CommentSold Code Test - Full Stack

CommentSold Code Test - Full Stack

This document outlines a code evaluation which will test your ability to put together a simple full-stack web application. There are a few core components that you will be expected to implement, and some room for your own approach to tackling the problem.

Data

`https://s3.amazonaws.com/commentsold-share/data.zip`

Requirements

You have been given a data dump of fake ecommerce data for users, inventory, products and orders (data format will be detailed below) from a hypothetical ecommerce system. Your task is to take that data and to build a web application that will provide a set of basic features for interacting with the data.

You may use any web framework or libraries that you wish. The deliverable is the code of your application which can be delivered directly or made available via something like GitHub.

If you have time to make the application available from web host feel free to do so but it's not a strict requirement. Provide any notes for getting the app up and running that may be required.

There are a number of features outlined below—the goal of this exercise is to demonstrate your capabilities with web development. Please do not feel obligated to implement all features listed.

Features:

PLEASE NOTE: You should plan to implement some version of **B** along with one of **D** or **E** as a minimum feature set.

Ideally this should be an exercise that takes in the 3-4 hour range.

> 💡 **Frontend candidates:** Consider B (authentication) an optional task, although it would be a nice-to-have if time permits. We would like to see a reasonable amount of styling, although functionality takes precedence. Using a component library or other styling framework is fine. Finally, it's encouraged to generate a good portion of the HTML client-side to illustrate comfort with modern frontend development (a SPA, for instance).

A. Setup

As a starting point you should import the data into some format that allows you to interact with it from a web framework you may use any storage system you see fit (e. g. database, etc.).

B. Authentication

- There are 50 user accounts and it should be possible to login with any one of them using the email and plain text password
- In order to allow for reduced scoping you can alternatively implement a basic login that does not verify password but instead simply creates a session based on the entered email address.

C. Products

1. Provide a list of products tied to the user account
    - name, style, brand, (optionally) available sku's
- *(optionally)* Allow the user to create a new product, edit an existing product or delete a product

---

Then one of the following areas of functionality:

D. Inventory Display and interaction

1. List all inventory records for the authenticated user and allow for navigation within the set

    - Display the Product Name, sku, quantity, color, size, price and cost
- Show the total count of inventory items in the system for the user
- Allow the user to filter the list for a specific product id or sku

Optionally:

- Allow the user to filter based on items with inventory below a threshold

OR

E. Order Display and Interaction

1. List all orders in the system for products on the logged in user account and allow for navigation within the set
    - Display the Order - customer name, email address, product name, color, size, order_status, order total, transaction id, shipper (if applicable), tracking number (if applicable)
- Show a total of sales for all orders
- Show the average sale total across all orders

Optionally:

- Allow the user to filter orders based on product or SKU
    - Optionally show the filtered totals for the filtered order set
- Show a breakdown of the number of orders in different states based on order_status

---

Guidelines:

- You may translate the CSV data into other formats if you wish, you are also free to modify the schema as long as it doesn't alter the nature of the task at hand.
- You are free to use whatever libraries or frameworks you find useful.
- It's OK if you do not have time to build all features, but be prepared to walk us through your thoughts and strategy.
- Performance issues should be a concern to you

- You own whatever code you write, and are free to do whatever you want with it.

Data Format:

Data is available from the following link in the form of a zip file containing csv files with the raw data. Each csv file includes a header row before the data rows matching the following data definition.

Data available: `https://s3.amazonaws.com/commentsold-share/data.zip`

Users:

The user data includes both password plaintext and password hash so that you can verify that your password hashing works or feel free to generate new hashes for the passwords. The hashes present were generated using the golang bcrypt library an example of plaintext/hash:

Password: `testing`

Hash: `$2a$10$.IM7LoxN3zNdzlicXhCpkuct2S2xuuMhuKtWRk0Wgr1zelSG1F/G6`

```
Users

Columns:
    id int
    name string
    email string
    password_hash string
    password_plain string
    superadmin boolean
    shop_name string
    remember_token varchar(100)
    created_at timestamp
    updated_at timestamp
    card_brand string
    card_last_four string
    trial_ends_at timestamp
    shop_domain string
    is_enabled boolean
    billing_plan string
    trial_starts_at timestamp
```

Products

```
Products

Columns:
    id int
    product_name string
    description text
    style text
    brand text
    created_at timestamp
    updated_at timestamp
    url string
    product_type string
    shipping_price int
    note text
    admin_id int
```

Inventory

```
Inventory

Columns:
    id int
    product_id int
    quantity int
    color text
    size text
    weight double
    price_cents int
    sale_price_cents int
    cost_cents int
    sku string
    length double
    width double
    height double
    note text
```

Orders

Orders represent an order in the system linked to a product along with

```
Orders

Columns:
    id int
    product_id int
    street_address text
    apartment text
```

```
    city text
    state text
    country_code string
    zip text
    phone_number string
    email text
    name string
    order_status string
    payment_ref text
    transaction_id string
    payment_amt_cents int
    ship_charged_cents int
    ship_cost_cents int
    subtotal_cents int
    total_cents int
    shipper_name text
    payment_date timestamp
    shipped_date timestamp
    tracking_number text
    tax_total_cents int
    created_at timestamp
    updated_at timestamp
```