SRS

❖ Introduction:

- **Purpose:** The purpose of the project is to develop a digital to-do list application that allows users to manage and organize their tasks efficiently.
- **Scope:** The application will provide features for creating, editing, and deleting tasks, updating,setting due dates and priorities, and organizing tasks into categories or projects.

❖ Assumptions and Dependencies:

- The application will require an internet connection to function.
- Users will have access to compatible devices (computers, smartphones, tablets) to use the application.

❖ Functional Requirements:

- **User Registration and Authentication:**
  > The application should allow users to create an account or log in with existing credentials.
  > User authentication should be implemented to ensure secure access to the application.

- **Task Management:**
  > Users can be able to create a new task by providing a Task name,start date,end date,status.
  > Users can be able to update task details such as start date,due date,status.
  > Users can be able to mark tasks as Not Started,completed.
  > Users can be able to delete the Task.
  > Users can be able to search for specific tasks using Task name.
  >Users can also be able to search Task by Date.

- **Task status Tracking:**
  > The application shall allow users to mark tasks as completed ,Not started,Yet to start.
  > The application shall provide visual indicators to differentiate between completed,Not started and Yet to start.

- **Reminders and Notifications:**
  > Users can receive notification about the Task.Notification can be sent through email as soon as the task is added by the user.

- **User Interface:**
  > The application should have a user-friendly interface.
  > It should provide clear navigation and easy access to various features.
  > The design should be responsive and adaptable to different screen sizes and devices.

❖ Non-Functional Requirements:

- **Performance:** The application should be responsive and provide quick response times.
- **Security:** User data should be securely stored and transmitted.
- **Reliability:** The application should be reliable and available for use at all times.
- **Scalability:** The application should be able to handle a growing number of users and tasks.
- **Compatibility:** The application should be compatible with major web browsers and mobile platforms.

❖ External Interface Requirements:

- **USER INTERFACES:**
  > Front-end software:HTML+CSS,BootStrap,JS

- **HARDWARE INTERFACES:**
  > Windows.
  > A browser that supports HTML

- **SOFTWARE INTERFACES:**
  > JDK-1.8
  > Apache Tomcat-9.0
  > MySQL-mysql-connector-java-8.0.30.jar
  > J2EE
  > Back-end software:SQL

- **TECHNOLOGY USED:**
  > JSP
- **PROGRAMMING LANGUAGES:**
  >Java

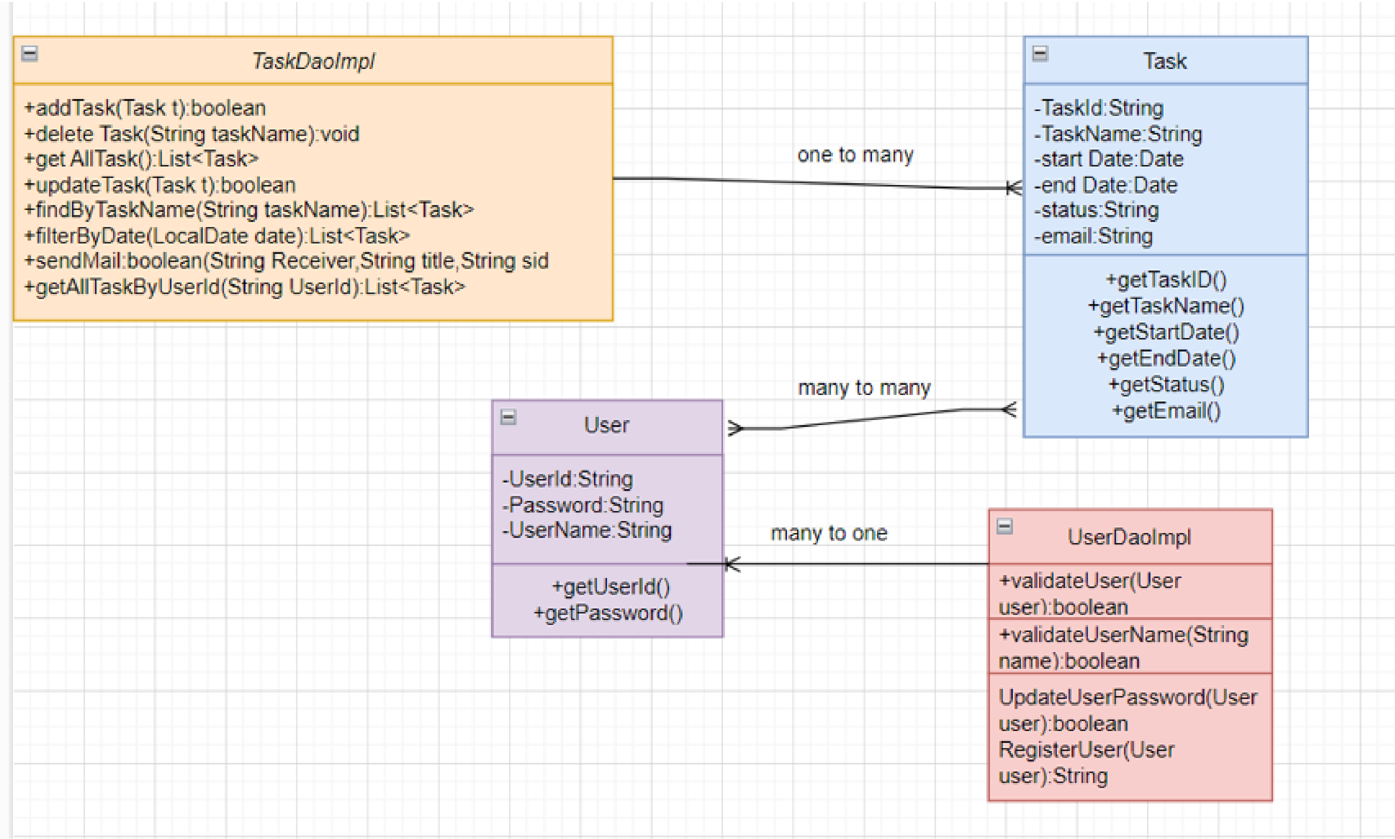
❖ Constraints:

