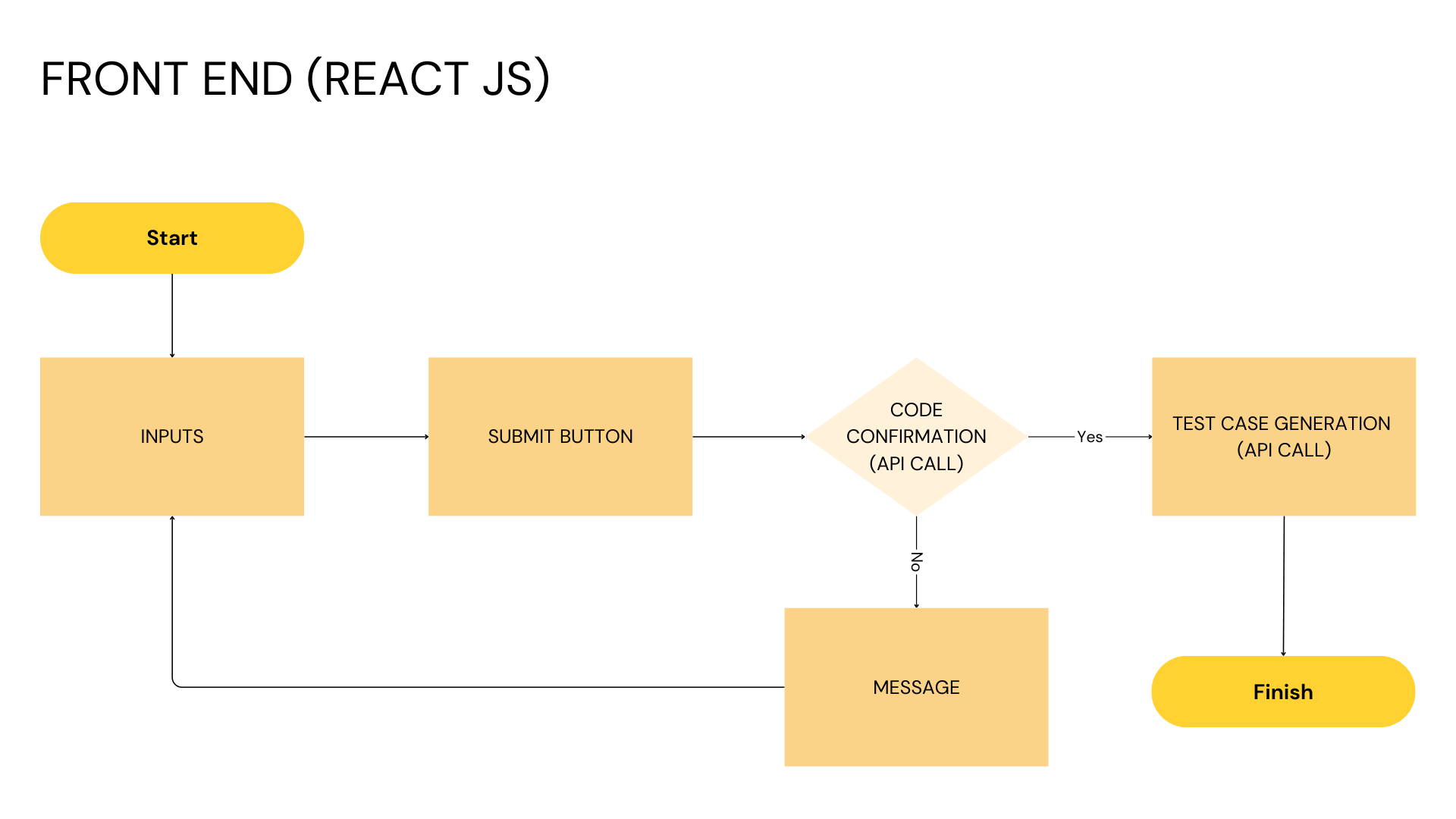
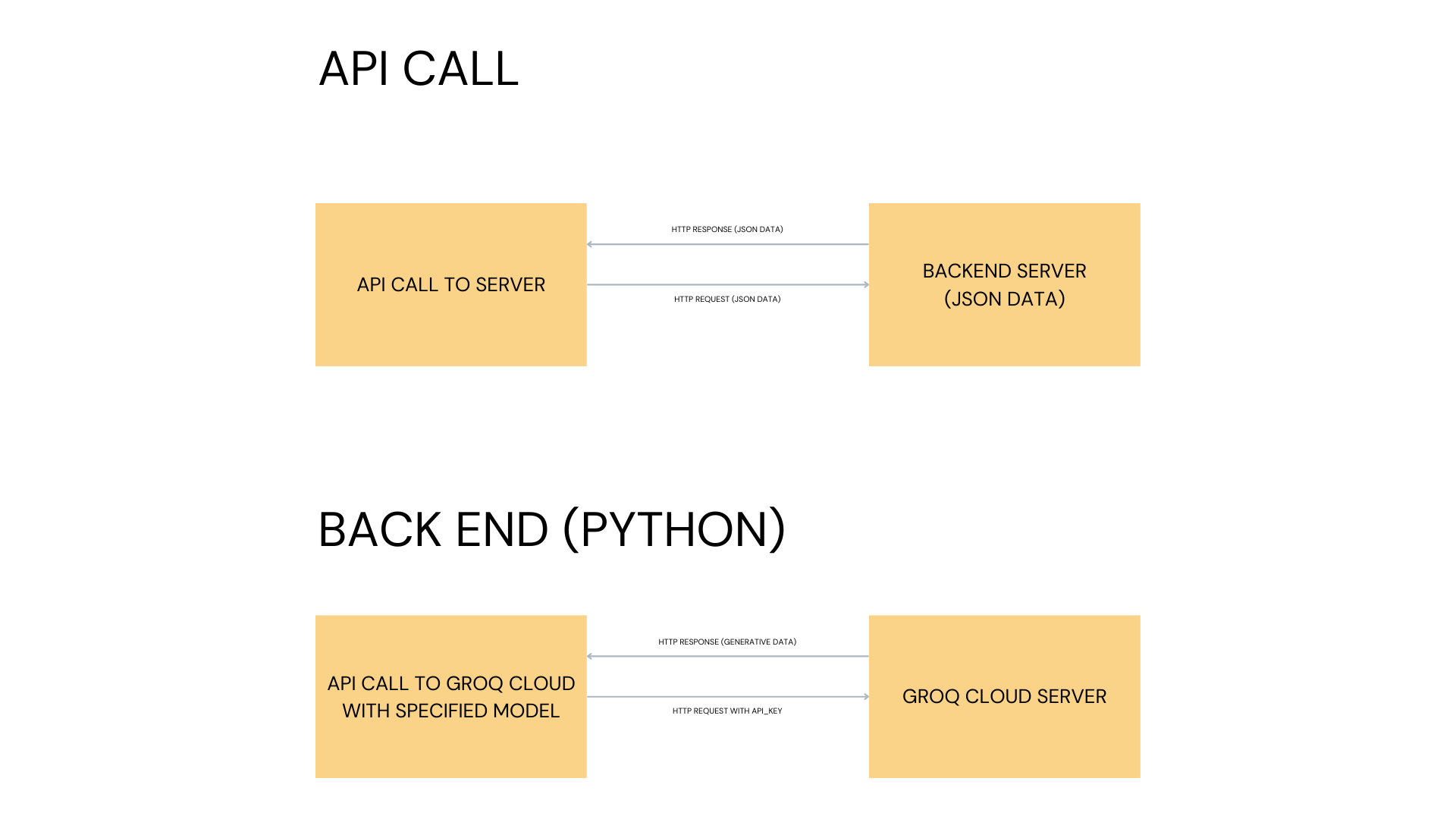
TEST CASE GENERATOR

DESIGN AND APPROACH





**APPROACH**

### Define the Requirements

* **Functionality**: Unit Test case generation for the code snippete
* **Input**: Code snippets, total number of test cases, positive and negative test cases, Scenario(Optional), Association text(Optional).
* **Output**: Unit Test Code

### Set Up the Backend (Python Flask)

1. **Create a Flask App**:
   * Set up a basic Flask app with endpoint to handle requests.
2. **Integrate GROQ CLOUD API**:
   * Install the library: pip install groq.
   * Use the API to generate test cases based on user input.
   * Handle authentication and API key securely.
3. **Define an Endpoint**:
   * Create a POST endpoint (/data) that accepts user input and returns generated test cases.

### Set Up the Frontend (React)

1. **Create a React App**:
   * Use create REACT VITE to bootstrap the frontend.
2. **Build the User Interface**:
   * Divide the page into 2 one for input and another for output
3. **Handle API Requests**:
   * Use fetch to send user input to the Flask backend.
   * One one request for confirming the code and another for test case generation in a prompt.
   * Display the response in the UI for the second request.