

Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Introducción a la Programación y Computación 2
Escuela de Vacaciones Diciembre 2024

Ing. Jose Manuel Ruiz Juarez
Tutor de curso: Rodrigo Alejandro Hernández de León



PROYECTO 2

OBJETIVO GENERAL

Desarrollar una solución integral que implemente una API que brinde servicios utilizando el protocolo HTTP bajo el concepto de programación orientada a objetos (POO).

OBJETIVOS ESPECÍFICOS

- Implementar una API a través de lenguaje Python que pueda ser consumida utilizando el protocolo HTTP.
- Utilizar el paradigma de programación orientada a objetos para construir software.
- Utilizar archivos XML como insumos para la comunicación con el API desarrollado.
- Utilizar expresiones regulares para extraer contenido de texto.

DESCRIPCIÓN GENERAL

IPCArt-Studio, la destacada plataforma para la creación y gestión de arte en píxeles, está en proceso de transformación hacia una moderna aplicación web diseñada para ofrecer una experiencia global y accesible. Este proyecto revolucionará la plataforma mediante la implementación de una API desarrollada en Flask, que garantizará una gestión eficiente de datos y operaciones, y un frontend desarrollado con Django, proporcionando una interfaz intuitiva, funcional y accesible desde cualquier navegador.

El sistema utilizará matrices dispersas para un almacenamiento eficiente y ágil de las imágenes de pixel art, optimizando el manejo de proyectos sin sacrificar recursos. Además, los datos del sistema se almacenarán en una base de datos estructurada mediante archivos XML, lo que permitirá una organización legible, portable y fácilmente integrable. Esta solución garantiza una gestión robusta de proyectos, usuarios y actividades sin depender de bases de datos tradicionales.

La plataforma no solo se centrará en las herramientas de creación y edición de pixel art, sino que también ofrecerá potentes capacidades de análisis mediante estadísticas visuales dinámicas. Estas estadísticas serán generadas utilizando Plotly, una biblioteca interactiva que permitirá a los usuarios explorar y comprender datos clave a través de gráficos y visualizaciones modernas y atractivas. Los artistas podrán visualizar métricas relacionadas con sus proyectos, solicitudes y actividades, obteniendo una visión clara del uso y desempeño del sistema.

Este nuevo enfoque tecnológico consolidará IPCArt-Studio como una herramienta de vanguardia, permitiendo a la comunidad de artistas digitales no solo colaborar y compartir sus creaciones, sino también analizar y optimizar su trabajo en un entorno digital avanzado y accesible.

IMPLEMENTACIÓN

IPCArt-Studio se transformará en una aplicación web basada en arquitectura cliente-servidor, diseñada para ser consumida como un servicio desde internet. El sistema permitirá convertir los archivos XML que actúan como base de datos en un formato JSON estandarizado, almacenando temporalmente estos datos antes de su envío al servidor, con el objetivo de garantizar la consistencia y modernizar la gestión de la información. La solución incluirá una API en Flask para manejar eficientemente las operaciones de transformación y almacenamiento, complementada con un frontend desarrollado en Django que ofrecerá una interfaz intuitiva y funcional. Esta evolución permitirá a IPCArt-Studio adaptarse a las necesidades actuales de accesibilidad y escalabilidad, consolidándose como una herramienta moderna para la creación y gestión de pixel art.

ARQUITECTURA

La arquitectura cliente – servidor habla acerca de una relación entre un programa (cliente) que solicita un servicio o recurso de otro programa (el servidor).

Se ha solicitado que construya un software que pueda ser consumido desde internet como un servicio. Dicho software será capaz de transformar los archivos XML actuales hacia un formato JSON, el JSON debe de ser almacenado previo al envío hacia el servidor ya que al pretender la estandarización del almacenaje se espera migrar la aplicación de IPCArt hacia un entorno virtual.

