ASU W. P. Carey School of Business
Arizona State University

CIS 345 – Business Information Systems Development II

Project: KitchenSupply Order Management System

Final: See Blackboard for Due Date

KitchenSupply is commissioning a prototype of an order management system from a group of talented CIS programmers studying object-oriented design and programming! They sell products to customers and want an application to process orders, manage inventory, and customers. A prototype would give KitchenSupply the ability to see both possibilities and shortcomings, refine their own requirements and expectations, and then go to a software development consultant. In their basic prototype, KitchenSupply would like a system, which allows the management of customers, products, and order logging. This includes the ability to add, edit, delete products to/from an order. Upon order submission, a receipt will be logged and inventory will be updated deducting the products sold from inventory and a customer's balance will be updated subtracting total order cost.

The company wants to maintain a list of Products, which they refer to as their available Inventory. Each Product should have an ID, Description, Quantity on hand, and price. Some products are attachments for other products. It is important that Attachment Products include the product ID of the product that they attach to and the material type for the attachment. This additional data is in combination of the standard Product data. All product information needs to be stored in a file to ensure inventory tracking is persistent, meaning if the application is closed and re-opened the Product inventory tracking will not reset.

KitchenSupply would like the prototype to track customer data. Customer data includes an Account number, Name, PIN, and Balance. A customer cannot ever have a negative balance which means an order will not be placed when a customer's balance is less than the total bill of the order. Customer data must also be persistent and stored in a file.

The application should have a Graphical User Interface (GUI), which enables the customer user to select their account from a list and see a list of products. Upon selecting a Customer Account, the application shall show all customer data on the form such as ID, Name, Balance, etc. but not the PIN as that is private. Customers must be able to select products and view the product details. Customers shall add products to an order list and be able to enter the quantity of each product they wish to add to the order. When the customer has added all the desired products and quantities of products, to their order list, they shall start the purchase process. The GUI must have the ability to make a purchase which will compute a total order amount and charge the amount to the customer account upon the customer entering their PIN. Deny any purchase if the customer account has insufficient funds or an incorrect PIN.

This same GUI must provide Employee users the ability to edit product or inventory. An employee account will exist in the customer file with the same object attributes. One difference is the balance will be zero because this account will not be making purchases. The employee account should activate inventory-edit-mode. In this mode employee users shall use the same GUI to select products and edit the quantities of any product. Same as customers, they will select a product and quantity and add to a order list. When done editing inventory, they will complete the edit clicking the same button to process an order. This button will still require a PIN, but it will be the employee PIN. If the employee account PIN is entered correctly, the application will add the entered quantities for each product to the internal product objects rather than computing a total and subtracting the entered quantities. The employee user will also have the ability to add new products to the list of products along with the quantity on hand of the new item. The entry fields on the GUI to add new products should not usable/visible when customers are using the program. The application shall provide the ability to add both products and attachment products.

KitchenSupply has heard that Lists are a good way to store a series of logically related objects although they don't know much about them and apparently there are many versions of Lists out there. You, the developers have considerable leeway in choosing options. In the end, all the functionality must work with whichever options chosen.

For its inventory management purposes, KitchenSupply would like the following. The main window should present the user with options using a neat menu system as is usual with GUI applications. The menu should be organized logically and allow the user to perform all functionality that the application provides including additions such as quit, save customers, switch to employee mode, and save products. They do want the ability for the users to be able to easily see question details by clicking or double-clicking products (such as id, description, quantity on hand, price, item it attaches to, and material type).

For the order mechanism, KitchenSupply would like a demonstration of multiple orders occurring in a row. This will ensure inventory and customer balances are properly managed.

While messageboxes are okay for providing some feedback to users, KitchenSupply doesn't want all feedback to be provided via message boxes since they are intrusive. On the other hand, status labels at the bottom of the screen are so hard to see! They were wondering if their development team could at least show the feedback a little more prominently and still have the application look good. Let's face it – functionality is important, but nobody wants to use an ugly app! The application should look nice and professional. The team should experiment with Fonts, Sizes, and Colors.

Since the program is supposed to be a professional application, if there was a nice looking-graphical logo, KitchenSupply would be very pleased (all graphic file sizes must be small to prevent long upload times to course website). I suggest using image processing to get the logo graphic down to a size of no bigger than 128x128 thumbnail size. I also suggest making an icon version for the upper left corner of your application window.

KitchenSupply would prefer if its application did not crash – ever. If the system goes down, it might take down other systems with it and KitchenSupply does not want its users to be left waiting to make an order! That would result in a bad evaluation for the development team.

REQUIREMENTS

1) General Requirements

- The project needs to be implemented as a Python tkinter Application**.**
- Use object-oriented design: have classes, methods and utilize properties and methods.
  - There should be little to no use of static variables and methods.
  - Inheritance is required to eliminate duplicate coding
- Passing data between widgets and objects should either be done using the Event Handling mechanism using bind or callback functions (depending on data transmission).
  - Data variables should be protected as much as possible from public access using techniques covered in the course
- Submit the project to Blackboard as a zip file, named in the format: "Project_<yourASUrite>.zip." All students must submit the project individually.
- Follow the CIS 345 Programming Conventions and CIS345 Commenting Guide documents for producing good, readable, and easily maintainable code.

