# ACITY GO PROJECT (INTERFACE STAGE)

This is a **detailed step-by-step guide** covering everything from setting up the frontend and backend to connecting it with our **SQL database**. This will include:

1. **Tech Stack Selection** (Frontend, Backend, Database, Hosting)
2. **Setting Up the Development Environment**
3. **Database Connection & API Design**
4. **Building the Web Interface (Frontend)**
5. **Deploying & Testing the Web App**

I’ll also include **resources (courses, tutorials, documentation, and sample projects)**

**Step-by-Step Guide: Adding a Web Application Interface to ACITY-GO**

This guide will take us from **planning to deployment**, covering **frontend, backend, and database integration**.

**🛠 Step 1: Choose the Tech Stack**

**Frontend (User Interface)**

* **React.js** (Recommended) → Easy to use, fast, and scalable.
* **Tailwind CSS** (Optional) → For beautiful, responsive UI.
* **Axios** → To fetch data from the backend.

**Backend (APIs & Business Logic)**

* **Node.js (Express.js)** (Recommended) → Fast, scalable, and widely used.
* **Django/FastAPI** (Alternative) → If you prefer Python.

**Database (Data Storage)**

* **MySQL / PostgreSQL** (Since you already use SQL).
* **Prisma ORM / Sequelize (for Node.js)** → Helps interact with the database easily.

**Hosting & Deployment**

* **Frontend** → Vercel / Netlify
* **Backend** → Render / Railway
* **Database** → Supabase / PlanetScale / Self-hosted MySQL

📌 **Resources:**

* [React.js Docs](https://react.dev/)
* [Express.js Docs](https://expressjs.com/)
* [MySQL Docs](https://dev.mysql.com/doc/)
* [PostgreSQL Docs](https://www.postgresql.org/docs/)