رواد مصر الرقمية

وزارة الاتصالات
وتكنولوجيا المعلومات

Task Management System

1. Project Planning & Management

1.1 Project Proposal

Overview: The Task Management System is a web-based application designed to allow users to create, and manage tasks efficiently.

Objectives:

- Provide an intuitive UI for task management.
- Implement task categorization, priority levels, and due dates.
- Enable task searching.

Scope:

- Frontend: React with Redux toolkit for state management.

1.2 Project Plan

Timeline :

Week	Tasks
1	Setup development environment, create wireframes, build basic layout.
2	Implement task creation, deletion and integrate Redux.
3	Implement task filtering, sorting, search functionality, and testing.
4	UI enhancements, deployment, and final documentation.

Milestones & Deliverables:

- Week 1: Project setup, wireframes, and basic layout.
- Week 2: Task creation, deletion, and Redux integration.
- Week 3: Task filtering, search, and testing.
- Week 4: UI improvements, deployment, and documentation.

1.3 Task Assignment & Roles

Role	Responsibility

Project Manager	Oversee development and ensure milestones are met.
Frontend Developer	Develop UI components using React and Redux.
Backend Developer	Implement REST API using Node.js and Express.
Database Administrator	Manage SQL DB schema and queries.
QA Tester	Test features and report bugs.

1.4 Risk Assessment & Mitigation Plan

Risk	Mitigation Strategy
API failures	Implement proper error handling and logging.
Data loss	Use database backups and implement recovery mechanisms.
UI responsiveness issues	Ensure mobile-friendly UI using CSS media queries.

1.5 Key Performance Indicators (KPIs)

- System uptime: 99%+
- Average response time: < 300ms
- User adoption rate: 70%+

---

2. Literature Review

2.1 Feedback & Evaluation

Lecturers and stakeholders will evaluate the system's usability, functionality, and performance.

2.2 Suggested Improvements

Potential enhancements include:

- Adding notifications for due tasks.
- Implementing user authentication.

2.3 Final Grading Criteria

Criteria	Weight
Documentation	20%
Implementation	40%
Testing	20%
Presentation	20%

---

3. Requirements Gathering

3.1 Stakeholder Analysis

Stakeholder	Needs
End Users	Easy task management and tracking.
Admin	User management and monitoring.

3.2 User Stories & Use Cases

- **User Story:** As a user, I want to create tasks so that can manage my workload efficiently.
- **Use Case:** A user logs in, creates a task, assigns a priority, and marks it as completed upon finishing.

3.3 Functional Requirements

- Users can create, view, edit and delete.
- Tasks have priorities and due dates.

3.4 Non-Functional Requirements

- The system should load within 3 seconds.
- Data should persist across sessions.

---

4. System Analysis & Design

4.1 Problem Statement & Objectives

Problem: Managing tasks manually is inefficient.
Objective: Provide a digital solution for effective task management.

4.2 Use Case Diagram

Illustration of user interactions with the system.

4.3 Software Architecture

- **Frontend:** React with Redux toolkit
- **Backend:** Node.js with Express
- **Database:** SQL DB

---

5. Database Design & Data Modeling

5.1 ER Diagram

Diagram showcasing entities (Users, Tasks, etc.) and relationships.

5.2 Logical & Physical Schema

Table	Attributes
User	ID, Name, Email

Task ID, Title, Description, Priority, Status, Due Date

---

6. Data Flow & System Behavior

6.1 Data Flow Diagram (DFD)

Context-level DFD illustrating data movement.

6.2 Sequence & Activity Diagrams

Diagrams representing interactions and workflows.

---

7. UI/UX Design & Prototyping

7.1 Wireframes & Mockups

Screens showcasing task creation, and lists.

7.2 UI/UX Guidelines

- Color scheme: Violet & White.
- Typography: Poppins.
- Accessibility: High contrast mode available.

---

8. System Deployment & Integration

8.1 Technology Stack

- Frontend: React, Redux
- Backend: Node.js, Express
- Database: SQL DB

8.2 Deployment & Component Diagram

Diagrams illustrating deployment environment and component relationships.

---

9. Additional Deliverables

9.1 API Documentation

Endpoint	Method	Description
/tasks	GET	Fetch all tasks
/tasks	POST	Create a new task
/tasks/:id	DELETE	Remove a task

9.2 Testing & Validation

Test Case	Expected Outcome
Create Task	Task is added successfully
Filter Tasks	Correct tasks are displayed

