CORK
UNIVERSITY
BUSINESS
SCHOOL

IS6217 CA 2 – In Class Brief

Individual Coding Project: 60 Marks

Lecturer: Mr Richard Harris

Project Context

Share Management Application

Building on the work your learnings in Semester 1 you are going to create an object oriented, database-driven, Python GUI share management application for the CUBS Investment Bank Ltd. The application must interact with a Database (e.g. CUBS_Investment.sqlite) that contains information about customer and account details. Sample Data is presented below. You are not obliged to use this data structure, but the database should support the functionality of your application.

Sample Data is shown Below:

Field	Data Type	Sample Data
Customer_ID	Autonumber	1
Cust_Name	Text	Joe Bloggs
Cust_Address	Text	Text
Cust_Telephone	Text	(021)12345678

Field	Data Type	Sample Data
Account_Number	Autonumber	1001
Cash_Balance	Numeric	1000€
Customer_ID	Numeric	1
PIN	Numeric	1234
Share_Balance	Numeric	100

You are required to create a share management application that will allow the customer to interact with and update the database accordingly. For the purposes of meeting the Minimum Requirements noted below you will need to populate the database with relevant customer and account records. Advanced requirements for this application require you to create administration functionality within your application. For this exercise, you may use a share value of 1 euro per share, or any other value you see fit.

Project Minimum Requirements (70%)

Using object-oriented programming, create a Python GUI application, leveraging three-tier programming techniques, that satisfies the requirements highlighted in the project context.

You should use the skills you have learned in this module to review the context and identify all required functionality, class attributes, and class methods to satisfy the client's request.

At Minimum, your application should include:

- Pin Access.
- A customer should be able to withdraw money if sufficient funds available (updating balance in database as appropriate).
- A customer can deposit funds (updating balance in database).
- A customer can check their balance, both share and cash (retrieve balance from database).
- A customer can purchase shares if they have sufficient cash balance.
- A customer can sell shares and have proceeds sent to their Cash Balance.
- Code should be commented as required.

Advanced Requirements (15%)

- Integrated administration functionality allowing for CRUD operations on all customer records.

- Ability to read customer data from a CSV file and upload data from said file into the relevant database for use within the application.

  Note: Students are expected to use their own judgment and creativity when implementing advanced requirements. Special attention should be paid to data governance, user access roles and database structure. Students are encouraged to attempt all requirements. Use comments to highlight any work that, while not functional, shows an attempt made to implement a specific requirement, highlight any commented functionality in accompanying documentation. **It is expected that your program will execute and run with no syntax errors. Any code causing errors should be commented out and noted in documentation.**

Presentation Video (15%):

- You are required to record a self-reflection video presentation, highlighting how you have archived the requirements noted above. Ensure that your video includes references to your code, highlighting where you feel you have done well, and area's you need improvement. The format of the video is your choice, but please note the below:

  - Maximum video time: 5 minutes

  - Format: MP4
  - Ensure video can be run on "VLC Player"

Marking

The code elements will be assessed based on the criteria below. Please see the rubric below for more details.

- Coding Standard and logic (60%)
- Sophistication, Integration and Completeness (20%)
- Usability (20%)

Submission

- Project to be submitted via Canvas
  - Include a "readme" Microsoft word file explaining your work, highlighting how you have achieved the requirements, and any other relevant details.
- Zip the entire solution folder and upload
- Late submissions are subject to penalty as per the book of modules.
- Code created/generated using AI applications (such as ChatGPT) must **NOT** be included in your final submission, with the exception of your UI Module, ensuring you include appropriate documentation in your "ReadMe" if AI is used.
- All code will be analysed for plagiarism.
  - All students are required to read and understand the UCC plagiarism policy, found here: https://libguides.ucc.ie/academicintegrity/plagiarism. Presenting a project for submission and grading constitutes acceptance of and compliance with the policy.

Marking Rubric

Criteria	1H	2H1	2H2	Pass	Fail
Coding standard and logic	Code is object oriented, commented. Appropriate data types are used. Logic as appropriate.	Efforts at object orientation with some minor implementation errors. Code commented infrequently. Repetition of code. Little effort to customise code used in class.	Inefficient coding standard and logic in places. Repetition of code. Basic level of commenting.	Data input and calculations are not facilitated via methods and property procedures. No commenting	Coding standard and logic inadequate. Basic logic errors.
Sophistication and Completeness	All minimum requirements are met and exceeded with novel functionality	All minimum requirements met with some additional functionality.	Minimum requirements are met.	Some gaps in requirements or functionality	Large gaps in requirements and functionality
User Interface and Usability	Application is easy to navigate and functionality is easily accessible.	Application facilitates core functionality in an efficient manner.	Some difficulty navigating between functionality	Inefficient use of controls.	Poor usability and navigation