



Construcción de una Aplicación Web para la lectura de imágenes de fondo de ojo implementando la escala de colores Teherán - Morales para determinar la palidez del disco ocular.

Jaime de Jesús Hernández Polo & Nellyth Jhon Arroyo Marrugo

Trabajo de investigación para optar al título de Sistemas

Director

Eugenia Arrieta Rodríguez

MSc en Ingeniería con énfasis en sistemas y computación

Universidad Del Sinú Elías Bechara Zainúm seccional Cartagena

Facultad de ciencias exactas e ingenierías

Escuela de Ingeniería de Sistemas

Enero 2023

CONTENIDO

RESUMEN.....	6
ABSTRACT.....	7
INTRODUCCIÓN.....	8
1. DISEÑO METODOLÓGICO.....	10
1.1 Planteamiento del problema.....	10
1.2 Justificación.....	12
1.3 Alcance.....	13
1.4 Objetivos.....	14
1.5.1 General.....	14
1.5.2 Específicos.....	14
1.5 Estado del Arte.....	15
1.6 Marcos de referencia.....	17
1.6.1 Marco teórico.....	17
1.6.2 Marco conceptual.....	21
1.6.3 Marco legal y consideraciones éticas.....	27
2. IMPLEMENTACIÓN.....	30
2.1 Planificación y estimación.....	30
2.2 Sprint 1.....	35
2.3 Sprint 2.....	36
2.4 Sprint 3.....	36
2.5 Sprint 4.....	37
3. DESARROLLO.....	38
3.1 Casos de uso del proyecto.....	38
3.2 Mockups del proyecto.....	39
3.3 Arquitectura del proyecto.....	41
3.3.1 Modelo de datos.....	42
3.3.2 Estructura de carpetas del proyecto.....	44
4. RESULTADOS.....	62
4.1 Página home del proyecto.....	62
5.2 Página de servicios.....	62
5.3 Página para la clasificación de imagen de fondo de ojo y su resultado.....	63
5.4 Pruebas automatizadas aplicadas a la aplicación web.....	65
DISCUSIÓN.....	73
CONCLUSIONES Y TRABAJOS FUTUROS.....	77
Referencias.....	79

ILUSTRACIONES

Ilustración 1 Fotografía de nervio óptico	18
Ilustración 2 Cuadrantes de la papila óptica	19
Ilustración 3 Flujo de la metodología Scrum	21
Ilustración 4 Principios Scrum	22
Ilustración 5 Esquema del Modelo - Vista - Template en Django	25
Ilustración 6 Canal HSV	26
Ilustración 7 Escala de graduación del color Teherán-Morales	27
Ilustración 8 Cronograma	30
Ilustración 9 Tabla de actividades en JIRA	33
Ilustración 10 Diagrama de casos de uso del proyecto	39
Ilustración 11 Mockup de la página inicial del proyecto	40
Ilustración 12 Mockup de la página de servicios del proyecto	40
Ilustración 13 Mockup de la página de clasificación de imagen del proyecto	41
Ilustración 14 Diagrama entidad-relación	43
Ilustración 15 Estructura del proyecto	44
Ilustración 16 Carpeta de configuración general del proyecto	45
Ilustración 17 Archivo de configuración de la app	45
Ilustración 18 Configuración de archivos ASGI y WSGI	46
Ilustración 19 Archivo de configuración settings.py	47
Ilustración 20 Archivo de configuración urls.py	48
Ilustración 21 Carpeta (app) imageReaderByHsv	48
Ilustración 22 Archivo de migración en Django	49
Ilustración 23 Archivo de init de la aplicación imageReaderByHsv	50
Ilustración 24 Modelo para la carga de imagen de la aplicación	51
Ilustración 25 Archivo admin.py de la aplicación	52
Ilustración 26 Interfaz de administración de la aplicación	52
Ilustración 27 Archivo forms.py	53
Ilustración 28 Archivo views.py	54
Ilustración 29 Archivo tests.py	55
Ilustración 30 Carpeta media del proyecto	56

Ilustración 31 Carpeta static del proyecto	56
Ilustración 32 Carpeta templates del proyecto	57
Ilustración 33 Plantilla dinámica reader_image.html	58
Ilustración 34 Archivo .gitignore	58
Ilustración 35 Archivo manage.py	59
Ilustración 36 Archivo README.md	60
Ilustración 37 Archivo requirements.txt	61
Ilustración 38 Archivo setup.cfg	61
Ilustración 39 Página home del proyecto	62
Ilustración 40 Página de servicios	63
Ilustración 41 Página para la clasificación de imagen de fondo de ojo	64

