COLLEGE CODE: 8203

COLLEGE: AVC COLLEGE OF ENGINEERING

DEPARTMENT: INFORMATION TECHNOLOGY

STUDENT NMID: C9C912D4BD1EB53B5A04BF0E7A44DD4E

ROLL NO: 23IT84

DATE: 03/10/2025

Completed the project named as Phase 4

TECHNOLOGY PROJECT NAME: To-Do App with React Hooks

SUBMITTED BY,

NAME: Ragulkumar A

MOBILE NO: 9360457452

Enhancements & Deployment

After completing the basic functionality (CRUD operations with React Hooks), **it** focuses on turning the To-Do App into a **scalable, user-friendly, and production-ready application**. This stage ensures the app is **feature-rich, secure, and deployable**.

Additional Features (Enhancing Functionality)

1. Subtasks Support
   - Each main task can have subtasks.
   - Example: *“Finish Project Report”* → Subtasks: *“Write Abstract”*, *“Create PPT”*, *“Proofread”*.
   - Managed using nested state in React (useState for main task + subtasks array).
2. Reminders & Notifications
   - Add a dueDate field for each task.
   - Show alert when a task deadline is close (using setTimeout or useEffect).
   - Push notifications (optional via service workers in PWA).
3. Task Prioritization
   - Tasks can be assigned High, Medium, or Low priority.
   - UI displays priority with color tags (red, yellow, green).
   - Helps users focus on critical tasks first.
4. Search & Filtering
   - Search bar to filter tasks by name or description.
   - Filters: *Completed*, *Pending*, *Overdue*, *By Category*.
   - Implemented using JavaScript array filter + controlled input state.
5. Dark/Light Mode Toggle
   - Theme toggle button stored in global state (useContext).

UI/UX Improvements

1. Responsive Design

   - Mobile-first design using Flexbox/Grid or Tailwind CSS.
   - Different layouts for small vs large screens.

2. Animations & Transitions

   - Smooth transitions when adding/removing tasks using Framer Motion.
   - Hover and click effects for better interactivity.

3. Color-Coded Categories

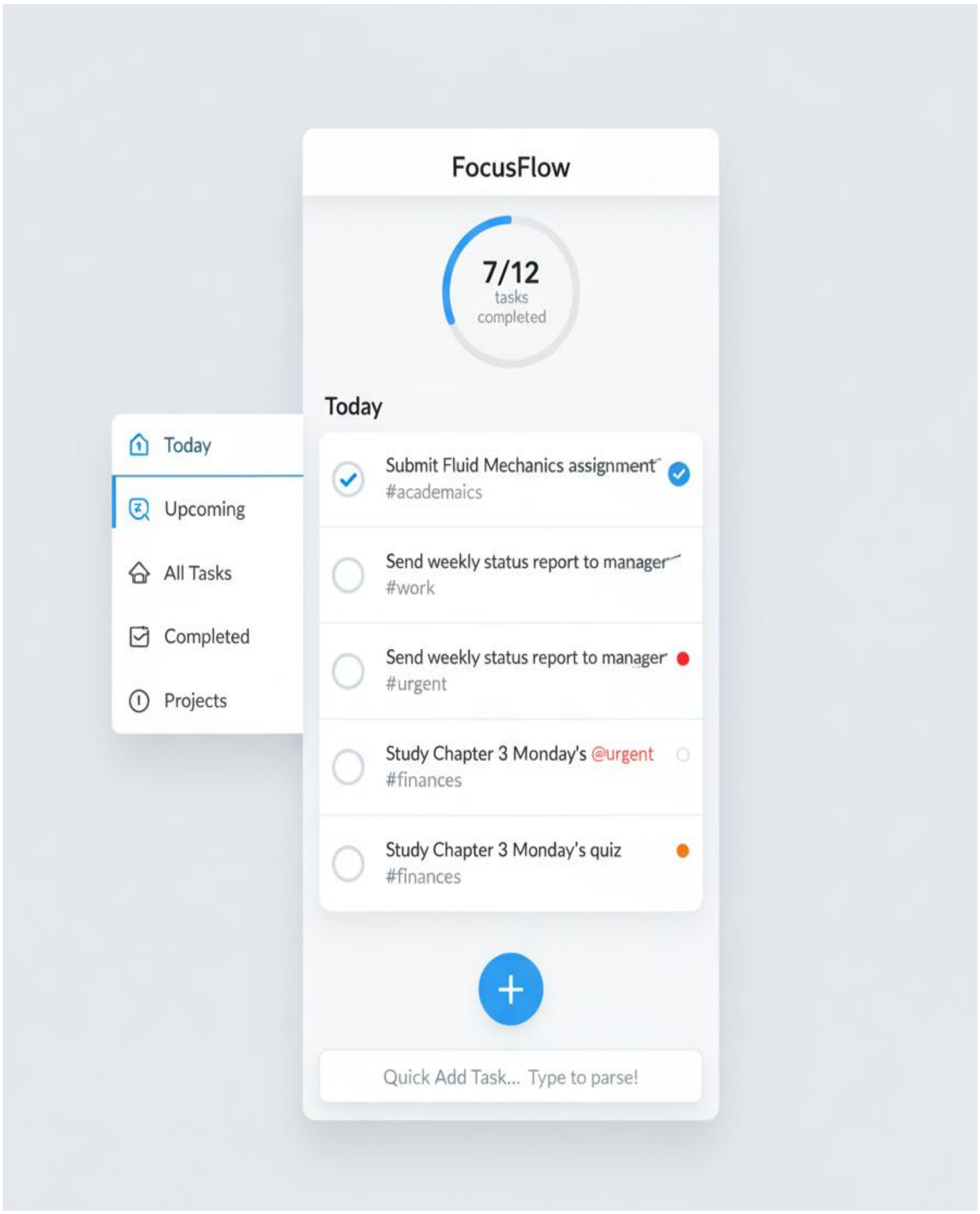
   - Work tasks (blue), Study (green), Personal (orange).
   - Helps users quickly identify task type.

