RapidQuest

Full Stack Developer Evaluation Task

Goal

Your mission, should you choose to accept it, will be to build a data visualization web application that can analyze e-commerce data from a sample Shopify store stored in MongoDB. You will build an API layer that reads the data from the database and performs the necessary queries to manipulate data and serve it to the front end through a REST API. The front end should connect to your API and visualize the data using Chart.js or a similar JavaScript visualization library.

Task 1: API Development

We have prepared a sample dataset in a MongoDB cluster for this evaluation task. Your task is to build JSON REST APIs that connect to this MongoDB and serves the data in format that your frontend will require. You may use Python (with Web2py, Flask or Django) or Node.js (with Express).

Connect to MongoDB Database with the following Connection String:
`mongodb+srv://db_user_read:LdmrVA5EDEv4z3Wr@cluster0.n10ox.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0`

MongoDB Collections: You will find the following collections in the RQ_Analytics database:

- shopifyCustomers
- shopifyProducts
- shopifyOrders

You are required to prepare and visualize data for the following charts:

1. **Total Sales Over Time:** Use shopifyOrders.total_price_set, grouped/aggregated by daily, monthly, quarterly, and yearly intervals.
2. **Sales Growth Rate Over Time**
3. **New Customers Added Over Time:** Track the addition of new customers based on the created_at field in the shopifyCustomers collection.
4. **Number of Repeat Customers:** Identify customers with more than one purchase across different time frames: daily, monthly, quarterly, and yearly.
5. **Geographical Distribution of Customers:** Utilize customers.default_address.city to visualize the distribution on a map.

6. **Customer Lifetime Value by Cohorts:** Group customers based on the month of their first purchase and visualize the lifetime value for each cohort.

Task 2: Frontend Development

Visualize the data from your APIs and plot the chart types mentioned previously.

Use any modern JavaScript framework (e.g., React, Angular, Vue.js) and a charting library like Chart.js, Highcharts, or similar. Do not use platforms like PowerBI or Tableau.

Task 3: Hosting & Deployment

Use any deployment / hosting service of your choice to publish your app and get a publicly accessible URL.

Deliverable:

- Publicly accessible URL of your working webapp. Submissions without a working URL of the app will not be evaluated.
- Public Github Repository of your application.

Evaluation criteria:

- Accuracy and Completeness of Charts
- Simplicity, Clarity of Visualizations and Ease of Use

How to Submit?

Submit ONLY using this form:
https://makelist.io/form.html?listid=e168ca8c-a195-4d12-a4b0-2942ef7302ca&orgid=440f21d1-ed57-4355-9082-552bfc409eb2