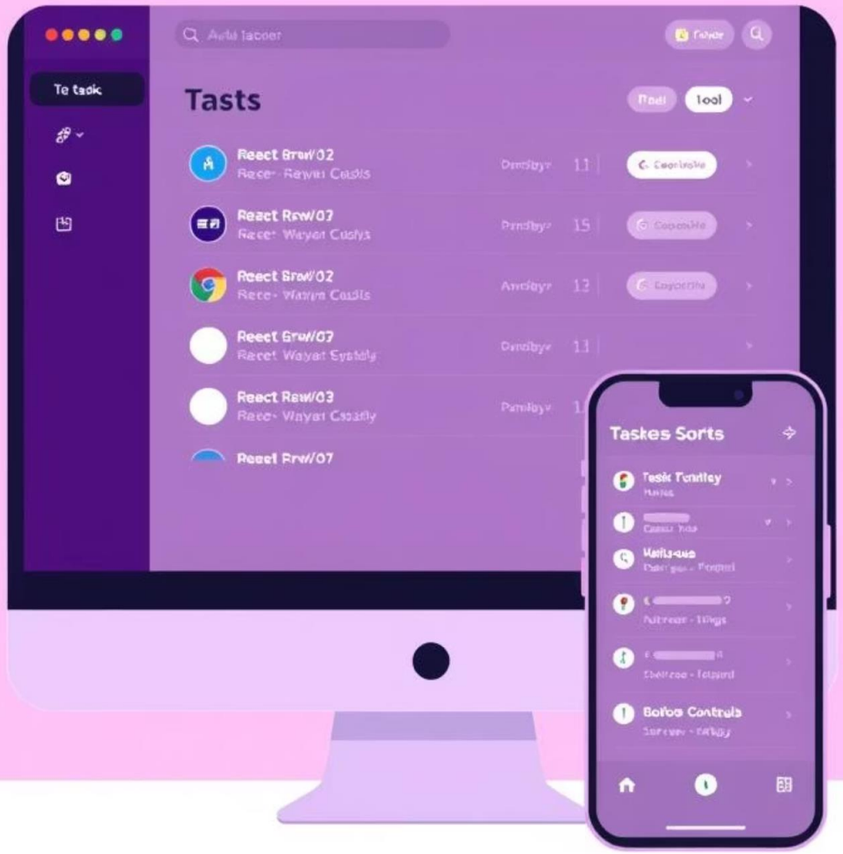
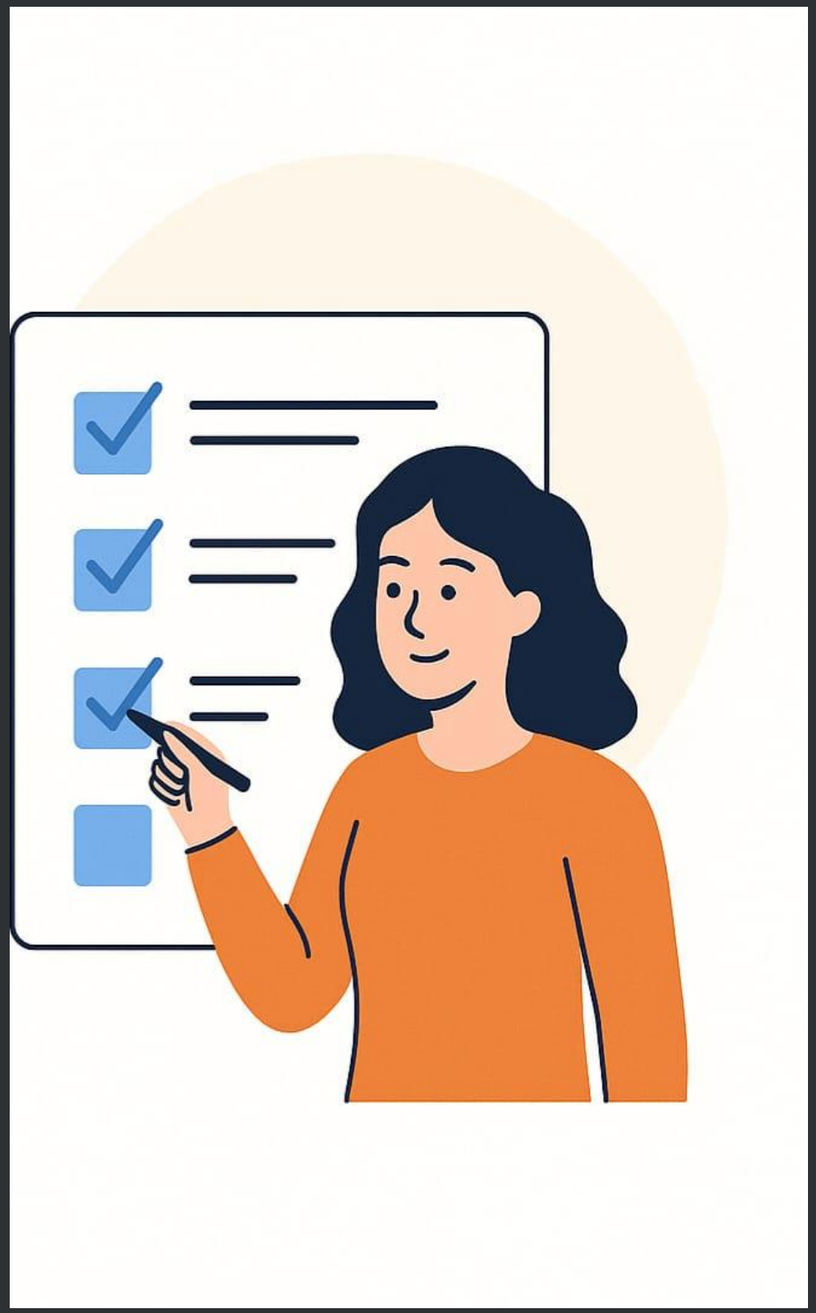


