**Notes application using React.js**

I have Developed a dynamic Notes application using React.js a fully functional note-taking app using React, designed to store and organize notes in real-time. The application offers a user-friendly interface with features such as creating, editing, deleting, and searching notes, login and signup features. It provides a seamless user experience with a responsive design. Implemented a robust backend for managing user data and notes with proper authentication and authorization.

**Key Features**:

* User registration and login with secure password storage.
* Ability to create, update, and delete notes.
* User-friendly interface for managing notes.
* Secure data storage for user information and notes.

Backend: [Node.js, Express.js]

Database: [MongoDB][ const mongoose = require('mongoose');

const mongoURI = "mongodb://localhost:27017/iNotebook"

async function connectToMongo() {

    await mongoose.connect(mongoURI).then(()=> console.log("Connected to Mongo Successfully")).catch(err => console.log(err));

  }

  module.exports = connectToMongo]

Authentication: [JWT, OAuth]

Frontend: [React.js, javascript, css]

Project Link: <https://drive.google.com/file/d/1J1ZKQ8hD6tv0FgRieObilbK1DPSYUdhi/view?usp=drive_link>

**Project Structure**

**src/**

* **components/**
  + Navbar.js – Handles navigation links and authentication status.
  + Login.js – User login form.
  + Signup.js – User signup form.
  + Notes.js – Displays a list of saved notes.
  + NoteItem.js – Represents a single note in the list.
  + NoteItem.js – Represents a single note in the list.
  + Home.js
  + Aboutg.js
  + Alert.js
* **services/**
  + authService.js – Handles API requests related to user authentication (login, signup, and logout).
  + noteService.js – Manages API requests for note creation, fetching, updating, and deletion.
* **context/**
  + NotesContext.js – Provides global state for user authentication.
  + Notestate.js
* **hooks/**
  + useAuth.js – Custom hook for accessing and managing authentication state.
  + useNotes.js – Custom hook for managing notes' state and fetching notes from the backend.
* **utils/**
  + ProtectedRoute.js
* **App.js** – The main application component that handles routing.
* **index.js** – Entry point that renders the app.
* **styles/** – Stylesheets ( CSS ).

**Backend/**

* **models/**
* **routes**
* **db.js**
* **index.js**
* **auth.js**
* **notes.js**