TABLAS

Tabla 1 Procesos fundamentales Scrum	24
Tabla 2 Roles Scrum	31
Tabla 3 Puntuaciones usadas en los sprints	32
Tabla 4 Requerimientos del proyecto	35
Tabla 5 Metodología tradicional vs Metodología ágil	75

RESUMEN

Este trabajo presenta el desarrollo de una aplicación web para la lectura de imágenes de fondo de ojo, con el objetivo de aplicar una escala llamada Teherán – Morales, con la cual se determina el grado de palidez en el disco ocular y a su vez esto le permite al profesional médico diagnosticar enfermedades relacionadas como el glaucoma. Para la elaboración de la aplicación se utilizó scrum (una metodología de desarrollo ágil con un énfasis inicial en el desarrollo de software) con Sprint de entre una y cuatro semanas. Para el desarrollo de las historias de usuario se utilizó la herramienta JIRA, que es una aplicación web para la gestión de proyectos, de tareas y de errores, mientras que las interfaces de usuario propuestas fueron diseñadas en la aplicación Lucidchart, que es una herramienta de diagramación basada en la web que permite crear todo tipo de diagramas. En la fase de codificación se implementa el lenguaje de programación Python acompañado del framework Django y para la persistencia de los datos se usó una base datos relacional en PostgreSQL. Como resultado de la implementación de la metodología mencionada, se obtuvo una aplicación web responsive con capacidad de carga de imágenes en menos de 5 segundos sin perder resolución, la cual permite seleccionar 4 puntos de la imagen en la que se aplica la escala y se entrega el valor promedio de la palidez del disco ocular.

Palabras claves

Aplicación web; Disco ocular; Escala Teherán – Morales; Fondo de ojo; Framework; Scrum; Python

ABSTRACT

This work presents the development of a web application for reading fundus images, with the objective of applying a scale called Teherán - Morales, with which the degree of pallor in the ocular disc is determined and, in turn, this allows the medical professional to diagnose related diseases such as glaucoma. Scrum was used to develop the application, which is an agile development methodology with an initial emphasis on software development with sprints of between one and two weeks. For the development of the user stories, the JIRA tool was used, which is a web application for project, task and error management, while the proposed user interfaces were designed in the Lucidchart application, which is a web-based diagramming tool that allows you to create all kinds of diagrams. In the coding phase, the Python programming language is implemented accompanied by the Django framework and for the persistence of the data, a relational database in PostgreSQL was used. As a result of the implementation of the aforementioned methodology, a responsive web application was obtained with the capacity to load images in less than 5 seconds without losing resolution, which allows selecting 4 points of the image in which the scale is applied and delivered. the average value of the pallor of the ocular disc.

Keywords

Eye Disk; Eye fundus; Framework; Scrum; Python; Teheran – Morales scale; Web application.

INTRODUCCIÓN

Una fotografía del fondo del ojo es la fotografía de la superficie interior del ojo, el cual abarca la retina, el polo posterior, el nervio óptico y la mácula; comúnmente se utiliza un microscopio de baja potencia con una cámara adjunta, este es utilizado para el seguimiento de la progresión de la enfermedad o en los programas de detección, en el cual las fotos se analizan después; es por eso su indicación para detectar y evaluar síntomas de enfermedades oculares como el glaucoma. Siendo, el glaucoma un conjunto de afecciones que pueden dañar el nervio óptico. Es la principal causa de ceguera en los Estados Unidos. Suele ocurrir cuando la presión del líquido que se encuentra dentro de los ojos aumenta lentamente y lesiona el nervio óptico. Es posible que al comienzo no haya síntomas. Sin tratamiento, las personas con glaucoma perderán la visión periférica (de lado). Sienten como que están mirando a través de un túnel. Con el tiempo, esta visión de túnel (el campo visual) puede disminuir hasta llevar a la ceguera [1].

Basándose en esos estudios del fondo de ojo, los investigadores de oftalmología de la universidad del Sinú han creado una escala de colores denominada Teherán-Morales que permite medir de forma cuantitativa el nivel de palidez del disco ocular cuando existe enfermedad glaucomatosa, basándose en la lectura de imágenes de fondo de ojo [2].

