UNIVERSIDAD DE TARAPACÁ FACULTAD DE INGENIERÍA INGENIERÍA CIVIL EN COMPUTACIÓN E INFORMÁTICA ARICA – CHILE

Sistema de gestión de imágenes histológicas digitalizado y moderno

Equipo de UTA: Sebastián Huidobro Rojas

José Ignacio Le Blanc Aravena

Empresa o Unidad: Facultad de Medicina

Equipo de empresa: Dr. Pedro Hecht

Curso: Proyecto IV ICCI

Profesor: Diego Aracena Pizarro

Arica, Chile
2024

Tabla de Contenidos

Índice de Tablas

Índice de imágenes

Introducción

El estudio de la histología, una disciplina fundamental en el ámbito de las ciencias de la salud, ha estado tradicionalmente limitado al análisis de muestras físicas en entornos de laboratorio. Sin embargo, los avances tecnológicos abren nuevas posibilidades para mejorar la accesibilidad y el aprovechamiento de este valioso material educativo y de investigación. En este contexto, la Facultad de Medicina UTA presenta el proyecto de digitalización de muestras histológicas, una iniciativa que busca transformar la forma en que se imparte y se desarrolla la histología en la institución.

Para ello, cuentan con un microscopio Leica DM500 binocular con cámara digital Eakins, que permite capturar imágenes en formato .jpeg en alta resolución.

Objetivos

Objetivo general

Crear un sistema digital para acceder eficientemente a muestras histológicas en la Facultad de Medicina de la Universidad de Tarapacá

Objetivo específico

- Convertir todas las muestras de histología física en imágenes digitales de alta resolución.
- Establecer una base de datos estructurada y con capacidad de búsqueda para almacenar y organizar imágenes histológicas digitales.
- Crear una interfaz intuitiva para que los usuarios vean, amplíen y analicen imágenes histológicas digitales.
- Permitir que los usuarios autorizados accedan y utilicen el repositorio de imágenes de histología digital desde cualquier lugar con conexión a Internet.
- Permitir a los usuarios agregar anotaciones, etiquetas y comentarios a imágenes de histología digital para mejorar la investigación y los fines educativos.

Restricciones

- La base de este proyecto será ejecutada en un plazo de 3 meses.
- Proyecto para el uso de la facultad de Medicina de la Universidad de Tarapacá.

Entregables

- Bitácoras para registros de reuniones, acuerdos y acciones bisemanalmente.
- Documento de requisitos firmado por el cliente.
- Informes de avance.
- Informe final.
- Manual de usuario.
- Producto final.

Definición del Proyecto

Contexto

La Facultad de Medicina de la Universidad de Tarapacá posee un banco de más de 200 muestras histológicas, fundamentales para la enseñanza e investigación en el campo de la medicina. Con el objetivo de modernizar el acceso y aprovechar al máximo este recurso, se busca implementar soluciones innovadoras que permitan una gestión eficiente y un acceso amplio a estas importantes muestras para uso educativo e investigativo en la facultad.

Problema

El problema radica en la falta de un sistema moderno y eficiente que permita gestionar y acceder de manera adecuada a las más de 200 muestras histológicas de la Facultad de Medicina de la Universidad de Tarapacá. La ausencia de un sistema digitalizado dificulta la exploración detallada de estas muestras, limitando su uso en actividades educativas y de investigación. Esto representa una barrera para el aprovechamiento pleno de este valioso recurso en el ámbito académico y científico.

Solución

La solución propuesta es desarrollar e implementar un sistema de gestión de imágenes histológicas digitalizado y moderno para la Facultad de Medicina de la Universidad de Tarapacá. Este sistema permitirá la digitalización, organización y acceso eficiente a las más de 200 muestras histológicas disponibles. Además, incluirá funcionalidades como etiquetado de imágenes, niveles de permisos para usuarios y acceso remoto desde diferentes dispositivos, facilitando así su uso en actividades educativas y de investigación. Con esta solución, se busca iniciar la digitalización, mejorar la utilidad y disponibilidad de las muestras histológicas, contribuyendo al avance académico y científico en el campo de la medicina en la facultad.

Requisitos del proyecto

Requisitos funcionales

ID	Requisito funcional	Descripción
RF-1	Almacenamiento confiable de las imágenes histológicas	Se utilizarán tecnologías de almacenamiento confiables para garantizar la disponibilidad de las imágenes.
RF-2	Gestión de Usuarios y Permisos	Se deben administrar usuarios con distintos permisos desde “superusuarios”, administradores, ayudantes y alumnos.
RF-3	Búsqueda y categorías	Ofrecer opciones de búsqueda avanzada para encontrar rápidamente imágenes específicas, además de permitir el uso de filtros para ordenar y visualizar imágenes según criterios específicos.
RF-4	Posibilidad de modelar y etiquetar las imágenes	Permitir a los usuarios, específicamente a superusuarios, administradores y ayudantes, agregar información adicional a las imágenes histopatológicas. Dando la posibilidad a futuro de que cada usuario tenga sus propias etiquetas.
RF-5	Compatibilidad con diferentes plataformas	El sistema se podrá acceder desde diferentes dispositivos, como computadoras de escritorio, laptops, tablets y smartphones.
RF-6	Autenticación y control de acceso	Los usuarios deberán autenticarse en el sistema utilizando un nombre de usuario y una contraseña.
RF-7	Seguridad	El sistema debe protegerse contra Accesos no autorizados y datos confidenciales.
RF-8	Usabilidad	El sistema debe ser fácil de usar, aprender y comprender para los usuarios.

Tabla 1. Requisitos funcionales

Requisitos no funcionales

ID	Requisito no funcional	Descripción
RnF-1	Calidad	El sistema debe tener las imágenes y video con una alta fidelidad.
RnF-2	Compatibilidad	El sistema tiene que tener la capacidad de visualizarse correctamente y realizar sus operaciones correspondientes en SmartPhones y en PCs (Independiente del sistema operativo).
RnF-3	Confiabilidad	El sistema debe ser confiable y funcionar sin fallos ni errores durante un período de tiempo específico.
RnF-4	Escalabilidad	El sistema debe poder adaptarse a un aumento en el número de imágenes, video y/o usuarios.
RnF-5	Mantenibilidad	El sistema debe ser fácil de modificar, actualizar y corregir errores.

Tabla 2. Requisitos no funcionales

Metodología de trabajo

Para este proyecto se utilizará la mitología ágil bajo el marco de trabajo SCRUM, pero con grandes modificaciones, teniendo en cuenta que solo hay dos integrantes en el equipo.

Marco de trabajo SCRUM

El marco de trabajo SCRUM trabaja en iteraciones llamadas sprint, el cual debe durar entre 1 a 4 semanas (bisemanal para este proyecto) durante el cual se desarrolla un incremento, el cual debe dar como resultado un prototipo funcional.

Artefactos de SCRUM

SCRUM utiliza 3 artefactos importantes:

- Product backlog: es la lista de las historias de usuarios ordenadas por importancia, dicho de otro modo es la lista de cosas por desarrollar.
- Sprint backlog: es la lista de historias de usuarios que se desarrollarán en un sprint en específico, o sea es la lista de tareas que se entregarán en el incremento.
- Incremento: es la suma de todas las historias de usuarios completadas en el sprint.

Eventos de SCRUM

SCRUM tiene varios eventos que se consideran imprescindibles:

- Planificación de la iteración (Sprint Planning): En este evento que se da al inicio del sprint se crea el sprint backlog.
- Ejecución de la iteración (Sprint)
- Reunión diaria de sincronización del equipo (Scrum Daily Meeting): Es una reunión diaria caracterizada por ser de corta duración donde los integrantes del equipo informan respondiendo a estas 3 preguntas, ¿Qué hiciste ayer? ¿Qué harás hoy? ¿Hay algún impedimento?, por motivos de que no se trabaja todos los días en el proyecto, esto se limitará a las sesiones en clases y a las reuniones para avanzar sincrónicamente.
- Demostración de los requisitos completados (Sprint Review): Es la presentación y entrega del prototipo hecho en el sprint.
- Retrospectiva (Sprint Retrospective): Es una actividad donde se registran las experiencias del equipo al final del sprint, se responde a las preguntas¿Qué

salió bien en el último sprint? ¿Qué salió mal? ¿Qué aprendimos? ¿Qué deberíamos hacer diferente la próxima vez?

Roles SCRUM

En scrum hay 3 roles imprescindibles los cuales conforman al Scrum Team:

- Scrum master: Se encarga de que se ejecuten los eventos de SCRUM como es debido.
- Product owner: Se encarga de que se prioricen los intereses de los stakeholders
- Developer: Se encarga de completar las historias de usuarios, no solo se limita a desarrolladores de software sino a todos aquellos que aportan en el desarrollo del proyecto.

Además, existen los stakeholders, que son aquellas personas con interés o a los cuales se les impacta con el proyecto.

Creación del product backlog

Para trabajar con el SCRUM es necesario definir historias de usuarios, las cuales son una representación de los requerimientos, pero teniendo como eje principal a los usuarios del producto, para crear las historias de usuarios se utilizan una serie de herramientas

Impact mapping

El impact mapping es una técnica que nos permite ver de manera visual los impactos que se quieren generar en los usuarios.
El impact mapping tiene cuatro partes:

- La primera es la meta, la cual es el objetivo del proyecto y tiene que responder a la pregunta ¿Para qué es el proyecto?
- La siguiente sección son los actores, es a quien va a impactar en el proyecto, por lo que tiene que responder a la pregunta ¿A quiénes impacta?
- La tercera sección son los impactos y son las maneras en cómo el proyecto va a impactar a los actores y cómo estos impactan o ayudan en nuestro proyecto, tiene que responder al ¿Cómo impactará el proyecto al usuario? Y viceversa.
- La última sección del impact mapping es el entregable y es el que haremos para lograr el impacto, por lo que debe responder a ¿Qué podemos hacer y entregar para lograr el impacto deseado en los actores? Esta última sección es importante dado que los entregables se utilizan como épicas del user story map.