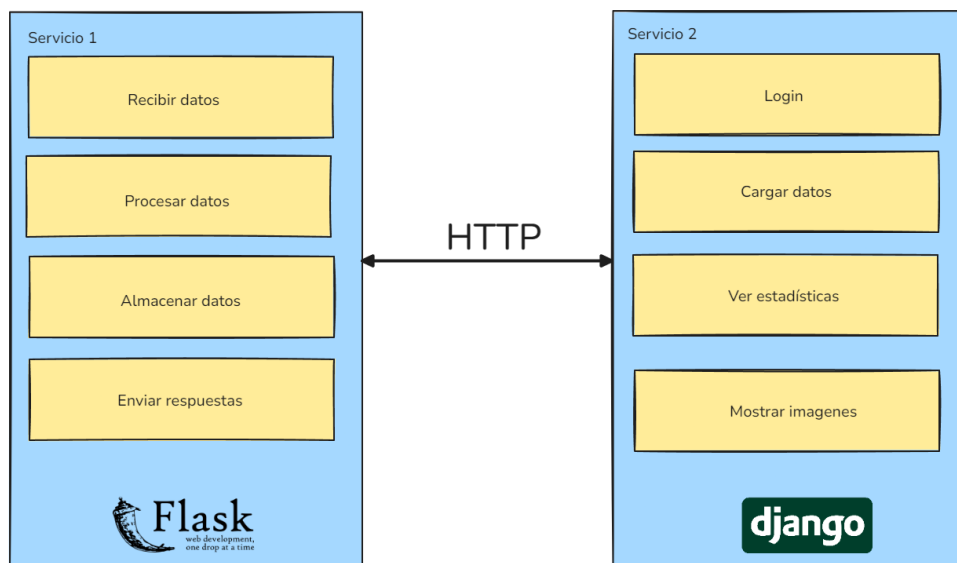


Figura 1: Arquitectura de la aplicación

SERVICIO BACKEND

Este servicio consiste en una API que brindara servicios utilizando el protocolo HTTP, su funcionalidad principal es procesar los datos recibidos del servicio de frontend, luego de procesar los datos es necesario que estos sean almacenados en un archivo XML, este servicio también tiene la funcionalidad de devolver los datos que fueron almacenados para que sean mostrados como anteriormente se indica.

Dentro de los endpoints que se espera como mínimo que realice son:

- /cargarUsuarios: Se espera un XML de entrada con datos de usuarios.
- /cargarImagen: Se espera un XML de entrada con las características de la imagen.
- /login: Para iniciar sesión como administrador o usuario común.
- /imagenes: Devolverá una galería de imágenes de las imágenes procesadas para el usuario.

Y cualquier endpoint que usted considere necesario.

PERSISTENCIA

La aplicación contendrá una capa de persistencia donde la información cargada será almacenada en una carpeta de archivos XML. Queda a discreción del estudiante definir el formato específico en que se estructurará la información dentro de estos archivos. Sin embargo, queda **PROHIBIDO** el uso de bases de datos relacionales o sistemas de gestión de bases de datos como SQL, MySQL, PostgreSQL, Oracle DB o similares para el almacenamiento de datos.

Debe de tomar en cuenta que durante la calificación se solicitará probar sus endpoints desde postman y además que muestre su persistencia de archivos XML.

SERVICIO FRONTEND

Consiste en una aplicación web, que contará con las funcionalidades tales como:

- Carga de archivos XML.
- Inicio de sesión de usuarios.
- Procesar imágenes.
- Mostrar imágenes.
- Ver estadísticas.

Se espera que en la aplicación de pueda visualizar los archivos XML previo al envío, así como el XML correspondiente con los datos que el servidor retorna.

En esta ocasión se contará con dos tipos de usuario:

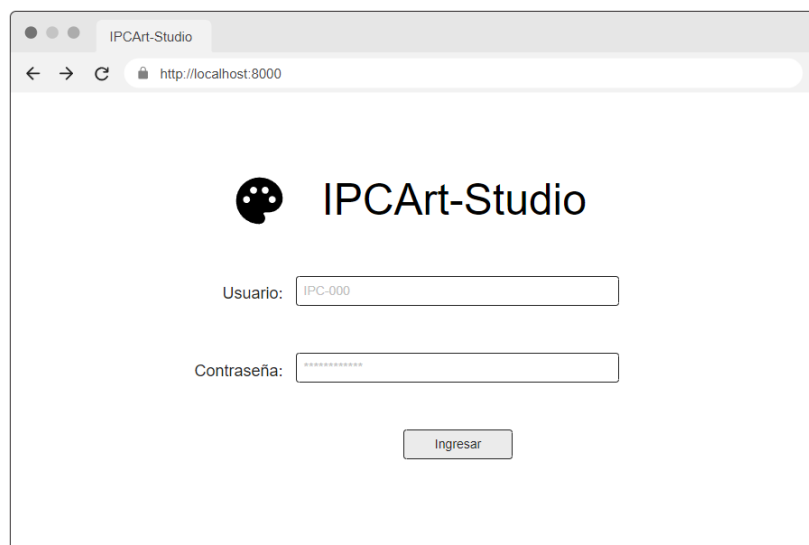
- Administrador
- Usuario común (quien previamente en el proyecto 1 fue solicitante)

Para el manejo de la carga de información se le solicitara a usted que únicamente acepte la carga de archivos XML, procese la información por medio de estructuras de datos y expresiones regulares y almacene la información por medio de XML.