Esta escala ha sido implementada con un grupo de pacientes en la clínica Cartagena del Mar, y como resultado de dicha experimentación se obtuvo un conjunto de recomendaciones que permiten identificar la necesidad de construir una aplicación de software que implemente la escala mencionada con el fin de acortar los tiempos para la predicción o el análisis de los resultados y lograr una valoración objetiva calculada por la misma aplicación, en vez de una valoración subjetiva emitida por el proceso observacional realizado por el médico y cálculo manual.

Para esto los interesados (Oftalmólogos) y el equipo Scrum realizaron el entendimiento de las necesidades y definieron los requerimientos con los que debe cumplir la herramienta. En consenso de forma general, se decidió desarrollar una aplicación web que pudiera hacer el cálculo de palidez del disco ocular basado en la escala Teherán-Morales con tecnologías de ágil desarrollo como Python y Django, aplicando una metodología de desarrollo ágil llamada Scrum.

El documento se encuentra distribuido en 4 capítulos en los que se podrá encontrar toda la información necesaria para mantener y entender la herramienta desarrollada, en el capítulo 1 se explica de manera detallada el planteamiento del problema, las necesidades que surgieron para el desarrollo del proyecto, la justificación y la metodología aplicada. seguido del capítulo 2, donde se presenta la forma cómo se utilizó la metodología de desarrollo ágil Scrum: planificación eficiente del proyecto, generando así el sistema de puntuación (utilizando la secuencia Fibonacci) para los requerimientos; y el seguimiento del trabajo. Además muestra un cronograma en el que se puede evidenciar las fechas y los entregables en los diferentes Sprint y la prioridad de cada requerimiento. Por otro lado, en el capítulo 3 se evidencia y explica todo el desarrollo a nivel de ingeniería y codificación, mostrar su infraestructura y los patrones de diseño utilizados. Finalmente, en el capítulo 4 se muestran los resultados, detallando las ilustraciones y la descripción que permite aclarar el funcionamiento.

1. DISEÑO METODOLÓGICO

En este capítulo se describen los aspectos metodológicos que soportan la importancia de este trabajo de investigación: el planteamiento del problema en el cual se describen las necesidades planteadas por los interesados, la justificación, objetivos, alcance, estado actual de la literatura alrededor de tema investigado, se detalla la metodología aplicada para la construcción de la aplicación y el patrón de diseño utilizado.

1.1 Planteamiento del problema

Según la Organización Mundial de la Salud, la principal causa de ceguera crónica irreversible en el mundo después de las cataratas, es el glaucoma, en la mayoría de los casos, esta patología es indetectable. Por lo tanto, muchos pacientes notan que padecen de glaucoma cuando presentan una disminución del campo visual notoria (Congdon,2004). Siendo el glaucoma una de las principales causas de ceguera para las personas mayores de 60 años, puede producirse a cualquier edad, pero es más común en adultos mayores y según la fisiopatología de la enfermedad glaucomatosa la teoría vascular se basa en la premisa de que existen alteraciones de base en el flujo sanguíneo de la cabeza del nervio óptico [3].

Se estima que, en el mundo, para el año 2040, habrá aproximadamente 111.8 millones de personas afectadas con esta enfermedad [4]. Se estima que, en el mundo, para el año 2040, habrá aproximadamente 111.8 millones de personas afectadas con esta enfermedad. Por tanto, la salud visual hace un aporte muy importante al cumplimiento de algunas de las metas y objetivos de desarrollo sostenible propuestos por la OMS [5]. Como, por ejemplo, la mejora en las limitaciones visuales de la población puede contribuir al aumento de la productividad, disminución de la pobreza, la misma salud física y mental, entre otras, ya que es una enfermedad que indirectamente afecta la vida en general del paciente y cuidadores. En consecuencia, las afecciones asociadas al

glaucoma, pueden desencadenar en patologías avanzadas en la visión y limitar la vida en general de los pacientes.

Teniendo en cuenta lo anterior, y profundizando en el manejo actual y esfuerzos realizados por los profesionales de la salud, se identifica que, en la práctica, el examen clínico oftalmológico es la principal herramienta para la detección y búsqueda de anomalías que afecten la salud visual de los pacientes, en la actualidad es común el uso ayudas diagnósticas para la lectura de imágenes de fondo de ojo, x, y, z, entre otras, que faciliten esta tarea.

