Sarah Gaines

November 2, 2023

Software Engineering

Assignment 4

**Instructions on how to build the software.**

Step 1: Setting Up Your Development Environment

Begin by ensuring that you have Python installed on your computer. If Python is not already installed, you can download the latest version from the official Python website (https://www.python.org/downloads/) and follow the installation instructions. Having Python installed is essential for running your restaurant management software.

Step 2: Creating the Python Script

Open a text editor or integrated development environment (IDE) such as Visual Studio Code, PyCharm, or IDLE. Create a new Python script by selecting "New File" or "New Python File" and give it a suitable name, for example, restaurant\_manager.py. This file will serve as the main script for your restaurant management software.

Step 3: Writing the Python Code

In your newly created restaurant\_manager.py script, begin writing the Python code. To start, you'll need to import the necessary libraries. In this software, we are using tkinter for creating the graphical user interface and messagebox for displaying messages or pop-up dialogs.

Step 4: Initializing the Application Window

Within your Python script, create a class named RestaurantManager. This class will encapsulate the functionality and features of your restaurant management software. In the constructor method \_\_init\_\_(self, root), the root parameter represents the main application window. You can create the main application window by calling tk.Tk() to instantiate a Tkinter application object. Set the window title to "Sarah's Restaurant" using self.root.title("Sarah's Restaurant").

Step 5: Defining the Restaurant Menu

Define the restaurant menu using a Python dictionary. This dictionary structure organizes menu items into categories, such as "Food," "Drinks," and "Dessert." Each category contains a list of menu items, where each item is represented as a dictionary with keys like "name" (for the item's name) and "price" (for its price).

Step 6: Creating GUI Components

To construct the graphical user interface, develop several methods within the RestaurantManager class. These methods create various components:

create\_menu(): Generates a menu display with categories and item buttons.

create\_order(): Creates a listbox to display the customer's order.

create\_feedback\_button(): Generates a "Give Feedback" button.

create\_total\_button(): Adds a "Calculate Total" button.

create\_delete\_button(): Implements a "Delete Item" button for removing menu items from the order.

create\_call\_server\_button(): Adds a "Call Server" button for calling a server.

create\_exit\_button(): Includes an "Exit Application" button for closing the software.

Step 7: Implementing Functionality

To make your restaurant management software fully functional, you'll need to implement various methods:

add\_to\_order(item): This method allows you to add selected menu items to the order list.

calculate\_total(): Calculates the total cost of the customer's order, including tax, and displays it as a pop-up message.

take\_feedback(): Provides a feedback window where customers can provide feedback. This window is optional.

delete\_item(item): Implements the functionality to remove a specific menu item from the order.

call\_server(): Enables customers to call the server for assistance if needed.

exit\_app(root): Adds a function to close the application gracefully.

Step 8: Running the Application

In the if \_\_name\_\_ == "\_\_main\_\_": block, create an instance of the RestaurantManager class and start the main application loop with root.mainloop(). The Tkinter main loop keeps the application running and responsive to user interactions.

Step 9: Customization and Testing

To tailor the application to your specific restaurant, you can customize various aspects such as menu items, colors, and button labels. Testing the application with sample menu items and use cases will help ensure it functions as intended.

Step 10: Running the Application

Save your restaurant\_manager.py script after making any customizations. Open a terminal or command prompt and navigate to the directory where your script is located using the cd command. Then, run the application by executing the command python restaurant\_manager.py.

Step 11: Adding a Welcome Message

If you'd like to provide a warm welcome to your customers, you can include a welcome pop-up message at the beginning of the application. This message can be displayed using messagebox.showinfo within the \_\_init\_\_ method.

Congratulations, you've successfully built a restaurant management software application with a user-friendly graphical interface that allows customers to interact with your restaurant's menu, place orders, and even provide feedback or request assistance.

Changes Made from System Architecture Design

In comparison to the system architecture design outlined in the previous assignment, a few changes and additions have been made to enhance the functionality and usability of the restaurant management software:

1. **User-Friendly Graphical Interfaces:**
   * **Enhanced Staff Console:** The staff console now boasts a more user-friendly graphical interface that streamlines order management, bill handling, and reservation management. The inclusion of a delete button allows staff to efficiently remove menu items from orders.
   * **Guest-Focused Display:** The guest display on tablets or table screens has been optimized for a better user experience, enabling guests to view and manage their bills seamlessly.
2. **Mobile Application Integration:**
   * **Mobile Application Interface:** The architecture now accommodates a mobile application interface, allowing customers to use smartphones or tablets for making reservations, browsing menus, and providing feedback conveniently.
3. **Internal System Interfaces:**
   * **Payment Gateway Integration:** To ensure secure and efficient payment processing, the architecture now incorporates interfaces with payment gateways such as Stripe or PayPal.
4. **Enhanced Database Integration:**
   * **Improved Database Management:** A robust connection with a database management system (e.g., MySQL or PostgreSQL) has been emphasized to store and retrieve diverse data types efficiently. This includes menu items, orders, bills, reservations, and user profiles.
5. **Additional Features:**
   * **Server Call Button:** The architecture includes a new feature for customers to call a server, enhancing communication between guests and staff.
   * **Exit Application Button:** An exit application button has been added for a convenient way to close the application.
   * **Welcome Message:** A pop-up welcome message greets users upon starting the application.
6. **External Software Interfaces:**
   * **Menu Management Portal:** For administrators, an external interface has been introduced for effortless menu updates, description changes, and price adjustments. This can be web-based for ease of use.
   * **Customer Feedback Systems:** The architecture now supports external connections to platforms or APIs that offer efficient and robust customer feedback and rating mechanisms.

These changes and additions aim to create a more intuitive, feature-rich, and user-friendly restaurant management software, enhancing both staff and guest interactions while ensuring efficient data management and external integrations for seamless operations.

Estimation of Completed Project

I estimate that I have completed about 90% of the tasks compared to the initial proposal. Most of the features and functionalities have been implemented, including the graphical user interfaces, menu management, feedback collection, and external interfaces. The remaining tasks involve final refinements, adding the exit application button, implementing the server call feature, and ensuring that all elements are well-integrated and thoroughly tested.

Video Link:

<https://www.loom.com/share/58524e7e616f488d9a8d732d327ececb?sid=7623f167-82e8-4edf-96d4-7113ee776991>

Github

[Sarah-Restaurant-App/Sarahs Restaraunt App.py at main · sarahgaines4/Sarah-Restaurant-App (github.com)](https://github.com/sarahgaines4/Sarah-Restaurant-App/blob/main/Sarahs%20Restaraunt%20App.py)