En el caso de los usuarios tienen que ingresar sus credenciales y estos usuarios deben de estar previamente registrados en el sistema por medio del administrador.

LOGIN

En este apartado únicamente inician sesión los usuarios que solicitaran su diseño de Pixel Art o el administrador y tiene que ser la primera vista en la cual el usuario mire cuando abra la aplicación.



Fuente: Elaboración propia

Para ingresar con IPCArt-Studio se tomará en cuenta las siguientes consideraciones:

- El administrador iniciará sesión con el usuario "AdminIPC" y la contraseña "ARTIPC2"
- Los usuarios comunes, su usuario comenzara con las palabras de IPC-#ID.

MODULO ADMINISTRADOR

En este módulo únicamente ingresa el administrador y contará con las siguientes funcionalidades:

CARGA MASIVA

La aplicación contará con una opción que permitirá que el administrador pueda cargar archivos de tipo XML donde se pueden cargar de 1 a muchos archivos XML y no borrarán el contenido que tengan en las listas al cargar un nuevo archivo XML. Dentro de los archivos XML que se cargarán serán los siguientes:

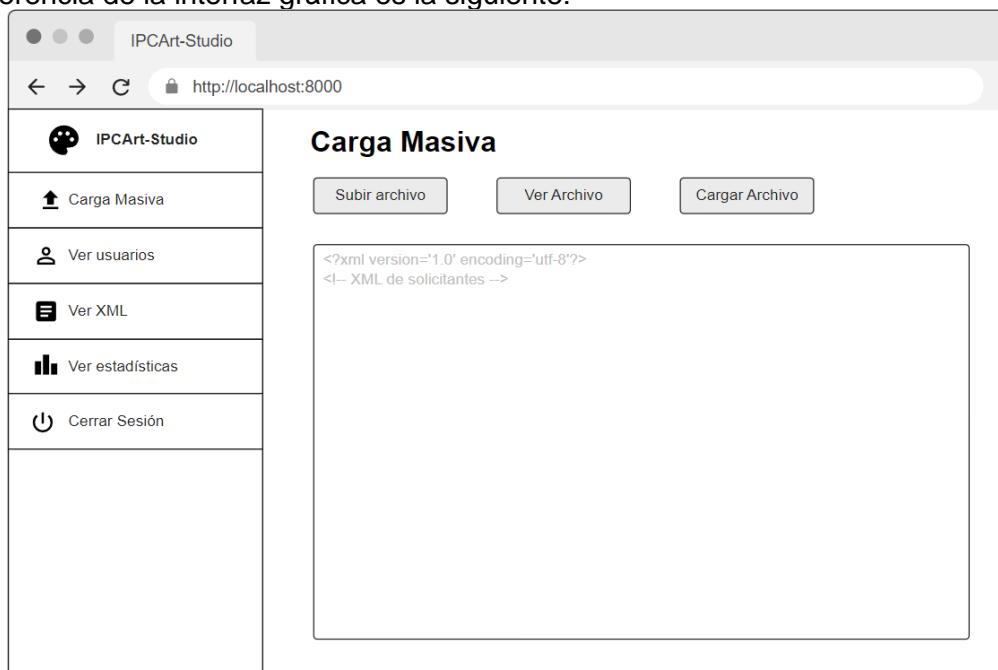
- Solicitantes:

```
<?xml version='1.0' encoding='utf-8'?>
<solicitantes>
  <solicitante id="IPC-001" pwd="3267">
    <NombreCompleto>Carlos Ruiz</NombreCompleto>
    <CorreoElectronico>carlos.ruiz@example.com</CorreoElectronico>
    <NumeroTelefono>67984298</NumeroTelefono>
    <Direccion>Ciudad de Guatemala, Guatemala</Direccion>
    <perfil>https://cdn-icons-png.flaticon.com/512/3135/3135768.png</perfil>
  </solicitante>
  <!--y muchos más solicitantes-->
</solicitantes>
```

