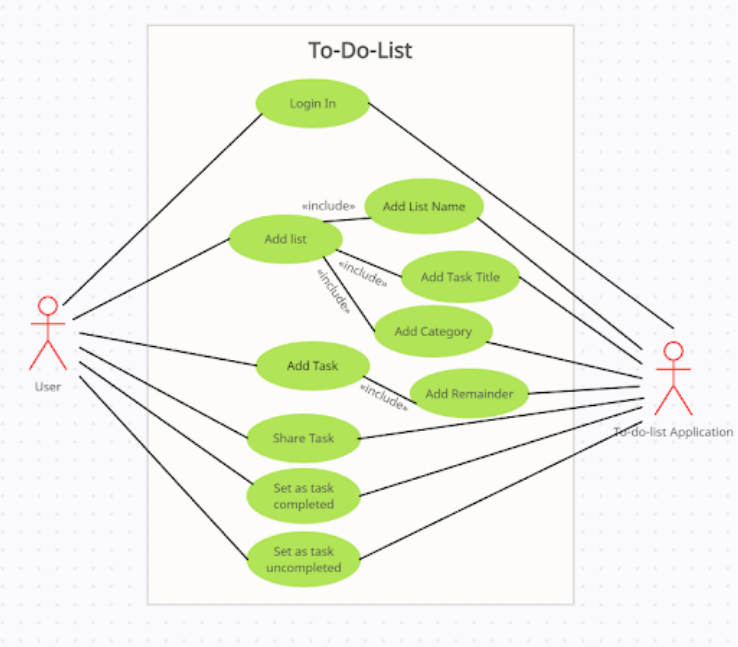
Websites

http://www.slideshare.net/

2. Overall Descriptions

2.1 Product Perspective

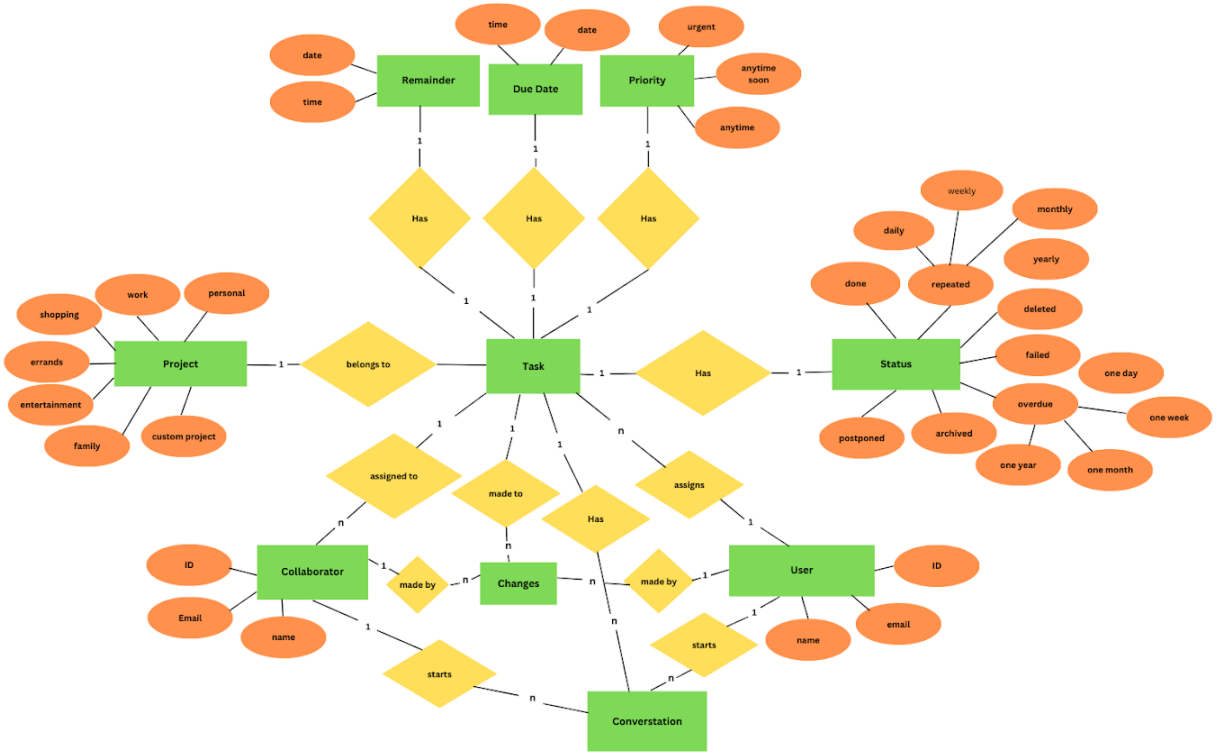
Use Case Diagram of To-Do-List



This is a broad level diagram of the project showing a basic overview. The users can be someone. This System will provide a functionality to schedule the list of works to be done. This project will be based on scheduling the task based on each priority or purpose. Further the user can add/update/modify the task which they would schedule.

