Programación II

Evaluación 2 (25% del ramo)

**Problema**

Estamos desarrollando una aplicación para un restaurante que permita gestionar **carga de ingredientes, stock, carta restaurant, pedidos y generar boletas.** La solución debe proporcionar una interfaz gráfica para los usuarios, permitir la gestión de pedidos con diferentes productos y generar boletas detalladas en formato PDF que puedan ser impresas o guardadas.

**Se ha dispuesto un video**

Falta el video aun

**software en funcionamiento para que puedan revisar con más detalle.**

los menús para ofrecer están fijados por el cliente y son:

|  |  |  |
| --- | --- | --- |
| Menús | precio de venta | Ingredientes |
| Papas fritas | 500 | 5 x papas |
| Pepsi | 1100 | 1 x Pepsi |
| Completo | 1800 | 1x vienesa  1x pan de completo  1x tomate  1x palta |
| Hamburguesa | 3500 | 1x pan de hamburguesa  1x lamina de queso  1x churrasco de carne |

**Descripción de la Pestaña 1: Carga de ingredientes**

**Propósito:** La pestaña 1, titulada " **Carga de ingredientes** " (ver video), está diseñada para permitir a los usuarios cargar los ingredientes desde un archivo .csv que será dado por el profesor.

Debe contar con un botón de carga de archivo y un treeview para visualizar la carga del archivo, el archivo tiene 3 columnas (nombre, unidad, cantidad)

Se debe agregar un botón llamado agregar al stock, que agregue los ingredientes de la tabla treeview al stock.

**Descripción de la Pestaña 2: Stock**

**Propósito:** La pestaña 2, titulada "**stock**" (ver video), está diseñada para permitir a los usuarios gestionar el stock de ingredientes. A través de esta interfaz, los usuarios pueden añadir nuevos ingredientes, eliminar ingredientes existentes, y visualizar la lista actual de ingredientes disponibles en el stock.

Además, hay un botón llamado generar menú que crear los menú prefijados a partir de la disponibilidad de ingredientes en el stock.  
**ej** si no hay palta, no se puede crear el menú completo.

**Descripción de la Pestaña 3: Carta restorant**

**Propósito:** La pestaña 3, titulada "**carta restaurant**", es visualizar un archivo en formato PDF a partir de los menús disponibles. Se debe crear un botón llamado generar carta pdf, que busque los menús disponible en tiempo real y después los convierta a pdf.

**Descripción de la Pestaña 4: Pedido**

**Propósito:** La pestaña 4, titulada "**Pedido**", está diseñada para gestionar la selección de menús a partir de los ingredientes disponibles en el stock. Esta pestaña permite a los usuarios ver qué menús pueden prepararse, añadirlos a un pedido, gestionar las cantidades, y finalmente, generar una boleta en formato PDF.

Como orden básico de entrega de los archivos del proyecto se esperan mínimamente estos archivos (puede separar en más archivos si lo considera más claro):

* Restaurante.py -> interface y funciones gráficas + carga de archivo csv con ingredientes (es el archivo principal del proyecto)
* Ingrediente.py -> Clase ingrediente considerando estructura del mismo.
* Stock.py -> Clase stock con todos los métodos asociados incluidos agregar, modificar y/o eliminar ingrediente al stock
* Pedido.py -> Clase pedido con todos los métodos asociados para su generación, considerando el concepto de agregar o sacar menús de un pedido, modificación de stock asociado y cálculo de totales.
* Menu.py -> indicando Clase Menu y métodos asociados + generación de menus por defecto.
* Menupdf.py -> funciones para generar el menú disponible respecto a stock en pdf.
* Boleta.py -> clase boleta que contiene los datos de la boleta y métodos de generación de datos de boleta y pdf asociado.

### **Criterios de Evaluación**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Criterio** | **Descripción** | **Excelente (10 pts)** | **Bueno (8 pts)** | **Suficiente (4 pts)** | **Insuficiente (1 pts)** | **Peso (%)** |
| **Implementación de la Orientación a Objetos** | Uso correcto de clases, atributos, comportamiento de objetos. | Utiliza todos los conceptos de OO correctamente y de manera eficiente. | Utiliza la mayoría de los conceptos de OO correctamente. | Utiliza algunos conceptos de OO, pero con errores. | Uso incorrecto o inexistente de los conceptos de OO. | 30% |
| **Funcionalidad** | El proyecto cumple con los requisitos funcionales especificados. | Cumple con todos los requisitos funcionales sin errores. | Cumple con la mayoría de los requisitos, con pocos errores. | Cumple con algunos requisitos, pero presenta varios errores. | No cumple con los requisitos funcionales especificados. | 20% |
| **Presentación Oral** | Claridad y organización en la presentación, descripción de la problemática, diagramas, solución, demostración de código y manejo de preguntas. | Excelente presentación oral; describe la problemática y solución de manera clara, responde adecuadamente las preguntas y cumple con el tiempo de 15 minutos. | Buena presentación describe la problemática y solución de manera clara, aunque con detalles menores; responde adecuadamente las preguntas; Excede el tiempo definido. | Presentación suficiente, pero con falta de claridad y organización; la descripción de la problemática y solución es incompleta; manejo básico de preguntas; Excede el tiempo definido. | Presentación pobremente organizada, falta de claridad y omisión de varios elementos clave; manejo ineficiente de preguntas; Excede el tiempo definido. | 25% |
| **Implementación de Interfaz Gráfica** | Uso adecuado de una interfaz gráfica para mejorar la experiencia del usuario. | Interfaz gráfica bien diseñada, intuitiva y completamente funcional. | Interfaz gráfica funcional y adecuada, con pequeños detalles por mejorar. | Interfaz gráfica básica, con varios aspectos que podrían mejorarse. | Interfaz gráfica pobremente implementada o inexistente. | 20% |
| **Evaluación del Diagrama de Clases** | Correcta representación de las clases, relaciones, herencia y otros elementos orientados a objetos en un diagrama de clases. | Diagrama de clases claro, completo y correctamente estructurado, mostrando todas las relaciones adecuadas. | Diagrama de clases funcional y correcto, con algunos detalles por mejorar. | Diagrama de clases con errores o ausencias en algunas relaciones. | Diagrama de clases incorrecto o incompleto, faltando elementos clave. | 5% |

**¿Qué debo hacer?**

Debes crear el diagrama de clases del problema propuesto y realizar la codificación usando programación orientada a objetos e interfaces graficas. Puedes basarte en el video de la aplicación dispuesto en la plataforma.

**¿Qué debo entregar?**

Debes preparar una presentación **PPT en grupos de 2-5** para el miércoles 15/10/2024 **seccion 1** y 17/10/2024 **seccion 2** la cual debe contener:

1. Presentación del grupo y tema (título del problema, nombre de los integrantes, fecha y logo del departamento).
2. El diagrama de clases con la solución propuesta del problema.
3. Presentación de la solución en código explicando el flujo principal de la solución.
4. La presentación de la interfaz visual del problema.
5. La demostración de la funcionalidad total del programa.

**La presentación tiene un tiempo máximo de 15 minutos y luego una tanda de preguntas de 5 minutos.**