De acuerdo con [6], se pueden mencionar las siguientes como limitaciones asociadas a la detección del glaucoma:

- Es necesario realizar cinco pruebas incluidas en un examen completo de ojo para detectar glaucoma.
- Pocos pacientes se realizan los exámenes por prevención, generalmente acuden al oftalmólogo cuando su campo visual se encuentra comprometido.
- Mucha información obtenida depende de una retroalimentación subjetiva del paciente.
- Al ser exámenes relacionados con el estado de la retina, pueden ser afectados por la hora del día, o el estado de alerta de la persona.

Adicionalmente, aunque la imagen de fondo de ojo puede proporcionar la palidez del nervio óptico, se requiere que esta imagen sea lo suficientemente clara y a color. Esta ayuda diagnóstica permite con mayor facilidad la visualización del polo posterior, mejorando identificación, localización y graduación de alteraciones de la normalidad. Sin embargo esta lectura del fondo de ojo es subjetiva, y existen pocas maneras de evaluar el color del nervio óptico en una fotografía, sin embargo existen programas informáticos, diseñados para edición de imágenes que pueden medir cuantitativamente la graduación del color, abriendo la posibilidad de correlacionar los parámetros

cuantitativos obtenidos en la foto con los parámetros anatómicos cuantitativos de grosor de capas de fibras nerviosas y densidad vascular peri papilar obtenidos mediante AngioOCT (Combinación de la Angiografía y la tomografía por coherencia óptica). Aunque existen estas herramientas, los cálculos y mediciones se realizan manualmente y tomando como entrada los valores observados subjetivamente por el profesional de la salud.

Por lo anterior, se plantea el siguiente cuestionamiento que será resuelto al finalizar la investigación:

¿Cómo construir una aplicación web para la lectura de imágenes de fondo de ojo y cálculo automático de la escala Teherán-Morales?

1.2 Justificación

Teniendo en cuenta la problemática expuesta y la importancia de la salud visual de la población que es de gran preocupación y objetivo de la Organización mundial de la salud. Este proyecto identifica una oportunidad para aportar en los esfuerzos por disminuir las cifras de afectados gravemente con el glaucoma. El aporte se relaciona directamente con el apoyo al proceso diagnóstico de la severidad de la enfermedad aplicando la escala Teherán-Morales [2] planteada por el equipo de investigadores de la Universidad del Sinú, situados en campo en la Clínica Oftalmológica de Cartagena, esta última institución es un centro de referencia que atiende gran parte de la población con patología ocular de la ciudad, lo que permite recolectar información suficiente para el desarrollo de actividades investigativas.

Este proyecto busca el desarrollo de una aplicación web usando buenas prácticas de ingeniería, que permitan la implementación y automatización de la escala de colores Teherán- Morales, calculando el valor promedio de la palidez del disco ocular, esto apoyado en la imagen de fondo de ojo a color proporcionada como entrada a la aplicación.

Desde el punto de vista de la ingeniería de sistemas y ciencias computacionales, existe la necesidad de implementar metodologías de desarrollo de software que agilicen el proceso de construcción del prototipo, con equipos de pocas personas y que apliquen correctamente los principios de desarrollo de software que minimicen los errores de diseño y que implementen correctamente los patrones de diseño. Por lo cual, este trabajo permitirá implementar estas buenas prácticas de la ingeniería y lograr un producto que cumpla con criterios de calidad.

Con el desarrollo de este trabajo, se espera beneficiar directamente a los especialistas de la salud. Los cuales contarán con una app web que los ayudará en la clasificación de dicha enfermedad y esto ayudará a su vez a mejorar la precisión y los tiempos de todo este proceso manual que se hace para detectar la enfermedad glaucomatosa. y por ende, resultará beneficioso para los pacientes en tener un resultado rápido y preciso.

Es importante mencionar que, existe un convenio de la Clínica en mención, con la Universidad del Sinú seccional Cartagena, que favorece espacios de aprendizaje y desarrollo de estudios clínicos necesarios en la región. Con este escenario se plantea el apoyo en la realización y desarrollo de este trabajo de investigación.

Adicionalmente, el desarrollo de esta aplicación permitirá fortalecer la línea de desarrollo de software del grupo de investigación del programa de Ingeniería de Sistemas, logrando un prototipo funcional como producto de desarrollo tecnológico.

