NAVYA NARAYAN PANICKER

WEEK 3 DAY 3: 9/10/25 [FRIDAY]

TASK : Add error handlers (404, 400); validate inputs (e.g. non-empty strings, valid JSON), Enable CORS; from React frontend make calls; test from Postman / browser.

QUESTIONS/REFLECTIONS:

1. How do you test error flows (client sends invalid data)?
   If the client sends a wrong data or causes error flows:
   - API should respond with proper error.
   - {"error": "Invalid input data"}
   - Or 404 error

2. Describe flow of exception in Flask: what happens if an unhandled exception occurs?
   Execption in flask:
   - **Error occurs** — something unexpected happens in your code.
     Example: database connection fails, or a key doesn't exist.
     Ex: user = data["name"] # KeyError if "name" missing
   - **Flask catches the exception** — Flask automatically detects the error during request handling.
   -
   - **If no custom handler exists**, Flask returns:
   - A **500 Internal Server Error**
   - A default HTML error page (in debug mode, it shows the full traceback). **If you define error handlers**, Flask uses them instead.
   - **Response sent to client** — The client (frontend or Postman) receives the JSON or HTML error message.

3. What is CORS? Why browsers block cross origin requests; how to configure CORS in Flask.
   - **CORS (Cross-Origin Resource Sharing)** lets browsers control requests from different domains.
   - **To use it in flask:**
     ```
     from flask_cors import CORS
     CORS(app)
     ```

4. What is the flow of a fetch/Axios request from React to Flask → response → error handling.
   - React sends request (fetch/Axios)
   - Flask API receives it, queries DB
   - Flask sends JSON response
   - React:
     - If success → display data
     - If error → show message like "Something went wrong"

5. How to log or debug failed requests.

    In flask:

        import logging

        logging.basicConfig(filename='app.log', level=logging.ERROR)

    in react:

        console.error(error);

        alert("Request failed");