Task Management System

A Modern Solution for Productivity

Presented by: Priyanka Jaybhaye

Date: August 6, 2025



Introduction:

Purpose: improve user Productivity and task organization

Features:

Add, update, and delete tasks

Access tasks from any device

Built with: React.js and Firebase

Key Benefit: Real-time updates and user-friendly interface

Why Task Management?

Organize Efficiently

Streamline daily tasks and manage your workload with ease, ensuring no important detail is missed.

Track Progress

Monitor the completion status of each task in real-time, providing clear visibility into your productivity.

Accessible Anywhere

Leverage cloud-based storage for seamless access to your tasks across all devices, wherever you are.

Built With Modern Tools

Our task management application leverages a robust and scalable technology stack, designed for performance and real-time capabilities.

React.js

A powerful JavaScript library for building dynamic and interactive user interfaces.

Firestore

A flexible, scalable NoSQL cloud database for storing and syncing data in real-time.

Authentication

Firebase Authentication offers secure and easy-to-use identity management (future scope).

Key Advantages

Real-time Updates: Instant synchronization of task data across all connected devices.

Scalable Backend: Firebase's infrastructure supports effortless scaling as user base grows.

Developer-Friendly: React's component-based architecture simplifies development and maintenance.

What Can It Do?

Add New Tasks

Quickly create and save new tasks with intuitive input fields.