4. Accessibility Enhancements

   - ARIA labels for screen readers.
   - Keyboard navigation (tab, enter, delete key for tasks).
   - Proper contrast for color-blind users.

5. Minimal Dashboard Design

   - Home screen shows task summary: *“10 Tasks Pending, 5 Completed”*.
   - Graphical view (pie chart using Recharts/Chart.js).



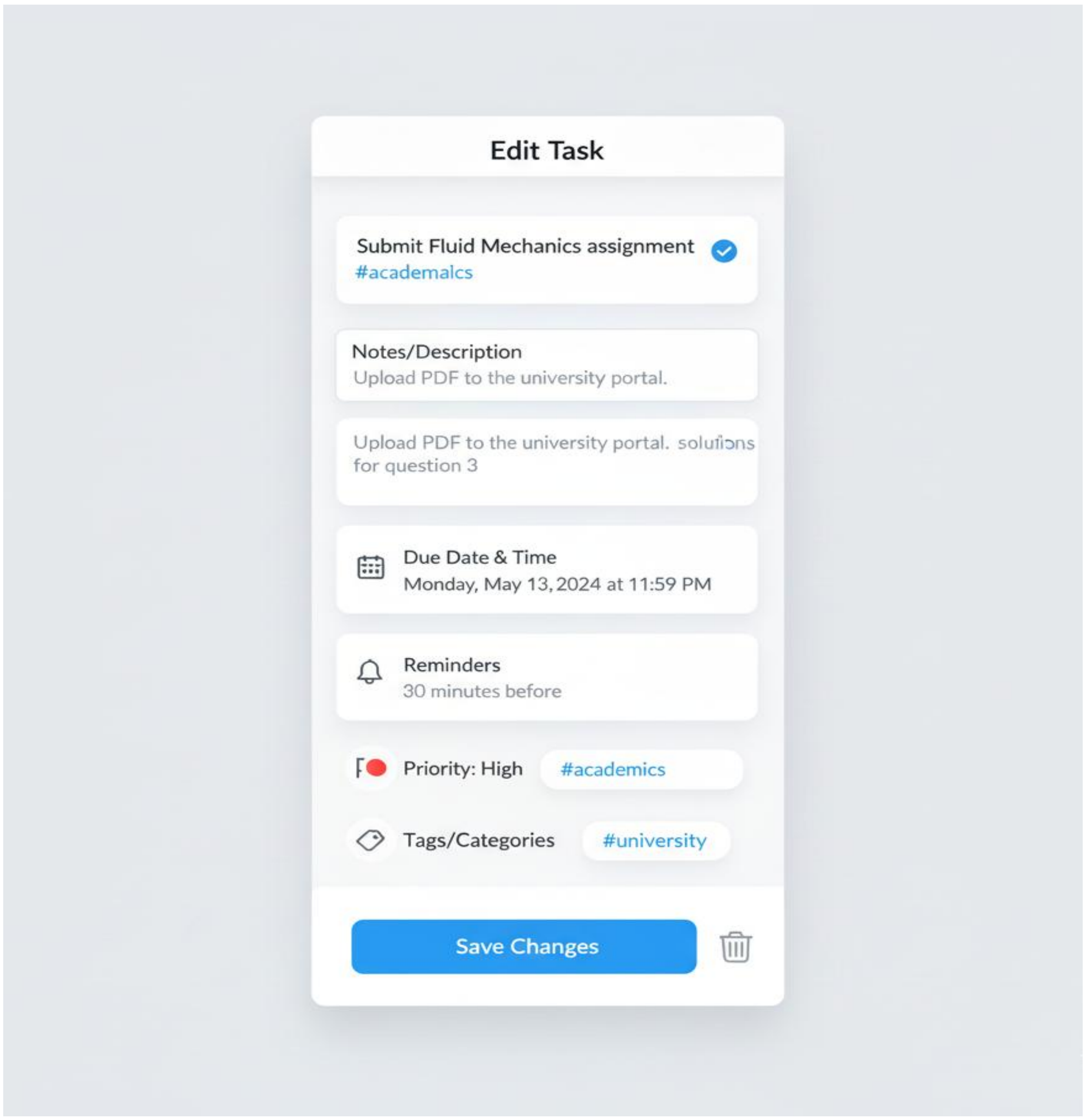
FocusFlow

7/12 tasks completed

Today

- Submit Fluid Mechanics assignment #academaics ✓
- Send weekly status report to manager #work
- Send weekly status report to manager #urgent ●
- Study Chapter 3 Monday's @urgent #finances
- Study Chapter 3 Monday's quiz #finances ●

Quick Add Task... Type to parse!



Edit Task

Submit Fluid Mechanics assignment
#academaics

Notes/Description
Upload PDF to the university portal.

Upload PDF to the university portal. solutions for question 3

Due Date & Time
Monday, May 13, 2024 at 11:59 PM

Reminders
30 minutes before

Priority: High #academics

Tags/Categories #university

API Enhancements (For Scalable Backend)

1. Backend Integration
    - Replace localStorage with REST APIs (Node.js + Express).
    - Store tasks in MongoDB/MySQL for persistence.
2. Authentication & User Accounts
    - Signup/Login using JWT or Firebase Auth.
    - Each user has their own task list.
3. Task Synchronization

- Sync tasks across devices.
- Example: Add task on laptop → automatically visible on mobile.

4. Advanced Filtering APIs
    - /tasks?status=completed → fetch only completed tasks.
    - /tasks?dueDate=today → fetch tasks due today.
5. Data Validation & Error Handling
