Task Management System

Introduction to the Task Management System:

The Task Management System is a simple yet powerful web-based application designed to help users efficiently manage their tasks. Each task can have a title, description, priority, status, and due date. The system allows users to register, log in, and perform operations such as creating, updating, deleting, and viewing tasks. It also features task sorting by priority and due date, ensuring users can prioritize their workloads effectively.

Technologies Used:

- **Backend**: Django (with Django Rest Framework) for RESTful API development, PostgreSQL for the database, JWT for token-based authentication.
- **Frontend**: ReactJS for building a responsive and interactive user interface, React Router for navigation.
- **Additional Tools**: Axios for API calls, Bootstrap/Material-UI for UI components.

Key Features Implemented:

- User registration and authentication (sign-up, sign-in, sign-out)
- CRUD operations for tasks: create, read, update, and delete
- Task sorting by priority and due date
- Token-based authentication using JWT for managing user sessions
- Fully responsive UI for mobile and desktop

API endpoints used:

HTTP Method	Endpoint	Description
POST	/api/users/signup/	Register a new user
POST	/api-token-auth/	Log in a user and return an auth token
GET	/api/tasks/	Retrieve all tasks for a logged-in user
POST	/api/tasks/	Create a new task
PUT	/api/tasks/${currentTask.id}/	Update a task by ID
DELETE	/api/tasks/${id}/	To delete the specific task of user by id

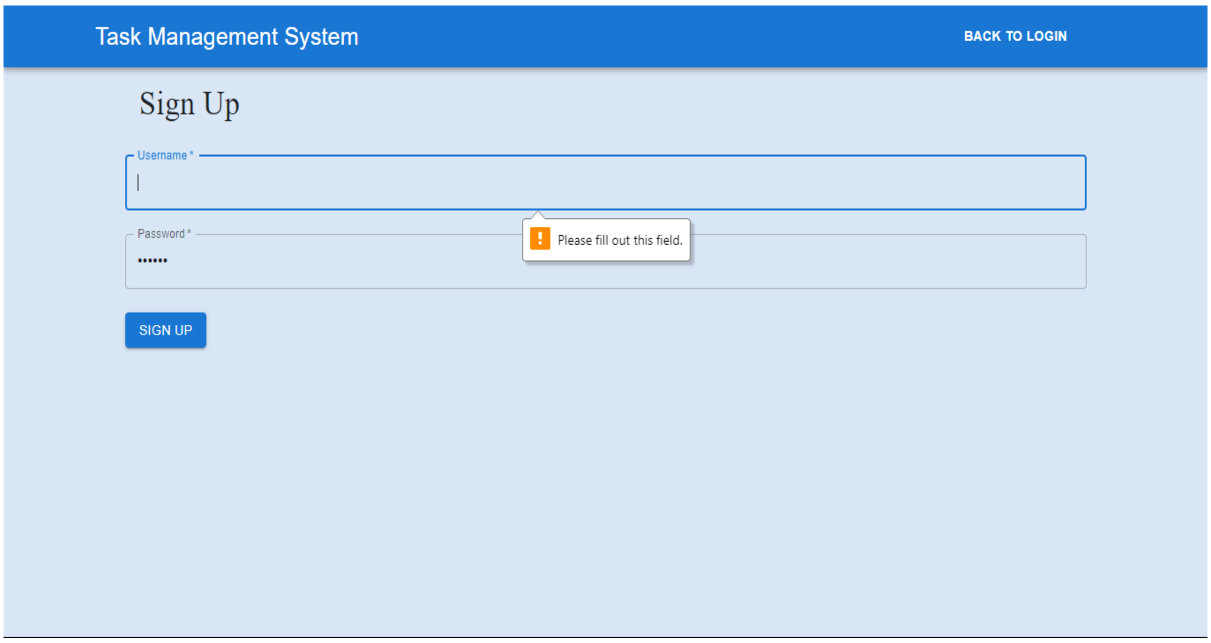
Functionality Demonstration

1. **User Registration Process:**
   - Users fill out the registration form with their username, password, and email.
   - Upon successful registration, they receive a confirmation message.
2. **Creating a New Task:**
   - After logging in, users navigate to the task creation form, fill in the details, and submit.
   - The task is added to the list, and a success message is displayed.
3. **Updating an Existing Task:**
   - Users can click on a task in the list, make changes in the update form, and save the updates.
   - Confirmation of successful update is shown.
4. **Deleting a Task:**
   - Users can delete a task by clicking the delete button, confirming the action in a prompt, and seeing the task removed from the list.
5. **Sorting Tasks by Priority and Due Date:**
   - Users can select sorting options to arrange tasks based on priority and due date, improving task management.

System Flow:

Step1:

Sign Up: Signup with error handing


Task Management System

BACK TO LOGIN

Sign Up

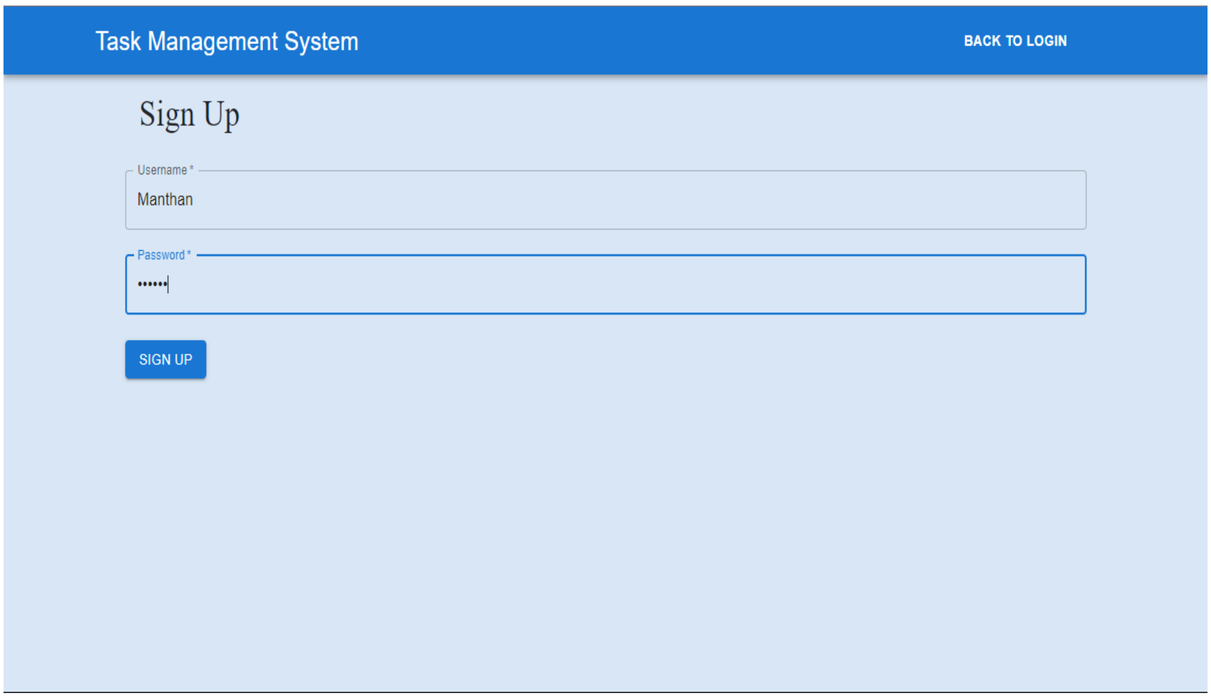
Username *

Password *

••••••

Please fill out this field.