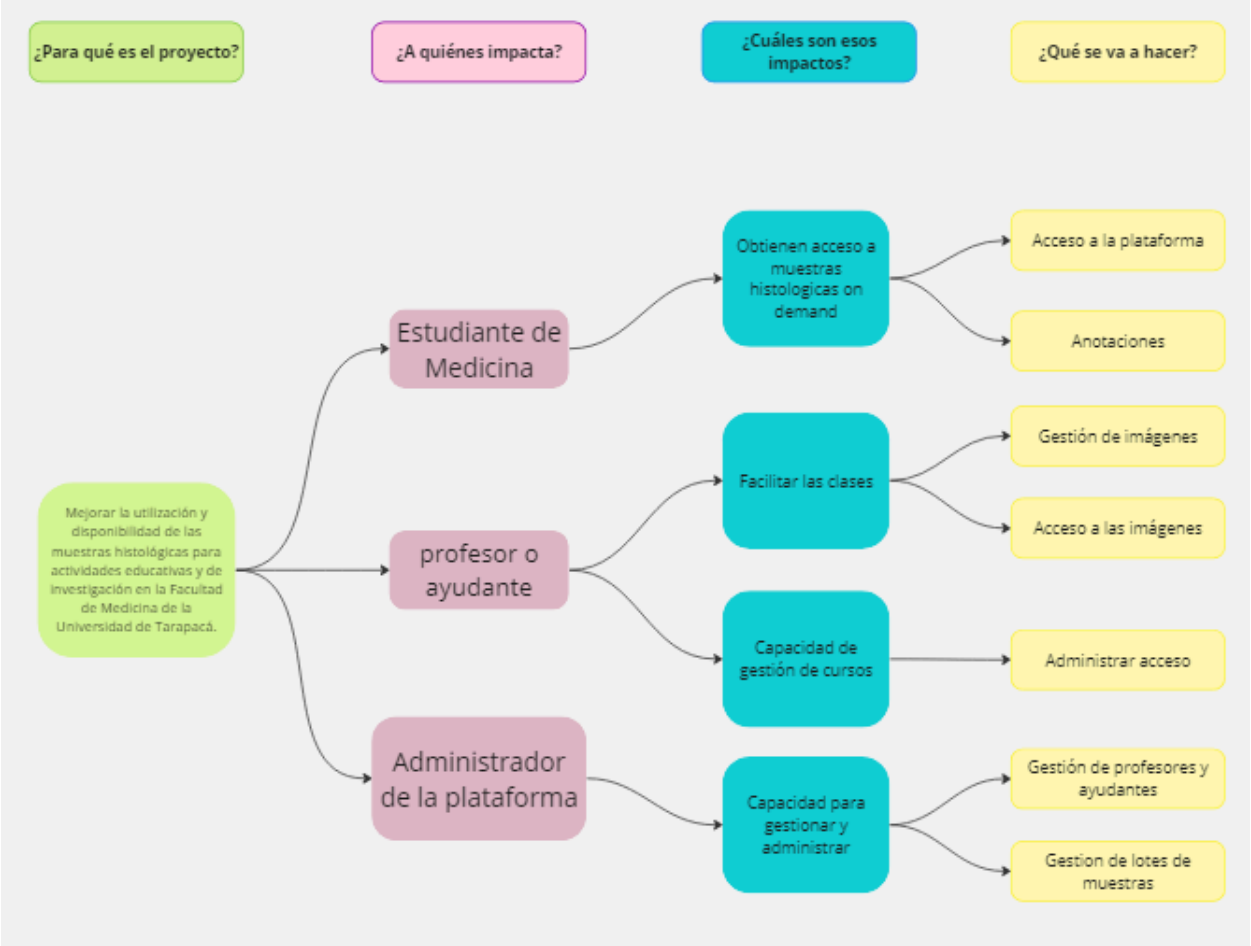

Imagen 1. Impact mapping

User story mapping

El user story map es una técnica que nos permite desglosar las épicas en historias de usuario a partir de las épicas
En la parte superior se colocan las épicas (verde oscuro en la imagen 2), las cuales obtuvimos como resultado del impact mapping
debajo de ellas se colocan historias de usuario de alto nivel (verde claro en la imagen 2), las cuales son demasiado generales aún por lo que se desglosan hacia abajo las distintas historias de usuario específicas (color crema en la imagen 2) aunque están aún no están bien redactadas para ser consideradas historias de usuarios.

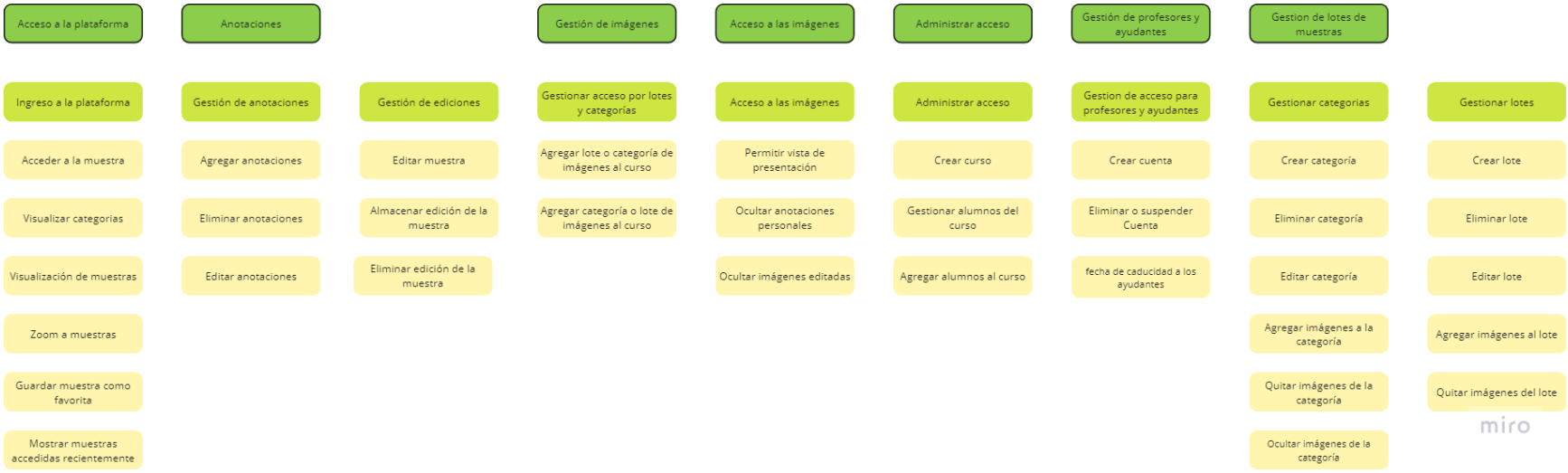

Imagen 2. User story mapping

MOSCOW

La técnica MOSCOW nos permite ordenar las historias de usuarios por importancia en el proyecto, es importante que esta actividad se lleve a cabo con el cliente del proyecto, consiste en una grilla de 4 áreas donde se deben poner las historias de usuario, las cuales representan cuatro niveles de importancia:

- Must have (debe tener): Son aquellas historias de usuario más importantes sin las cuales el proyecto se considera incompleto.
- Should have (debería tener): Son aspectos críticos pero no imprescindibles
- Could have (podría tener): Son aspectos que estaría bien tener, pero no son críticas.
- Won't have (no tendrá): Son características que no merecen la inversión y que no aportan beneficio en este momento, se podrían considerar más adelante.

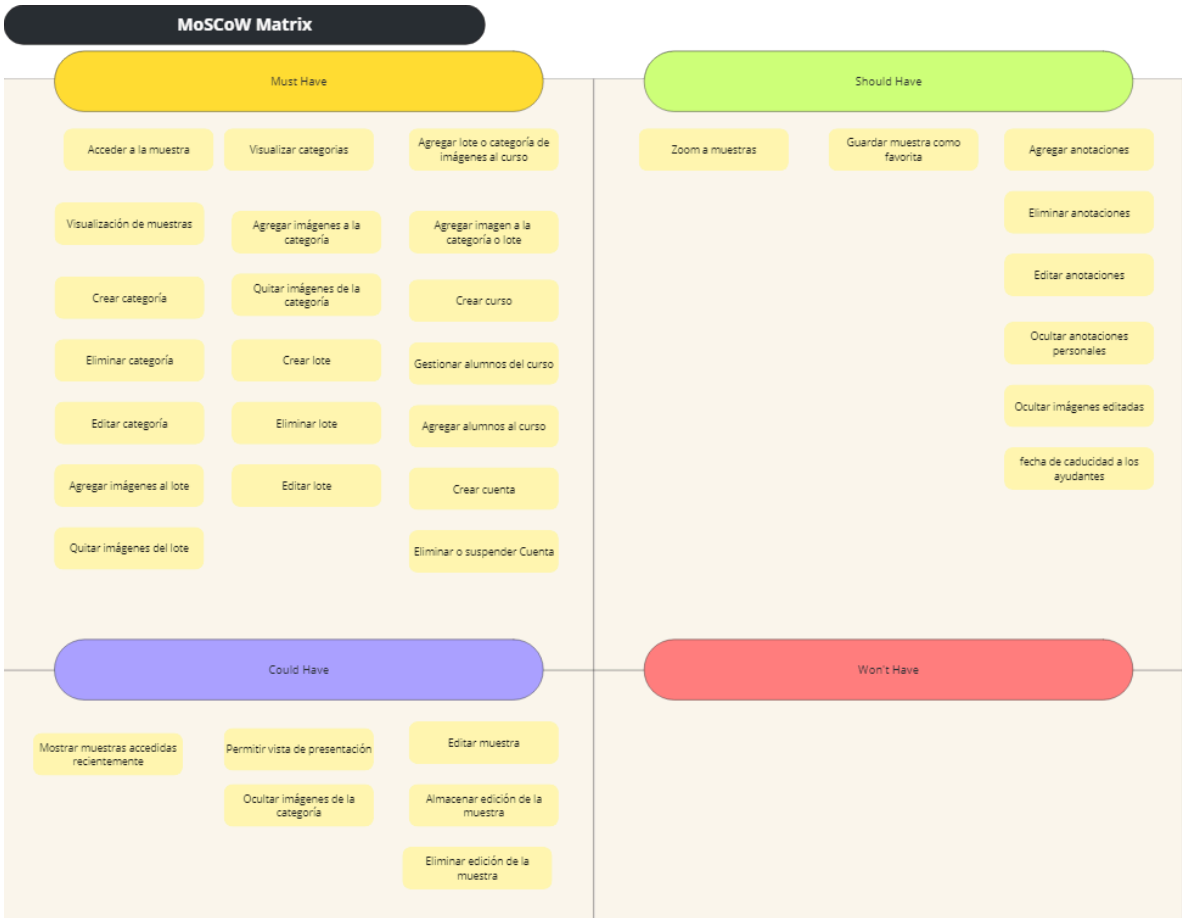

Imagen 3. Aplicación de MOSCOW

Product backlog

Teniendo ordenadas las prioridades se puede volver a ordenar en el user story map y representamos la importancia por colores, de más importante a menos importante naranja, verde y azul.
Ahora el user story map ordenado es nuestro product backlog.

Imagen 4. Product backlog

Carta Gantt

Como se puede observar en la imagen 4 se incluyeron los sprints para subdividir la carta Gantt.
Dado el marco de trabajo donde las actividades de un sprint se especifican en el sprint planning es difícil resolver una carta Gantt complemente.