Los usuarios se guardarán en la persistencia y cada usuario contará con los datos de:

- Id: Comienza con las letras de IPC- y posteriormente con números y va a ser único en toda la aplicación y valide que no se repita el mismo id en un usuario ya que con este id ingresa a la aplicación al iniciar sesión.
- Pwd: Es la contraseña con la que el usuario podrá iniciar sesión.
- Nombre Completo: Nombre del solicitante.
- Correo Electrónico: Es el correo del solicitante y tiene que validar con una expresión regular que si sea la sintaxis correcta del correo electrónico.
- Número de Teléfono: Es el número del solicitante, tiene que validar que sean enteros y que obligatoriamente debe de contener 8 dígitos.
- Dirección: Es la dirección donde vive el solicitante.
- Perfil: Será un link de la foto de perfil de cada usuario.

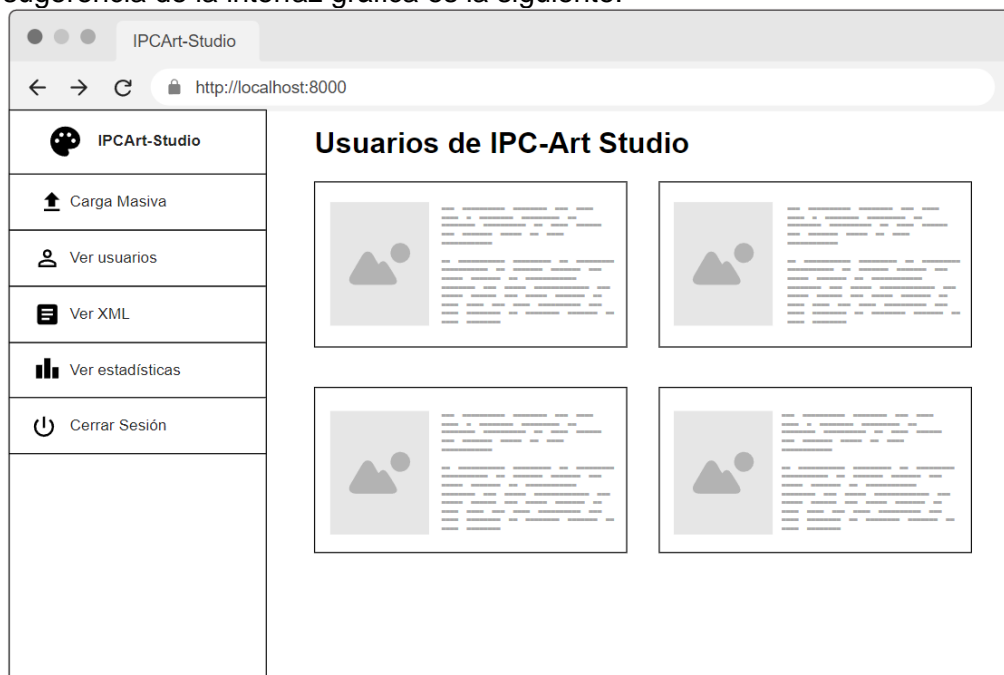
Una sugerencia de la interfaz gráfica es la siguiente:



VER USUARIOS

El administrador tiene que ver los usuarios de 2 maneras:

