SOFTWARE REQUIREMENTS SPECIFICATION

**For**

**TASK MANAGEMENGT SYSTEM**

**Prepared By:-**

*GUHAN.K*

*DEEPAK PRAKASH.S*

*BHARANIDAR.S*

*DHARNIDHAR.S*

# Introduction

## Purpose

A task management system is designed to help individuals, teams, or organizations organize, track, and complete tasks efficiently. Its primary purpose is to streamline workflow, improve productivity, and ensure that work is completed in a timely manner.

It also provides a centralized location for organizing tasks, making it easy for individuals and teams to access and manage their work even the users can assign priority levels to tasks, helping them focus on high-priority items first and ensuring that critical work is addressed promptly.

## Document Conventions

* + - Entire document should be justified.
    - Convention for Main title

Font face: Times New Roman Font style: Bold

Font Size: 1

* + - 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

## Scope of Development Project

A task management system is a software application that helps teams and individuals manage their tasks and projects more efficiently. It allows users to create, assign, track, and manage tasks, set deadlines, and monitor progress. A task management system can be used for short-term, long-term, or permanent processes.

The scope of a development project for a task management system would include defining the goals, constraints, workflow management strategies, tasks, and deliverables. The project scope statement should identify the project deliverables and organize them into smaller elements called work packages, making them easier to manage.

## 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

## References

* + - Books

 Software Requirements and Specifications: "Getting Things Done: The Art of Stress-Free Productivity" by David Allen

"The Checklist Manifesto: How to Get Things Right" by Atul Gawande

"Scrum: The Art of Doing Twice the Work in Half the Time" by Jeff Sutherland

* + - Websites

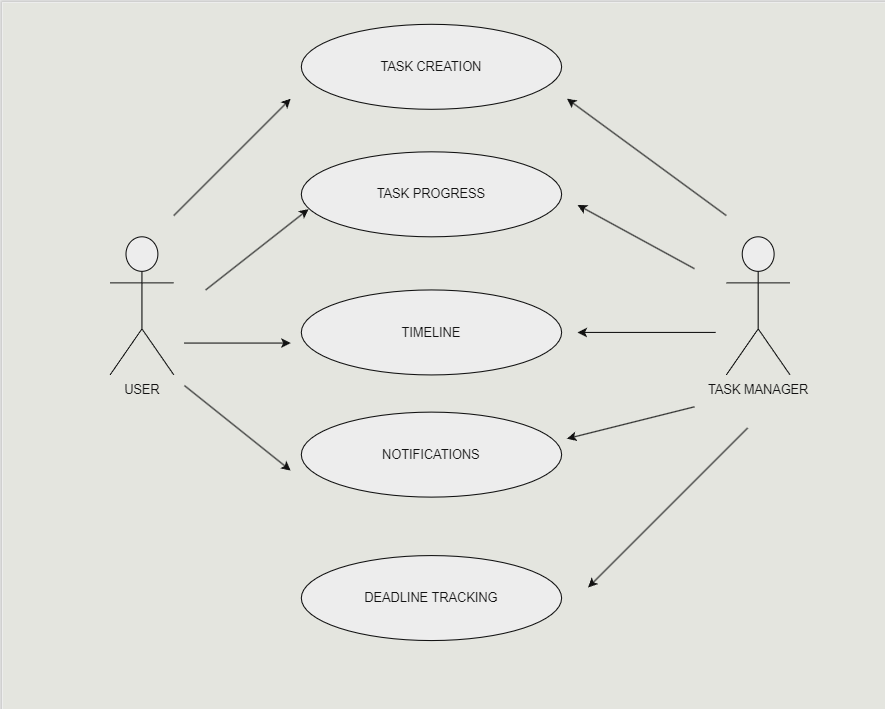
[**https://solr.apache.org/guide/solr/latest/deployment-guide/task-management.html/**](http://www.slideshare.net/)

[**https://hive.com/blog/task-management-software/**](http://ebookily.net/doc/srs-library-management-system)

