The GitHub repository I’ve provided already contains **both the frontend and backend** of my project.  
I want you to **integrate them completely inside Replit** so that everything runs smoothly together.

Here’s exactly what needs to happen:

1. **Project Structure & Setup**
   * The repo already includes separate folders for frontend and backend.
   * Configure the project so that both can run together inside Replit.
   * When I press “Run” on Replit, the app should open in **one browser tab**, where both the frontend and backend work seamlessly.
   * Use a **proxy setup** or a **combined Express + static file serve** method so that the frontend is served by the backend in production.
2. **Backend (Node.js + Express + MongoDB)**
   * The backend already includes JWT- and bcrypt-based authentication for **signup** and **login**.
   * Ensure that the /signup and /login routes correctly hash passwords with bcrypt, generate JWT tokens, and return them to the frontend.
   * Enable **CORS** properly for local and deployed environments.
   * Make sure environment variables like JWT\_SECRET, MONGO\_URI, and AI\_API\_KEY are set using Replit’s Secrets tab.
3. **Frontend Integration**
   * Connect the frontend authentication UI (Signup/Login forms) to the backend routes using axios or fetch().
   * Store the JWT token securely in localStorage or cookies.
   * Protect private pages (like dashboard or AI section) by checking for valid JWT before rendering.
4. **AI Therapist Agent Connection**
   * There’s an AI section in the frontend where users can chat with an AI therapist.
   * The backend already has an endpoint (e.g., /api/ai) that takes a POST request with the user’s prompt.
   * Connect this endpoint to the frontend’s AI section so that:
     + When the user types a prompt and hits send, it triggers a POST request to /api/ai.
     + The backend processes the prompt using the existing AI logic and returns a JSON response (the AI’s reply).
     + The frontend should then dynamically display this reply in the chat interface.
5. **Run & Build Behavior**
   * In development, you can run the frontend and backend on separate ports using proxy.
   * In production (Replit deploy), serve the frontend build folder directly from the backend’s Express app.
   * Example: app.use(express.static(path.join(\_\_dirname, "frontend/dist")))
   * Ensure the Replit “Run” command starts the backend and serves the frontend automatically.

**Deliverables**

* ✅ Fully integrated code (frontend + backend) that runs on a single Replit instance.
* ✅ Authentication (signup/login) works using JWT and bcrypt.
* ✅ AI chat section connected and functioning (frontend ↔ backend).
* ✅ Environment variables properly configured.
* ✅ CORS and routing correctly handled.
* ✅ Clear comments explaining each integration step.

Please refactor or reorganize files if needed to make the integration clean and smooth.