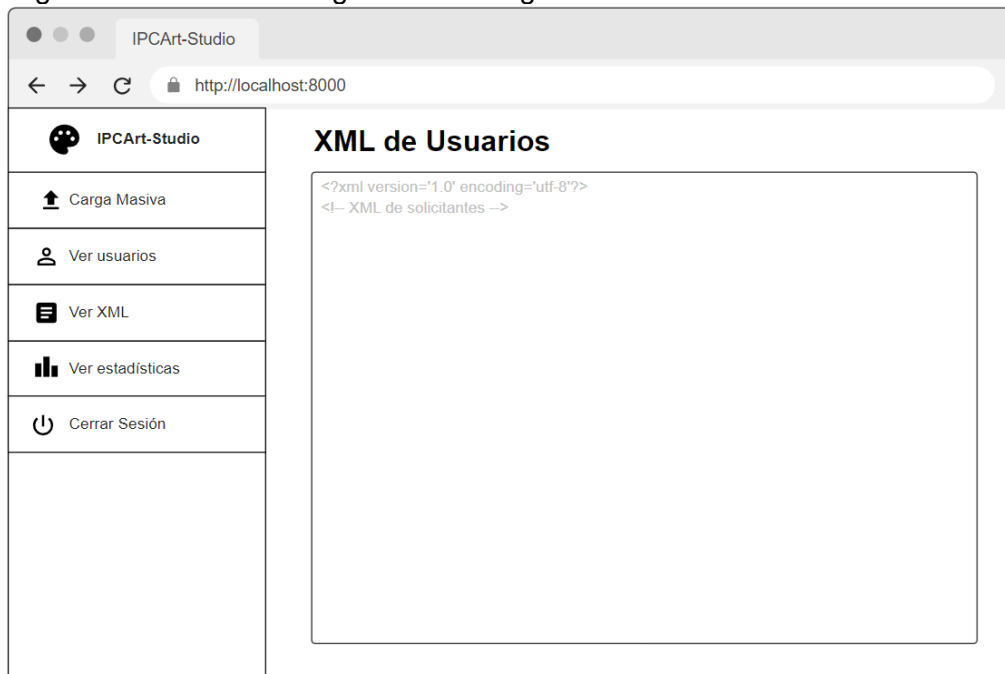
1. En una página web donde se muestre toda la información de los usuarios.
Una sugerencia de la interfaz gráfica es la siguiente:



Fuente: Elaboración propia

2. En un XML detallando la información de cada usuario y con ello se enlace las imágenes procesadas.

Una sugerencia de la interfaz gráfica es la siguiente:



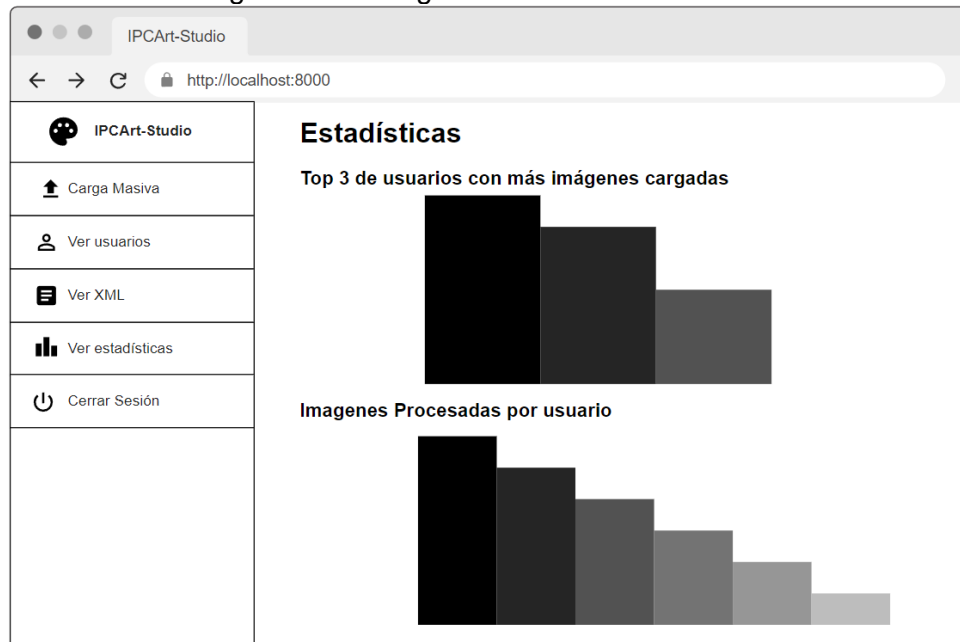
Fuente: Elaboración propia

VER ESTADÍSTICAS

El administrador puede ver las siguientes estadísticas, representadas en gráficos de barras:

- Top 3 de usuarios con más imágenes cargadas.
- Cantidad de imágenes editadas por cada usuario en orden descendente.