SIGN UP


Task Management System

BACK TO LOGIN

Sign Up

Username *

Manthan

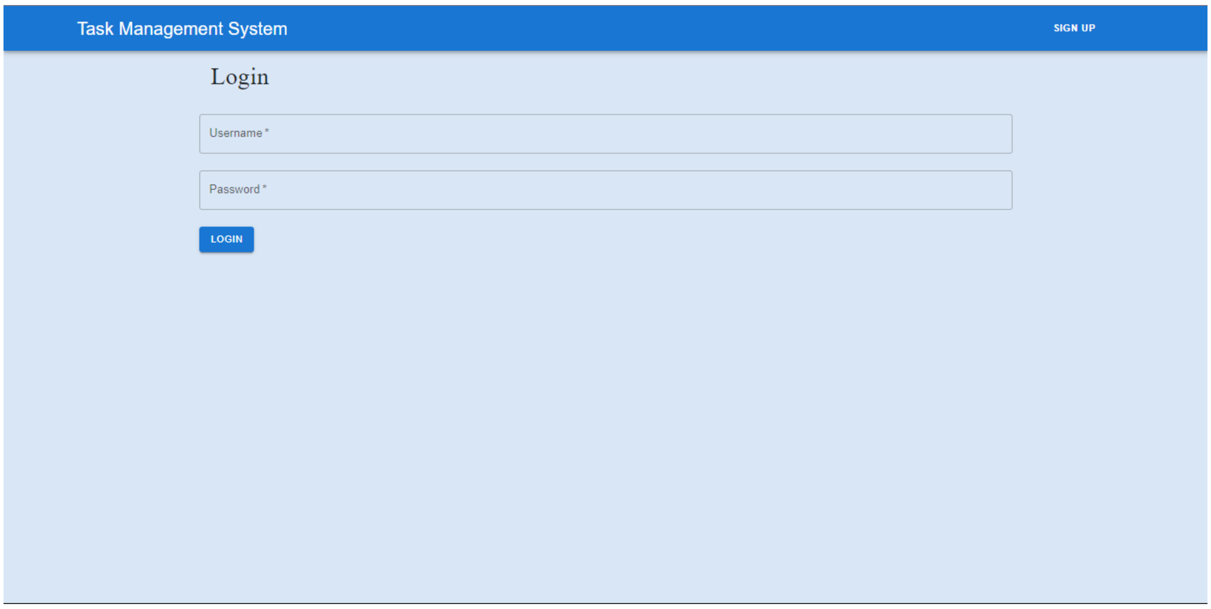
Password *

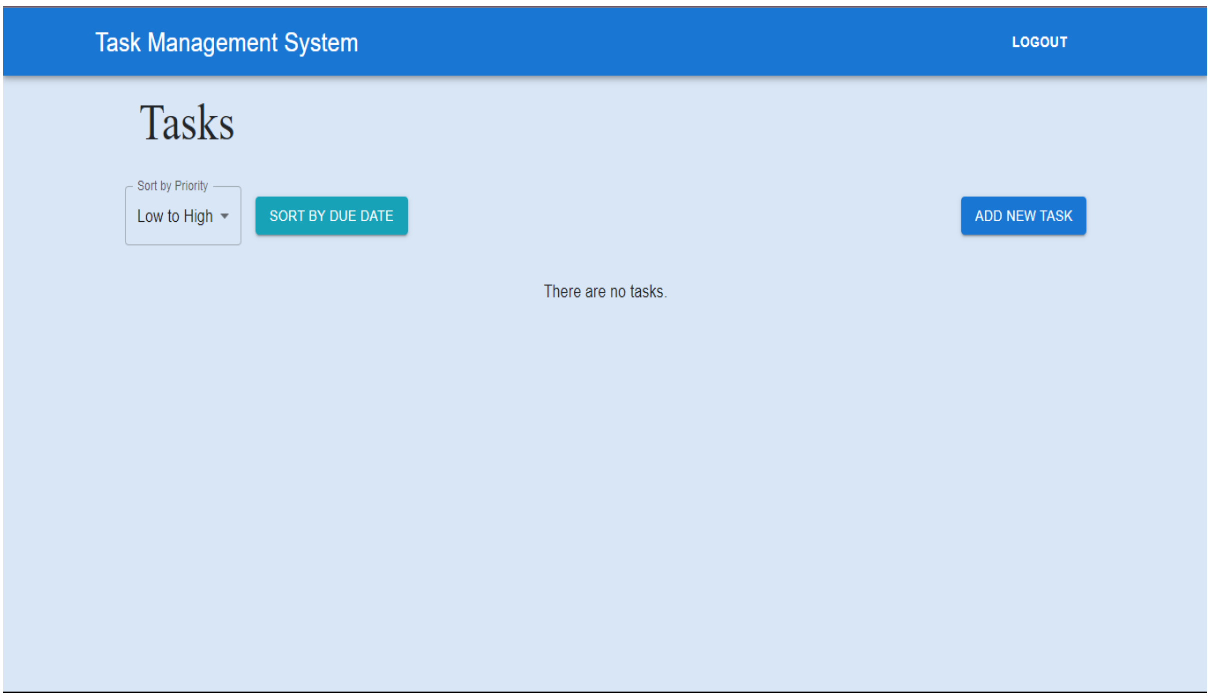
••••••

SIGN UP

Step2:

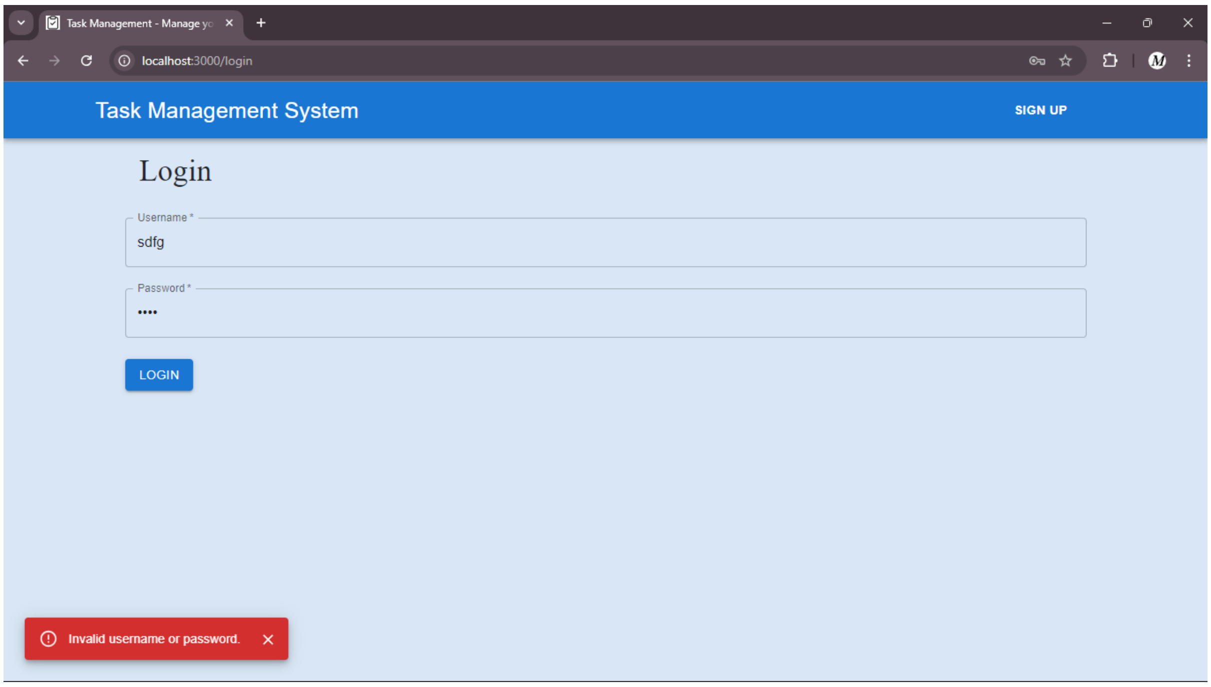
Login :





There are no tasks.