# Overall Descriptions

## Product Perspective

Use Case Diagram of Task Management System

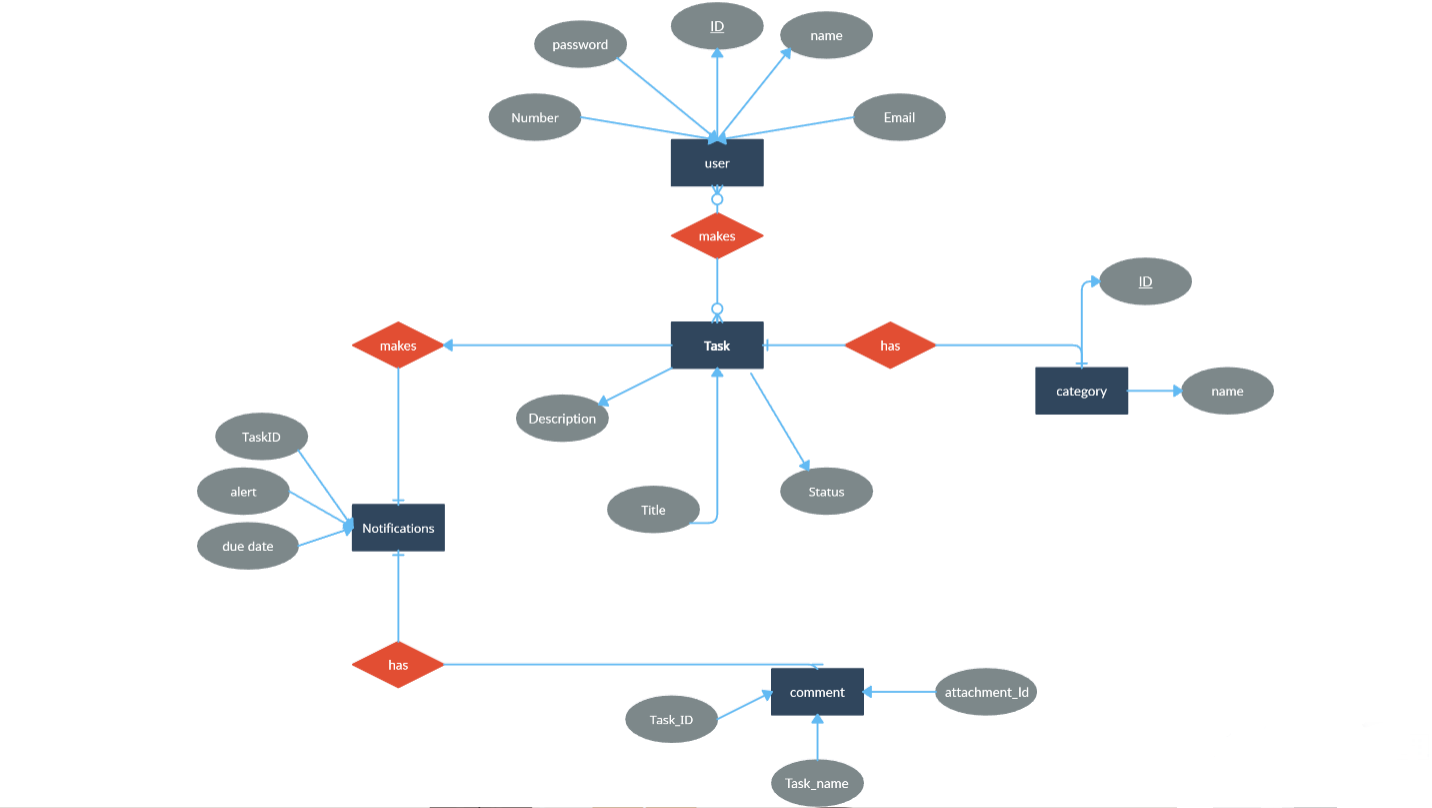
**

A Use Case Diagram offers a high-level visual representation of the interactions between users or external systems and a software system, showcasing various use cases and the relationships among them. The Use Case Diagram for the Task Management System provides a concise overview of the system's functionalities and user interactions.

The primary actors, such as "User" and potentially "Administrator," are identified, each associated with specific use cases. Common use cases include "Create Task," "Assign Task," "View Task Details," and "Generate Task Report." Relationships between actors and use cases illustrate the flow of interactions, showcasing how users engage with the system to perform various tasks. The diagram highlights the essential features, such as task creation, assignment, and reporting, emphasizing the core functionalities that users are expected to utilize.

## Product Function

Entity Relationship Diagram of Task Management System



An Entity-Relationship (ER) diagram provides a visual representation of the entities and the relationships between them in a database system. For a Task Management System (TMS), an ER diagram helps outline the structure of the database and how different components are related.

## User Classes and Characteristics

The system provides different types of services based on the type of users:

**User Class:**

UserID: Unique identifier for each user.