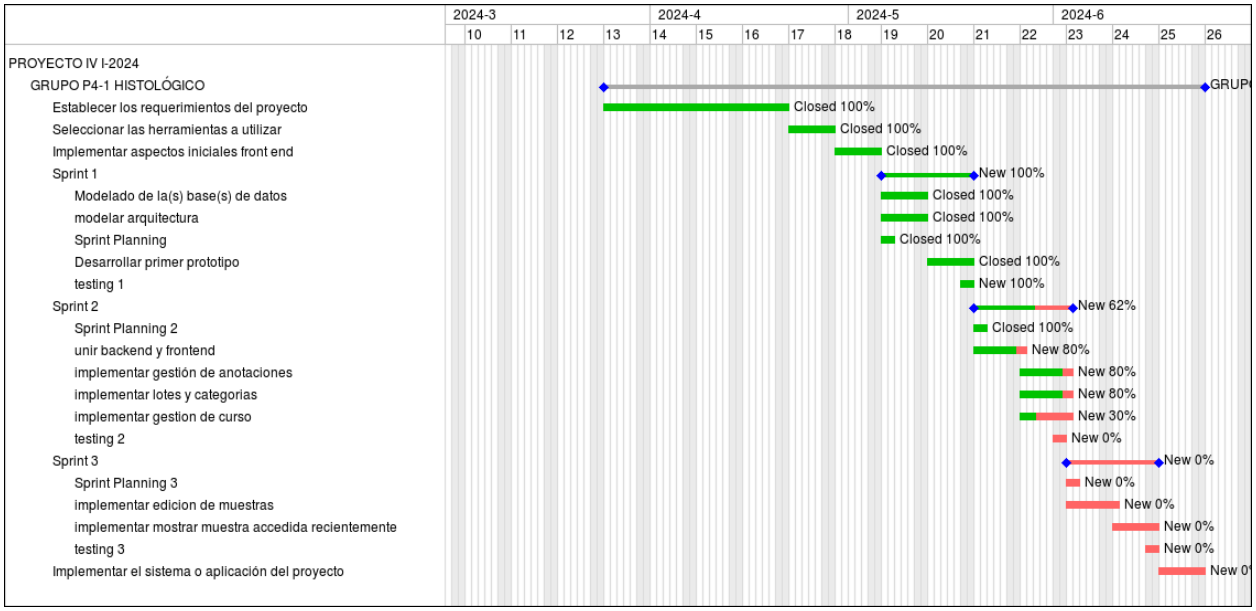

Imagen 5. Carta Gantt

Herramientas a utilizar

Herramienta	Descripción
Docker	Plataforma de virtualización de código abierto que permite crear, empaquetar y ejecutar aplicaciones en contenedores aislados.
MySQL	Sistema de gestión de bases de datos relacionales. Es reconocido por su facilidad de uso, escalabilidad y confiabilidad
Nginx	Servidor web que se encargará de guardar las imágenes y servir las imágenes.
Node.js	Entorno de ejecución de código abierto y multiplataforma para el lenguaje de programación JavaScript. Está diseñado para ejecutar código JavaScript del lado del servidor
Express.js	Marco web minimalista y flexible para Node.js que facilita la creación de aplicaciones web y APIs RESTful.
Angular	Es uno de los marcos web más populares para crear aplicaciones web modernas, conocido por su estructura modular, componentes reutilizables y enfoque en la creación de aplicaciones web de una sola página.
SQLite	Sistema de base de datos local que se utilizará durante la etapa de desarrollo.
Django	Framework web para python que puede trabajar tanto de frontend como de backend.

Tabla 3. Herramientas a utilizar

Diseño del proyecto

Diagrama de casos de uso

Se tomaron las historias de usuario de alto nivel como base para crear el diagrama de casos de uso.

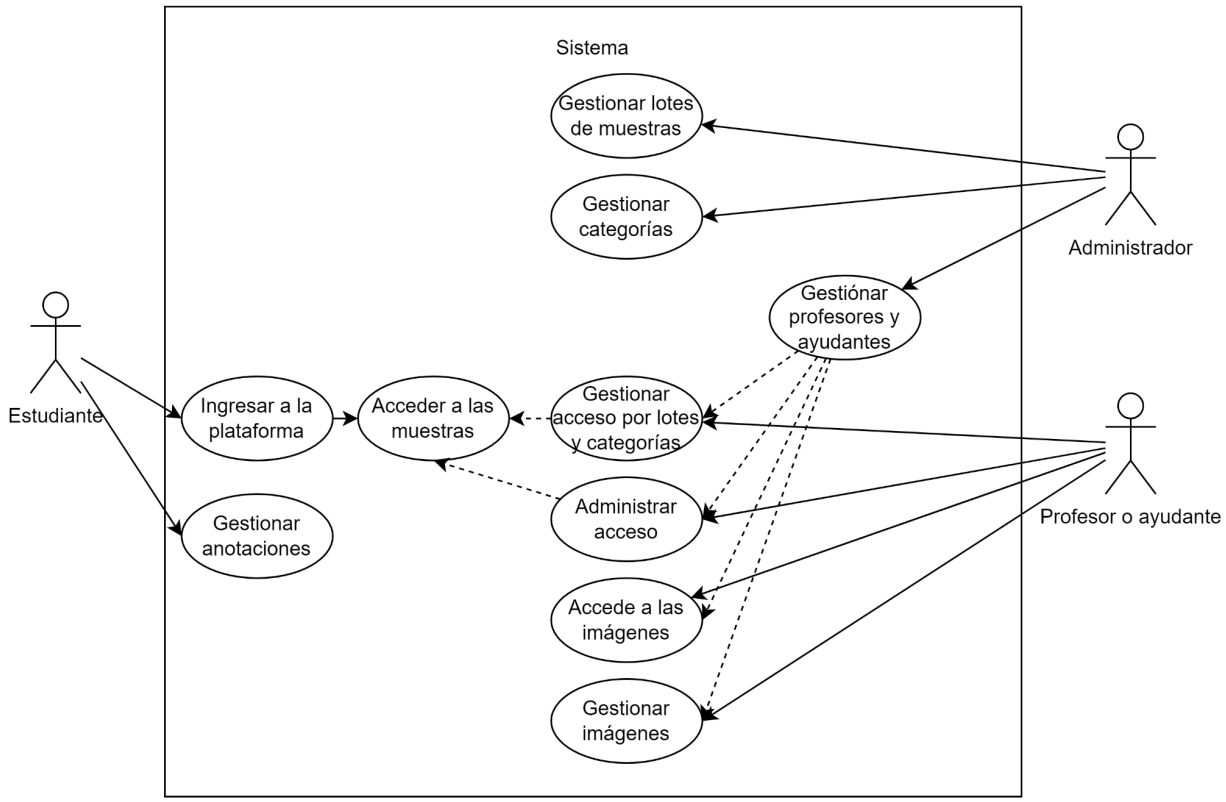

Imagen 6. Diagrama de casos de uso

ID	Caso de uso	Actor	Definición
CU1	Ingresar a la plataforma	Estudiante	El estudiante ingresa a la plataforma
CU2	Gestionar anotaciones	Estudiante	El estudiante gestiona sus anotaciones personales
CU3	Acceder a las muestras	Estudiante	Si está autorizado el estudiante puede acceder a las muestras
CU4	Gestionar lotes de muestras	Administrador	El administrador gestiona las muestras
CU5	Gestionar categorías	Administrador	El administrador gestiona las categorías de las muestras histológicas
CU6	Gestionar profesores y ayudantes	Administrador	El administrador puede gestionar las cuentas de los profesores y ayudantes, esto incluye crear cuentas.
CU7	Gestionar acceso por lotes y categorías	Profesor o ayudante	Los profesores y ayudantes pueden permitir el acceso de los estudiantes a las distintas categorías y/o lotes
CU8	Administrar acceso	Profesor o ayudante	Los profesores y ayudantes pueden gestionar qué estudiantes tienen acceso a la plataforma
CU9	Accede a las imágenes	Profesor o ayudante	Los profesores y ayudantes pueden acceder a las muestras y utilizarlas en clases y ayudantías
CU10	Gestionar imágenes	Profesor o ayudante	Los profesores deben poder agregar nuevas muestras, así como gestionarlas

Tabla 4. Caso de uso

Diagrama de contexto

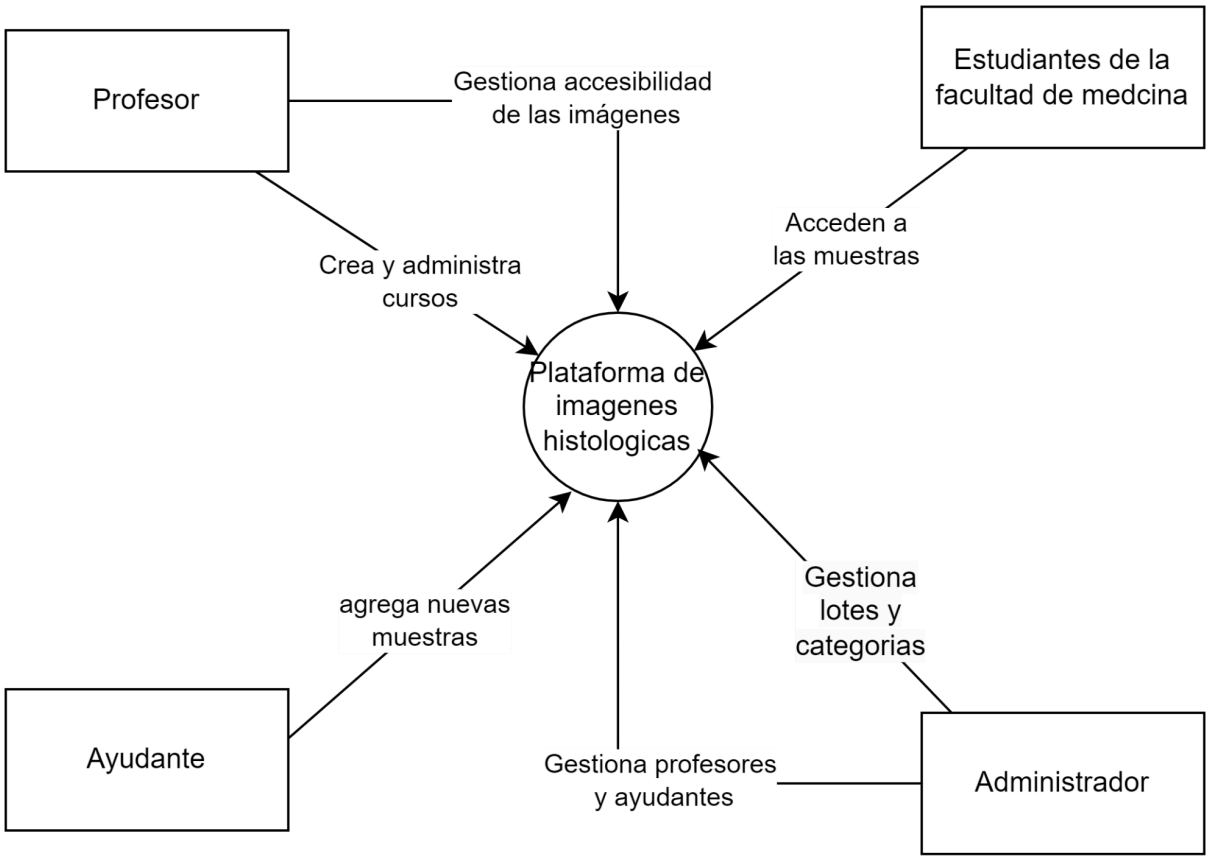

Imagen 7. Diagrama de contexto

Diagramas de flujo

En este anexo se mostrarán los eventos que ocurrirán en el sistema en forma de diagrama.