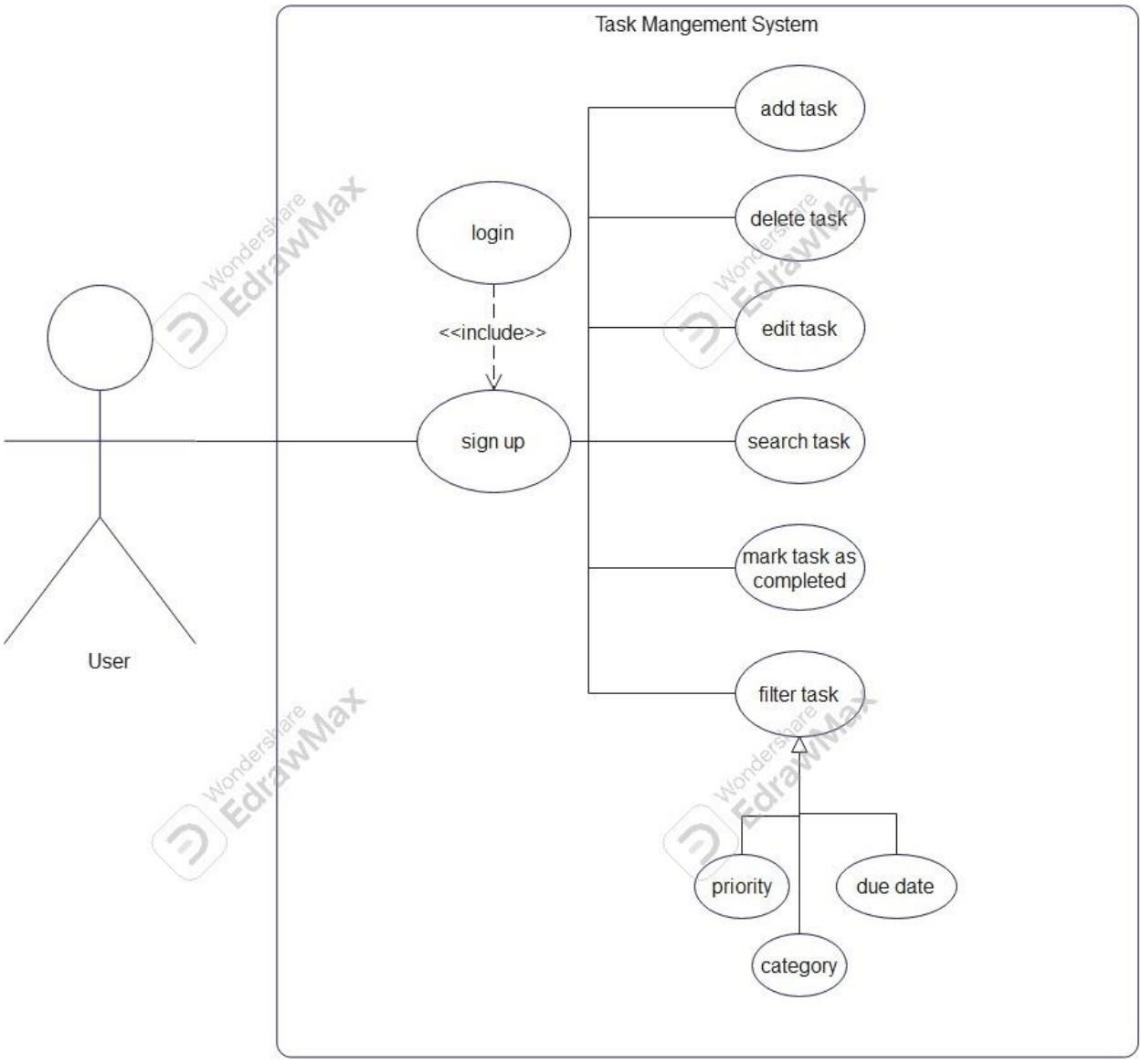
9.3 Deployment Strategy

- Hosting on Vercel (Frontend) and Heroku (Backend).
- Continuous Integration (CI/CD) pipeline for seamless updates.

