INTERNET AND WEB SYSTEMS - 1 PROJECT PROPOSAL

PRATHAP RAMACHANDRA (#0212954)

1. Introduction

The Advanced To-Do List App is a web-based task management solution that allows users to manage, track, and collaborate on projects more efficiently. It includes features like as work categories, priorities, real-time collaboration, repeating tasks, and smart reminders to keep users productive and organized. The software also includes task metrics and customizable dashboards to improve the user experience.

2. What?

This website lets users create, update, delete, and manage tasks. It includes features for assigning priorities, categorizing tasks, sharing tasks with colleagues, tracking task progress, and getting notifications. The key features include:

- Task categorization (Work, Personal, School)
- Recurring tasks (Daily, Weekly, Monthly)
- Real-time collaboration
- Smart reminders and notifications
- Analytics dashboard for task completion rates and productivity tracking

3. Why?

This Advanced To-Do List App was designed to meet the growing demand for a comprehensive, efficient, and collaborative task management solution. Many users, particularly professionals and teams, encounter substantial issues with current solutions that are either overly simplistic, lack advanced functionality, or do not support real-time collaboration. Here's why creating this website is important, and how it benefits users:

Solving Real-Time Collaboration Challenges

Problem: Many to-do list apps lack significant real-time collaborative features. Users cannot collaborate effectively on projects, resulting in repeated work and confusion.

Solution: This app supports real-time collaboration, allowing numerous users to work on tasks concurrently. It guarantees that work updates are immediately visible to everyone, eliminating delays and enhancing team communication.

4. List of Supported Features

- Task creation, editing, and deletion
- Classifying tasks and setting priorities
- Smart reminders and notifications
- Recurring tasks
- Task sharing and real-time collaboration
- Analytics dashboard featuring productivity tracking and task completion rate
- Customizable themes (light/dark mode)
- Filtering and sorting tasks

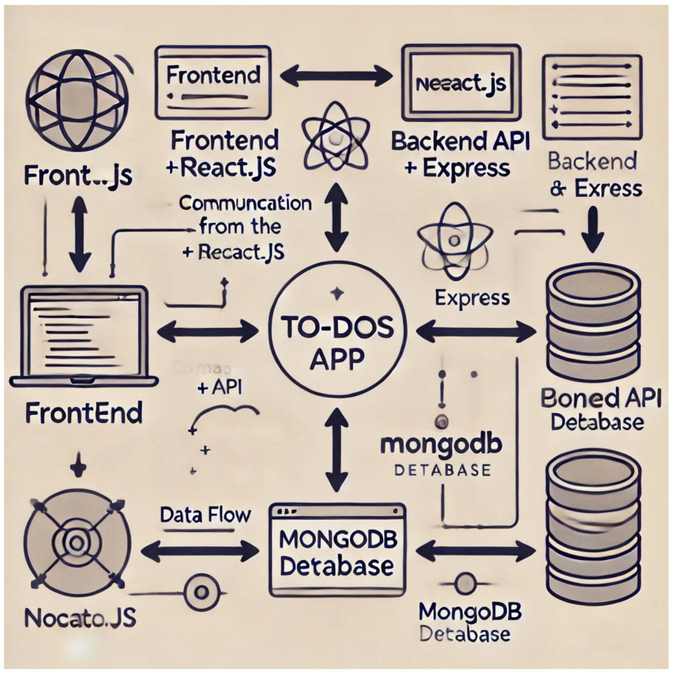
5. List of Not Supported Features

- Tasks cannot be managed in an offline state.
- Integration of mobile apps (there isn't one as of yet)
- Voice command integration (complete command integration is not available, however voice input is possible for tasks)

6. List of Future Planned Features

- iOS and Android mobile applications
- sophisticated analytics and AI-powered task recommendations
- Integration with Microsoft Outlook and Google Calendar
- Task management with voice command integration
- Task boards with milestones depending on projects

7. How?



1. **High-Level Diagram**:
   A high-level diagram of the system will include user interaction with the frontend (React.js), backend API (Node.js + Express), and MongoDB as the database.

2. **List of Components/Modules**:
   - **Frontend**: HTML, CSS, JavaScript, React.js for dynamic UI
   - **Backend**: Node.js, Express.js for API development
   - **Database**: MongoDB for task storage and management
   - **Notification Module**: Push notifications using Web Push API and emails using Nodemailer
3. **Languages to be used for each module**:
   - **Frontend**: HTML, CSS, JavaScript (React.js)
   - **Backend**: JavaScript (Node.js, Express.js)
   - **Database**: MongoDB
4. **List of 3rd Party/Open Source Modules**:
   - **Nodemailer**: For sending email notifications
   - **Web Push API**: For push notifications
   - **Mongoose**: For MongoDB integration
   - **Socket.io**: For real-time collaboration
5. **List of 3rd Party Services/APIs**:
   - **Paid/Free**: All services used (Nodemailer, Web Push, Socket.io) will be free.
6. **REST API Endpoints with Payloads**:
   - **POST `/api/tasks`**: Creates a new task with title, description, priority, category, and due date.
   - **GET `/api/tasks`**: Fetches all tasks.
   - **PUT `/api/tasks/:id`**: Updates an existing task.
   - **DELETE `/api/tasks/:id`**: Deletes a task.
7. **Build Steps**:
   - **Frontend**: Use React's `npm run build` for creating production-ready code.
   - **Backend**: Deploy with Node.js server (`npm start`).
8. **Install Steps**:
   - Install the app using `npm install`.
   - Run the MongoDB server using `mongod`.

- Run the app using `npm start`.

8. GitHub Information

9. References

1. Node.js Documentation
2. MongoDB Documentation
3. React.js Documentation