Iniciar sesión

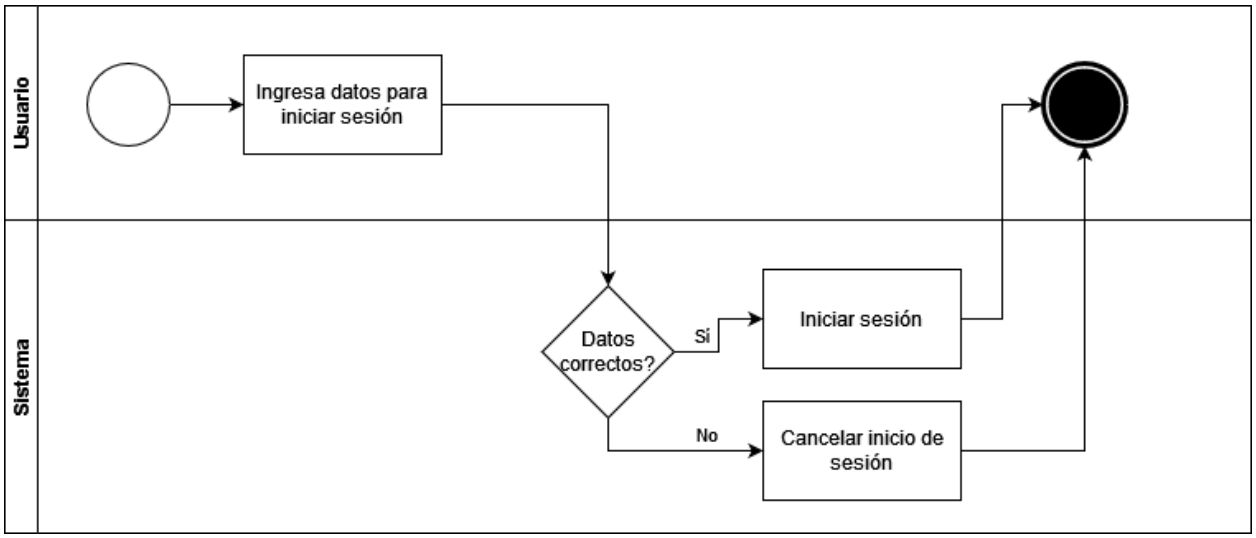

Imagen 8. Iniciar sesión

Visualizar Tejido

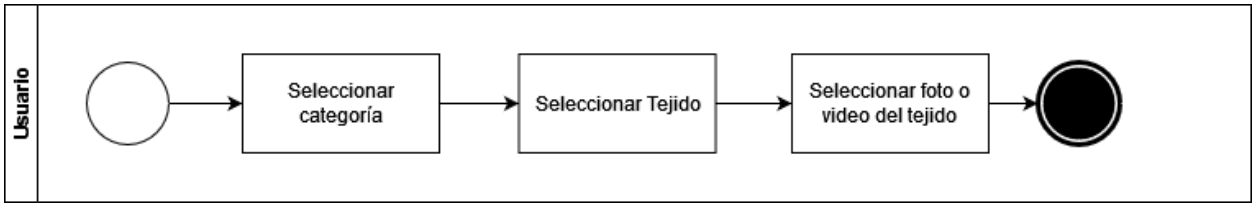

Imagen 9. Visualizar tejido

Gestión de usuarios

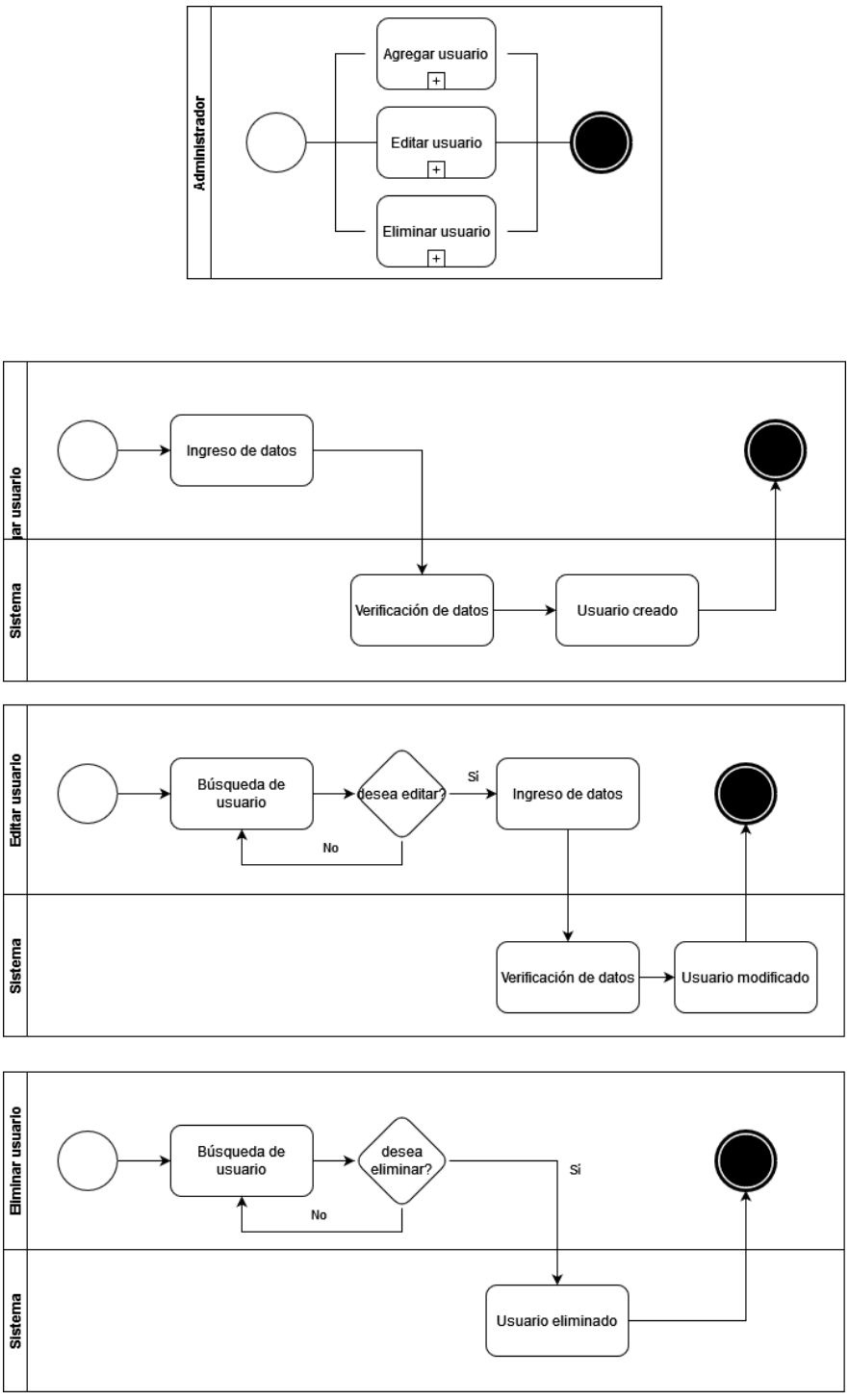

Imagen 10. Gestionar usuarios

Gestión de notas

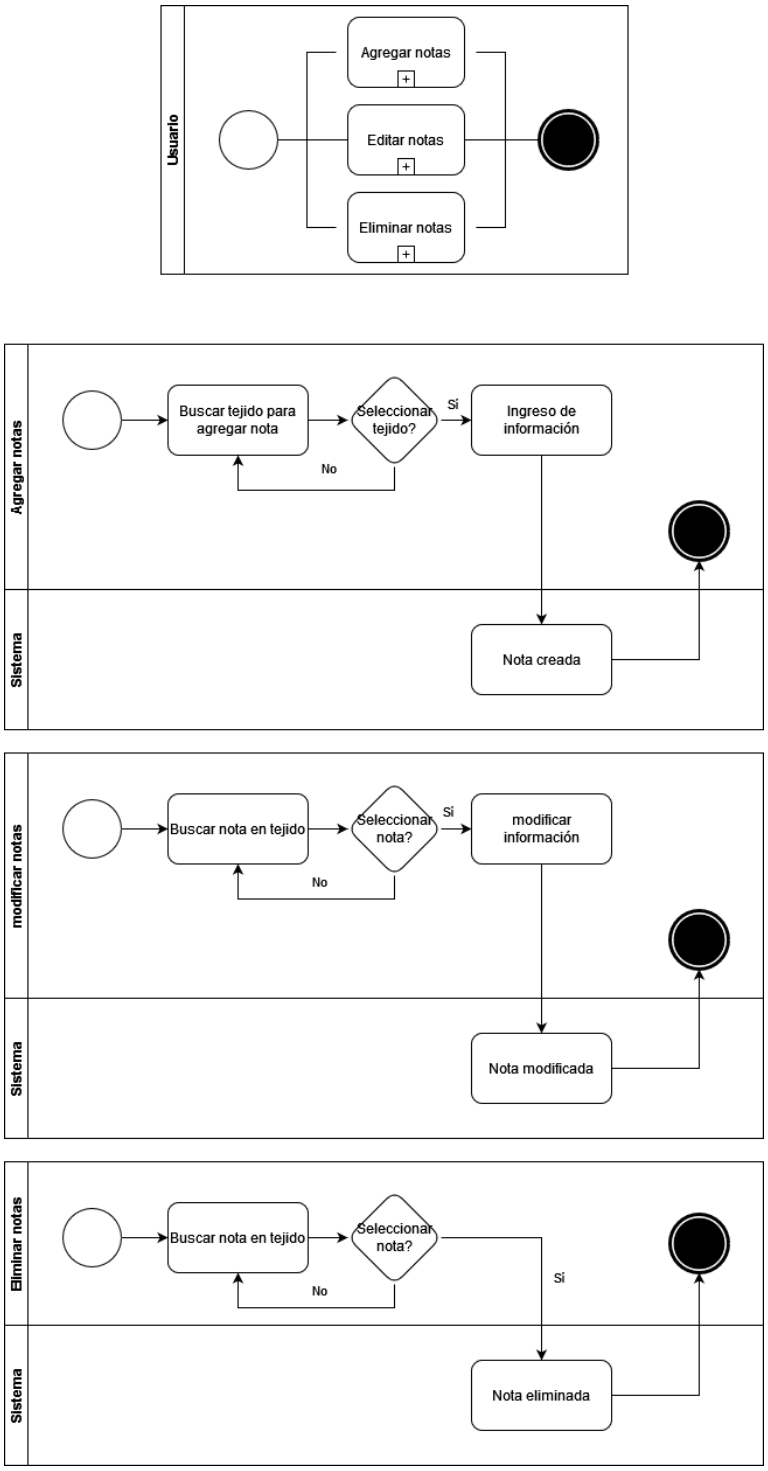

Imagen 11. Gestión de notas

Arquitectura del sistema

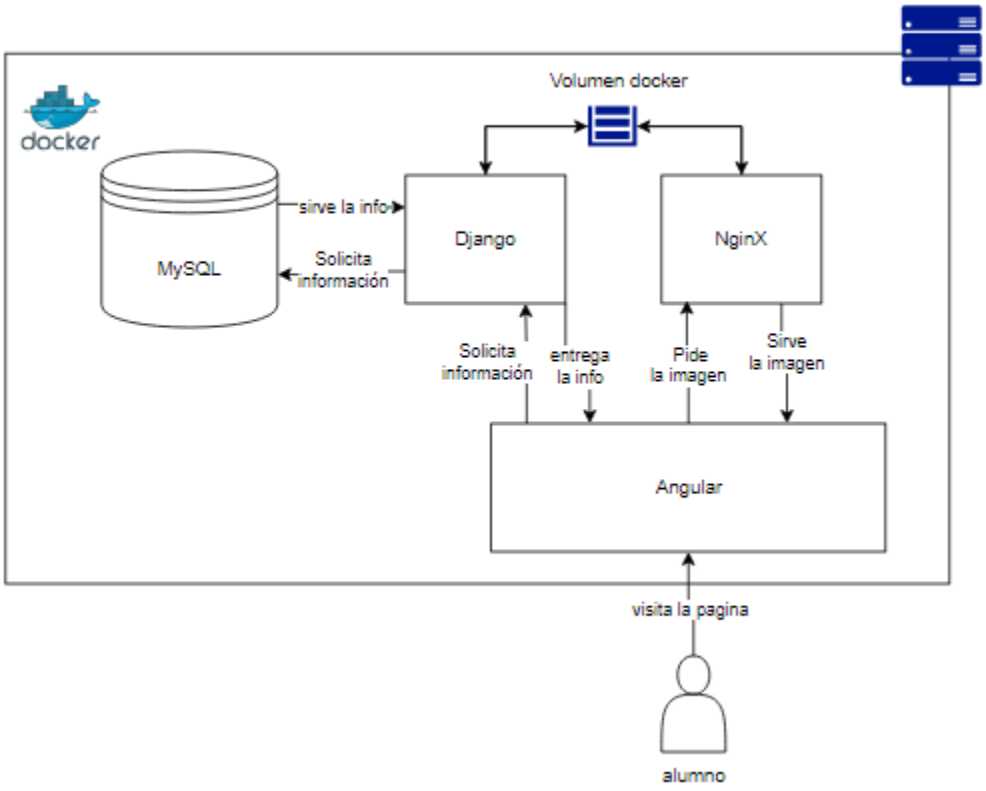

Imagen 12. Arquitectura del sistema

Como se ve en la imagen 12, el sistema está dividido en 4 subsistemas MySQL encargada de almacenar la información de los usuarios, así como la información relacionada con las muestras y capturas, las imágenes se guardan en un volumen Docker el cual nos da resistencia de datos y son servidas a través del servidor de Nginx y administradas por Django por último se tiene Angular para servir el front-end

Modelo de la base de datos

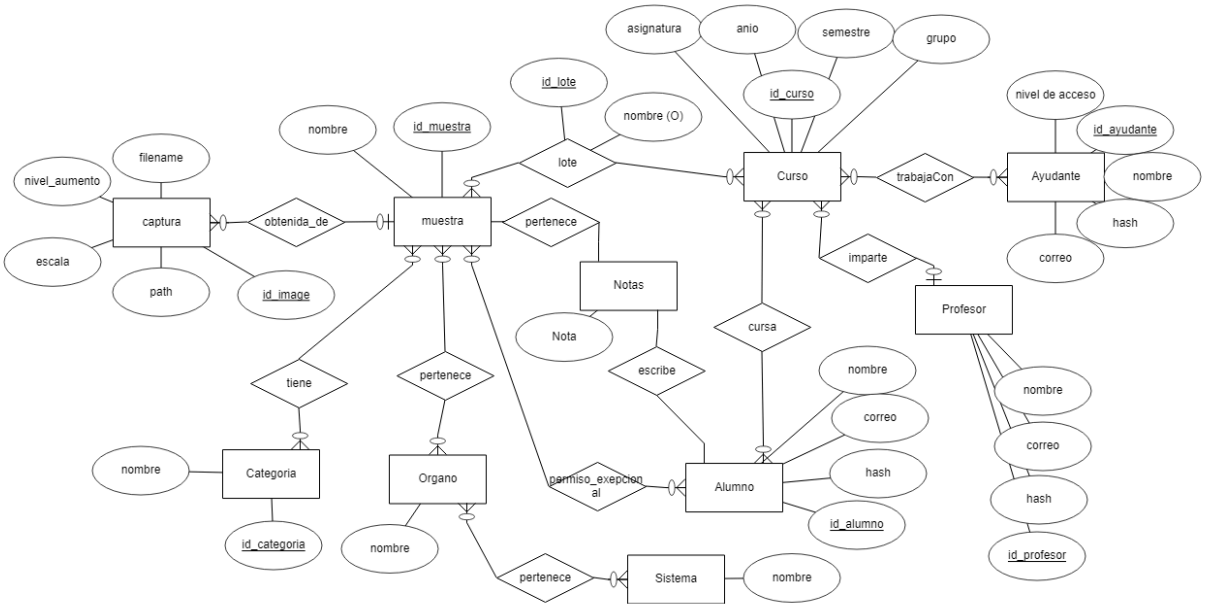

Imagen 13. Modelo base de datos

Desarrollo del Backend

Nginx

La implementación de nginx como servidor web para alojar las imágenes del proyecto se utilizó una imagen de docker de nginx donde se cargan los ficheros de configuración

```
FROM nginx
COPY nginx/nginx.conf /etc/nginx/nginx.conf
COPY nginx/conf.d /etc/nginx/conf.d
```

El archivo de configuración general es nginx.conf y en conf.d están las configuraciones de cómo trata nginx las configuraciones.

```
server {
    listen       80;
    listen  [::]:80;
    server_name  localhost;
    location / {
        root   /usr/share/nginx/html;
        index  index.html index.htm;
    }
    location /images/ {
        root /usr/share/nginx/html;