1.3 Alcance

Con la elaboración de este prototipo web se busca que a través de la carga de

imágenes de fondo de ojo (con formato JPG, JPEG y PNG) al seleccionar el disco ocular o 4 puntos exactos en el disco ocular, se pueda obtener de forma cuantitativa el nivel de palidez, y en base a ese nivel de palidez, una clasificación precisa en la escala Teherán-Morales. Con esto se podrá brindar un diagnóstico más preciso y objetivo a los pacientes con enfermedad glaucomatosa.

El prototipo web busca desarrollar con herramientas tecnológicas como Django y Python para el Backend aplicando todas las normativas como patrones de diseño de software para la correcta escritura y estructura del código en el proyecto, también se utilizará PostgreSQL para guardar la información en una base de datos y algunas herramientas para el Frontend como Bootstrap.

1.4 Objetivos

1.5.1 General

Construir una aplicación web para la lectura de imágenes de fondo de ojo que permita determinar la palidez del disco ocular aplicando la escala de colores Teherán - Morales.

1.5.2 Específicos

- Analizar los requerimientos del experto en optometría para determinar los requisitos funcionales y no funcionales de la aplicación.
- Diseñar la aplicación web dando respuesta a los requerimientos previamente definidos.
- Desarrollar la aplicación web usando el lenguaje de programación Python con framework Django aplicando el patrón de diseño Modelo - Vista - Template.
- Validar la aplicación con el experto evaluando el cumplimiento de los requisitos funcionales y no funcionales.

1.5 Estado del Arte

Las investigaciones realizadas sobre el uso de las tecnologías de software aplicadas en el ámbito de las enfermedades oculares nos muestran un crecimiento exponencial en cuanto al desarrollo de sistemas de diagnósticos asistidos por computadoras y el tratamiento de los malestares que se logran diagnosticar. Es por esto que ha tenido una gran popularidad aplicar estas tecnologías en los procesos médicos actuales, de acuerdo a la revisión literaria realizada, también se logró encontrar una gran variedad de estudios y empresas en los que se aplican estos conceptos y donde se pueden observar resultados favorables. A continuación, algunos de estos estudios.

Un estudio presentado en el año 2020 por el Ing. Eduardo Bernal Catalán en el Instituto Tecnológico de Acapulco, describe el desarrollo de un dispositivo capaz de capturar retinografías para así diagnosticar de manera confiable la retinopatía diabética la cual es una complicación que afecta a la población diabética, que de ser detectada a tiempo puede ser tratada y evitar su progresión que conlleva a la pérdida total de la vista de manera irreversible. Este dispositivo incluye un algoritmo para el entrenamiento de una red neuronal con un valor de especificidad que si supera el 80 % (0.8). El modelo de aprendizaje obtenido por el entrenamiento de la red neuronal ha sido evaluado y se obtuvo una especificidad del 82% que es un valor aceptable para el apoyo brindado a los médicos [7].

Otro trabajo relevante corresponde a un artículo científico presentado en el año 2021, en donde se detalla que las imágenes retinianas a menudo tienen problemas como, bajo contraste, detalles borrosos o artefactos, que pueden afectar gravemente el diagnóstico. Por lo tanto, es muy importante mejorar la calidad visual de tales imágenes, y se plantea que un preprocesamiento aplicado a las imágenes para mejorar su calidad visual. Esta técnica mejora la identificación de las estructuras retinianas en imágenes retinianas degradadas. En este trabajo se presenta un novedoso algoritmo basado en la morfología

matemática multiescala. En primer lugar, la imagen original se difumina mediante el filtro Abrir-Cerrar-Cerrar-Abrir (OCCO) para reducir cualquier artefacto en la imagen. A continuación, la transformación Top-Hat extrae múltiples características brillantes y oscuras de la imagen filtrada. Finalmente, los valores máximos de brillo se agregan a la imagen original y los valores máximos de oscuridad se restan de la imagen original, previamente ajustada por un peso. El algoritmo se probó en 397 imágenes retinianas de la base de datos pública *STARE (Structured Analysis of the Retina)*. El algoritmo propuesto se comparó con algoritmos de última generación y los resultados muestran que la propuesta es más eficiente en la mejora del contraste, manteniendo la similitud con la imagen original e introduciendo menos distorsión que los otros algoritmos. Según los oftalmólogos, el algoritmo, al mejorar las imágenes retinianas, proporciona una mayor claridad en los vasos sanguíneos de la retina y facilita la identificación de patologías [8].