Error handling for login for unauthorized users:


Task Management - Manage yo

localhost:3000/login

Task Management System

SIGN UP

Login

Username *

sdfg

Password *

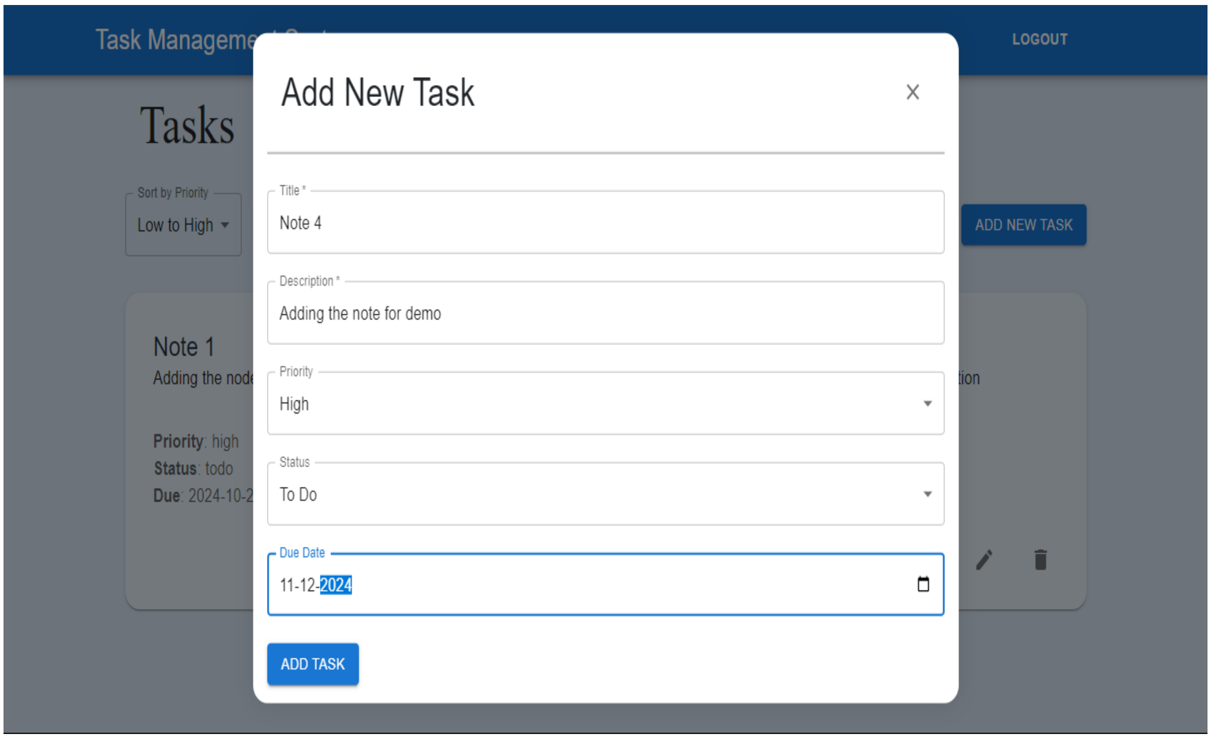
••••

LOGIN

Invalid username or password.