        autoindex on;
        location ~* \.(jpg|jpeg|png|gif|ico|bmp|svg)$ {
            expires 30d;
            add_header Cache-Control "public, no-transform";
        }
    }
    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
        root   /usr/share/nginx/html;
    }
}
```

Aquí se expone el directorio de /usr/share/nginx/html a través de la dirección /images/

Django

Para la implementación de django en docker se utiliza una imagen de alpine para python como base:

```
FROM python:3.10.14-alpine3.20
ENV PYTHONUNBUFFERED=1
WORKDIR /app
RUN  apk update \
    && apk add --no-cache gcc musl-dev postgresql-dev python3-dev
libffi-dev \
    && pip install --upgrade pip
COPY ./requirements.txt ./
RUN pip install -r requirements.txt
COPY ./ ./
CMD ["python","manage.py","runserver","0.0.0.0:8000"]
```

En el contenedor de docker se crea un directorio llamado /app, se instalan librerías necesarios, se copia e instala con pip el archivo requirements.txt del proyecto de django luego se copian todos los archivos del proyecto menos las librerías de python las cuales se instalaron con él requirements, por último en el container de docker y se ejecuta el servidor django
Se utilizó el ORM de Django el cual es una herramienta que permite interactuar con bases de datos utilizando objetos de Python en lugar de escribir directamente consultas SQL.

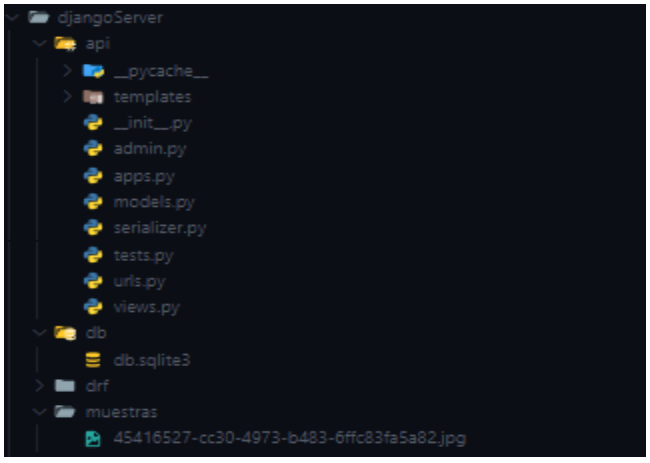

Imagen 14. Archivos django

Se utilizó Django REST framework para desarrollar la api

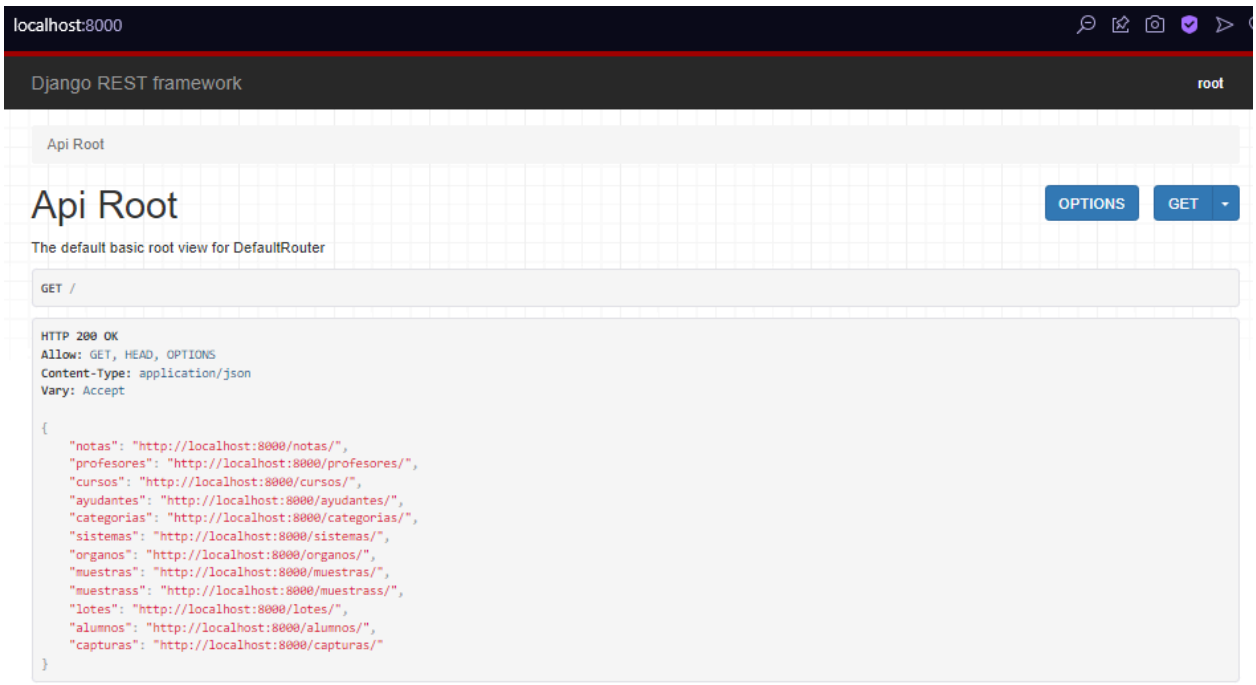

Imagen 15. API del proyecto

Además se configuraron peticiones específicas para la implementación en el frontend

