Plan for the software developer interview

The following tasks have been designed to evaluate the candidates' technical skills, learning ability, and adaptability. Candidates will be given one week to research, complete, and submit their solutions. Each task is tailored to test practical knowledge, problem-solving abilities, and the willingness to learn new tools or technologies.

Task 1: Database Design for Library Management System

• **Objective:** Design the database model for a library management application developed by 5IG Solutions for the Universidad de los Andes.
• **Requirements:**
o The database must support the following functionalities:

1. Display a complete list of books, including both available and currently loaned items.
2. Track which student has borrowed which book.
3. Store and manage loan information, including:

- Loan date
- Estimated return date
- Actual return date

4. Organize books into shelves, where each shelf must include the following characteristics:

- Location (e.g., Section A, B, C)
- Main topic (e.g., Economics, Science, Politics)
- Material the shelf is made of
- Total number of books it holds

o Document the entity-relationship (ER) model clearly, identifying key entities (e.g., Book, Student, Loan, Shelf), attributes, and relationships.
o Ensure the model supports future scalability (e.g., adding new categories or managing digital books).
o Optional: Implement the model using a database management system (e.g., MySQL, PostgreSQL, SQLite) and provide SQL scripts to create the schema.

Task 2: Working with REST APIs

- **Objective**: Build a Python script to interact with a REST API.
- **Requirements**:
    - Use a publicly available REST API (e.g., JSONPlaceholder or any similar free API).

- The script should:
    1. Send a GET request to retrieve data.
    2. Process and extract specific information from the response.
    3. Send a POST request to add new data.
- Include error handling for invalid requests (e.g., handling HTTP status codes).
- Provide a brief explanation of how the API works and describe your approach.

Task 3: Front-End Development Challenge

- **Objective**: Create a simple front-end web application to display data retrieved from an API.
- **Requirements**:
    - Use **HTML**, **CSS**, and **JavaScript** (you can use a framework like Vue.js or React if comfortable).
    - The application should:
        1. Fetch and display data from a REST API (can use the same API as Task 2).
        2. Allow users to search or filter the displayed data.
        3. Be visually appealing.
    - Share the source code and a hosted demo if possible (e.g., GitHub Pages).

Task 4: DevOps and CI/CD

- **Objective**: Simulate a CI/CD pipeline for deploying a simple web application.
- **Requirements**:
    - Use a platform like GitHub Actions or Jenkins.
    - The pipeline should:
        1. Automatically deploy the application (e.g., a basic HTML page) when a change is pushed to the repository.

2. Include steps like testing, building, and deploying the application.

- Document the setup process, including any scripts or configuration files used.

Submission Guidelines

1. Complete as many tasks as possible. Focus on quality over quantity.
2. Submit:
    - Source code/scripts.
    - Simple Documentation for each task, including explanations of your approach, challenges faced, and solutions.