homework_instructions.md 7.79 KB

Node.js & MySQL

Overview

In this activity, you'll be creating an Amazon-like storefront with the MySQL skills you learned this week. The app will take in orders from customers and deplete stock from the store's inventory. As a bonus task, you can program your app to track product sales across your store's departments and then provide a summary of the highest-grossing departments in the store.

Make sure you save and require the MySQL and Inquirer npm packages in your homework files--your app will need them for data input and storage.

Submission Guide

Make sure you use the normal GitHub. Because this is a CLI App, there will be no need to deploy it to Heroku. This time, though, you need to include screenshots, a gif, and/or a video showing us that you got the app working with no bugs. You can include these screenshots or a link to a video in a `README.md` file.

- Include screenshots (or a video) of typical user flows through your application (for the customer and if relevant the manager/supervisor). This includes views of the prompts and the responses after their selection (for the different selection options).

- Include any other screenshots you deem necessary to help someone who has never been introduced to your application understand the purpose and function of it. This is how you will communicate to potential employers/other developers in the future what you built and why, and to show how it works.

- Because screenshots (and well-written READMEs) are extremely important in the context of GitHub, this will be part of the grading.

If you haven't written a markdown file yet, click here for a rundown (https://guides.github.com/features/mastering-markdown/), or just take a look at the raw file of these instructions.

Submission on BCS

- Please submit the link to the Github Repository!

Instructions

Challenge #1: Customer View (Minimum Requirement)

1. Create a MySQL Database called `bamazon`.

2. Then create a Table inside of that database called `products`.

3. The products table should have each of the following columns:

- item_id (unique id for each product)

- product_name (Name of product)

- department_name

- price (cost to customer)

- stock_quantity (how much of the product is available in stores)

1. Populate this database with around 10 different products. (i.e. Insert "mock" data rows into this database and table).

2. Then create a Node application called `bamazonCustomer.js`. Running this application will first display all of the items available for sale. Include the ids, names, and prices of products for sale.

3. The app should then prompt users with two messages.

- The first should ask them the ID of the product they would like to buy.
- The second message should ask how many units of the product they would like to buy.

1. Once the customer has placed the order, your application should check if your store has enough of the product to meet the customer's request.

- If not, the app should log a phrase like `Insufficient quantity!`, and then prevent the order from going through.

1. However, if your store *does* have enough of the product, you should fulfill the customer's order.
   - This means updating the SQL database to reflect the remaining quantity.
   - Once the update goes through, show the customer the total cost of their purchase.

- If this activity took you between 8-10 hours, then you've put enough time into this assignment. Feel free to stop here -- unless you want to take on the next challenge.

Challenge #2: Manager View (Next Level)

- Create a new Node application called `bamazonManager.js`. Running this application will:

- List a set of menu options:
- View Products for Sale
- View Low Inventory
- Add to Inventory
- Add New Product
- If a manager selects `View Products for Sale`, the app should list every available item: the item IDs, names, prices, and quantities.
- If a manager selects `View Low Inventory`, then it should list all items with an inventory count lower than five.
- If a manager selects `Add to Inventory`, your app should display a prompt that will let the manager "add more" of any item currently in the store.
- If a manager selects `Add New Product`, it should allow the manager to add a completely new product to the store.

- If you finished Challenge #2 and put in all the hours you were willing to spend on this activity, then rest easy! Otherwise continue to the next and final challenge.

Challenge #3: Supervisor View (Final Level)

1. Create a new MySQL table called `departments`. Your table should include the following columns:

- department_id

- department_name

- over_head_costs (A dummy number you set for each department)

1. Modify the products table so that there's a product_sales column, and modify your `bamazonCustomer.js` app so that when a customer purchases anything from the store, the price of the product multiplied by the quantity purchased is added to the product's product_sales column.

- Make sure your app still updates the inventory listed in the `products` column.

1. Create another Node app called `bamazonSupervisor.js`. Running this application will list a set of menu options:

- View Product Sales by Department

- Create New Department

1. When a supervisor selects `View Product Sales by Department`, the app should display a summarized table in their terminal/bash window. Use the table below as a guide.

department_id	department_name	over_head_costs	product_sales	total_profit

department_id	department_name	over_head_costs	product_sales	total_profit
01	Electronics	10000	20000	10000
02	Clothing	60000	100000	40000

1. The `total_profit` column should be calculated on the fly using the difference between `over_head_costs` and `product_sales`. `total_profit` should not be stored in any database. You should use a custom alias.
2. If you can't get the table to display properly after a few hours, then feel free to go back and just add `total_profit` to the `departments` table.

- Hint: You may need to look into aliases in MySQL.
- Hint: You may need to look into GROUP BYs.
- Hint: You may need to look into JOINS.
- **HINT**: There may be an NPM package that can log the table to the console. What's is it? Good question :)

Reminder: Submission on BCS

- Please submit the link to the Github Repository!

Minimum Requirements

Attempt to complete homework assignment as described in instructions. If unable to complete certain portions, please pseudocode these portions to describe what remains to be completed. Adding a README.md as well as adding this homework to your portfolio are required as well and more information can be found below.

Create a README.md

Add a `README.md` to your repository describing the project. Here are some resources for creating your `README.md`. Here are some resources to help you along the way:

- About READMEs (https://help.github.com/articles/about-readmes/)
- Mastering Markdown (https://guides.github.com/features/mastering-markdown/)

Add To Your Portfolio

After completing the homework please add the piece to your portfolio. Make sure to add a link to your updated portfolio in the comments section of your homework so the TAs can easily ensure you completed this step when they are grading the assignment. To receive an 'A' on any assignment, you must link to it from your portfolio.

One More Thing

If you have any questions about this project or the material we have covered, please post them in the community channels in slack so that your fellow developers can help you! If you're still having trouble, you can come to office hours for assistance from your instructor and TAs.

Good Luck!