- The project should be developed within a specific time frame and budget.
- The application should utilize suitable technologies and frameworks that meet the project requirements.
- The project team should adhere to relevant coding standards and best practices.

❖ **Class Diagram, an ER Diagram, and the corresponding database table names and relationships for a to-do list** :

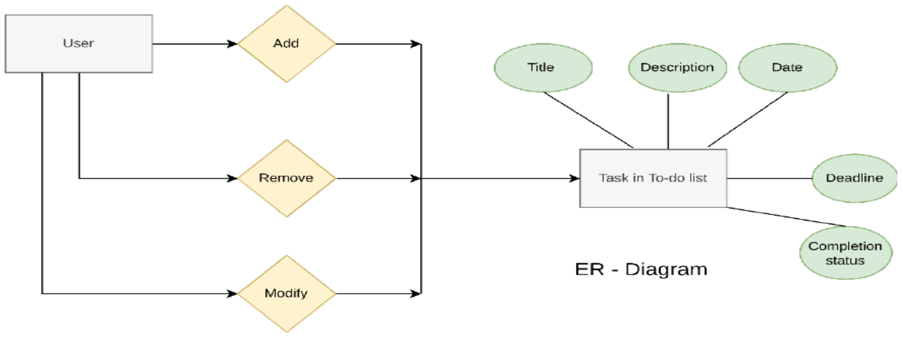
- In this representation, the class diagram shows the class structure in Java, where the data types are represented as Java data types.
- The ER diagram depicts the relationships between the entities.
- The database table names are represented with the corresponding columns and their data types.

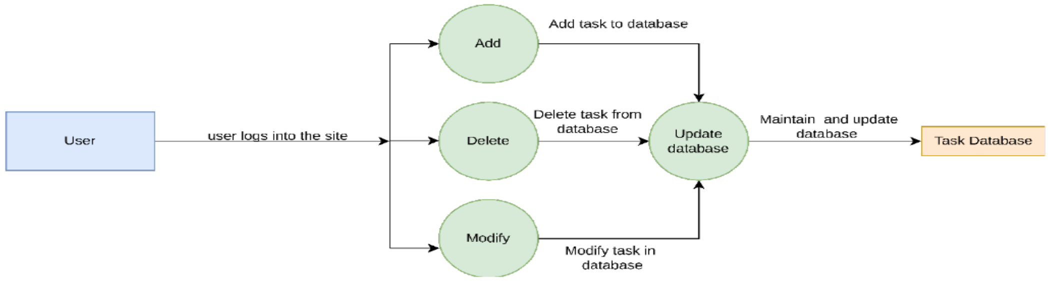
- **Class Diagram:**


TaskDaoImpl
+addTask(Task t):boolean
+delete Task(String taskName):void
+get AllTask():List<Task>
+updateTask(Task t):boolean
+findByTaskName(String taskName):List<Task>
+filterByDate(LocalDate date):List<Task>
+sendMail:boolean(String Receiver,String title,String sid
+getAllTaskByUserId(String UserId):List<Task>
one to many
Task
-TaskId:String
-TaskName:String
-start Date:Date
-end Date:Date
-status:String
-email:String
+getTaskID()
+getTaskName()
+getStartDate()
+getEndDate()
+getStatus()
+getEmail()
many to many
User
-UserId:String
-Password:String
-UserName:String
+getUserId()
+getPassword()
many to one
UserDaoImpl
+validateUser(User user):boolean
+validateUserName(String name):boolean
UpdateUserPassword(User user):boolean
RegisterUser(User user):String