Toggle Completion

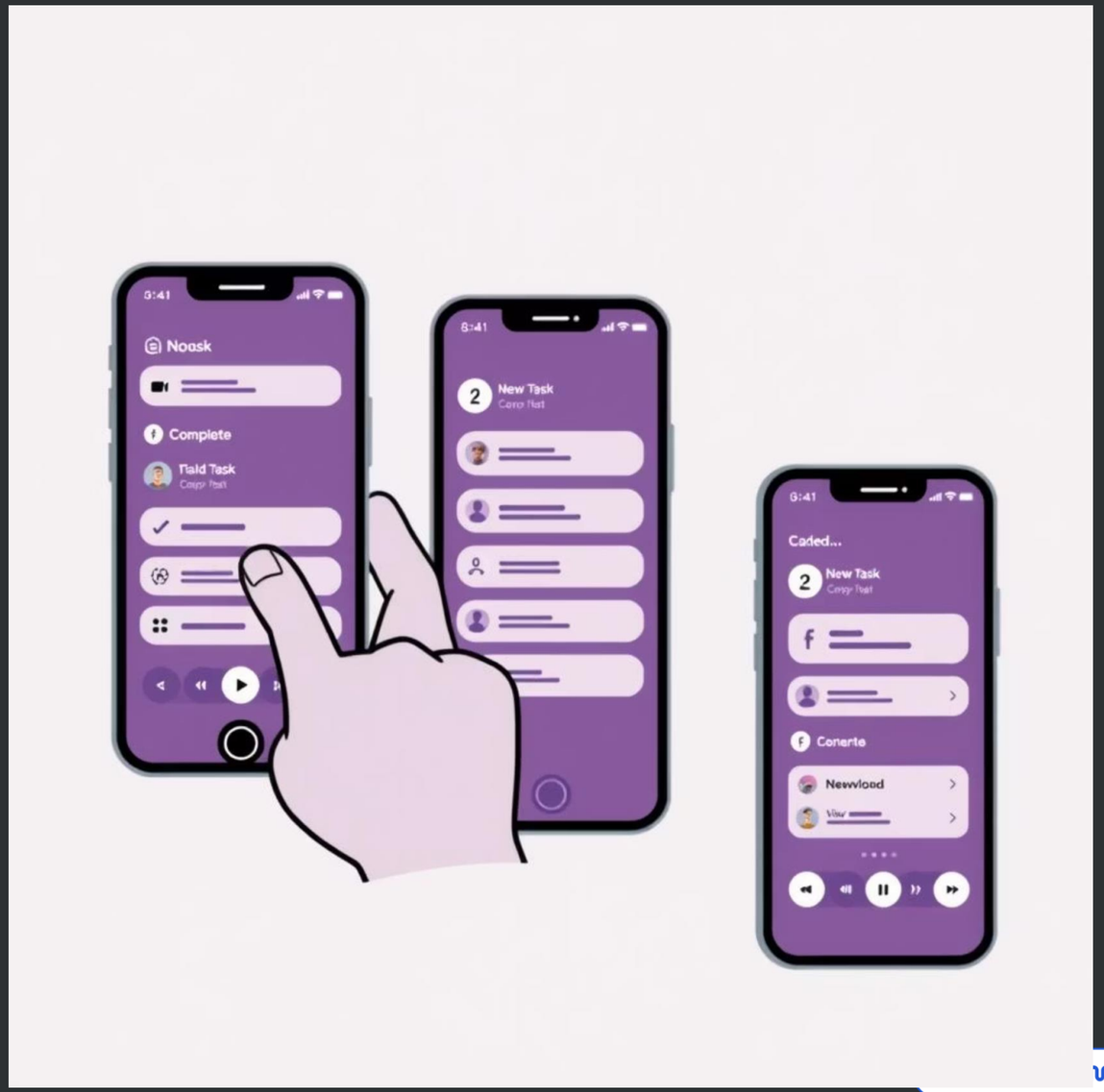
Easily mark tasks as complete or incomplete with a single click.

Delete Tasks

Remove unnecessary tasks to keep your list clean and focused.

Responsive Design

Access and manage tasks seamlessly on any device, from desktops to smartphones.



How It Works: Code Highlights

The core functionality of our application relies on efficient data manipulation using Firebase and responsive UI updates with React Hooks.

Firebase Operations

Seamlessly interact with Firestore for data persistence.