2.2 Product Function

Entity Relationship Diagram of To-Do-List



The purpose of this project is to create a comprehensive to-do list for developing a product diagram. A product diagram is a visual representation of a product, highlighting its key features, components, and functionalities. This project aims to guide the process of creating a product diagram by breaking it down into manageable tasks. By following this to-do list, you can ensure a systematic and organized approach to developing a product diagram.

2.3 User Classes and Characteristics

End Users:

- May have varying levels of expertise.
- Typically use the product/service to accomplish specific tasks or goals.
- May have different levels of familiarity with technology.

Administrators:

- Responsible for configuring and managing the system.
- Need access to advanced settings and controls.
- Often have a higher level of expertise compared to regular users.

Managers/Supervisors:

- Monitor and oversee the work of other users.
- Require access to reports, analytics, and performance metrics.
- Need to make decisions based on the information provided.

Customers/Clients:

- External users who interact with the product or service.
- Seek specific information, make purchases, or provide feedback.
- May have varying levels of familiarity with the product or service.

Guest Users:

- Users who have limited access or no user account.
- Often restricted to basic features or public information.
- May include potential customers exploring the product/service.

Technical Support/Helpdesk:

- Provide assistance and troubleshoot issues for end users.
- Need access to detailed system information and logs.
- Require good communication and problem-solving skills.

Developers/IT Personnel:

- Responsible for system maintenance, updates, and customization.
- Require access to backend components and APIs.
- Need coding and technical skills to work with the system.

Compliance and Security Officers:

- Ensure that the product or service complies with legal and security standards.
- Monitor data privacy and security.
- Need access to audit logs and security settings.

Business Executives/Decision-Makers:

- Use data and insights from the product/service for strategic decisions.
- Require high-level reports and dashboards.
- May not be deeply involved in day-to-day operations.

Accessibility Users:

- Users with disabilities who require assistive technologies.
- Need accessible design, keyboard navigation, and screen reader support.
- May have specific preferences or requirements for customization.

Mobile Users:

- Access the product or service on mobile devices.
- Need responsive design and a mobile-friendly interface.
- May have limited screen real estate and different interaction patterns.

International Users:

- Users from different countries and language backgrounds.
- Require support for multiple languages, currencies, and cultural considerations.
- May have varying legal and regulatory requirements.

Power Users:

- Highly skilled and experienced users.
- Often use advanced features and shortcuts.
- May provide valuable feedback for product improvement.

2.4 Operating Environment

The operating environment of a computer system or application is a complex ecosystem consisting of hardware, software, and external factors. Hardware components like servers, storage devices, and client devices provide the foundational infrastructure. The choice of an operating system (OS) is critical as it manages hardware resources and serves as a platform for software applications. Software frameworks, libraries, and applications run on top of the OS, facilitating the development of functional systems. Additionally, networking infrastructure, security measures, and environmental factors like temperature and power stability play vital roles in ensuring the system's performance and reliability.

2.5 Assumptions and Dependencies

Assumptions:

- **Definition:** Assumptions are statements or conditions that project planners consider to be true, even though they may lack complete information or certainty.
- **Purpose:** Assumptions serve as the basis for decision-making and planning when there is uncertainty, enabling project teams to move forward.
- **Identification:** Assumptions should be explicitly identified, documented, and regularly reviewed throughout the project's lifecycle.
- **Examples:** Common assumptions include resource availability, technology functionality, and stakeholder cooperation.
- **Risks:** Unverified assumptions can introduce risks if they prove false, potentially impacting project timelines and outcomes.

Dependencies:

- **Definition:** Dependencies are relationships between project tasks, activities, or components where the timing or success of one is reliant on another.
- **Types:** There are four main types of dependencies: finish-to-start, start-to-start, finish-to-finish, and start-to-finish, each specifying how tasks are related.
- **Documentation:** Dependencies should be clearly documented in a dependency log, outlining the nature, direction, and impact of each dependency.
- **Critical Path:** Understanding dependencies is crucial for identifying the critical path, which determines the minimum duration of the project and highlights tasks that can't be delayed.
- **Risk Management:** Dependencies introduce potential risks; if a dependent task is delayed or encounters issues, it can impact subsequent tasks, so they need continuous monitoring and management.

