Your TodoApp — Final Summary

Tech Stack

Frontend: React (Vite), React Router, Custom CSS

Backend: Node.js, Express

Database: MongoDB (Mongoose)

Auth: JWT + bcryptjs

Icons: React Icons

Key Features You've Implemented

User Authentication System

Signup + Login with JWT token creation

Passwords hashed using bcryptjs

Token saved in localStorage and used for API protection

Protected Routes

Only logged-in users can access /home

Login/Signup pages redirect if already logged in

✅ Todo CRUD Functionality

Add, Delete, Update, and Mark Completed

Sorted todos by **priority**

Added **deadline support**

Each todo linked to a user (userId)

Sorting System

Sort todos by **priority**, **deadline**, or **createdAt**

Navbar & Footer

Navbar dynamically shows username & logout

Footer inspired by Todoist — clean, modern, responsive

Global State Management

AuthContext + useAuthContext for login/logout state

Centralized authentication logic

⚡ Error Handling

Frontend + backend error messages

Validation for missing fields, duplicate users, invalid tokens

---

Project Structure

```
todoApp/
│
├── backend/
│   ├── models/
│   ├── routes/
│   ├── controller/
│   ├── middleware/
│   ├── server.js
│
├── frontend/
│   ├── src/
│   │   ├── components/
│   │   ├── context/
│   │   ├── hooks/
│   │   ├── pages/
│   │   ├── App.jsx
│   │   ├── main.jsx
│   │   ├── index.css
│
└── README.md
```

What You Learned

How **frontend and backend communicate** through REST APIs

How **JWT authentication** secures routes

How to use **React Context** for global state

Proper structure of a **full-stack MERN app**

How to deploy-ready a project step-by-step