Task Management System Documentation

Contents

1 Overview **2**

2 Features **2**

3 API Endpoints **3**
3.1 Authentication Endpoints 3
3.2 Task Endpoints 3

4 Database Schema **3**
4.1 Users Table 3
4.2 Tasks Table 3

5 React Frontend Components **4**
5.1 Task List Page 4
5.2 Task Form 4
5.3 Task Details Page 4

6 Deployment **4**
6.1 Local Deployment 4
6.2 Docker Deployment 4

7 SQL Query **4**
7.1 Get Task Count Grouped by User and Status 4

8 Future Improvements **5**

1 Overview

The Task Management System is a full-stack application that allows users to manage tasks effectively. Users can register, log in, and create tasks. They can also update, delete, filter, and view their tasks based on status and due dates. Each task is private to the user who created it, ensuring data security and privacy.

2 Features

- **User Authentication:**
  - Users must register and log in to access the application.
  - Only authenticated users can create, retrieve, update, or delete tasks.
- **Task Management:**
  - Users can perform CRUD operations (Create, Read, Update, Delete) on their tasks.
  - Task attributes include:
    - **id:** Auto-generated primary key.
    - **title:** Max length 100.
    - **description:** Optional.
    - **status:** Pending, In Progress, Completed.
    - **due_date:** Date format.
    - **created_at:** Auto-generated timestamp.
- **Task Filtering:**
  - Users can filter tasks by their status (Pending, In Progress, Completed).
- **Overdue Tasks:**
  - A dedicated endpoint allows users to view tasks with a past due date.
- **Data Privacy:**
  - Each user can only view or manage their own tasks.
  - Users cannot access tasks created by others.
- **Frontend Integration:**
  - A ReactJS frontend provides an intuitive interface for interacting with the API.
  - Features include a task list page, task form, and task details page.
- **Deployment:**
  - The project is deployed locally or in Docker containers using Docker Compose for easy setup.

3 API Endpoints

3.1 Authentication Endpoints

- **POST /auth/register/**: Register a new user.
- **POST /auth/login/**: Authenticate a user and return a token.

3.2 Task Endpoints

- **GET /tasks/**: Retrieve a list of tasks created by the authenticated user.
  - Optional filtering by status (`?status=pending`).
- **POST /tasks/**: Create a new task.
- **PUT /tasks/¡id¿/**: Update a specific task by ID (only if the task belongs to the authenticated user).
- **DELETE /tasks/¡id¿/**: Delete a specific task by ID (only if the task belongs to the authenticated user).
- **GET /tasks/overdue/**: Retrieve tasks with due dates in the past.

4 Database Schema

4.1 Users Table

Field	Type	Constraints
id	Serial	Primary Key
name	String	Not Null
email	String	Not Null, Unique
password	String	Not Null

4.2 Tasks Table

Field	Type	Constraints
id	Serial	Primary Key
title	String	Not Null, Max Length 100
description	Text	Optional
status	Enum	Not Null (Pending, In Progress, Completed)
due_date	Date	Not Null
created_at	Timestamp	Auto-generated
user_id	ForeignKey	References Users(id), Not Null

5 React Frontend Components

5.1 Task List Page

- Fetch and display tasks for the authenticated user.
- Allow filtering tasks by status using a dropdown.

5.2 Task Form

- Create or update tasks using a form.
- Client-side validation ensures required fields are filled correctly.

5.3 Task Details Page

- Display the full details of a specific task.

6 Deployment

6.1 Local Deployment

1. Set up the backend and frontend on a local development server.
2. Use `CORS` middleware to allow communication between the React frontend and Django backend.

6.2 Docker Deployment

1. Use Docker Compose to orchestrate containers for Django, PostgreSQL, and React.
2. Run `docker-compose up --build`.

7 SQL Query

7.1 Get Task Count Grouped by User and Status

```
SELECT
    u.id AS user_id,
    u.name AS user_name,
    t.status,
    COUNT(t.id) AS task_count
FROM
    users u
LEFT JOIN
    tasks t ON u.id = t.user_id
GROUP BY
    u.id, u.name, t.status;
```

8 Future Improvements

- Add email reminders for overdue tasks.
- Enable task sharing with permissions.
- Add additional filtering options (e.g., filter by due date).