**COS2614 2024 Assignment 1**

**Question 1**

Develop a Qt Graphical User Interface (GUI) application that calculates a unique ID and a secure initial key for a new user based on the user's full name. The application should adhere to the following specifications for generating the unique ID and the initial key:

**Unique ID Generation Rules:**

- The unique ID should be 6 characters long.

- The first two characters are the uppercase initials of the first and last names. If there is a middle name, use the initial of the middle name as the second character instead of the last name.
- The remaining four characters are numeric, representing the count of letters in the full name (excluding spaces). If this count is less than 4 digits, pad the number with leading zeros to make it 4 digits.

**Initial Key Generation Rules:**

- The initial key should consist of 6 characters randomly selected from the user's full name.
- The selection should ensure that at least one character is a vowel (a, e, i, o, u), and at least one character is a consonant.
- No spaces are allowed in the initial key, and all characters should be in lowercase.

**User Interaction:**

- Use a QInputDialog to prompt the user to enter their full name. Assume the full name is provided as a single string, with each name part separated by spaces (e.g., "John Michael Doe").
- Display the generated unique ID and initial key using a QMessageBox.

**Assignment Objectives:**

This assignment aims to assess your ability to:

- Utilize Qt widgets like QInputDialog and QMessageBox for basic user input and output.
- Apply string manipulation techniques to parse and process user input.
- Implement logic to generate strings based on specific criteria, including the use of random selection for character generation.

**Submission Requirements:**

Your source code for the Qt application with screenshots of test runs of your program.

**Evaluation Criteria:**

- Correct implementation of the unique ID and initial key generation rules.
- Effective use of Qt for GUI components and user interaction.
- Code readability
- Ensure your application is user-friendly and robust against unexpected inputs

**Question 2**

Design and implement a simple console application in C++ that models an online store inventory system, based on the requirements and class relationships described below:

**Class Descriptions:**

**Item**: Represents a generic item sold in the online store. Each item is characterized by a unique identifier (ID), a name, and a price. The Item class should provide methods to get and set these properties. Additionally, it should include a method to print item details.

**Vendor**: Represents a vendor supplying items to the store. A vendor is described by a vendor ID, a name, and an address. This class should provide methods to get and set these properties. The **toString()** method should return a string representation of the vendor, including all its details.

**StoreItem**: Extends the Item class. In addition to the properties inherited from Item, StoreItem includes a property to store the vendor (supplier) of the item. The class should override the method to print item details to include vendor information if available. It should also include a method **getVendorName()** that returns the name of the vendor if set, or "Unknown" if not.

**Implementation Details**:

- The **StoreItem** class should be initialized with an ID, name, and price. Vendor details should be set using a separate method, **setVendor()**, which takes a Vendor object as a parameter.
- The **toString()** method in both Vendor and StoreItem should return a string representation of the object's state in a readable format. For StoreItem, implement two versions of **toString()**: one that only returns item details and another that also includes vendor details based on a boolean parameter.

**Application Flow**:

- Your console application should prompt the user to enter details for a list of items and their corresponding vendors.

- After entering the details, the application should display the information of each StoreItem using the **toString()** method, first without vendor details and then with vendor details.
- It should also demonstrate the use of **getVendorName()** for each StoreItem.

**Assignment Objectives**:

- Demonstrate understanding of class inheritance and polymorphism in C++.
- Implement class methods, including constructors, accessors, mutators, and methods for string representation of objects.
- Handle user input in a console application and display output based on the user input.

**Submission Requirements**:

Source code files for your Qt application and screenshots of test runs of your program.

**Evaluation Criteria**:

- Correct implementation of class hierarchy as described.
- Proper use of inheritance and method overriding.
- Code readability and adherence to C++ best practices.
- Ability to process and respond to user input in the console application.