Adicionalmente, en el trabajo de grado presentado en el año 2018 por Jorge José Rami Rodríguez en la Universidad Zaragoza, se presenta una estrategia innovadora para la comparación de técnicas de detección de microaneurismas, mediante la fusión de imágenes de OCT-A (Tomografía de Coherencia Óptica con Angiografía) de los plexos vascular superficial y profundo. La combinación de imágenes se realizó con el software GIMP 2.10.6 (GNU Image Manipulation Program), superponiendo las capas. Los resultados obtenidos mediante este enfoque proporcionarán una mejor comprensión de las capacidades relativas de la AGF (Angiografía Fluoresceínica) y la OCT-A en la detección de microaneurismas, contribuyendo así al avance del campo de la oftalmología y la detección temprana de enfermedades retinianas [9].

Finalmente, el artículo científico publicado en el año 2021, en donde se explica el uso de un modelo de aprendizaje profundo para asistir en el diagnóstico de TO (toxoplasmosis ocular) a partir de imágenes de fondo de ojo. Una red neuronal residual pre-entrenada fue ajustada a un dataset recolectado en el

Hospital de Clínicas de Asunción. Con sensibilidad y especificidad de 94% y 93% respectivamente, el modelo propuesto resulta prometedor [10].

De acuerdo con la literatura revisada, este proyecto tomará como referencia las investigaciones anteriormente mencionadas, con el fin de aplicar las técnicas que mejores resultados presentaron en cada uno de estos proyectos, y así, lograr minimizarla curva de error y maximizar la probabilidad de éxito en cuanto a resultados al momento de construir la aplicación web para la lectura de imágenes de fondo de ojo implementando la escala de colores Teherán - Morales para determinar la palidez del disco ocular.

1.6 Marcos de referencia

1.6.1 Marco teórico

La teoría vascular del glaucoma explica que una disminución de la densidad de vasos sanguíneos peris papilares disminuye también la nutrición que recibirá el nervio óptico, llevando a isquemia y muerte de células ganglionares [11]. Y es bien sabido el efecto hipotensor que producen los medicamentos antihipertensivos, que podrían ser contraproducentes para la circulación ocular y finalmente la del nervio óptico. Se han descrito factores como la edad, disminución de la presión arterial nocturna y la alta variabilidad de la presión arterial pueden ser factores de riesgo adicionales para la progresión de la pérdida de campo glaucomatoso [12]. Además, la marcada fluctuación circadiana de presión de perfusión ocular media se asocia con la reducción nocturna de la presión arterial. La fluctuación circadiana de presión de perfusión ocular media puede ser un factor de riesgo para el desarrollo de glaucoma de tensión normal [13].

La palidez del nervio óptico sea difusa o sectorial, acompaña como signo clínico a una variedad de neuropatías ópticas, y esta palidez se ha asociado, dentro de muchos factores de riesgo, con la hipotensión arterial marcada,

nocturna, o sostenida en el tiempo [13]. Dependiendo de la patología la palidez puede estar acompañada de estrechamiento arteriolar, como sucede en la Neuropatía óptica isquémica anterior no artrítica o idiopática; o disminución del espesor de la capa de fibras nerviosas papilares, como sucede en las neuropatías ópticas hereditarias [13]. Esta hipotensión arterial (relativa) nocturna puede producir la situación de hipoperfusión de las pequeñas arterias ciliares posteriores cortas que irrigan la papila llevando a disminución de la densidad capilar peri papilar. Esta densidad puede ser medida mediante AngioOCT, y la palidez del nervio óptico se mide comúnmente por una evaluación subjetiva clínica.

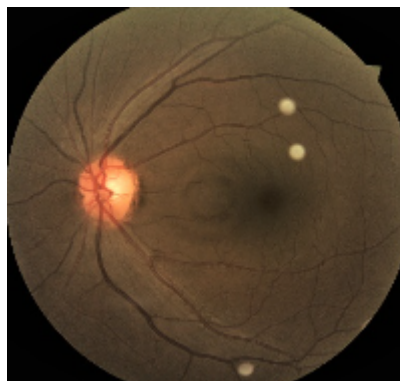


Ilustración 1 Fotografía de nervio óptico.