Username: User's display name or username.

Email: User's email address.

Password: User's login password.

Preferences: User-specific settings (e.g., theme, notification preferences).

**Task Class:**

TaskID: Unique identifier for each task.

Title: Title or name of the task.

Description: Detailed description of the task.

Due Date: Deadline for completing the task.

Priority: Priority level of the task (e.g., high, medium, low).

Status: Current status of the task (e.g., not started, in progress, completed).

CreatorID: UserID of the user who created the task.

**Project Class:**

ProjectID: Unique identifier for each project.

Title: Title or name of the project.

Description: Detailed description of the project.

StartDate: Start date of the project.

EndDate: Expected end date of the project.

Members: List of UserIDs for users involved in the project.

**Notification Class:**

NotificationID: Unique identifier for each notification.

UserID: UserID of the user to whom the notification is directed.

Type: Type of notification (e.g., task assignment, due date approaching).

Content: Information or message associated with the notification.

Timestamp: Date and time when the notification was generated.

**Task Manager Class:**

Users: A collection of User objects.

Tasks: A collection of Task objects.

Projects: A collection of Project objects.

Notifications: A collection of Notification objects.

Methods for managing and manipulating tasks, projects, users, etc.

## Operating Environment

The operating environment for a task management system is a dynamic and interconnected digital landscape that encompasses various platforms and devices to facilitate seamless user interaction. It is designed to operate efficiently across diverse environments, including mobile applications, and desktop software.

Native mobile applications for iOS and Android platforms provide a responsive and tailored experience for users on the go. The task management system is adaptable to different screen sizes and resolutions, ensuring a consistent and user-friendly interface across devices. Integration capabilities with third-party tools, calendars, and communication platforms enhance the system's versatility, allowing users to synchronize tasks with their existing workflows. Security measures are embedded to safeguard user data, and regular updates ensure compatibility with evolving technologies, reinforcing the system's resilience in a rapidly evolving digital environment.

## Assumptions and Dependencies

The Assumptions are:

* + - The users of the application have basic computer literacy and can navigate the application with ease.
    - The application is compatible with the operating system and hardware of the user’s device.
    - The users have a stable internet connection to access the application.
    - The users are aware of the project’s goals and objectives.
    - The users are aware of the tasks assigned to them and their deadlines.
    - The users have the necessary skills and knowledge to complete the tasks assigned to them.
    - The users will provide timely updates on the progress of their tasks.
    - The users will communicate effectively with other team members.

The Dependencies are:

* + - The application must be developed and tested before it can be deployed.
    - The application must be compatible with the database management system used by the organization.
    - The application must be integrated with other tools used by the organization.
    - The application must be scalable to accommodate future growth.
    - The application must be secure to prevent unauthorized access to sensitive information.

## Requirement

**Software Configuration:-**

This software package is developed using java as front end which is supported by sun micro system. Microsoft SQL Server as the back end to store the database.

Operating System: Windows NT, windows 98, Windows XP Language: Java Runtime Environment, Net beans 7.0.1 (front end)

Database: MS SQL Server (back end)

**Hardware Configuration:-**

* Processor:

Pentium(R)Dual core CPU

* Hard Disk: 40GB
* RAM: 256 MB or more

## Data Requirement

Reporting: Task management tools can create reports that show completed work and team member performance.

Collaboration: Task management software can help teams collaborate without having to switch between tools.

Communication: A communication plan can help manage expectations for team members and stakeholders.

Prioritization: Task boards can help organize tasks by priority.

Project tracking: Task management software can help monitor task status, assign tasks, set dependencies, and send reminders.

Timesheets: Timesheets can show project progress using a bar chart.

# External Interface Requirement

## GUI

**User Interfaces:**

Web Interface: Specify the requirements for the web-based user interface. Consider the layout, navigation, and design elements.

Mobile Interface: If the system has a mobile app, outline the requirements for the mobile user interface, ensuring responsiveness and usability on various devices.

Accessibility: Define accessibility standards to ensure the system is usable by individuals with disabilities.

**Software Interfaces:**

Identify and describe any third-party software or systems that the Task Management System needs to integrate with. This could include email systems, calendars, or other business applications.

Outline the use of APIs (Application Programming Interfaces) for seamless integration with external systems.

**Communication Interfaces:**

Specify the protocols and communication methods used for data exchange between the Task Management System and external entities. This may include HTTP/HTTPS for web communication or specific protocols for integrations.