Step 3:

Add Task:


Task Manageme

LOGOUT

Tasks

Sort by Priority

Low to High

ADD NEW TASK

Note 1

Adding the node

Priority: high

Status: todo

Due: 2024-10-2

Add New Task

Title *

Note 4

Description *

Adding the note for demo

Priority

High

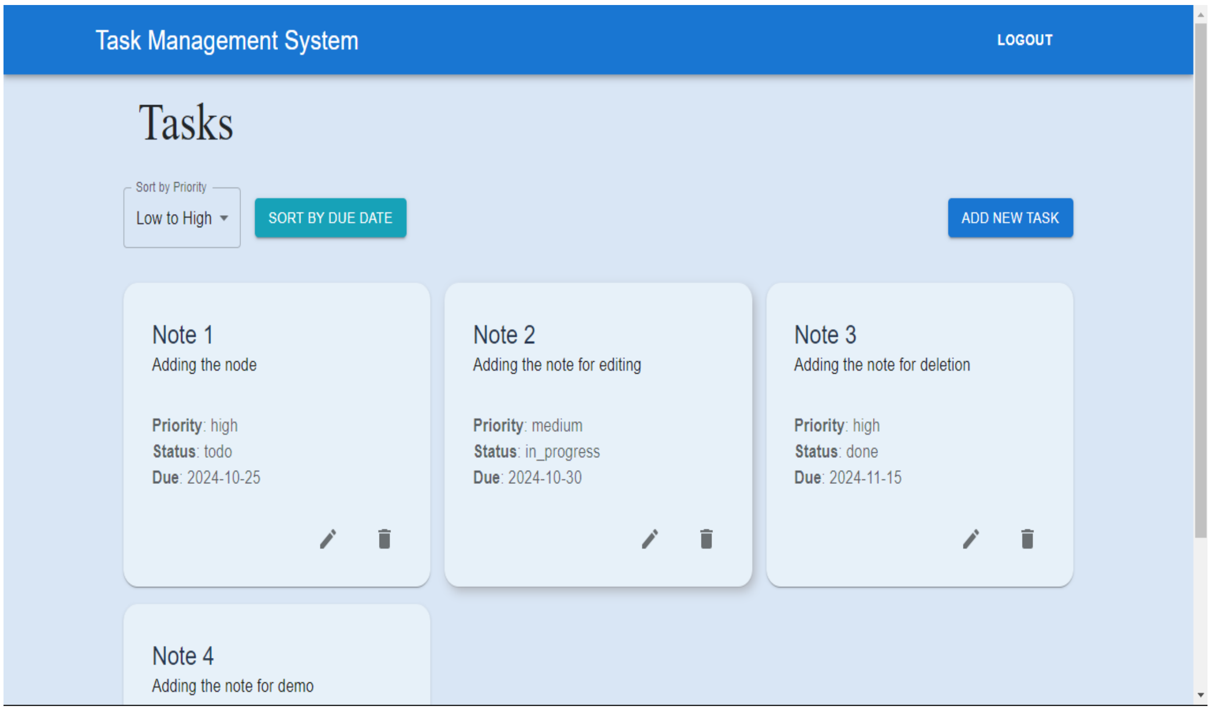
Status

To Do

Due Date

11-12-2024

ADD TASK


Task Management System

LOGOUT

Tasks

Sort by Priority: Low to High

SORT BY DUE DATE

ADD NEW TASK

Note 1
Adding the node

Priority: high
Status: todo
Due: 2024-10-25

Note 2
Adding the note for editing

Priority: medium
Status: in_progress
Due: 2024-10-30

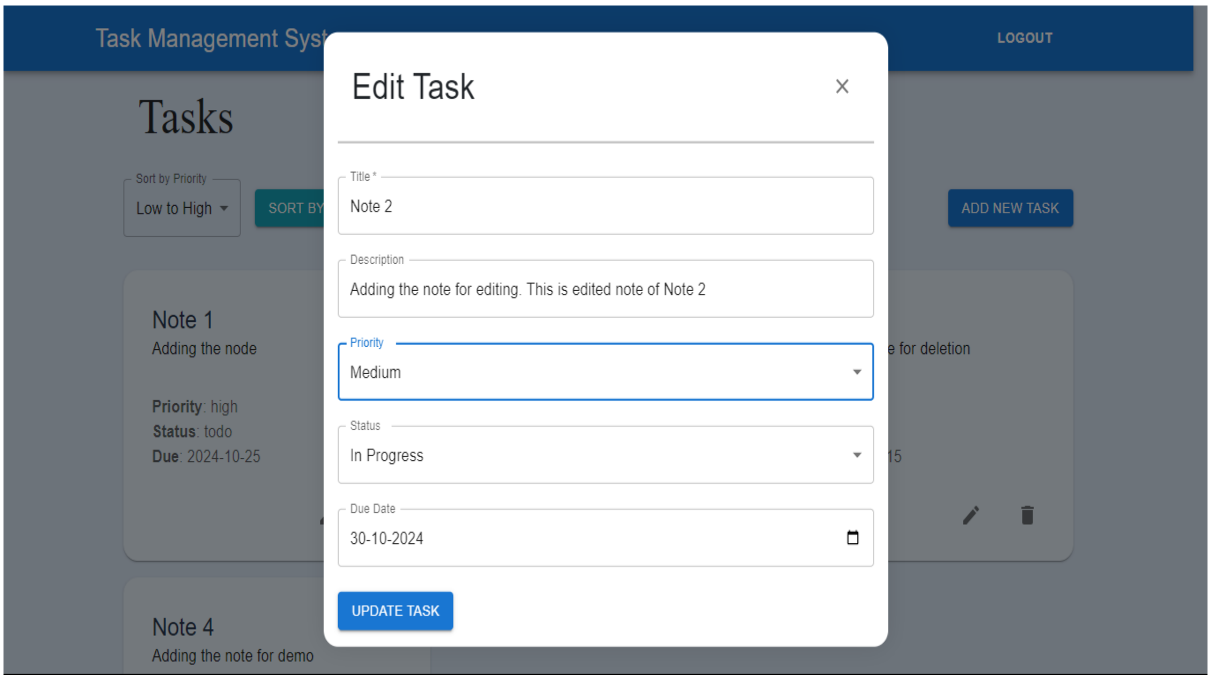
Note 3
Adding the note for deletion

Priority: high
Status: done