Fuente: Imagen de retina de campo ultra-amplio | Revista Mexicana de Oftalmología. Accedido el 16 de enero de 2023. [Imagen]. Disponible: <https://www.elsevier.es/es-revista-revista-mexicana-oftalmologia-321-articulo-imagen-retina-ampo-ultra-amplio-S0187451916300798>

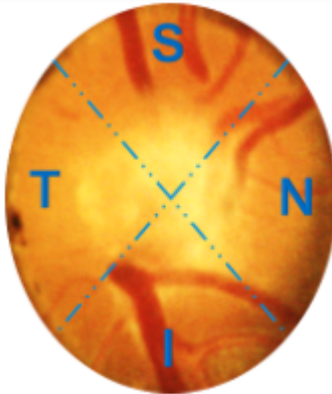


Ilustración 2 Cuadrantes de la papila óptica.

Fuente: Imagen de retina de campo ultra-amplio | Revista Mexicana de Oftalmología. Accedido el 16 de enero de 2023. [Imagen]. Disponible: <https://www.elsevier.es/es-revista-revista-mexicana-oftalmologia-321-articulo-imagen-retina-campo-ultra-amplio-S0187451916300798>

La imagen de fondo de ojo puede llegar a ser de gran ayuda a la hora de tener una imagen clara, y que perdure en el tiempo, permite con mayor facilidad la visualización del polo posterior, mejorando identificación, localización y graduación de alteraciones de la normalidad. Sin embargo esta lectura del fondo de ojo es subjetiva, y existen pocas maneras de evaluar el color del nervio óptico en una fotografía, sin embargo existen programas informáticos, diseñados para edición de imágenes que pueden medir cuantitativamente la graduación del color, abriendo la posibilidad de correlacionar los parámetros cuantitativos obtenidos en la foto con los parámetros anatómicos cuantitativos de grosor de capas de fibras nerviosas y densidad vascular peri papilar obtenidos mediante AngioOCT.

El nervio óptico se encuentra en la escala de tonos rojos, pero, cuando éste se torna pálido, pierde saturación, esto significa que sigue siendo rojo, pero con mayor cantidad de gris, por lo que no es correcto considerar un nervio óptico rosado o blanco.

Otros estudios metodológicamente similares a este utilizaron el software

ImageJ para evaluar la coloración midiendo por separado la intensidad de los píxeles rojos, verdes y azules marcando previamente el borde del anillo neuro retiniano, para poder obtener las imágenes RGB en el área seleccionada. Para cuantificar el color del nervio óptico definieron un índice de color (DCI) donde $DCI = R/(R+G+B)$ que requirió complejos algoritmos para estadificar el color; y encontraron que este índice fue mayor en pacientes jóvenes y en preoperatorios de catarata. Se considera que la fortaleza de la escala Teherán-Morales consiste en clasificar objetivamente la coloración del nervio óptico, con mayor facilidad y aplicabilidad en la práctica clínica.

El Nivel de Madurez Tecnológica (Technology Readiness Level, TRL) es una medida cuantitativa utilizada para evaluar el nivel de desarrollo y la viabilidad de implementación de una tecnología. El TRL se compone de nueve niveles, desde el 1 (investigación básica) hasta el 9 (implementación y operación comercial). Cada nivel representa un grado de avance tecnológico y sirve como indicador para comprender la madurez y la viabilidad de una tecnología en particular.

Es importante destacar que el TRL proporciona una herramienta objetiva para evaluar la madurez de una tecnología, lo cual es esencial para determinar la viabilidad y el potencial de éxito de un proyecto de investigación y desarrollo [14].

En este proyecto se ha determinado que la tecnología se encuentra en el nivel 4 del TRL. En este nivel, se ha demostrado la validez del concepto tecnológico a través de experimentos. Esto significa que el proyecto se encuentra en una etapa en la que se han superado las investigaciones preliminares y se ha demostrado la viabilidad de la tecnología en un entorno controlado. A medida que avance la investigación y desarrollo, se trabajará en llevar el proyecto a niveles superiores de madurez tecnológica, acercando así la implementación práctica y comercialización.

1.6.2 Marco conceptual

La metodología de desarrollo Scrum es una de las más conocidas. Es una herramienta adaptable, iterativa, rápida, flexible y eficaz, diseñada para ofrecer un valor considerable en tiempo récord a lo largo del proyecto. Scrum garantiza transparencia en la comunicación y crea un ambiente de responsabilidad colectiva y de progreso continuo.

Uno de los puntos clave de Scrum radica en el uso de equipos interfuncionales (cross-functional), auto organizados y empoderados que dividen su trabajo en ciclos de trabajo cortos y concentrados llamados Sprint [15].