    - Backend checks for empty tasks.
    - Prevent duplicate task entries.
    - Error handling with HTTP codes (400, 401, 404, 500).

Performance & Security Checks

1. Performance Optimization
    - Use React.memo to avoid re-rendering unchanged tasks.
    - useCallback for stable function references.
    - Code splitting and lazy loading (React.lazy + Suspense).
2. Security Improvements
    - Password hashing (bcrypt for Node.js backend).
    - JWT token expiry for secure sessions.
    - Input validation to prevent XSS or SQL Injection.
3. Database Indexing
    - Add indexes on dueDate and priority for fast queries.

Testing of Enhancements

1. Unit Testing
    - Test React components like TaskForm, TaskList using Jest/React Testing Library.
    - Example: Check if a task is added correctly.
2. Integration Testing
    - Verify frontend ↔ backend APIs.

- Example: Add task in React → stored in MongoDB → fetches correctly.

3. Manual Testing
    - Try adding, editing, deleting, and filtering tasks.
    - Test on different devices (mobile, tablet, desktop).
4. Cross-Browser Testing
    - Chrome, Firefox, Safari, Edge.

Deployment

1. Frontend Deployment
    - ✓ Deploy React app on Netlify or Vercel.
    - ✓ Automatic build from GitHub repo.
2. Backend Deployment
    - ✓ Deploy Node.js + Express backend on Render / Railway / Heroku / AWS.
    - ✓ Connect backend API URL to frontend.
3. Database Deployment
    - ✓ MongoDB Atlas or MySQL Cloud instance.
4. CI/CD Pipeline (Optional)
    - ✓ Use GitHub Actions to auto-deploy after every push.