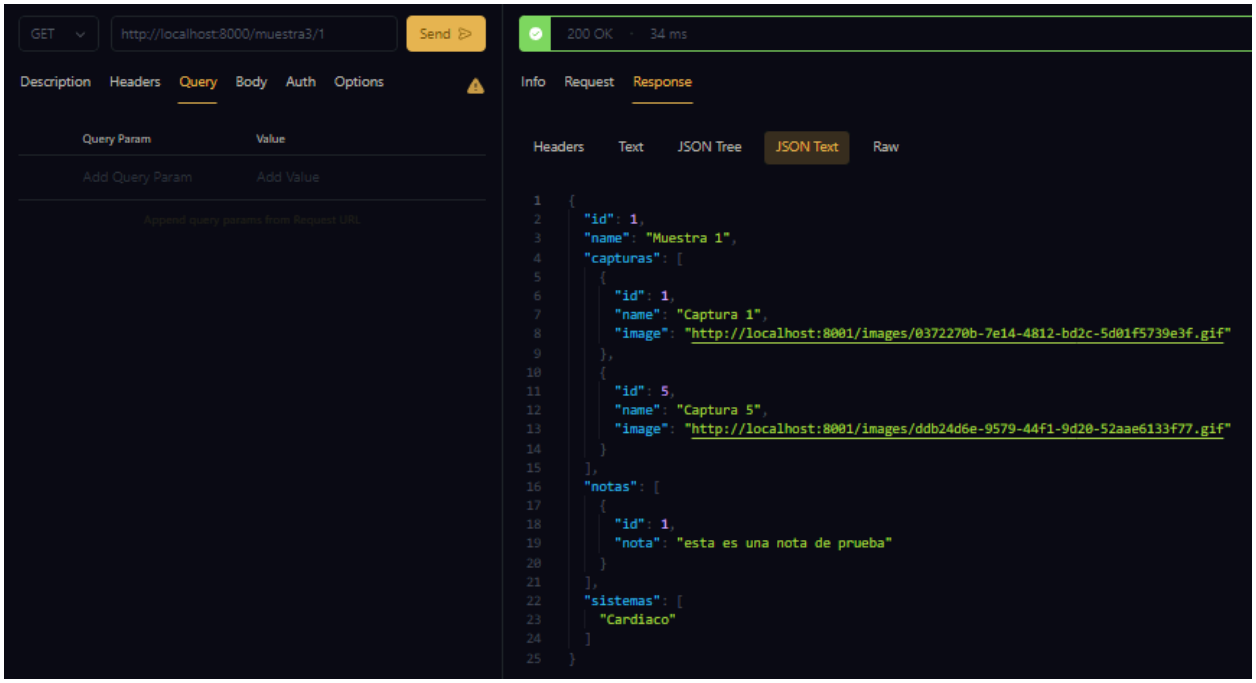

Imagen 16. Petición personalizada

Django proporciona Django administration la cual es una interfaz para hacer CRUD en la base de datos

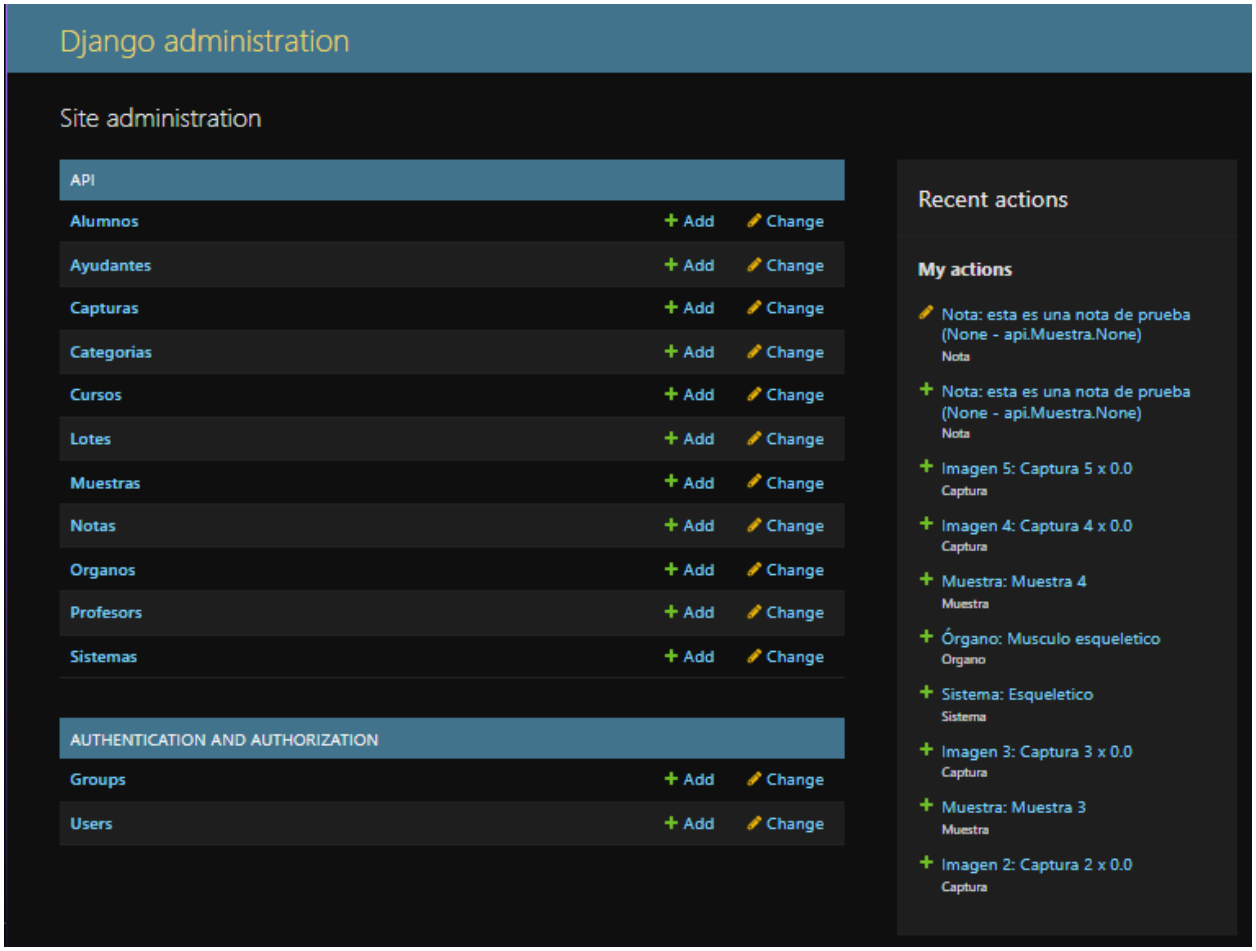

Imagen 17.Django administration

Integración con el frontend

La integración con el frontend de angular se puede realizar a través de de un servicio y configurando el CORS de django para permitir la conexion.

```
export class ApiService {
  private apiUrl = 'http://localhost:8000';

  constructor(private http: HttpClient,) { }
  getCategorias(): Observable<any[]>{
    return this.http.get<any[]>(`${this.apiUrl}/categorias`);
  }
  getTejidos(category: string): Observable<Tejido[]> {
    if (category === 'all') {
      return
this.http.get<Tejido[]>(`${this.apiUrl}/muestras/por_categoria/?category=a
ll`);
    } else {
      return
this.http.get<Tejido[]>(`${this.apiUrl}/muestras/por_categoria/?category=$
{category}`);
    }
  }
  getTejido(id: number): Observable<Muestra> {
    return this.http.get<Muestra>(`${this.apiUrl}/muestra3/${id}`);
  }
}
```

Visualización del Front-End

Se realizó en balsamiq algunos bocetos de como se verá el sistema una vez se encuentre implementado.
A continuación se mostrarán algunas secciones que se realizaron:

En la figura, se muestra como sería el inicio de sesión donde los usuarios deberán ingresar sus datos para validar sus credenciales.

Imagen 18. Login

Si el inicio de sesión es exitoso, se redirigirá a la siguiente pantalla. La pantalla consiste en mostrar las categorías de los tejidos en el lado izquierdo, el cual poseerá cada una con una cierta cantidad de tejidos almacenados para la visualización.

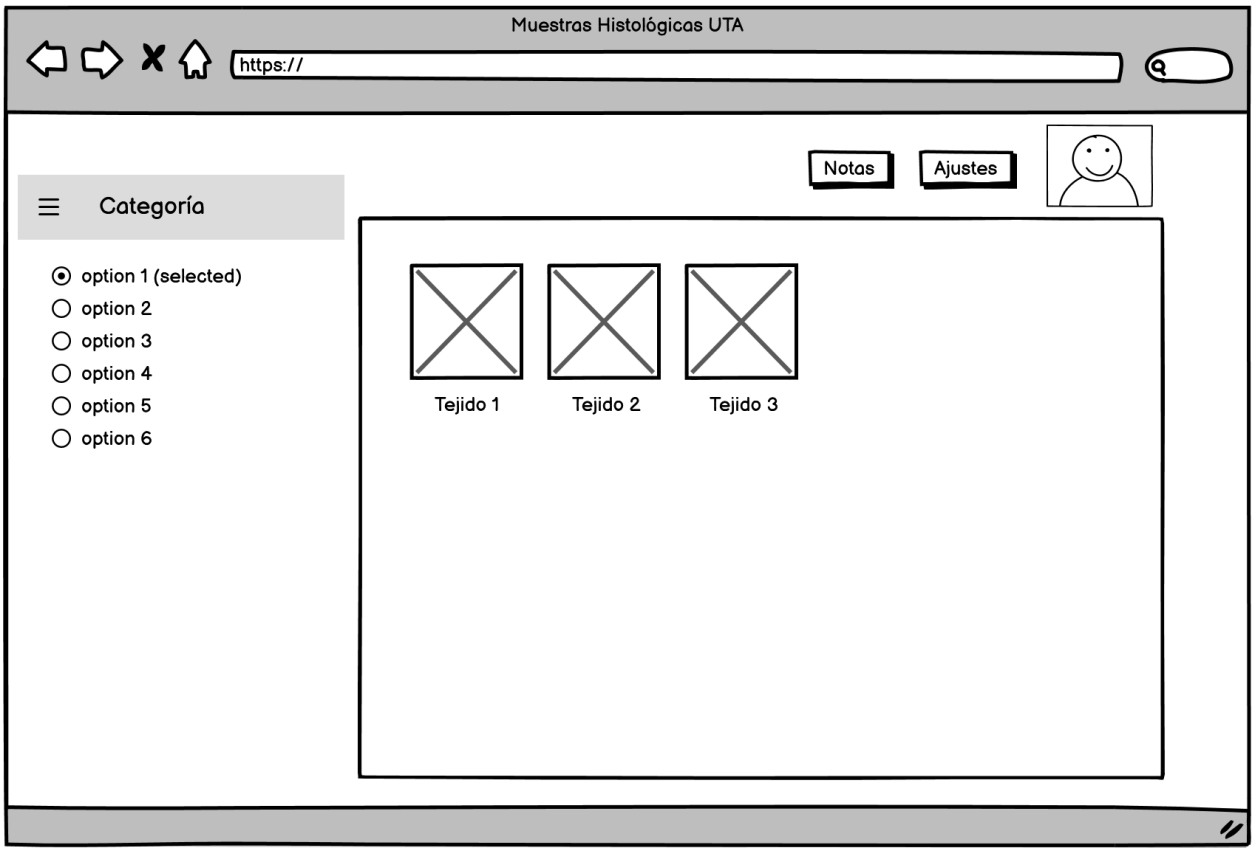

Imagen 19. Pantalla principal

Posteriormente, si se oprime algún tejido, se derivará a la siguiente pantalla donde se mostrará las fotos y/o videos del tejido, el cual cada usuario podrá colocar sus propias notas respectivas para su uso personal que no afectará al contenido original.