Una sugerencia de interfaz gráfica es la siguiente:



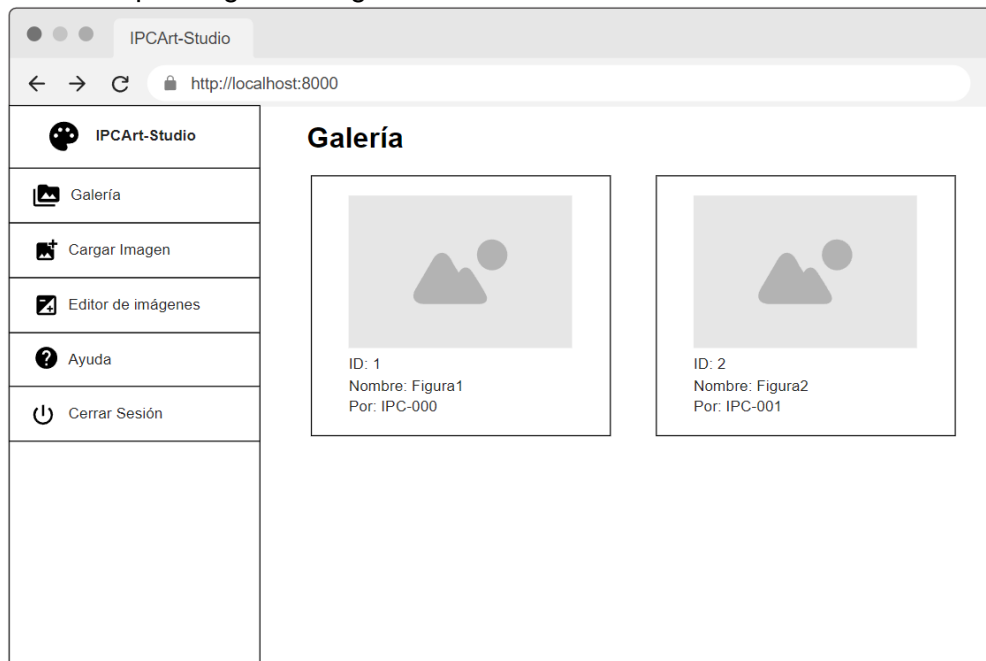
Fuente: Elaboración propia

MODULO DE USUARIO

En este módulo únicamente ingresan los usuarios comunes que iniciaron sesión en la aplicación y contendrá lo siguiente:

GALERIA

Los usuarios pueden ver todas las imágenes píxel art que existen en el sistema, junto con los datos del usuario que cargó su imagen.



Fuente: Elaboración propia

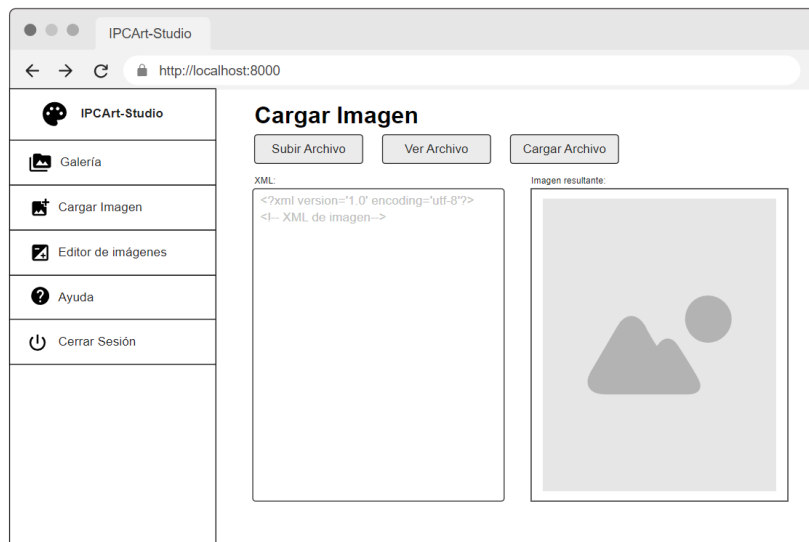
CARGAR IMAGEN

Los usuarios pueden cargar su imagen por medio de un formato de archivo XML. El XML que tiene que cargar el usuario tiene que contener la siguiente estructura:

```
<figura>
  <nombre>Figura 1</nombre>
  <diseño>
    <pixel fila="1" col="14">#000000</pixel>
    <pixel fila="1" col="15">#000000</pixel>
    <pixel fila="1" col="16">#000000</pixel>
    <pixel fila="2" col="13">#000000</pixel>
    <!-- Más pixeles aqui -->
  </diseño>
</figura>
```

Al cargar el archivo XML, se procederá a procesar dentro de un generador de imágenes usando la matriz dispersa. Además, al presionar el botón de cargar se debe de enviar el XML y retornar el código de graphviz y mostrar la imagen procesada a lado del texto del archivo XML. Para visualizar la imagen, le recomiendo usar quickchart. El servidor debe ser capaz de asignar un único ID a cada figura.