2.6 Requirement

Software Configuration:-

- **Operating System:** Windows 10
- **Language:** JRE, Net beans 7.0.1 (front end) Microsoft SQL (back end)
- **Database:** MS SQL Server (back end)

Hardware Configuration:-

- **Processor:**Intel Core i5 or AMD Ryzen 5
- **Hard Disk:** 120-256GB
- **RAM:** 4GB or more

2.7 Data Requirement

The data requirements for a to-do list application include user data for authentication and personalization, task data with details like title, due date, and priority, category data for organizing tasks, relationships to link users with tasks and tasks with categories, collaboration data to enable shared lists, activity logs for tracking user actions, and configuration data for app settings and integrations. These elements collectively ensure the application's functionality, user experience, and collaborative capabilities.

3. External Interface Requirement

3.1 GUI

- The task list is where users view their to-do items at a glance, including essential details such as task names, due dates, and priorities. It serves as the central hub for task management, allowing users to quickly assess their workload.
- Clicking on a task within the list should open a detailed view, providing users with comprehensive information about the task, including descriptions, due dates, and current status. This feature enables users to access and modify task specifics easily.
- Users should have the ability to create, edit, and delete categories to help organize their tasks effectively. Categories add an extra layer of structure to the task management process, making it easier to sort and find tasks based on their classifications.
- Implementing a search bar and filtering options enables users to locate specific tasks efficiently. Users can search by keywords, filter tasks by due date or priority, and customize their view to suit their immediate needs.
- A notification system is essential to keep users informed of upcoming task due dates or any changes in shared tasks, promoting timely action and collaboration among users. Notifications serve as reminders and real-time updates, enhancing the overall usability of the to-do list application.
-

Login Interface:-

A login interface includes fields for users to enter their username or email and password, along with options like "Remember Me" and "Forgot Password." Users can also register or log in using social accounts.

Search:-

A search feature allows users to find specific tasks quickly by entering keywords or applying filters, enhancing task management efficiency.

Categories View:-

Categories view organizes tasks based on user-defined categories, helping users easily sort and access their tasks by grouping them into different categories or tags.

Control Panel:-

A control panel is a centralized interface that enables users to access and manage various settings, configurations, and controls within the application, providing a hub for customization and control.

4. System Features

System features refer to the specific functionalities and capabilities that a software or application provides to its users. These features are designed to fulfill specific tasks or requirements within the system.

- **Task Management:** Users can create, edit, and organize tasks, including setting due dates, priorities, and categories.
- **Collaboration:** The ability to share tasks and task lists with others, facilitating teamwork and project management.
- **Search and Filtering:** Users can search for specific tasks by keywords or apply filters to quickly find and access relevant tasks.

5. Other Non-functional Requirements

5.1 Performance Requirement

- **Response Time:** The application should respond to user actions within milliseconds or seconds to ensure a smooth and responsive user experience.
- **Scalability:** The system should be able to handle a growing number of users and tasks without performance degradation. It should scale horizontally or vertically as needed.
- **Load Handling:** The application should support a specified maximum number of simultaneous users or tasks without significant slowdowns or crashes, ensuring reliable performance under heavy usage.
- **Database Efficiency:** Database operations, such as task retrieval and updates, should be optimized to ensure fast response times, even as the amount of data stored in the system increases.

5.2 Safety Requirement

- All data should be transmitted securely over HTTPS to prevent eavesdropping.
- Sensitive data, such as user passwords, should be stored securely using strong encryption algorithms (e.g., crypt for password hashing).
- Implement user authentication to ensure that only authorized users can access the application.
- Use proper authorization mechanisms to control access to specific tasks or lists, ensuring that users can only modify their own data.

5.3 Security Requirement

- **Authentication and Authorization:** Implement user authentication to ensure that only authorized users can access the application**.**
- **Secure Communication:** Use HTTPS to encrypt data in transit between the client and server to prevent eavesdropping and data interception.

- **Input Validation:** Validate and sanitize user inputs to prevent SQL injection, cross-site scripting (XSS), and other injection attacks.

- **Session Management:** Implement secure session management to prevent session fixation, session hijacking, and session replay attacks.

- **Data Protection:** Encrypt sensitive user data stored in the database using encryption libraries or database features.

