**Total Life Take Home Project**

**Objective:** The purpose of this take-home test is to assess your technical skills in frontend and backend development. You will have the opportunity to demonstrate your coding abilities, problem-solving skills, and attention to detail. *We estimate that this task should take approximately 4-5 hours to complete. However, please don't be discouraged if you don't finish everything within this time frame. We're more interested in seeing your approach and understanding of the concepts rather than a fully completed task.*

**Submission:** Please submit your completed test as a link to a GitHub repository. Ensure that your submission includes all the necessary files to run your application, along with a README.md file containing setup instructions and any other relevant documentation.

**Evaluation Criteria:**

After submission, we will schedule a call for evaluation with a developer from Total Life. Please be ready to demonstrate your code during this call and discuss any questions.

1. **Code Quality:** We are looking for clean, readable, and well-organized code.
2. **Functionality:** Your application should meet all the specified requirements and work as expected.
3. **Problem-Solving:** Your approach to solving the given tasks will be evaluated.
4. **Communication:** You must be able to explain your codebase and your design choices.

**Notes:**

You are encouraged to use online resources and documentation, but the work you submit should be your own. We highly suggest that you try to answer any questions that you might have on your own. Trust your intuition and work within the requirements and details provided.

Good luck, and we look forward to seeing your submission!

**Project Part 1. Backend Development**

**Objective:** Create a simple REST API using a framework of your choice. We recommend Node.js/Express or a Python backend. The application you are building is an internal tool used to manage patients and their appointments.

**Requirements:**

1. Design a clinician, patient, and an appointments table.
   * Try to select a few fields that might be useful in the context.
   * Setup relevant relationships between the two resources.
2. Implement endpoints to create, read, update, and delete for all resources.
3. Use an SQLlite database to store the data.
4. Validate incoming request data.
5. Be sure to include an NPI number in the clinician table.
   * *An NPI number is a unique identification number for covered health care providers in the US.*
   * When a new clinician is added onto the system, we will want to validate the NPI number, and check the clinicians first and last name, as well as their state using the <https://npiregistry.cms.hhs.gov/api-page> API.

**Evaluation Criteria:**

1. Code quality and organization.
2. Correct implementation of CRUD operations.
3. Proper validation of request data.
4. Understanding of the code base

**Project Part 2. Frontend Development**

**Objective:** Create a simple frontend web app (ie. React app) that fetches and displays a list of appointments from the API in part 1.

**Requirements:**

1. Display appointments in a list, showing the patient’s name, appointment time, and appointment status.
2. Implement a time range filter.
3. Style the application using any minimal styling framework (Tailwind CSS etc.)
4. You must be able to demonstrate a working list by leveraging the backend in part 1a and populating it with data.

**Evaluation Criteria:**

1. Code quality and readability.
2. Correct use of state management.
3. Implementation of APIs for data fetching.
4. Ability to organize a basic UI to display the information.
5. Low priority: styling