Una sugerencia de la interfaz gráfica es la siguiente:



Fuente: Elaboración propia

EDITOR PERSONALIZADO DE IMAGENES

Los usuarios seleccionan alguna de las imágenes ya cargadas en el sistema y puede cambiar a las siguientes tonalidades:

- Escala de grises

Para esto utilice la siguiente formula:

$$\text{Nuevo gris} = 0.2989 * R + 0.5870 * G + 0.1140 * B$$

(Nuevo gris, Nuevo gris, Nuevo gris)

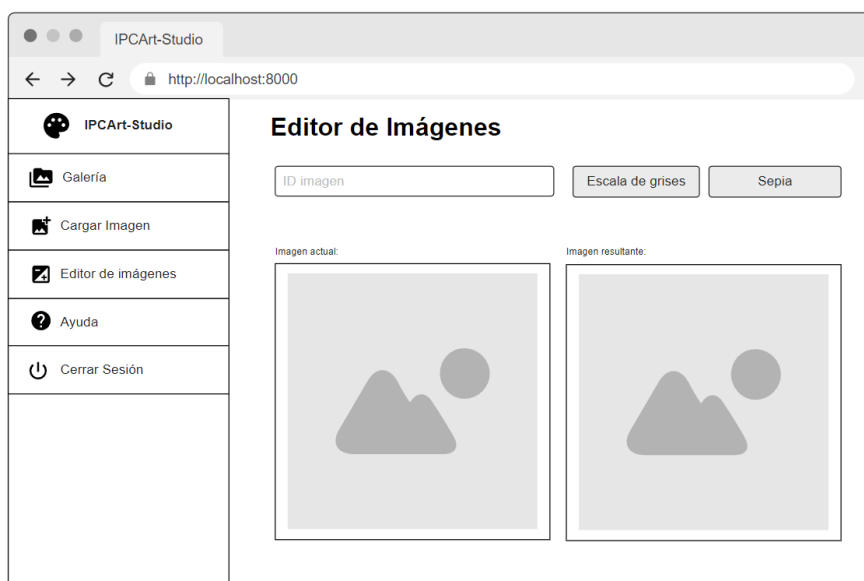
- Tonalidad sepia

Para esto utilice la siguiente formula:

$$\begin{aligned} \text{Nuevo Rojo} &= 0.393 * R + 0.769 * G + 0.189 * B \\ \text{Nuevo Verde} &= 0.349 * R + 0.686 * G + 0.168 * B \\ \text{Nuevo Azul} &= 0.272 * R + 0.534 * G + 0.131 * B \end{aligned}$$

(Nuevo Rojo, Nuevo Verde, Nuevo Azul)

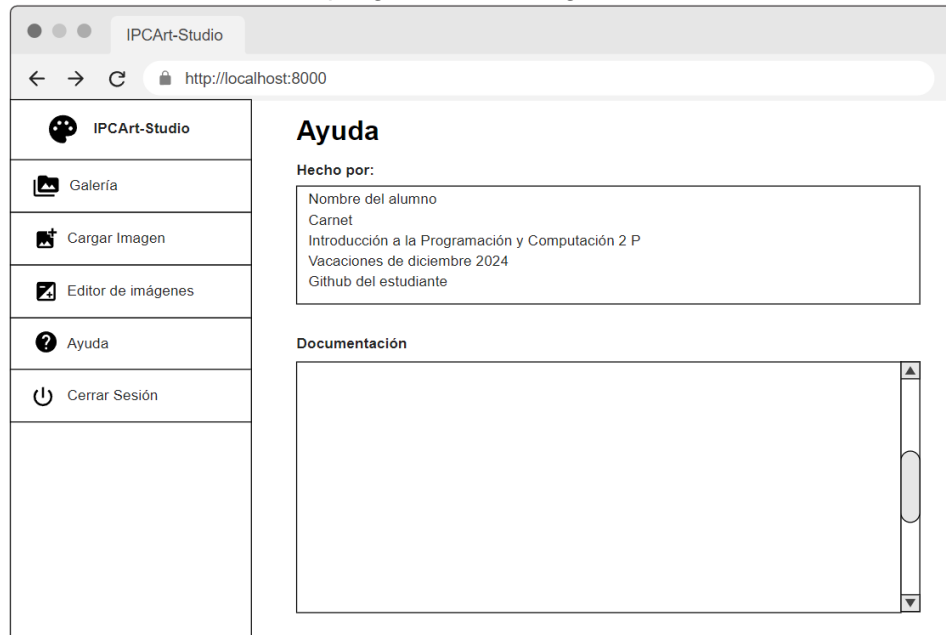
Y genera un nuevo ID de la imagen y la guarda en su persistencia. Además, se mostrará el resultado en graphviz y graficándolo a lado con la imagen ya procesada. Junto con ello esta imagen aparecerá como propiedad del editor en la galería. Una sugerencia de la interfaz es la siguiente:



