**Tech Stack Used in This Project**

* **Backend:** Node.js with Express.js
* **Frontend:** React with TypeScript
* **Database:** MongoDB (accessed via Mongoose)

**Why This Stack?**

**Node.js + Express.js (Backend)**

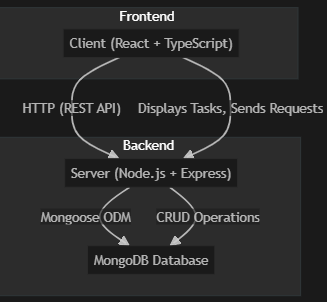
* **Reason:** Node.js with Express provides a fast, efficient, and scalable way to build RESTful APIs using JavaScript, which matches the language used on the frontend for consistency and easier collaboration.

**React + TypeScript (.tsx) (Frontend)**

* **Reason:** React enables building interactive, component-based user interfaces. TypeScript (.tsx files) is used instead of plain JavaScript or JSX because it adds static type checking, which helps catch errors early, improves code quality, and makes the codebase easier to maintain as the project grows.

**MongoDB + Mongoose (Database)**

* **Reason:** MongoDB is a flexible NoSQL database that works well with JavaScript/Node.js. Mongoose provides a straightforward way to define data models and interact with MongoDB, making data management easier and more reliable.

**High Level Architecture Diagram  
**

**Basic Schema**

title: { type: String, required: true },

  description: String,

  completed: { type: Boolean, default: false },

  dueDate: Date

* Each Task is a document in the tasks collection.
* There are no foreign keys or relations in this basic schema.
* All task data is stored in a single document per task.