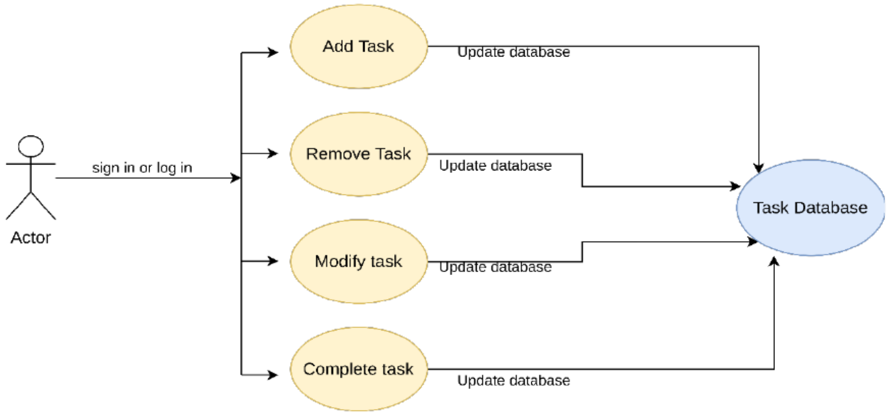
❖ Design Flow:

● ER Diagram & Data Flow Diagram:

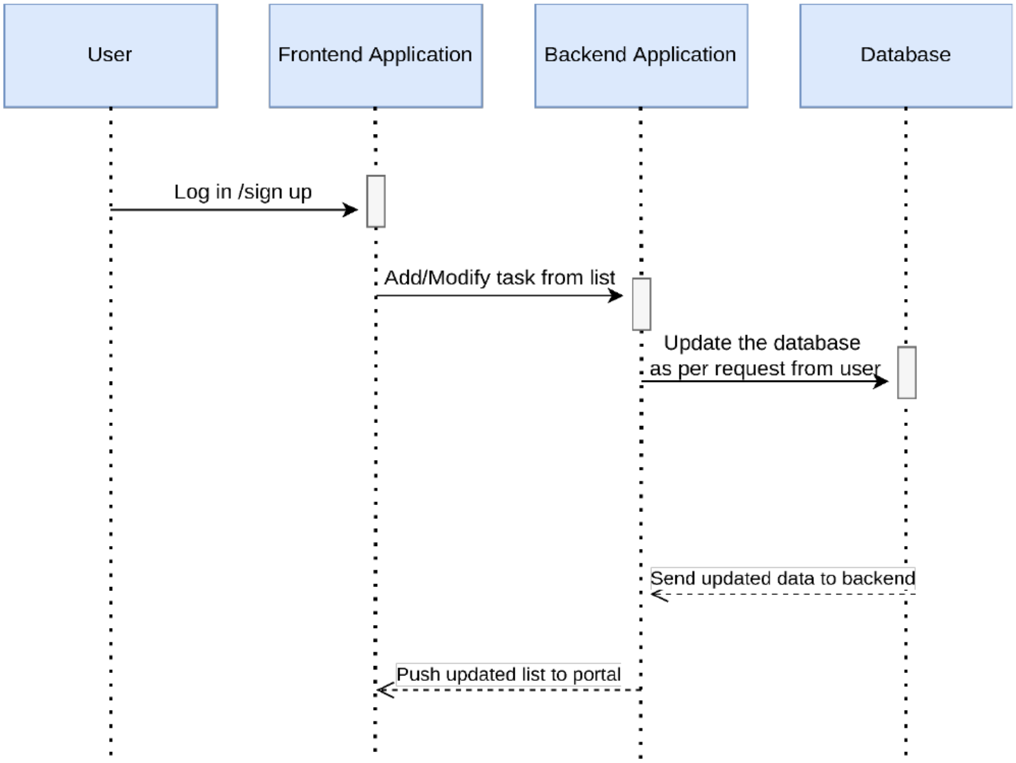




● Use Case Diagram:



- **Graphical Abstract:**



- **Database Table Names and Relationships:**

  - **Table: user**

    Columns:
    - userID (VARCHAR)
    - password (VARCHAR)
    -username(VARCHAR)
  - **Table:Task**

    Columns:
    -TaskID(primary key,INT)
    - TaskName(VARCHAR)
    - startDate (DATE)
    - endDate (DATE)
    - status (VARCHAR)
    -email(VARCHAR)

- **Relationships:**
  - Multiple users can have multiple Tasks (many-to-many relationship between users and tasks)
  -TaskDaoImpl class manages the creation,modification,and deletion of Tasks.
  -One-to-many relationship between TaskDaoImpl and Task.
  -Many-to-one relationship between User and UserDaoImpl.