Fuente: Elaboración propia

AYUDA

Se desplegará dos opciones, una para visualizar información del estudiante y otra para visualizar la documentación del programa. Una sugerencia de la interfaz es la siguiente:



Fuente: Elaboración propia

CONSIDERACIONES

Debe utilizarse versionamiento para el desarrollo del proyecto. Se utilizará la plataforma Github en la cual se debe crear un repositorio en el que se gestionará el proyecto. Se debe realizar dos releases: el primero realizado el 26 de diciembre con nombre (v.0.2.0) y el segundo realizado antes de la fecha de entrega con nombre (v2.0.0). Además, cada estudiante debe trabajar una rama la cual tendrá de nombre *develop*(*no_carnet*). A la hora de entregar deben de unir sus cambios de la rama *develop* a la rama *main* y con ello poder generar su release. También debe de aplicar las buenas prácticas para realizar commits aprendidas en laboratorio. Para la realización del primer release es obligatorio que contenga código fuente la carpeta.

DOCUMENTACIÓN

Para que el proyecto sea calificado, el estudiante deberá entregar la documentación utilizando el formato de ensayo definido para el curso. En el caso del proyecto, el ensayo puede tener un mínimo de 4 y un máximo de 7 páginas de contenido, este máximo no incluye los apéndices o anexos donde se pueden mostrar modelos y diseños utilizados para construir la solución. Este informe debe expresar con claridad el diseño de objetos ideado para resolver este proyecto por lo que debe expresar el diagrama de clases y una tabla de los endpoints creados. Debe de tomar en cuenta que es un ensayo formal, por lo que se calificará tanto la redacción, como ortografía y presentación. Al no presentar el formato del ensayo del curso o no presenta la documentación, no tendrá derecho a nota del proyecto.

RESTRICCIONES

- Solo se permitirá la utilización de los IDEs discutidos en el laboratorio.
- La persistencia debe ser elaborada por medio de archivos XML.
- Uso obligatorio de programación orientada a objetos (POO).
- El nombre del repositorio debe de ser IPC2_ProyectoVD2024_#carnet. Trabajando dentro de la carpeta llamada "Proyecto 2".
- Agregar al auxiliar al repositorio: **rodrialeipc**. Si al día de la entrega, no se encuentra el auxiliar agregado al laboratorio, no tendrá derecho a calificación.
- Se trabajará de forma individual.
- El estudiante debe subir la documentación solicitada al repositorio para poder optar a la calificación, si entrega otro formato que no sea el ensayo del curso no se calificará el proyecto.
- Los archivos de entrada no podrán modificarse.
- Se calificará en base al release realizado previo a la fecha de entrega. No se calificará dado que se dé el caso que existan modificaciones de código en fechas posteriores a la entrega.
- Se penalizará con el 50% de la nota por cada release no realizado.
- Para dudas concernientes al proyecto se utilizarán los foros en UEDI de manera que todos los estudiantes puedan ver las preguntas y las posteriores respuestas.
- No es permitido el uso de JavaScript.
- Para el backend debe utilizarse el framework Flask mientras que para el frontend debe utilizarse Django, **NO ESTA PERMITIDO EL USO DE LIBRERIAS PARA DISEÑO DE INTERFACES GRAFICAS DE ESCRITORIO.**
- **COPIAS TOTALES O PARCIALES SERÁN REPORTADOS A LA ESCUELA Y OBTENDRÁN NOTA DE 0 PUNTOS.**
- **NO HABRÁ PRÓRROGA.**

ENTREGA

- La entrega será el lunes 30 de diciembre antes de las 23:59.
- La entrega será por medio de la UEDI.
- Se debe de entregar el link del repositorio.