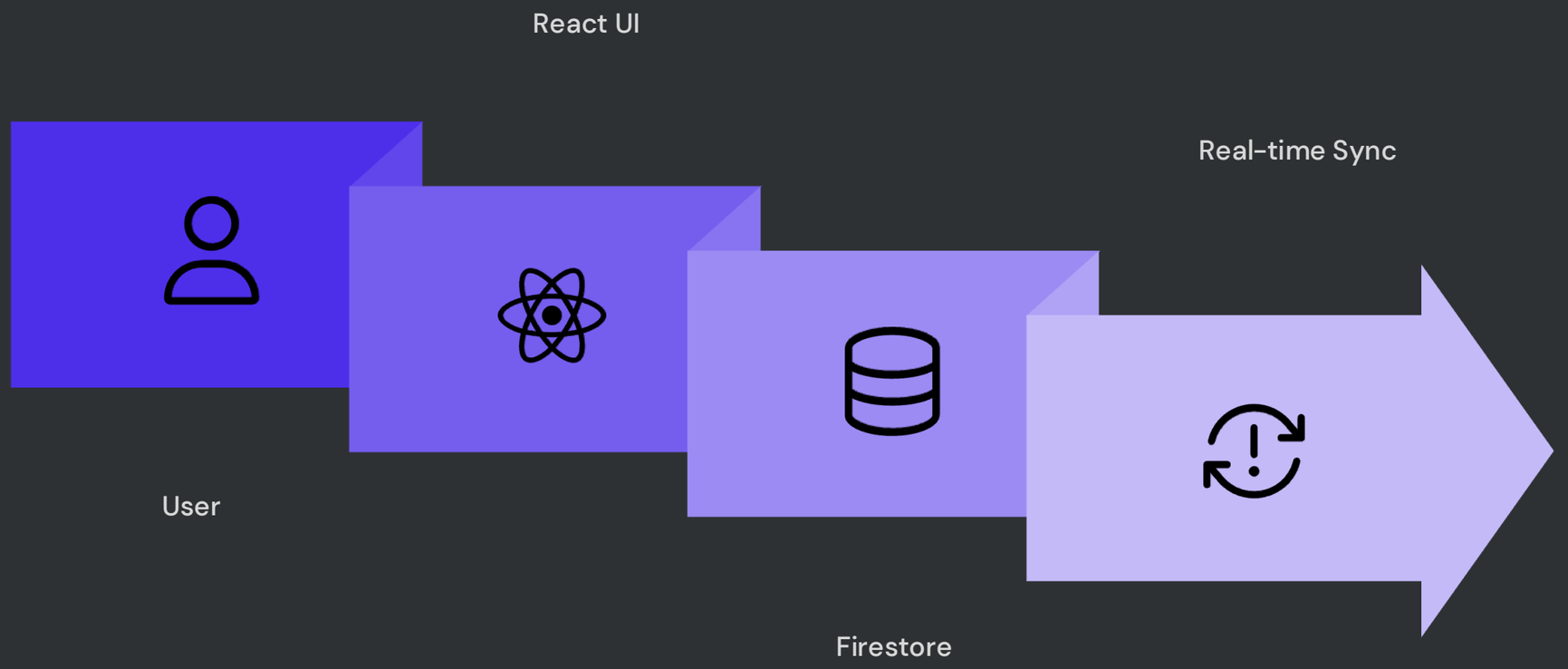
```
import {   collection,   addDoc,   updateDoc,   deleteDoc,   doc } from "firebase/firestore";import { db } from "./firebaseConfig";// Add a new taskconst addTask = async (text) => {  await addDoc(collection(db, "tasks"), { text, completed: false });};// Update task statusconst toggleTask = async (id, completed) => {  const taskRef = doc(db, "tasks", id);  await updateDoc(taskRef, { completed: !completed });};// Delete a taskconst deleteTask = async (id) => {  await deleteDoc(doc(db, "tasks", id));};
```

