Coding Test

Full Stack Engineer Role - Coding Test

Objective

Create a basic application with a Python backend and a React frontend. The focus is on functionality rather than aesthetics.

Prerequisites

Before diving into the tasks, make sure you have:

- Set up a local database (SQLite, PostgreSQL, etc.) for use.
- Familiarized yourself with Postman or similar tools for testing API calls.

Backend (Python)

Overview

In this section, you will set up a basic Python backend API, integrate it with the SellerCloud API, and establish a connection to a local database.

Task 1: Setup

1.1. Set up a basic Python backend API. You can use lightweight frameworks or libraries like Flask, FastAPI, Bottle, or Falcon.

```
- Note: You can name the endpoint whatever you like.
```

Task 2: SellerCloud API Integration

1.2. **Automated Authentication and Product Creation**: Your endpoint for creating a new product should first authenticate with the SellerCloud API and then use the obtained access token to create a product—all within the same API request.

```
- **Authentication Step**: The endpoint should make a POST request to `https://smartline-trial.api.sellercloud.us/rest/api/token` to obtain an access token. For this test, use the following hardcoded authentication details:
  - **Username**: logs@krameramerica.com
  - **Password**: KramerAmericaTest1

- [SellerCloud Authentication Documentation](https://developer.sellercloud.com/dev-article/authentication/)

- **Product Creation Step**: Upon successful authentication and retrieval of the access token, the endpoint should immediately proceed to create a new product in SellerCloud. Make a POST request to `https://smartline-trial.api.sellercloud.us/rest/api/Products` using the received access token as a Bearer token in the Authorization header.

- **Note on ProductSKU**: Ensure that the `ProductSKU` is unique for each product creation request. If the SKU is not unique, the SellerCloud API will return an error message.

- **JSON Payload**: Use the following format for the product creation step:
```json
{
    "CompanyId": 163,
    "ProductName": "product_name",
    "ProductSKU": "product_sku",
    "ProductTypeName": "product_type_name"
}
```
- Note: The `CompanyId` will always be `163`.

- **Error Handling**: Should any of the API calls fail or return an error, your API should handle this gracefully and return a generic error message.
```

Task 3: Database Connection

1.3. Establish a connection to a local database (e.g., SQLite, PostgreSQL).

1.4. Once you receive a 200 status code from SellerCloud for part creation, save the `ProductName`, `ProductSKU`, and `ProductTypeName` to the local database.

Notes

- Keep error handling simple. Encapsulate the entire endpoint with a generic error message for any failures.

Frontend (React)

Overview

In this section, you will create a single-page application using React to capture product details.

Task 4: UI Setup

2.1. Create a single screen or page.

2.2. Implement a form to capture the required information for creating a new product in SellerCloud (`ProductName`, `ProductSKU`, `ProductTypeName`).

Functionality

Overview

Here, you will bring the backend and frontend together. The frontend will send data to the backend, which will then interact with the SellerCloud API and update the local database.

Task 5: Form Submission

3.1. On form submission, send the captured data to the backend.

3.2. Backend should interact with the SellerCloud API and save the returned information to the local database.

Sample Product Data for Testing

For your convenience, here's a list of sample product data you can use while testing your implementation:

1. **ProductName**: "ElectroGadget 2000"
   - **ProductSKU**: "EG2000"
   - **ProductTypeName**: "Electronics"
2. **ProductName**: "UltraWrench Pro"
   - **ProductSKU**: "UWPRO"
   - **ProductTypeName**: "Tools"
3. **ProductName**: "AquaPure Filter"
   - **ProductSKU**: "APF100"
   - **ProductTypeName**: "Home Appliances"
4. **ProductName**: "SportsMax Shoes"
   - **ProductSKU**: "SMS100"
   - **ProductTypeName**: "Footwear"
5. **ProductName**: "PixelBook Laptop"
   - **ProductSKU**: "PBL300"
   - **ProductTypeName**: "Computers"

Feel free to test with any other data this is a sandbox account.

Validation

To confirm that the product has been added successfully to SellerCloud:

1. Visit https://smartline-trial.delta.sellercloud.us/
2. Type the product name you just created into the search bar and verify its presence.

Additional Instructions

- Keep the backend lightweight; we're focusing on functionality.
- Once completed, add the project(s) to a Git repository and send us the link.
- Don't forget to handle CORS if you have if your frontend and backend are running on different origins.

Should you encounter any challenges or have questions as you work through this test, please don't hesitate to reach out to me at dguardado@krameramerica.com. I'm here to assist.

For those who may not have extensive experience with Python, there's a wealth of resources and tutorials available online that can guide you. Ultimately, the objective of this test is not just to gauge your current expertise in the mentioned areas, but also to understand how you approach and learn new concepts when faced with unfamiliar challenges. Your resourcefulness and adaptability are equally valuable to us.

Best of luck!