**User Authentication and Authorization:**

Define the external interfaces related to user authentication and authorization. This could involve integrating with LDAP (Lightweight Directory Access Protocol), OAuth, or other authentication mechanisms.

**Notification Interfaces:**

Describe how the system will interact with external notification services (e.g., email, SMS, push notifications) to keep users informed of task updates, reminders, or other relevant information.

# System Features

The Task Management System (TMS) is a comprehensive platform designed to streamline and enhance the organization and execution of tasks within a given workflow. With user-friendly authentication and authorization mechanisms, the TMS ensures secure access for individuals and teams. Users can effortlessly create, assign, and prioritize tasks, fostering efficient collaboration through real-time communication features like comments and file attachments. The system offers advanced capabilities such as task dependencies, due dates with reminders, and progress tracking, empowering users to stay organized and meet deadlines. Seamless integration with calendars, email, and other productivity tools enhances overall user experience, while reporting and analytics provide valuable insights into task completion and user performance. The TMS also emphasizes customization, allowing users to tailor views, layouts, and dashboards according to their preferences. Whether accessed via web or mobile devices, the TMS ensures a responsive and accessible experience, supporting offline functionality for uninterrupted productivity. Security features, such as data encryption and audit trails, guarantee the protection of sensitive information. Through continuous improvement initiatives, including user feedback mechanisms and gamification elements, the TMS evolves to meet the evolving needs of its users, making it an indispensable tool for effective task management.

# Other Non-functional Requirements

## Performance Requirement

1. Reliability: - Consistent Performance: The system should perform consistently without errors or unexpected failures. Users rely on the system's stability to manage their tasks effectively without worrying about data loss or system crashes. - Error Handling: Robust error handling mechanisms should be in place to gracefully manage errors or exceptions, ensuring they don't disrupt the overall system functionality. This includes providing informative error messages to users and logging errors for system administrators to troubleshoot. - Fault Tolerance: Design the system to tolerate faults or failures in components, ensuring that if one part of the system fails, it doesn’t bring down the entire system. Redundancy in critical components and failover mechanisms can contribute to fault tolerance.
2. Availability: - Uptime and Accessibility: The system should be available and accessible to users whenever they need to interact with it. This involves maintaining high uptime percentages and minimizing downtime for maintenance or unexpected issues. - Load Balancing: Distribute incoming user requests across multiple servers or resources to prevent overloading any single component, ensuring consistent availability even during periods of high usage. - Monitoring and Proactive Maintenance: Implement monitoring tools to continuously track the system's health and performance. Proactive maintenance and regular checks help identify potential issues before they cause system downtime, ensuring maximum availability. - Disaster Recovery and Redundancy: Have contingency plans and redundant systems in place to handle unexpected disasters or hardware failures. This might involve data backups, redundant servers, or cloud-based failover systems to ensure continuous availability in case of outages or disasters. Ensuring high reliability and availability involves a combination of robust design, fault-tolerant architecture, proactive maintenance, and responsive support mechanisms. It's about instilling confidence in users that the system will consistently perform as expected and remain accessible whenever they need to manage their tasks.

## Safety Requirement

Protect sensitive data, such as user credentials and task details, with robust encryption and access controls. Implement regular data backups and a disaster recovery plan to ensure data resiliency. Validate task creation, modification, and completion to ensure authorized users are performing actions correctly. Manage task dependencies to prevent conflicts or errors in task execution. Prioritize tasks based on urgency and allocate resources effectively to ensure timely completion. Monitor task progress and alert users of potential issues or delays. Ensure the system can handle the expected volume of tasks and users without performance degradation. Design the system to scale horizontally to accommodate increasing demand. Implement fault tolerance mechanisms to handle hardware or software failures without disrupting task management operations. Maintain a disaster recovery plan to restore system functionality in case of major outages.

## Security Requirement

Protect sensitive data, such as user credentials and task details, with robust encryption and access controls. Implement regular data backups and a disaster recovery plan to ensure data resiliency. Validate task creation, modification, and completion to ensure authorized users are performing actions correctly. Manage task dependencies to prevent conflicts or errors in task execution. Implement role-based access control to restrict user privileges based on their roles and responsibilities. Enforce multi-factor authentication for user logins to prevent unauthorized access. Continuously monitor system activity and log all events, including user actions, task changes, and system events. Use these logs for auditing, troubleshooting, and security incident investigations Regularly scan the system for vulnerabilities and promptly apply security patches to address any identified weaknesses. Keep the system software up-to-date to minimize security risks.