2) Feature Requirements

You have the freedom to visually structure your application according to your preference. All applications should allow users to:

- View a list of Products
- Add new Products in Employee user mode
  - Ask for details of each Product or product attachment.
- Edit existing Product Inventory or Quantities
- Implement data persistence: the application should load a list of products and customers from data files and save the them to their respective data files (Open/Save functionality).
- Customers can complete an order adding multiple products, a selected quantity to an order list, and completing the order triggering updates to both customer and product data.
- Demonstrate use of Graphics (**submit code in separate modules within project**)
  - Use Graphics to create a company logo
  - Use Image processing to create a thumbnail and icon version of the logo
  - Show Icon in upper left of window and thumbnail on the Application Window

- Provide the ability to cleanly exit the application.

3) Technical Requirements:

- Use the module name: kitchen_supply
- The project should have at minimum a Tk object to manage the application's interface, classes for customer, product, and attachment product.
- The project should have a Save Data menu option, which allows the user to save customer or inventory data.
- All submissions need to provide a sample data files called customers.csv, products.json, and an order_log which you choose the extension/file type. **The files should have at least 5 customers (employee is one of them), 6 products, 4 attachment products already loaded in it.** Of the 4 non-employee customers, 1 must have $2.50 balance, and 1 must have a lot of money to buy everything. The other two can be balances of your choosing.
- The class project is meant to be a demonstration of concepts learned in class. It is a technical requirement that the classes, widgets, data structures, and event-handling architecture covered in class be utilized for this project.
  - Barring small methods not covered in class, concepts covered not in class should not be used for the project. If in doubt, ask before using!
  - **No credit will be given for code, which significantly departs from class concepts and utilizes data structures, tools, functions, widgets, and techniques not covered in class.**

DELIVERABLES

1. Final Version **(100% of Project Grade)**

- Completed Product, Attachment, and Customer Classes
    - Has Constructor(s)
    - Instance variables for all attributes and properties for each attribute.
    - Methods to process data and provide public access to the functionality
- Main application Window with menu system.
    - Functioning Menu System
- File I/O should be completed
    - The open menu item should load the List of Questions from data.dat file.
    - The close menu item save the List of Questions to the data.dat file.
- A sample files with test data per the requirements above.
- Follow the CIS 345 Programming Conventions and CIS345 Commenting Guide documents for producing good, readable, and easily maintainable code.
- Fully working project application with all features as detailed in this document.
- A Readme.txt file, which documents how the application should be used. It should explain the steps users should take to operate your application, what buttons/menu items to click, the sequence of events, etc., which are needed to run the program properly.

The submissions are expected on-time and on Blackboard. If there is no file submitted on time on Course Website, or if your file cannot be opened or is corrupt in any way, it will receive a score of zero. *Check your files three times before submission. If you do not check your files, be prepared to do that at the risk of receiving a zero.*

HINTS AND RECOMMENDATIONS

- You can choose to implement the project with the look that you prefer.
- You have a fair degree of flexibility in modeling your project design, implementing methods, etc.
- Use tkinter PEs and Assignments as your template in getting started.

Suggested Sequence of Activities

- Select an Account, customer or employee.
- Once you have your Account selected, start select products to view details. Pick products and enter the quantity you wish to add to your order.
- After that, you can select purchase and must enter the correct Account PIN.
- Repeat the steps selecting different and the same account to ensure all data is updating correctly.
- Select the Employee account.
- Use the employee account to modify inventory levels.
- Use the employee account to add both a new product and attachments
- Close the application and re-open to ensure all previous data was saved and loaded in.
- Look at your code again after it is working and take out all redundant and duplicate code – create reusable classes and methods.
- Finally, clean up variable and method names, comment code properly and make the code look clean and readable!

GETTING HELP

Skill levels vary and while some people may get done much quicker, you should expect to spend 10-20 hours on this project. If you start early, you can get help early. You are welcome to come to me for help in approaching the project, design issues, solving problems, implementing specific portions, or debugging code. I am here to help you in all aspects of implementing the project! You only need to make sure that you start early enough so that you can get help. Help does need to be provided in person. Very little help can be provided over e-mail since it needs to be done interactively.

You should also to go the Tutor for help. The tutor also helps CIS 340 students, so if you wait till the end, getting assistance will be very tough due to capacity constraints. Do NOT wait till the end of semester to get started. Do not struggle alone! ***Get started early, get stuck early, and get help early!***

Grading Criteria for Project Final

The following tables represents a general rubric covering the major areas of evaluation, which are foreseen. **All submissions will be evaluated according to these criteria PLUS any additional issues**, which arise in the submission. If an error or issue of concern is uncovered in the process of grading a submission, it WILL be factored into the score regardless of whether it is listed below or not. Therefore, you can use the tables below as a guide but not as a set-in-stone list of evaluation items.