Due: 2024-11-15

Note 4
Adding the note for demo

Step 4:

Edit Task:


Edit Task

Title *: Note 2

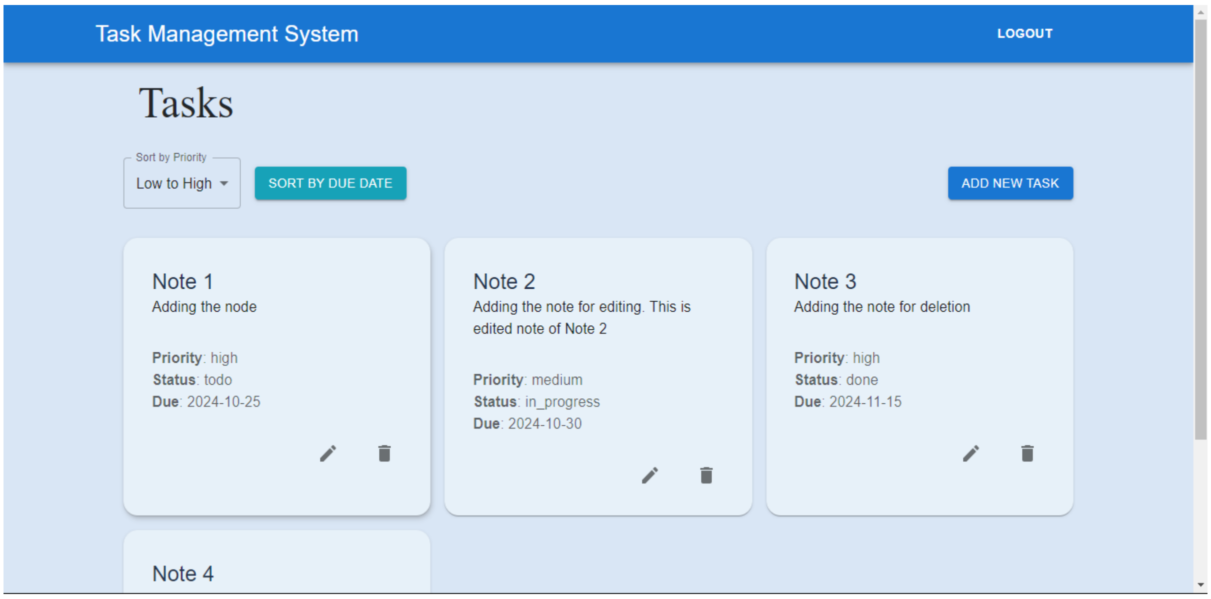
Description: Adding the note for editing. This is edited note of Note 2

Priority: Medium

Status: In Progress

Due Date: 30-10-2024

UPDATE TASK


Task Management System LOGOUT

Tasks

Sort by Priority: Low to High | SORT BY DUE DATE | ADD NEW TASK

Note 1
Adding the node

Priority: high
Status: todo
Due: 2024-10-25

Note 2
Adding the note for editing. This is edited note of Note 2

Priority: medium
Status: in_progress
Due: 2024-10-30

Note 3
Adding the note for deletion

Priority: high
Status: done
Due: 2024-11-15

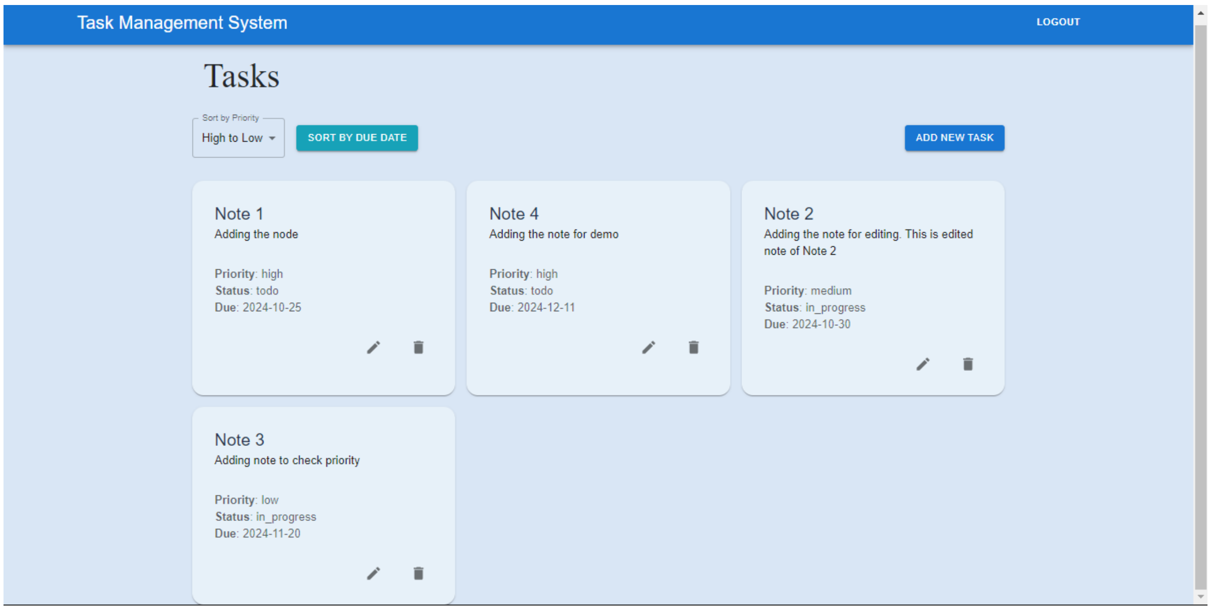
Note 4

Step 5:

Sortings:

1. Based on Priority

a) High to Low (High-Medium-Low)


Task Management System LOGOUT

Tasks

Sort by Priority: High to Low | SORT BY DUE DATE | ADD NEW TASK

Note 1
Adding the node

Priority: high
Status: todo
Due: 2024-10-25

Note 4
Adding the note for demo

Priority: high
Status: todo
Due: 2024-12-11

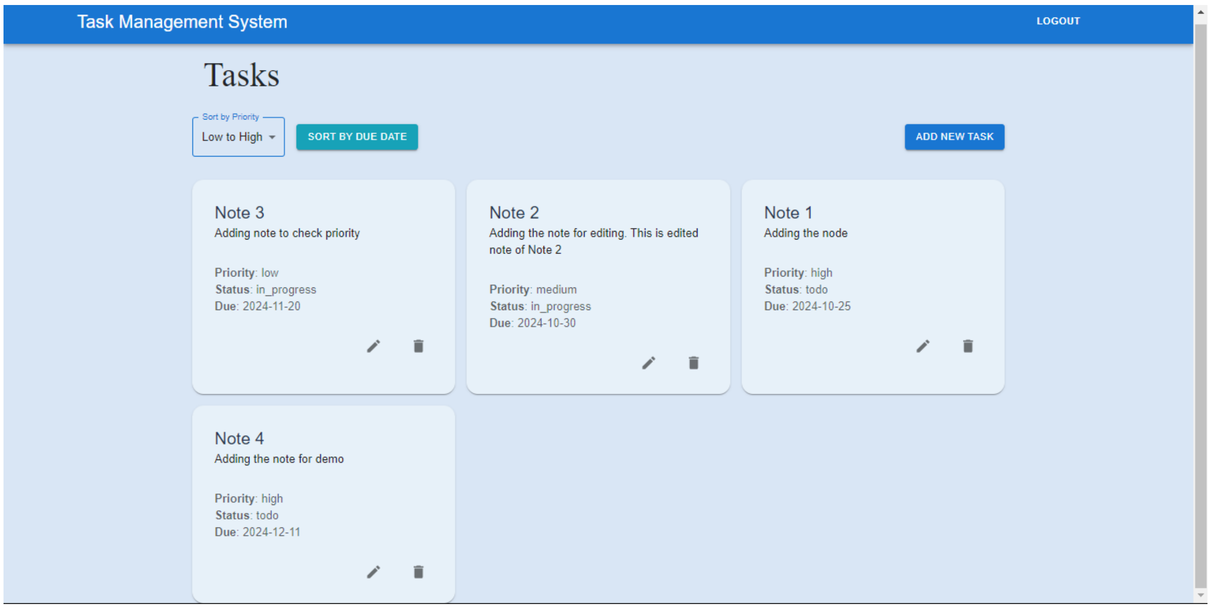
Note 2
Adding the note for editing. This is edited note of Note 2

Priority: medium
Status: in_progress
Due: 2024-10-30

Note 3
Adding note to check priority

Priority: low
Status: in_progress
Due: 2024-11-20

b) Low to High (Low-Medium-High)


Task Management System

LOGOUT

Tasks

Sort by Priority
Low to High

SORT BY DUE DATE

ADD NEW TASK

Note 3
Adding note to check priority

Priority: low
Status: in_progress
Due: 2024-11-20

Note 2
Adding the note for editing. This is edited note of Note 2

Priority: medium
Status: in_progress
Due: 2024-10-30

Note 1
Adding the node

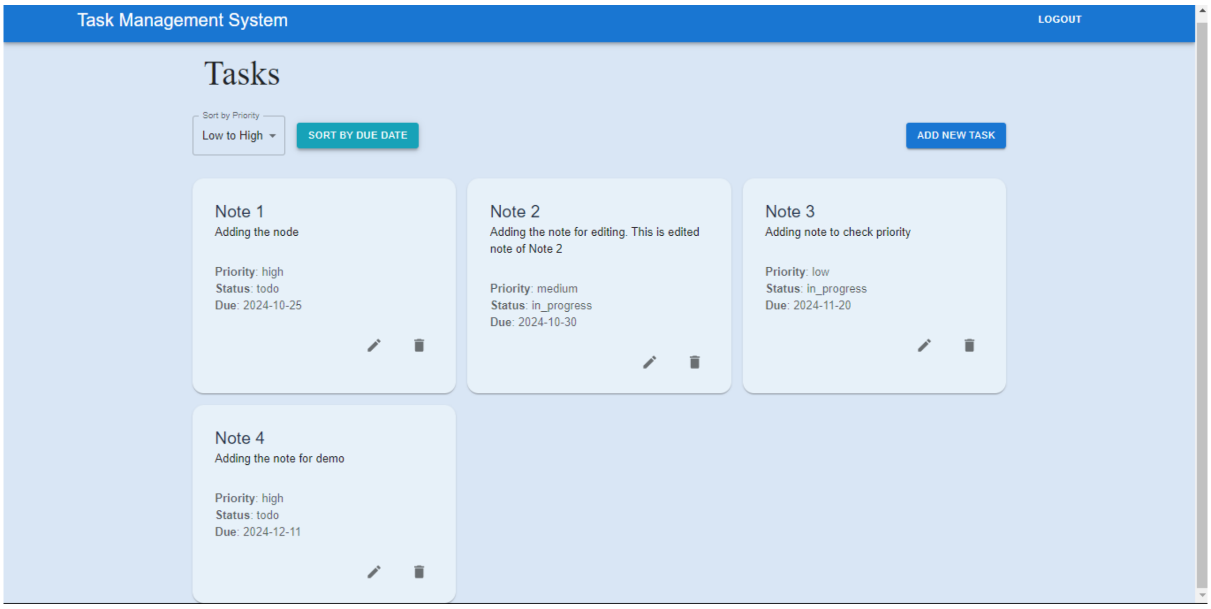
Priority: high
Status: todo
Due: 2024-10-25