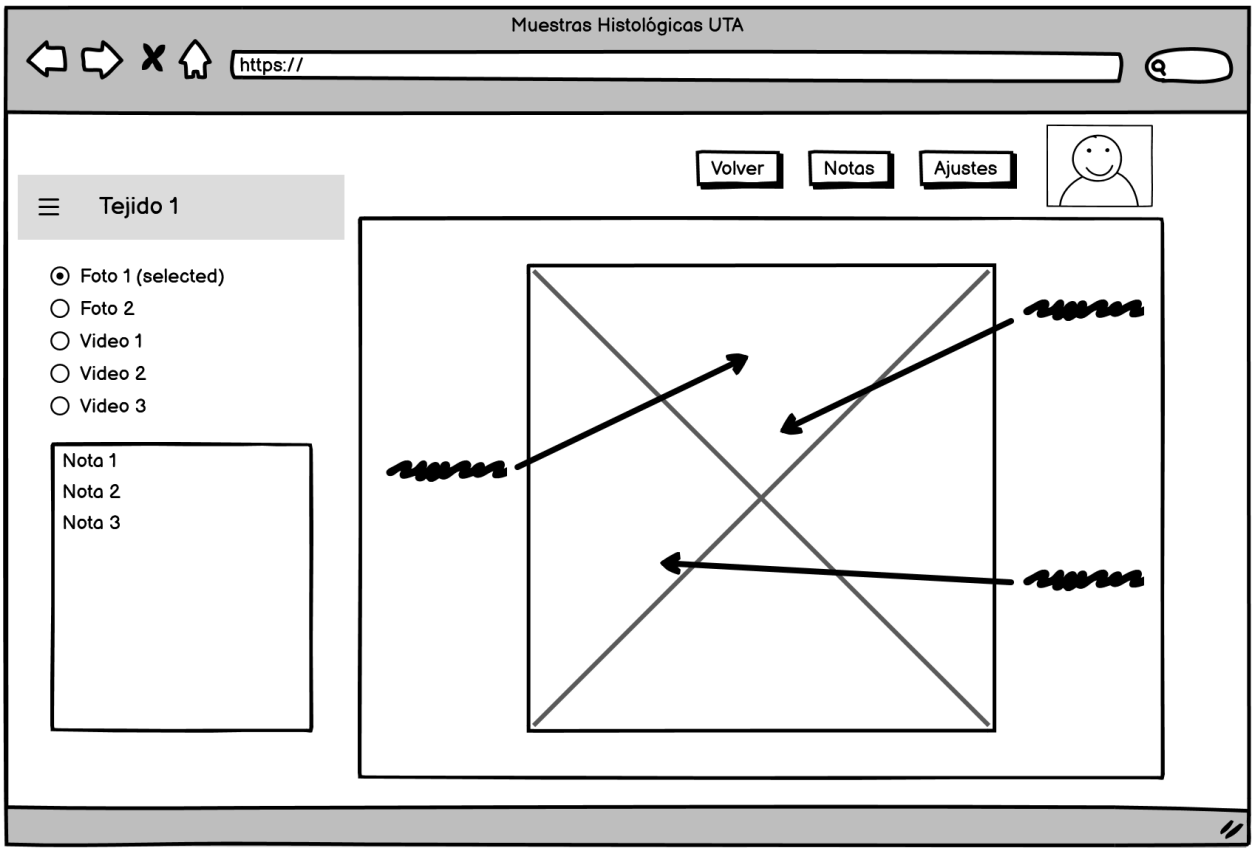

Imagen 20. Ver tejido

Adicionalmente, están las pantallas de ajustes del usuario (como datos personales, ver si dispone de superusuario o no, etc.) y una visualización de todas las notas personales hechas ordenadas por categoría principalmente.

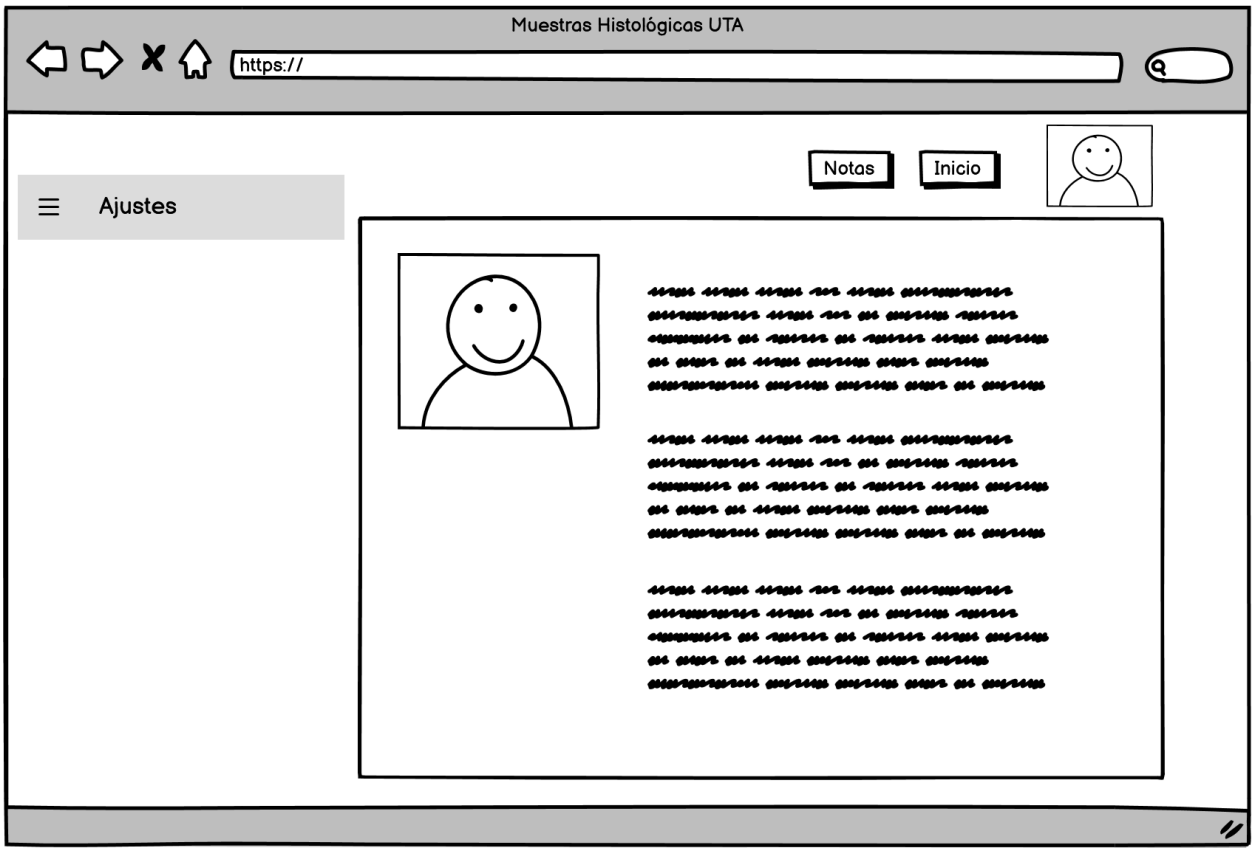

Imagen 17. Datos personales

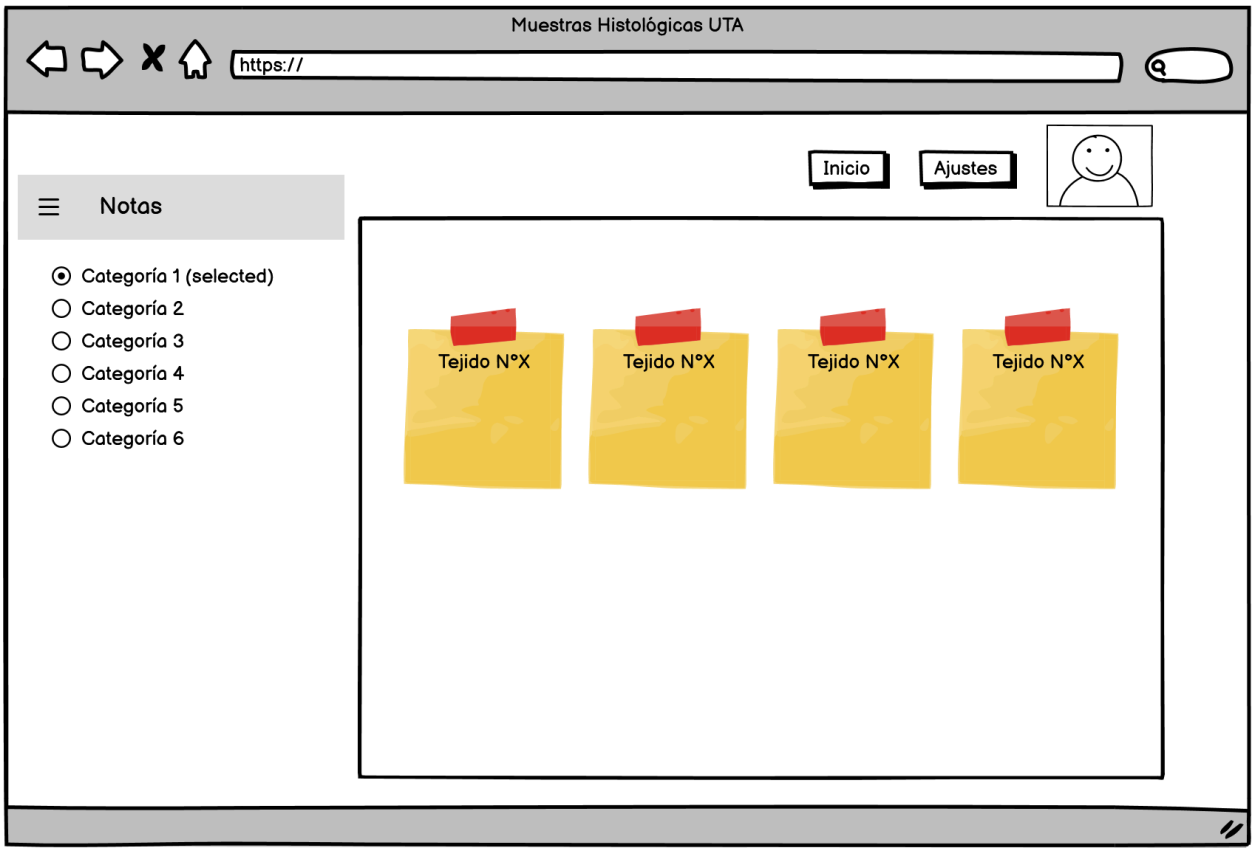

Imagen 18. Visualizar notas

Para poder hacer el FrontEnd, inicializamos un proyecto de angular y procedemos a agregar nuevas carpetas que reflejan cada ventana del sistema