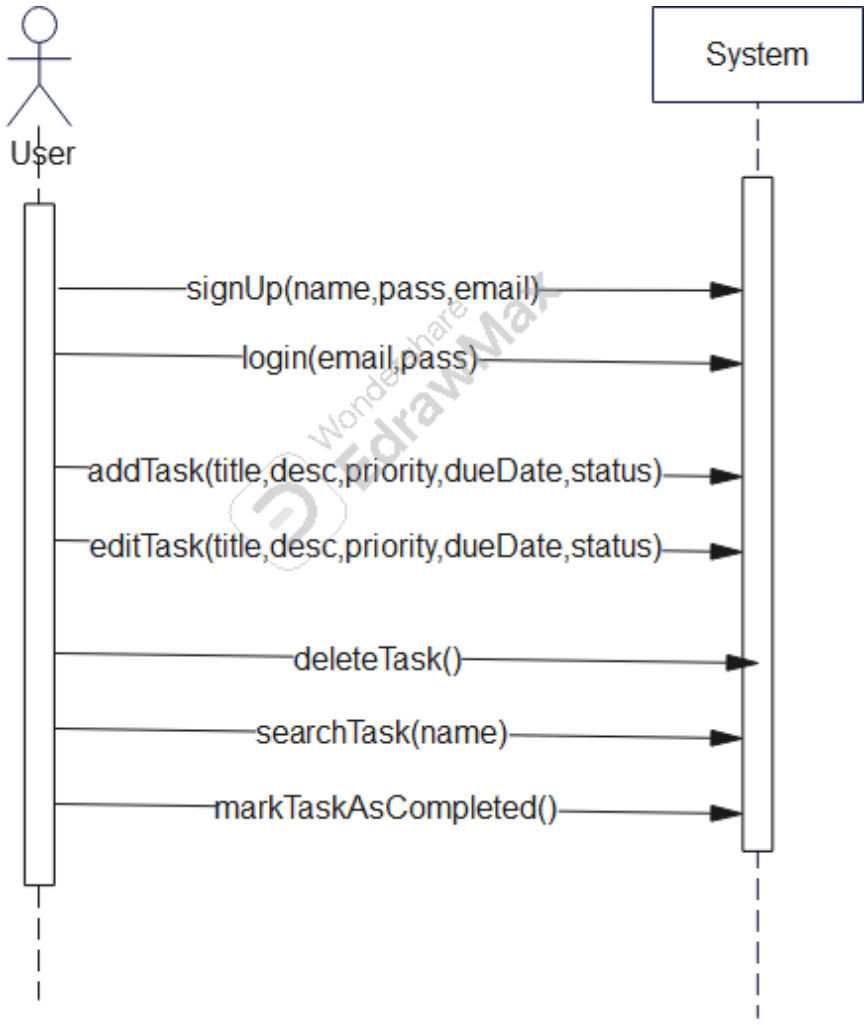
Conclusion

This document outlines the complete process of developing the Task Management System using React, Node.js, and SQL DB. The system provides efficient task organization, prioritization, and management features while ensuring scalability and usability.

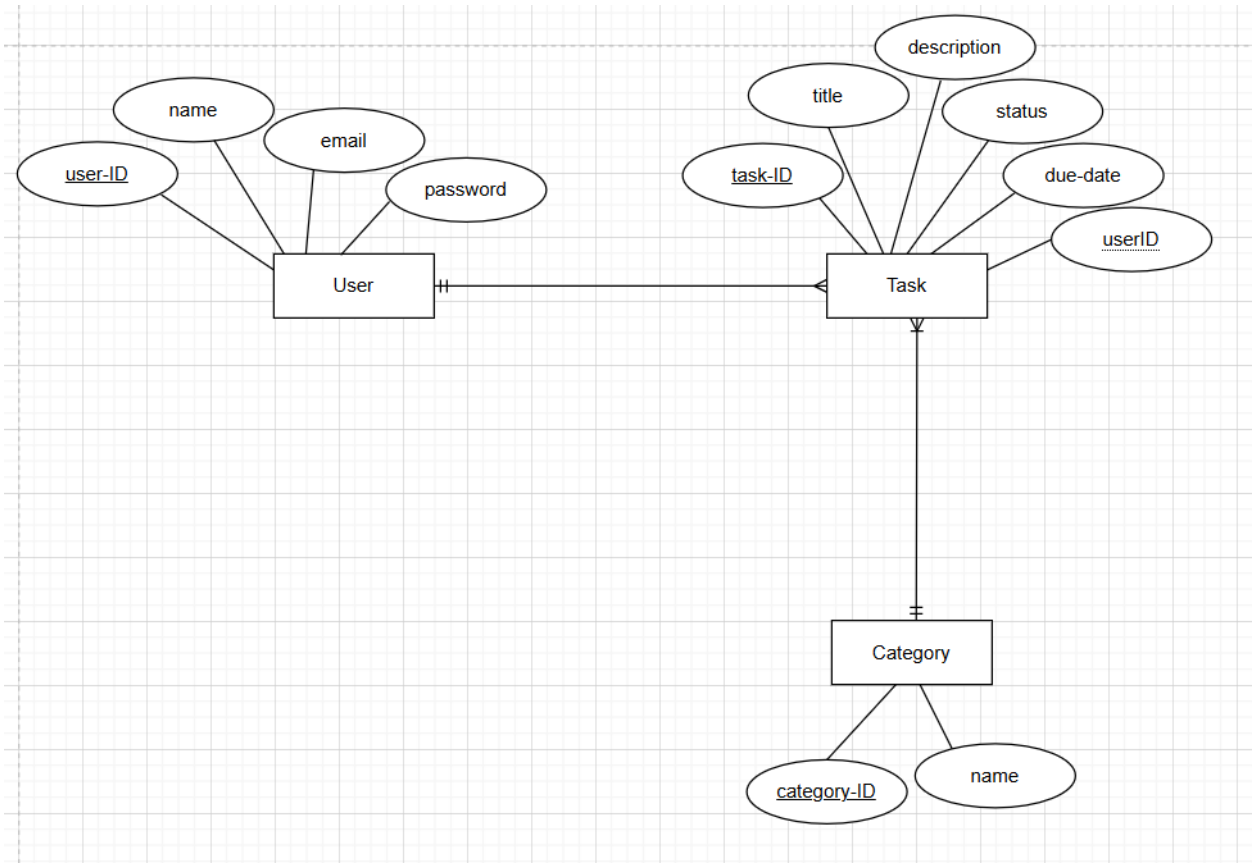
UML Use Case Diagram



UML sequence diagram



ERD Diagram


description
title
status
name
email
user-ID
task-ID
due-date
password
userID
User
Task
Category
category-ID
name

Class Diagram