## Requirement attributes

Requirement attributes: These requirements define the core functionalities of the system, such as creating, assigning, modifying, and completing tasks. They should be specific, measurable, achievable, relevant, and time-bound (SMART). These requirements address the system's performance, usability, security, and other aspects that impact its overall effectiveness.

They may include aspects like response time, scalability, accessibility, and error handling. These requirements capture the needs and expectations of the system's users, including task managers, team members, and stakeholders. They should be gathered through user research, interviews, and surveys to ensure the system aligns with user expectations. These requirements align the task management system with the organization's goals, objectives, and strategies. They may include requirements for improved productivity, collaboration, efficiency, and cost savings. These requirements specify the technical specifications, hardware, software, and network capabilities needed to implement and operate the task management system. They should consider the organization's existing IT infrastructure and resources.

## Business Rules

Clearly define the ownership and responsibility for each task, ensuring that there is accountability for task completion. Establish a mechanism for prioritizing tasks based on urgency and importance, and track the status of tasks to monitor progress and identify potential delays.

Identify and manage task dependencies to prevent conflicts or errors in task execution. Ensure that dependent tasks are not completed until their prerequisites are fulfilled. Implement notification mechanisms to alert users of relevant task updates, upcoming deadlines, and potential issues. Establish escalation procedures to address critical tasks that fall behind schedule. Enforce role-based access control to restrict access to tasks based on user privileges and responsibilities. Prevent unauthorized users from viewing, modifying, or completing tasks.

## User Requirement

The system should have a user-friendly interface that is easy to navigate and understand, even for users with limited technical expertise. Users should be able to quickly create, assign, and modify tasks, including setting due dates, priorities, and descriptions. The system should facilitate collaboration among team members, enabling them to share tasks, provide feedback, and track each other's progress. Users should have access to real-time task updates, allowing them to monitor progress, identify potential delays, and make informed decisions. The system should provide customizable notification options to alert users of relevant task events, such as deadlines, assignments, and status changes.

# Other Requirements

## Data and Category Requirement

The Task Management System (TMS) necessitates specific data and category features to ensure a robust and organized structure. Data requirements encompass the need for a secure and scalable database infrastructure capable of efficiently storing and retrieving task-related information. The system should support the storage of diverse data types, including text, dates, and attachments, with appropriate indexing for quick data retrieval. Additionally, data backup and recovery mechanisms are imperative to safeguard against data loss.

## Appendix

A: Admin, Abbreviation, Acronym, Assumptions; B: Books, Business rules; C: Class, Client, Conventions; D: Data requirement, Dependencies; G: GUI; K: Key; L: Library, Librarian; M: Member; 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;

## Glossary

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

* + - Administrator: A login id representing a user with user administration privileges to the software
    - User: A general login id assigned to most users
    - 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

## Class Diagram

A class is an abstract, user-defined description of a type of data.

**User Class:**

Represents users of the system. Attributes include userID, username, email, and password. Methods may include creating tasks and assigning tasks.

**Task Class:**

Represents tasks created within the system. Attributes include taskID, title, description, due Date, priority, status, and assigned To (a User object). Methods include creating tasks, assigning tasks to users, and completing tasks.

**Project Class:**

The Project class serves as a blueprint for individual projects within the Task Management System. Each project is uniquely identified by a ProjectID and possesses attributes such as Title, Description, StartDate, and End Date, which define the project's basic details and timeline. The Members attribute is a list of UserIDs representing the individuals involved in the project, facilitating collaboration and task assignment within the project context.

**Notification Class:**

Notifications play a crucial role in keeping users informed about important events within the Task Management System. The Notification class includes attributes such as NotificationID for unique identification, UserID specifying the recipient of the notification, Type indicating the nature of the notification (e.g., task assignment, approaching due date), Content carrying the message or information, and Timestamp capturing the date and time of notification generation.

**Task Manager Class:**

The Task Manager class acts as a central orchestrator, bringing together collections of User, Task, Project, and Notification objects. This class encapsulates the overall functionality of the Task Management System, providing methods for managing and manipulating tasks, projects, users, and notifications. Through this class, users can interact with and control various aspects of the system, including creating tasks, assigning them to projects, managing user profiles, and handling notifications.

**CLASS DIAGRAM:**

