TASK

Capstone Project III — Databases

Visit our website

Introduction

WELCOME TO THE THIRD CAPSTONE PROJECT!

In this task, you will be modifying your previous Capstone Project so that the program now uses a database, instead of text files, to store the data needed for the program. You will be tasked with designing and implementing a database. You will also write the code needed to interact with the database.

Get in touch

Connect for support

Remember that with our courses, you're not alone! You can contact your mentor to get support on any aspect of your course.

The best way to get help is to login to **www.hyperiondev.com/portal** to start a chat with your mentor. You can also schedule a call or get support via email.

Your mentor is happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!

DEVELOPER PORTFOLIO

By the end of this project, you will be adding this program to your developer portfolio. Today most applications rely heavily on data! A prospective employer will want to see that you can persist and manipulate data. This project should make it clear to a potential employer that you can follow a development process to create a data-driven program that is debugged, tested, refactored and documented and that meets a client's specifications! Put extra time and effort into this project to make sure that it really showcases the skills you have acquired!

THE TASK AT HAND

You are asked to create a food delivery system for a company called "Food Quick". Food Quick is the company that receives the orders and distributes them to a driver based on their current load and their location. They want you to create a program that can help them keep track of the orders and distribute accordingly.

Food Quick stores the following information for each customer:

- Order number
- Customer name
- Contact number of the customer
- Address of the customer
- Location (city) of the customer
- Email address of the customer
- Name of the restaurant
- Location of the restaurant
- Contact number of the restaurant
- How many of each meal is being ordered
- The list of meals being ordered and their prices
- Any special preparation instructions given by the customer
- The total amount to be paid

Food Quick would like you to be able to create an invoice for a customer after the above information has been inputted into the program. The invoice should be a text file with the following format:

```
Order number 1234
Customer: Jill Jack
Email: jilljack@yahoo.com
Phone number: 123 456 7890
Location: Cape Town

You have ordered the following from Aesop’s Pizza in Cape Town:
```

```
1 x Pepperoni pizza (R78.00)
2 x Hawaiian pizza (R82.00)

Special instructions: Extra tomato base on the Pepperoni pizza

Total: R242.00

John Krill is nearest to the restaurant and so he will be delivering your
order to you at:

12 Cherry Road
Plumstead

If you need to contact the restaurant, their number is 098 765 4321.
```

Before you begin

A key focus of this project will be ensuring that your code is correct, well-formatted and readable. In this regard, make sure that you do the following before submitting your work:

1. Make sure that you have identified and removed all syntax, runtime and logical errors from your code.

2. Make sure that your code is readable. To ensure this, add comments to your code, use descriptive variable names and make good use of whitespace and indentation. See **this style guide** to see how classes and methods should be named and how your program should be formatted.

3. Make sure that your code is as efficient as possible. How you choose to write code to create the solution to the specified problem is up to you. However, make sure that you write your code as efficiently as possible.

4. Make sure that all output that your program provides to the user is easy to read and understand. Labelling all data that you output (whether in text files or to the screen) is essential to make the data your program produces more user-friendly.

Compulsory Task 1

Follow these steps:

- Design and create a database called QuickFoodMS. Assume that each customer can only be assigned to one driver. Each customer will also only be ordering from one restaurant.

  Submit the following:

  - Dependency diagrams for each table in the database.
  - An Entity Relationship Diagram that shows the relationships between the tables in your database.
  - Screenshots of your console that show how each table was created.

- Add at least two rows of data to each table in the database. Submit screenshots of your console that show how data is added to the tables.

Compulsory Task 2

Follow these steps:

- Copy and paste the code that you wrote for the last Capstone Project into the Dropbox folder for this Capstone Project.

- Modify your code so that it:
  - Reads and writes data about restaurants and customers associated with drivers from your database instead of text files. Your program should not use any text files.
  - Capture information about new customers and add these to the database.
  - Update information about existing customers (e.g. if they change their address).
  - Finalise existing orders. When an order is finalised the following should happen:
    - An invoice should be generated for the customer. This invoice should contain the customer's contact details and the total amount that the customer must pay.
    - The project should be marked as "finalised" and the completion date should be added.
  - Find all orders that have missing information and still need to be completed from the database.

    - Find and select an entry by entering either the order number or customer name.

- Besides meeting the above criteria, you should also do the following:
    - Include exception handling. Use try-catch blocks wherever appropriate.
    - Remove all errors from your code. Take extra care to detect and remove all logical and runtime errors.
    - Adequately refactor your code.
    - Document your code. Adhere to the style guide found **here**.
    - Use Javadoc to generate API documentation from documentation comments for your program.
    - Follow the guidelines **here** to create a Readme file for this project.

- After receiving feedback from your mentor and improving your code based on this feedback, add your program to Github.

Rate us

Share your thoughts

HyperionDev strives to provide internationally-excellent course content that helps you achieve your learning outcomes.

Think that the content of this task, or this course as a whole, can be improved or think we’ve done a good job?

Click here to share your thoughts anonymously.