Title

Task Mgmt. App

Description

This proj. aims to dev. a Task Mgmt. App that allows users to create, manage, and track tasks efficiently. Users can add tasks, set deadlines, assign priorities, and collaborate with team members.

Scope

- Target Aud.: Professionals, teams, and individuals seeking task org.
- Content: Users can create tasks, categorize them (e.g., work, personal), and set reminders.
- Formats: The app will be available as a web app (WA) and mobile app (MA).
- Features:
    - Task creation and editing.
    - Deadline setting and reminders.
    - Collaboration tools (comments, file attachments).
    - Dashboard for task overview.

Objectives

- User Eng.: Provide an intuitive UI for task management to enhance productivity.
- Collaboration: Facilitate teamwork through shared tasks and comments.
- Productivity Growth: Help users increase efficiency in managing their tasks.

Architecture Details

- Architecture Pattern: Monolithic architecture.
- Containerization: Single container with a single component that encapsulates the entire app (frontend and backend).

Monolithic Architecture Overview

In a monolithic architecture, all components of the app (UI, business logic, data access) are integrated into a single codebase. This simplifies deployment and

scaling but may pose challenges in flexibility and maintainability as the app grows.

Technology Details

- Frontend Techs: HTML, CSS, JS (React or Angular).
- Backend Techs: Node.js with Express for server-side logic.
- DB: MongoDB or MySQL for storing user data and task metadata.
- Auth. Lib: Use libraries like JWT (JSON Web Token) for user authentication.
- Hosting: Deploy on platforms like Heroku or AWS.

Team Composition

- PM: Project Manager oversees proj. timelines and team coordination.
- FE Dev(s): Frontend Developer(s) responsible for UI/UX design and implementation.
- BE Dev(s): Backend Developer(s) manages server-side logic and DB interactions.
- Designer: Creates assets and ensures the app has an appealing visual design.
- QA Tester: Conducts testing to ensure functionality and UX are optimal.

Timeline Details

1. Week 1 (Jan 28 - Feb 3): Define proj. goals, scope, and assemble the team; finalize tech stack.
2. Week 2 (Feb 4 - Feb 10): Design UI/UX mockups; begin FE dev.
3. Week 3 (Feb 11 - Feb 17): Develop BE services; set up DB schema and API endpoints.
4. Week 4 (Feb 18 - Feb 24): Integrate FE with BE; implement auth. features using JWT.
5. Week 5 (Feb 25 - Mar 3): Conduct thorough testing; fix bugs and optimize performance.
6. Week 6 (Mar 4 - Mar 10): Launch the app; initiate mktg. strategies to promote user engagement.