Task for Software Development Interns: Building an Enhanced Task Management System with Advanced Features

Overview

Building on the foundation of your Task Management System, the next three weeks will focus on enhancing its capabilities, improving scalability, and introducing collaborative features. The tasks are structured weekly to ensure steady progress toward a polished, feature-rich application.

Week 4: Collaborative Features and Notifications

Objective:

Introduce collaborative features, allowing multiple users to interact with tasks and add a notification system.

Tasks

1. **User Collaboration**
    - ○ Update the database schema to support task ownership and sharing with multiple users.
    - ○ Add fields like `owner` and `sharedWith` (array of user IDs).
    - ○ Update API endpoints to ensure users can:
        - ■ Share tasks with other users (`PUT /tasks/:id/share`).
        - ■ Retrieve tasks shared with them (`GET /tasks/shared`).
2. **Real-Time Notifications**
    - ○ Implement a notification system using **Socket.IO** or a similar library.
    - ○ Notify users in real time when:
        - ■ A task is shared with them.
        - ■ A task's status is updated.
    - ○ Create an endpoint (`GET /notifications`) to fetch past notifications.
3. **Frontend Integration**
    - ○ Add a "Share Task" button to the Task Details component. ○

Display real-time notifications in a pop-up or sidebar.

4. **Testing and Error Handling**
   - Ensure only authorized users can share or edit tasks.
   - Handle errors gracefully for invalid sharing attempts.

Deliverables

- Real-time notifications working seamlessly.
- Functionality to share tasks with other users.
- Updated database schema and API endpoints.
- Code pushed to GitHub.

Week 5: Advanced Analytics and Reporting

Objective:
Introduce analytics to provide insights into task progress and completion trends.

Tasks

1. **Analytics Dashboard**
   - Develop a new **Dashboard** component showing:
     - Total tasks created, completed, and pending.
     - Weekly/monthly task trends (e.g., tasks completed vs. overdue).
     - Status breakdown (e.g., pie chart of Pending, In Progress, Completed).
   - Use **Chart.js** or **Recharts** for visualizing data.
2. **Backend Enhancements**
   - Add new endpoints for analytics:
     - `GET /analytics/overview`: Fetch summary stats.
     - `GET /analytics/trends`: Fetch weekly or monthly trends.
3. **Frontend Integration**
   - Build the analytics dashboard using dynamic data from the backend

   . ○ Display data visually in charts and graphs.
4. **Data Optimization**

- ○ Use database aggregations (e.g., MongoDB Aggregation Framework) to compute analytics efficiently.
- ○ Optimize API calls to minimize response time.

Deliverables

- A functional analytics dashboard with visualizations.
- Backend endpoints for analytics data.
- Optimized database queries for reporting.
- Code and dashboard screenshots uploaded to GitHub.

Week 6: Deployment, Documentation, and Final Enhancements

Objective:

Prepare the project for production with proper documentation, deployment, and final touches.

Tasks

1. **Deployment**

   - ○ Deploy the application to a cloud platform (e.g., **Render**, **Vercel**, or **Heroku**).
   - ○ Ensure both frontend and backend are deployed and accessible via a single URL

2. **Final Enhancements**

   - ○ Implement dark mode for the frontend.
   - ○ Add support for task attachments (e.g., images, files).
   - ○ Improve the UI/UX for mobile responsiveness.

3. **Documentation**

   - ○ Create a comprehensive **README** for the GitHub repository, including:
     - ■ Setup instructions for both backend and frontend.
     - ■ API documentation with example requests and responses.
     - ■ Screenshots and descriptions of key features.

4. **Testing**
    - Conduct end-to-end testing to ensure all features work as expected.
    - Fix any bugs identified during testing.

Deliverables

- Fully deployed project with live URL.
- A detailed README file in the GitHub repository.
- Dark mode and task attachment functionality implemented.
- A recorded video showcasing the deployed application and its features.

Evaluation Criteria

1. **Feature Implementation:**
    - Real-time notifications and task sharing.
    - Analytics dashboard functionality.
    - Final enhancements like dark mode and attachments.
2. **Performance:**
    - Optimized backend and responsive frontend. ○
Fast API response times and real-time features.

3. **Documentation:**
    - Clear and comprehensive README.
    - Well-structured and commented code.
4. **Deployment:**
    - Smoothly running application on a live URL.

Final Deadline

Complete Project Submission: 24th July, 2025