**Python Flask Backend - Question 1:**

Create a simple Flask API with two endpoints. The first endpoint should accept a JSON object containing a list of numbers and return the sum of the numbers. The second endpoint should accept a JSON object containing two strings and return the concatenated result. Demonstrate error handling for invalid input.

**Python Flask Backend - Question 2:**

Implement a simple user authentication system using Flask. Create an endpoint for user registration that accepts a username and password, stores them in a dictionary (as a stand-in for a real database), and returns a success message. Then create an endpoint for user login that checks if the provided username and password match the stored values, and if so, returns an "access granted" message; otherwise, return an "access denied" message.

**ReactJS Front-end - Question 1:**

Create a simple React app with two input fields and a submit button. The first input field should accept a string and the second should accept a number. On clicking the submit button, display the inputted string repeated the number of times specified in the second input field, in a new paragraph below the form. Ensure proper validation and error handling for the inputs.

**ReactJS Front-end - Question 2:**

Implement a simple to-do list application in React. The application should allow users to add and delete items from the list. The list items should be stored in the component's state. Demonstrate the use of React hooks to manage state and handle user actions.