React Hooks

Manage component state and side effects effectively.

```
import React, { useState, useEffect } from "react";import {  collection,  onSnapshot,  query } from "firebase/firestore";import { db } from "./firebaseConfig";function TaskList() { const [tasks, setTasks] = useState([]); useEffect(() => { const q = query(collection(db, "tasks")); const unsubscribe = onSnapshot(q, (snapshot) => { const tasksData = snapshot.docs.map((doc) => ({ id: doc.id, ...doc.data(), })); setTasks(tasksData); }); return () => unsubscribe(); // Cleanup }, []); return ( <ul> {tasks.map((task) => ( <li key={task.id}>{task.text}</li> ))} </ul> );}
```

Data Flow: System Architecture



Challenges & Solutions

Developing a real-time application comes with its unique set of challenges. Here's how we addressed them:

Challenge	Solution
Real-time Data Synchronization	Utilized Firestore's `onSnapshot` listeners to instantly reflect database changes in the UI.
Efficient State Management	Integrated React Hooks (`useState`, `useEffect`) with Firebase data fetching for optimized state updates.
Offline Capabilities	Leveraged Firestore's built-in offline support to allow users to interact with data even without internet access.
Scalability for Growth	Chose Firebase as a backend-as-a-service (BaaS) to inherently handle increased user loads and data volume.

Future Enhancements

Our vision extends beyond basic task management. Here's a roadmap of exciting features we plan to implement:

User Authentication

Implement secure login/signup with Firebase Authentication to enable personalized task lists.

Due Dates & Reminders

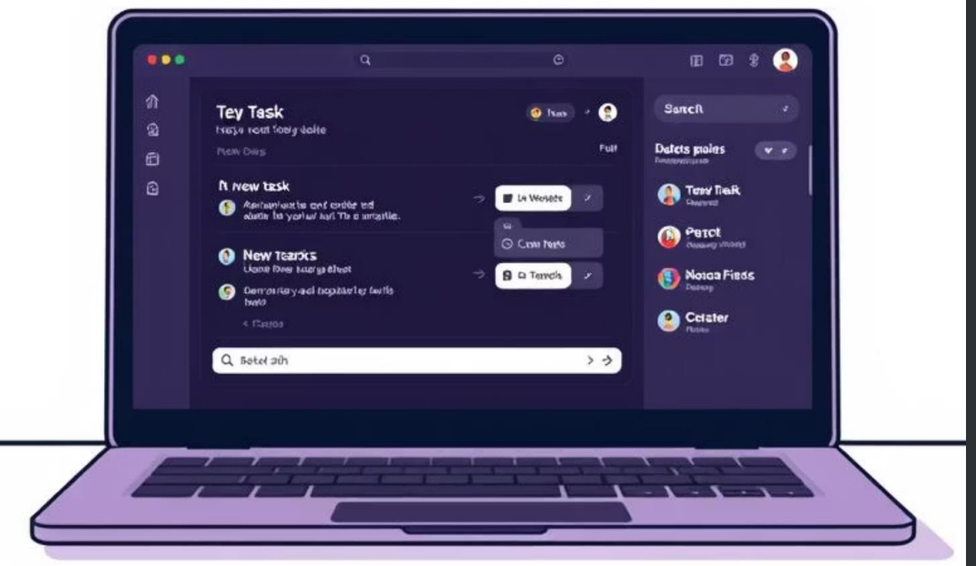
Add functionality for setting task deadlines and receiving timely notifications to stay on track.

Task Categories

Allow users to categorize tasks for better organization and filtering (e.g., Work, Personal, Shopping).

Task Sharing

Enable collaboration by allowing users to share tasks with others and work together on projects.



Demo

Let's see the application in action. This brief demonstration will showcase the core functionalities we've discussed.

Key features to observe during the demo:

- Adding new tasks and instant appearance in the list.
- Toggling completion status with visual feedback.
- Deleting tasks and real-time removal from the list.
- The seamless responsiveness across different simulated device sizes.

Questions?

Thank you for your time and attention. I am now happy to answer any questions you may have about the Task Management App, its development, or future prospects.