- **Cross-Site Request Forgery (CSRF) Protection:** Implement CSRF tokens to protect against CSRF attacks by ensuring that requests originate from trusted sources.

5.4 Requirement attributes

- User registration and login.
- User profile management.
- Secure password storage and authentication.
- Intuitive and user-friendly design.
- Task creation, editing, and deletion.
- Sorting and filtering options.
- Notifications for due dates and completed tasks.
- Drag-and-drop functionality for task prioritization.
- Task due dates and reminders.
- Task priority levels.

5.5 Business Rules

A business rule is anything that captures and implements business policies and practices. A rule can enforce business policy, make a decision, or infer new data from existing data.This includes the rules and regulations that the System users should abide by. This includes the cost of the project and the discount offers provided. The users should avoid illegal rules and protocols. Neither admit nor member should cross the rules and regulations.

5.6 User Requirement

Users should be able to register and log in securely, create, manage, and organize tasks with details like titles, descriptions, due dates, priorities, and statuses. Additionally, categorization and tagging of tasks, along with search and filtering capabilities, enhance task organization and retrieval. Reminders and notifications help users stay on top of deadlines, and a user profile feature adds personalization. For collaborative use, sharing and editing tasks with others is a valuable option.

1.User Registration and Authentication:

- Users should be able to register with a unique username and password.
- Users should be able to log in securely.
- Users should be able to reset their password if they forget it.
- There should be a "Remember Me" option for convenience.

2.Task Management:

- Users should be able to create, read, update, and delete tasks.
- Each task should have a title, description, due date, priority, and status (e.g., completed or pending).
- Tasks should be sortable by priority, due date, or status.
- Users should be able to mark tasks as completed.

3.Task Categorization and Tags:

- Users should be able to categorize tasks into different lists or categories
- Users should be able to add tags to tasks for better organization and search.

4.Search and Filtering:

- Users should be able to search for tasks using keywords.
- Users should be able to filter tasks based on categories, tags, or due dates.

5.Reminders and Notifications:

- Users should receive reminders or notifications for tasks that are due soon.
- Notifications can be sent via email or in-app notifications.

6. Other Requirements

6.1 Data and Category Requirement

There are categories that need to consider several key data categories. First, you'll manage user data, including usernames, email addresses, and securely hashed passwords for authentication. Tasks are the core of the application, each with a title, optional description, due date, priority level, and status (e.g., pending, in progress, completed). Implementing tags or categories can help users organize their tasks effectively. You can also include optional features like task lists or projects, collaboration capabilities with shared lists and permissions, reminders, task history, and notifications. Additionally, think about enabling search and filtering options for users to find tasks quickly. Your data will be stored in a chosen database system, with a backend server handling API requests, business logic, and database interactions. Create a user-friendly frontend interface for task management, secure user authentication, and ensure data security throughout. Finally, consider aspects like testing, deployment, documentation, scalability, performance, analytics, and backup procedures to make a robust and user-friendly to-do list application.

6.2 Appendix

A: Admin, Abbreviation, Acronym, Assumptions; B: Books, Business rules; C: Class, Client, Conventions; D: Data requirement, Dependencies; G: GUI; K: Key; N: Non-functional Requirement; O: Operating environment; P: Performance,Perspective,Purpose; R: Requirement, Requirement attributes; S: Safety, Scope, Security, System features; U: User, User class and characteristics, User requirement.

6.3 Glossary

The following are the list of conventions and acronyms used in this document and the project as well:

- **Administrator:** responsible for managing and overseeing the system, software, or platform they have access to.
- **User :** an individual who utilizes the to-do list application to create, manage, and track tasks.
- **Client:** 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 assignment referring to what the user interacts with directly
- **Application Logic Layer:** The section of the assignment referring to the Web Server. This is where all computations are completed
- **Data Storage Layer:** The section of the assignment referring to where all data is recorded
- Use Case: A broad level diagram of the project showing a basic overview
- **Class diagram:** It is a type of static structure diagram that describes the structure of a system by showing the system's cases, their attributes, and the relationships between the classes
- **Interface:** Something used to communicate across different mediums
- **Unique Key:** Used to differentiate entries in a database

6.4 Class Diagram

A class is an abstract, user-defined description of a type of data. It identifies the attributes of the data and the operations that can be performed on instances (i.e. objects) of the data. A class of data has a name, a set of attributes that describes its characteristics, and a set of operations that can be performed on the objects of that class. The classes' structure and their relationships to each other frozen in time represent the static model.