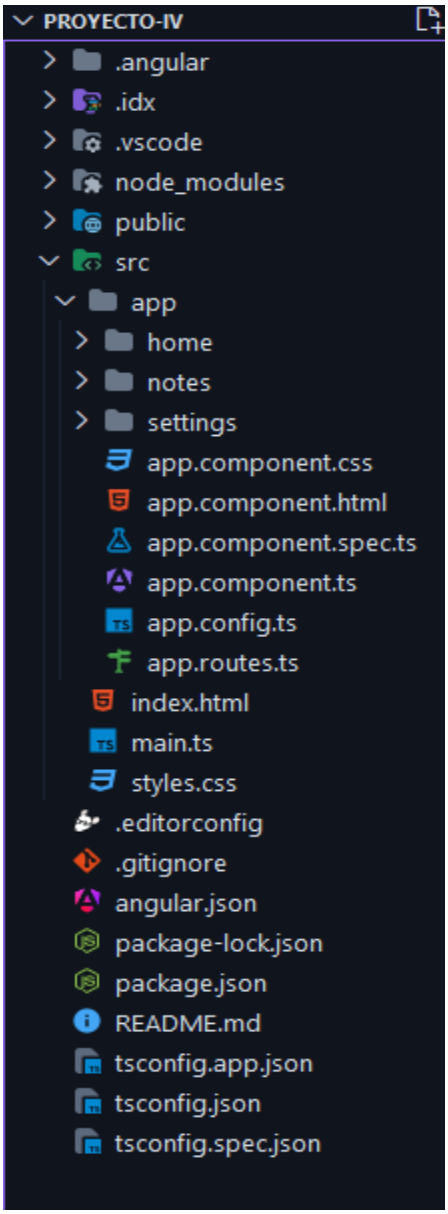

Imagen 18. Visualizar notas

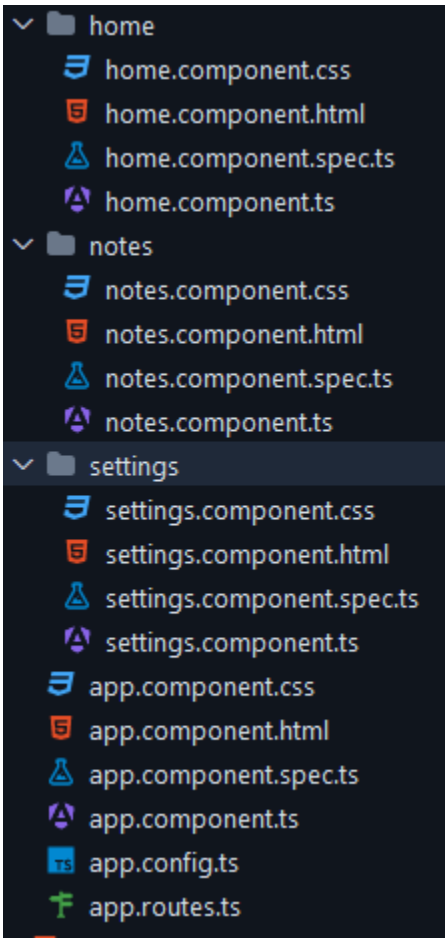

Imagen 19. Framework Angular

En el archivo app.component.html alberga el header y footer de todos los componentes que serán navegados del proyecto.

```
<header>
  <div class="nombre">
    <h1 class="titulo">Muestras Histológicas UTA</h1>
    <img src="favicon.png" alt="">
  </div>
  <div>
    <button routerLink="">Inicio</button>
    <button routerLink="/notes">Notas</button>
    <button routerLink="/settings">Perfil</button>
    <img
src="https://w7.pngwing.com/pngs/196/91/png-transparent-balsamiq-mockup-website
```

```
-wireframe-computer-icons-user-interface-mockups-logo-white-face-user-interface
-design-thumbnail.png" alt="">
  </div>
</header>

<router-outlet class="router-outlet"></router-outlet>

<footer>
  <h3>Facultad de Medicina</h3>

</footer>
```

El archivo llamado app.routes.ts utiliza el router-outlet del archivo app.component.html para poder navegar entre los diferentes componentes que se crean en el proyecto. Estas rutas se conectarán con la importación del componente, como en este caso home, settings y notes.

```
import { Routes } from '@angular/router';
import { HomeComponent } from './home/home.component';
import { SettingsComponent } from './settings/settings.component';
import { NotesComponent } from './notes/notes.component';

export const routes: Routes = [
    { path: '', component: HomeComponent },
    { path: 'settings', component: SettingsComponent },
    { path: 'notes', component: NotesComponent }
];
```

Vistas frontend

Vista de las muestras por categoría y sistema al que pertenece

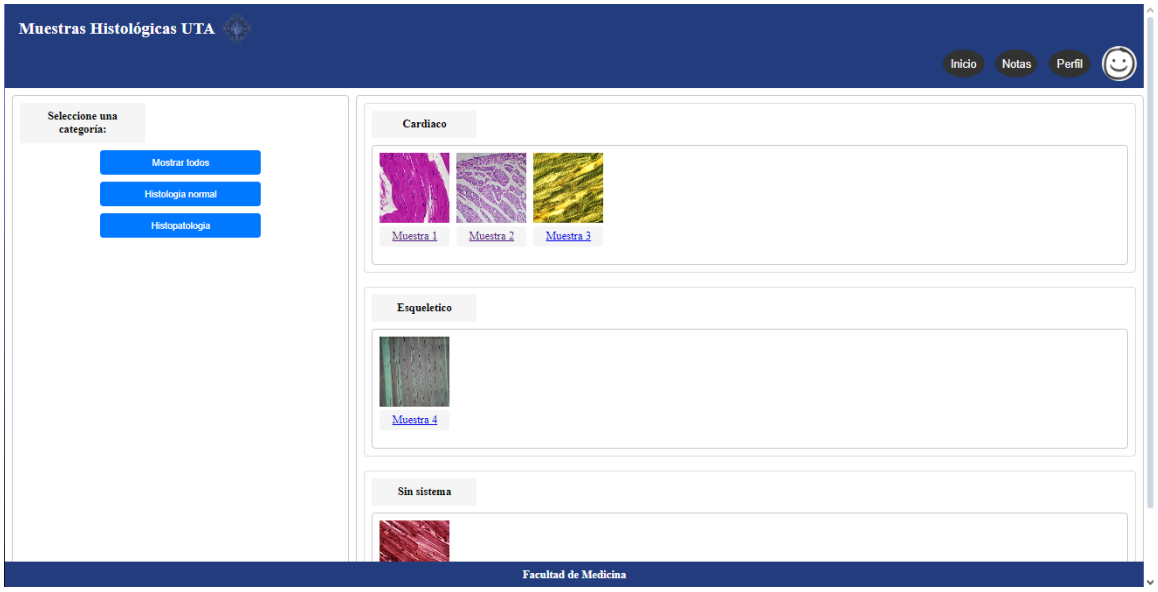

Imagen 20. Vista de todas las muestras

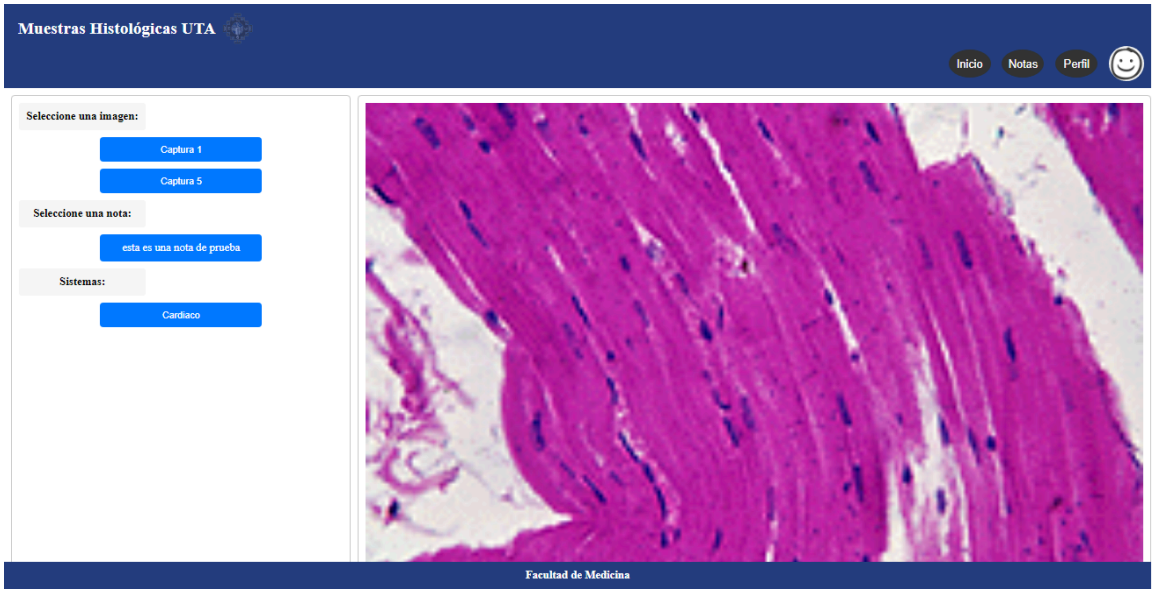

Imagen 21. Vista de una muestra

Conclusión

La digitalización del banco de muestras histológicas de la Facultad de Medicina UTA es un proyecto visionario que marca un hito en la modernización de la enseñanza y la investigación en el campo de la medicina. Este proyecto tiene el potencial de transformar la forma en que se imparte la histología, impulsar la investigación científica y optimizar el uso de recursos, contribuyendo al desarrollo de nuevas tecnologías y conocimientos en el ámbito de la salud.

Referencias

[1] S. Huidobro, "backendHistologia (branch: dev)," GitHub, https://github.com/TebaHuido/backendHistologia/tree/dev (accedido: julio 11, 2024).

[2] S. Huidobro, "angudocker," Docker Hub, https://hub.docker.com/r/minero420/angudocker (accedido: julio 11, 2024).

[3] J. Le Blanc, "Front-Histologia," GitHub, https://github.com/Joselito-informatico/Front-Histologia.git (accedido: julio 11, 2024).