Sistema automatizado de tickets para sala de cine

Autores		
Nombre	Padron	Email
Alaniz Patricio Javier	91465	palaniz@fi.uba.ar
Blas Andres Romero	111476	baromero@fi.uba.ar
Escobar Jonathan Nahuel	109764	jescobarv@fi.uba.ar

30 de noviembre de 2023

1. Palabras clave

Python, Aplicacion, Cine, QR, Totem.

2. Introducción

En el presente documento, presentamos nuestra solución para un sistema automatizado de tickets destinado a salas de cine, desarrollado íntegramente en el lenguaje de programación Python. Este proyecto comprende dos aplicaciones independientes: una encargada de generar códigos QR a partir de los datos recopilados durante la compra del usuario, y la segunda aplicación, diseñada para la lectura y validación de estos códigos QR en la entrada al cine, además de la posterior almacenamiento de la información en un archivo de texto.

3. Solución propuestas

La solucion propuesta consta de 2 aplicaciones, ambas basadas en la biblioteca Tkinter de Python, tienen un propósito común pero funciones distintas.

3.1. Aplicación 1 - Generador de Códigos QR

Esta aplicación permite a los usuarios ingresar datos de compra de entradas de cines y snacks. Luego, genera un código QR a partir de estos datos y lo guarda en una carpeta específica en formato PDF. La interfaz gráfica de esta aplicación incluye campos para ingresar información como detalles de la compra (entradas, snacks, etc.) y un botón para generar el código QR. Utiliza funciones para procesar los datos ingresados y convertirlos en un código QR utilizando la biblioteca cv2. Posteriormente, guarda este código QR como PDF en la carpeta QR.

3.2. Aplicación 2 - Lector de Códigos QR

Esta aplicación se encarga de leer los códigos QR generados previamente. Utiliza la cámara web para escanear un código QR y extraer la información codificada en él. La interfaz gráfica incluye un campo de entrada para ingresar manualmente el ID del código QR y un botón para escanear el código utilizando la cámara. Al detectar un código QR válido, extrae la información contenida en él y muestra los detalles de la compra (entradas, snacks, etc.) en el área de la interfaz gráfica destinada para ello. Hace uso de la biblioteca cv2 para la detección y decodificación de códigos QR a través de la cámara web.

3.3. Estructuras de Datos Utilizadas

Para el manejo de datos de compra y generación de códigos QR, se utilizan estructuras de datos como listas, diccionarios y cadenas de texto. Estas estructuras ayudan a organizar y procesar la información de manera eficiente para su posterior conversión en códigos QR. En la lectura de códigos QR, los datos extraídos se almacenan y manejan utilizando estructuras de datos similares, permitiendo mostrar la información decodificada de manera ordenada en la interfaz gráfica.

3.4. Dificultades y Desafíos

Integrar APIs externas requiere una gestión adecuada de las solicitudes y respuestas, así como el manejo de datos provenientes de fuentes externas. El diseño y la implementación de las estructuras de datos deben ser robustos y flexibles para gestionar la información de manera efectiva, considerando la posible variabilidad en los datos de entrada y salida.

4. Pruebas y/o validación

Se implementaron pruebas unitarias para verificar el funcionamiento de las funciones principales del programa, como la generación de códigos QR, la lectura de los mismos, el manejo de archivos y la integración con APIs externas. Cada función se sometió a pruebas individuales, proporcionando datos de entrada variados para asegurar que el código respondiera de manera correcta y adecuada a diferentes situaciones. Se verificó la integridad de la información después de la codificación y decodificación de los códigos QR, asegurando que los datos generados y leídos fueran consistentes.
Se realizaron pruebas funcionales que simularan el flujo completo de las aplicaciones. Esto incluyó desde la entrada manual de datos hasta la generación y lectura exitosa de los códigos QR. Se realizaron pruebas para verificar la integración correcta con APIs externas, si estas estaban presentes en las aplicaciones. Esto incluyó verificar la precisión de la información obtenida de las APIs y su integración adecuada en el flujo de la aplicación.
Se probó la aplicación en diferentes escenarios, como la ausencia de archivos necesarios (como la carpeta de códigos QR o archivos de registro) para asegurar que el programa manejara estas situaciones de manera adecuada y no se produjeran errores inesperados. Se realizaron pruebas de manejo de errores, como introducción de datos incorrectos o fallas en la lectura de códigos QR, para verificar que el programa respondiera de manera controlada y proporcionara mensajes de error claros y útiles.

5. Plan de actividades

Se adoptó un enfoque ágil para el desarrollo, aprovechando las funcionalidades de control de versiones de GitHub para realizar iteraciones continuas y colaborativas.
La comunicación y colaboración se llevaron a cabo principalmente a través de reuniones virtuales en plataformas como Google Meet, donde se discutían avances, obstáculos y próximos pasos del proyecto. Se emplearon canales de mensajería instantánea como WhatsApp para una comunicación fluida y rápida entre los miembros del equipo, facilitando consultas rápidas y coordinación cotidiana.

6. Hipótesis y Supuestos

Se asume que la API externa utilizada para enriquecer los datos de las películas y eventos estará siempre disponible y accesible durante el desarrollo y uso del programa. Esta suposición es fundamental para asegurar el correcto funcionamiento de la integración con la API.
Se parte del supuesto de que los usuarios finales poseen un conocimiento básico sobre el correcto funcionamiento del programa, incluyendo la generación y lectura de códigos QR, así como la interacción con la interfaz gráfica.
Se considera que cada cine cuenta con una única sala y que las programaciones de películas no se superponen en el tiempo. Esta suposición simplifica el proceso de generación de códigos QR y la gestión de la información asociada.
Se establece como premisa el propósito de continuar mejorando y evolucionando las aplicaciones en el futuro. Esta suposición es crucial para planificar cambios, adiciones de funcionalidades y correcciones en versiones posteriores del software, con el objetivo de mantener su relevancia y utilidad a lo largo del tiempo.

7. Referencias

Tkinter - https://docs.python.org/3/library/tk.html
OpenCV - https://docs.opencv.org/4.x/
Pillow - https://pillow.readthedocs.io/en/stable/
Request - https://requests.readthedocs.io/en/latest/
QRCode - https://pypi.org/project/qrcode/
Datetime - https://docs.python.org/3/library/datetime.html
PyMuPDF - https://pymupdf.readthedocs.io/en/latest/
GitHub - https://docs.github.com/es

8. Resumen/Abstract

El proyecto consistió en el desarrollo de dos aplicaciones basadas en Python utilizando la biblioteca Tkinter para la interfaz gráfica. La primera aplicación se enfocó en generar códigos QR a partir de datos de compra de entradas de cine y snacks, mientras que la segunda aplicación permitió la lectura de estos

códigos.
Se adoptó un enfoque práctico de desarrollo, utilizando GitHub para el control de versiones y la colaboración remota a través de reuniones virtuales en Google Meet y comunicación por WhatsApp. La metodología fue ágil y adaptable, permitiendo un desarrollo incremental y continuo.
Se tomaron supuestos importantes, como la disponibilidad permanente de una API externa, el conocimiento básico de los usuarios sobre el funcionamiento del programa y la singularidad de salas en los cines, sin superposición de películas. Además, se contempló la idea de mejorar continuamente las aplicaciones en el futuro.