Project – QA Café

This exercise is designed to check the knowledge and skills accumulated throughout this module. It will involve many of the processes and techniques you've covered and is an open brief rather than a lab with steps.

The exercise has a set of requirements and starting brief that should be met, and this exercise should be done over the course of a day.

Brief

You are a Software Engineer for a popular high street coffee chain 'QA Café', who are looking for an updated order management system. They require a method of storing and updating orders that are made in the café, and require a user interface to interact with the store of orders.

In order for the order management system to be effective, they need to do the following:

- Enter an order
- Read an order by ID
- Read all orders
- Update an order by ID
- Delete an order by ID
- Delete all orders

The owners of QA Café are requesting the data for orders be stored within some form of SQL database and are open to more functionalities being added.

Requirements

This project is recommended to be completed in one day.

- Layer 1 – Complete the project with a combination of Python and SQLite
- Layer 2 – Complete the project with a combination of Java and JDBC (you can skip Layer 1 and start here if you'd prefer)
- Layer 3 – Test the project via JUnit

You must consider the following basic requirements when designing your app:

- App with Terminal-based UI
- SQL Database, either locally hosted or cloud hosted
- SQL Table within your database that contains order_id, customer_name, drink, size, extras, price
- Python/Java code considering OOP when designing and using functions where appropriate
- Test suite for unit tests of the basic functionality
- Basic Documentation about the project and how to use in form of README.md

These are basic MVP requirements of this app; you are free to add extra functionality in whatever way you deem appropriate once the requirements have been met. Extra functionality could include:

- Graphical User Interface
- More complex table and/or relational tables
- More functionality to interact with the database
- Mock and integration testing
- Hosting the app on a cloud server for access
- More thorough Documentation utilising markdown language

Deliverable

You are expected to host your solution on a public GitHub and send it to your trainer. Your code must be merged to the main branch, and you should be using a proper feature-branch model (with the use of Dev and Feature branches) throughout.

You should ensure your link to your repo is submitted to the relevant activity within Bud by the deadline. You should make sure your repo is public, so anyone who accesses the link via Bud can see through the work you have done. You should expect feedback submitted via the same message board on Bud.