AI-Assisted Todo App – Candidate Checklist

Goal:

Build a Todo app using AI tools (**Cursor, GPT, Copilot, Claude**).

Assessed on **AI fluency**, **code quality**, **best practices**.

💡 **UI/UX:** You are free to design and structure the UI/UX however you want — be creative and make it your own.

Bonus: Deploy to production.

Tech Stack (pick one backend)

- **Frontend**: React + TypeScript + Vite
- **Data/Auth**: Supabase (Postgres, Auth: email/password or magic link)
- **Backend**: Node.js (Fastify/Express + Zod) **OR** Python (FastAPI + Pydantic)

Functional Requirements

Auth

- Sign up / Sign in (email/password or magic link)
- RLS – users only see their own data

Todos

- Fields: title*, description, due_date, priority (low/med/high), completed, created_at
- Create, read, update, delete todos

Views & UX

- Filters: all, active, completed, due-today, priority
- Bulk actions: toggle all, delete completed

- Loading states, empty states, error feedback

Quality

- Strict TypeScript (no any)
- Client + server validation
- Reusable, clean components
- Accessible forms/interactions

Deployment

- Frontend on Vercel or Netlify
- Backend (if used) on Render or Railway
- Document env vars in README

Bonus Features

(optional)

- Deploy the app
- Real-time updates across tabs (Supabase Realtime)
- Dark mode toggle with persistence
- Mobile-friendly layout improvements
- Sort todos by due date or priority

Submission Checklist

- GitHub repo with clean commits
- /prompts/ folder (AI prompts used)
- README with:
    - Setup instructions
    - Env variables
    - DB schema & RLS notes
    - Architecture overview
    - Deployment steps
    - AI usage summary

- Live URLs: frontend (+ backend if applicable)

References

- Cursor tutorial – Todo website in 5 min (Medium)
- Connect Supabase MCP (YouTube)
- More older video that doesnt use mcp - React Supabase CRUD Tutorial (YouTube)
- Deploy to Vercel – Step By Step (YouTube)

—------------------—------------------—------------------—------------------—--------------------—-----

Guidelines for Completing the Task

1. Approach the Task in Stages

Stage 1 – Setup
- Create a Supabase project and run your schema.
- Scaffold a React + Vite + TypeScript project.
- Choose your backend (or go frontend-only if RLS is enforced properly).

Stage 2 – Auth
- Implement Supabase Auth (email/password or magic link).
- Ensure users only see their own data.

Stage 3 – CRUD
- Implement Create, Read, Update, Delete for todos.
- Validate inputs on both client and server.

Stage 4 – UX
- Add filters, bulk actions, and loading/empty/error states.
- Ensure basic accessibility.

Stage 5 – Deploy (Bonus)
- Deploy frontend to Vercel/Netlify.
- Deploy backend (if used) to Render/Railway.
- Test everything in production.

2. Working With AI Effectively
- Be Specific – Tell the AI your exact stack, desired libraries, and output format.
- Iterate – Don't accept the first answer; ask for refinements.
- Ask for Explanations – Make sure you understand the code before using it.
- Debug Collaboratively – Share error messages and ask the AI to help

troubleshoot.
- Refactor with AI – Once it works, ask AI to improve structure and readability.

—----

3. AI Prompt Examples

<u>Project Scaffolding</u>

"Create a React + TypeScript + Vite project for a Todo app with Supabase Auth and CRUD functionality. Use Tailwind for styling and organize code into reusable components."

<u>Auth Setup</u>

"Show me how to integrate Supabase Auth (email/password) into my React app, including sign-up, sign-in, and session persistence."

Database Queries

“Write functions to create, read, update, and delete todos in Supabase, scoped to the authenticated user using Row Level Security.”

Validation

“Add Zod validation to ensure title is required (max 200 chars) and priority is one of low, med, high.”

UI Features

“Add filter buttons for all, active, completed, due-today, and priority. Ensure they update the displayed list without page reload.”

Deployment

“Guide me through deploying my Vite + Supabase frontend to Vercel, including setting up environment variables.”

---

4. Best Practices Checklist

- Commit often with meaningful messages.
- Keep AI prompts and responses you used in /prompts/.
- Avoid committing secrets (use .env files).
- Test all user flows before submission.
- Write a README that lets anyone set up your project from scratch.

Free tools you can use

Development & Hosting

- Vercel – Free tier for frontend hosting with CI/CD from GitHub.
- Render – Free tier for backend hosting (750 hours/month for one instance).
- Railway – Free tier with $5/month credit for backend hosting.

Database & Auth

- Supabase – Free tier includes:

Code Assistance (AI)

- Cursor – Free plan with daily AI-assisted coding requests.
- GitHub Copilot – 30-day free trial (after that paid).
- Claude – Free tier with limited daily requests.
- ChatGPT (OpenAI) – Free tier for GPT-3.5, paid for GPT-4.