Note 4
Adding the note for demo

Priority: high
Status: todo
Due: 2024-12-11

2. Based on Due Date

a) Ascending Order


Task Management System

LOGOUT

Tasks

Sort by Priority
Low to High

SORT BY DUE DATE

ADD NEW TASK

Note 1
Adding the node

Priority: high
Status: todo
Due: 2024-10-25

Note 2
Adding the note for editing. This is edited note of Note 2

Priority: medium
Status: in_progress
Due: 2024-10-30

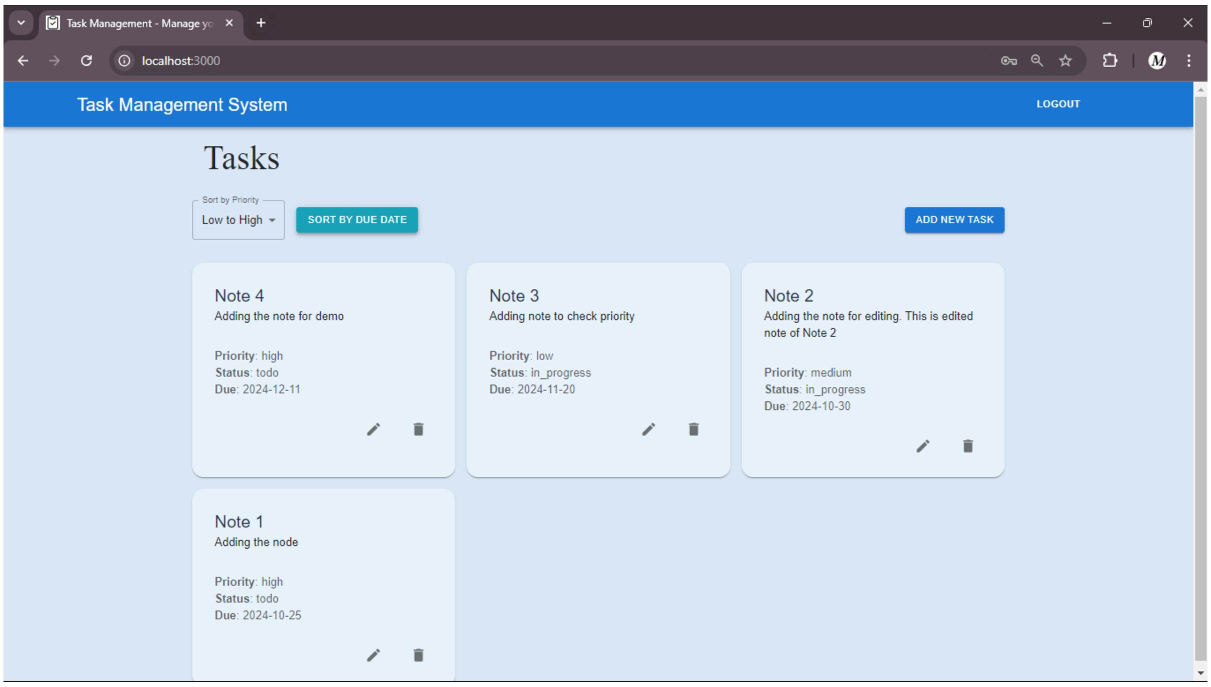
Note 3
Adding note to check priority

Priority: low
Status: in_progress
Due: 2024-11-20

Note 4
Adding the note for demo

Priority: high
Status: todo
Due: 2024-12-11

b) Descending Order


Task Management - Manage yo

localhost:3000

Task Management System

LOGOUT

Tasks

Sort by Priority
Low to High

SORT BY DUE DATE

ADD NEW TASK

Note 4
Adding the note for demo

Priority: high
Status: todo
Due: 2024-12-11

Note 3
Adding note to check priority

Priority: low
Status: in_progress
Due: 2024-11-20

Note 2
Adding the note for editing. This is edited note of Note 2

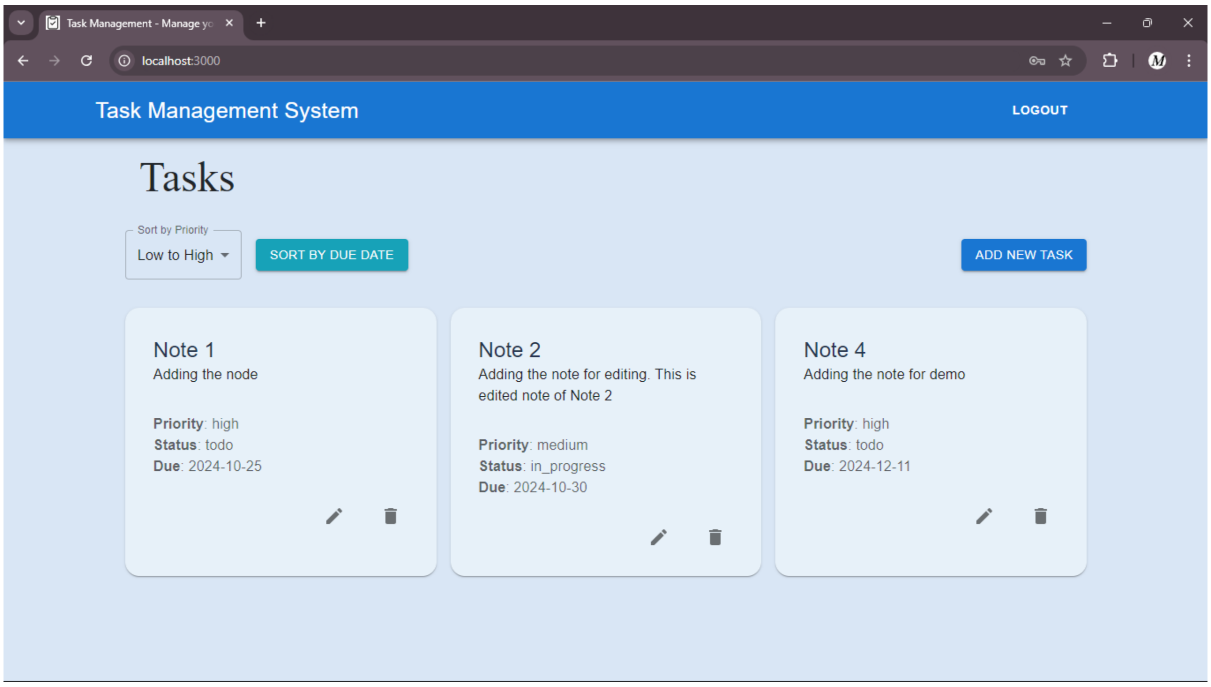
Priority: medium
Status: in_progress
Due: 2024-10-30

Note 1
Adding the node

Priority: high
Status: todo
Due: 2024-10-25

Step 6:

Delete Task


Task Management - Manage yo

localhost:3000

Task Management System

LOGOUT

Tasks

Sort by Priority
Low to High

SORT BY DUE DATE

ADD NEW TASK

Note 1
Adding the node

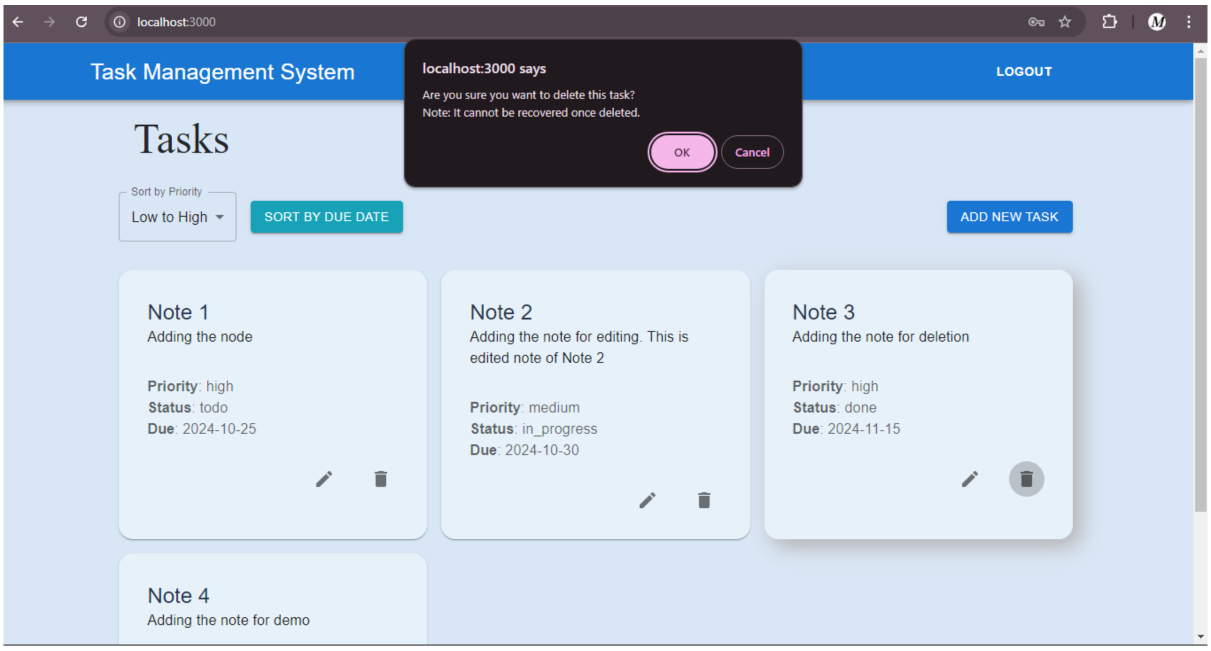
Priority: high
Status: todo
Due: 2024-10-25

Note 2
Adding the note for editing. This is edited note of Note 2

Priority: medium
Status: in_progress
Due: 2024-10-30

Note 4
Adding the note for demo

Priority: high
Status: todo
Due: 2024-12-11



localhost:3000 says

Are you sure you want to delete this task?
Note: It cannot be recovered once deleted.

Note 1
Adding the node

Priority: high
Status: todo
Due: 2024-10-25

Note 2
Adding the note for editing. This is edited note of Note 2

Priority: medium
Status: in_progress
Due: 2024-10-30

Note 3
Adding the note for deletion

Priority: high
Status: done
Due: 2024-11-15

Note 4
Adding the note for demo

Error Handling

Examples of Error Messages and How They're Displayed to the User:

- When trying to create a task without a title:
  - Error message: "Title is required."
- Invalid login attempts result in:
  - Error message: "Invalid credentials. Please try again."

Description of Backend Validation and Error Responses:

- The backend validates inputs and responds with appropriate HTTP status codes (e.g., 400 for bad requests).
- Detailed error messages guide users to correct their input.

Security Measures

Authentication Mechanism:

- JWT (JSON Web Tokens) is used for secure user authentication. Tokens are issued upon successful login and included in the headers of subsequent requests to validate user sessions.

Any Additional Security Features Implemented:

- Password hashing and salting for user passwords to enhance security.
- Rate limiting on API endpoints to prevent brute-force attacks.