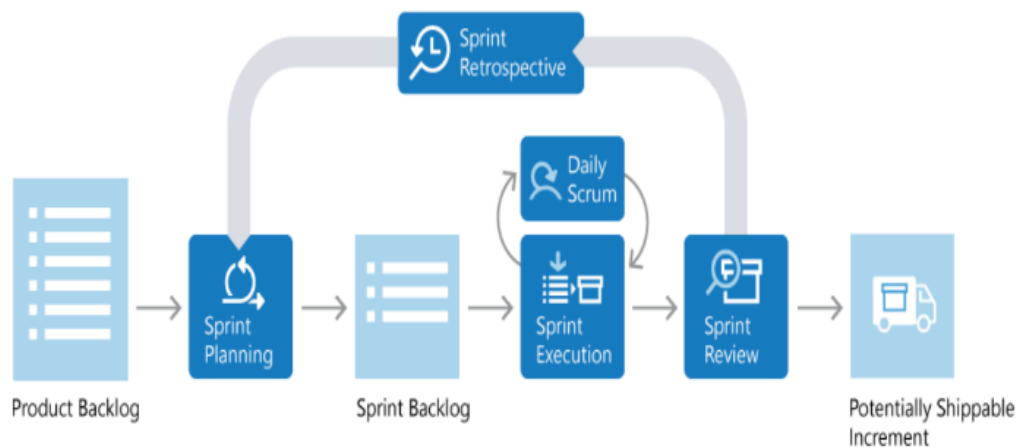


Ilustración 3 Flujo de la metodología Scrum

Fuente: S. Paduano. SCRUM y ser ágil. (5 de septiembre de 2019). Accedido el 16 de enero de 2023. [Imagen]. Disponible: <https://medium.com/@sebastianpaduano/scrum-y-ser-ágil-47af5649e15c>

Los principios de Scrum son las pautas básicas para aplicar la herramienta y deben implementarse en forma obligatoria en todos los proyectos Scrum.



Ilustración 4 Principios Scrum

Fuente: S. Paduano. SCRUM y ser ágil. (5 de septiembre de 2019). Accedido el 16 de enero de 2023. [Imagen]. Disponible: <https://medium.com/@sebastianpaduano/scrum-y-ser-ágil-47af5649e15c>

Los roles de Scrum se dividen en dos extensas categorías:

➤ **Roles centrales**

Son aquellos que se requieren de forma obligatoria para crear el producto o servicio del proyecto. Estos roles incluyen:

- El Product Owner es la persona que debe lograr que el proyecto obtenga un gran valor para la empresa. Este rol también es responsable de acordar los requisitos del proyecto con el cliente y de mantener la justificación del negocio para el proyecto. Representa la voz del cliente.
- El Scrum Master es quien asegura que el equipo Scrum cuente con un buen ambiente para desarrollar el proyecto. El objetivo de este rol es guiar, facilitar y enseñar las prácticas de Scrum a todos los colaboradores en el proyecto.

- El equipo Scrum es el grupo de personas que atienden los requisitos especificados por el Product Owner y quienes crean los entregables del proyecto [16].

➤ **Roles no centrales**

Son aquellos que no son necesariamente obligatorios para el proyecto, y se en estos roles se pueden incluir a miembros que estén interesados en el proyecto. Se incluyen los siguientes:

- Stakeholder(s) en este rol se incluye a los clientes, usuarios y patrocinadores, que van a interactuar con el equipo principal de Scrum y son influyentes en el proyecto a lo largo de su desarrollo.
- El Scrum Guidance Body (SGB) este rol es basado en un conjunto de documentos y/o grupo de expertos que usualmente son responsables de la definición de objetivos relacionados con la calidad, las regulaciones del gobierno, la seguridad y otros parámetros claves de la organización. El SGB guía el trabajo del Product Owner, el Scrum Master y el Equipo Scrum.

Proceso Scrum

Existen 19 procesos fundamentales de Scrum que se aplican a todos los proyectos.

FASE	PROCESOS FUNDAMENTALES DE SCRUM
INICIO	<ol style="list-style-type: none"> 1. Crear la visión del proyecto 2. Identificar al Scrum Master y Stakeholder(s) 3. Formar Equipos Scrum 4. Desarrollar épica(s) 5. Crear el Backlog Priorizado del Producto

	6. Realizar la planificación de lanzamiento
PLANIFICACIÓN Y ESTIMACIÓN	7. Crear historias de usuario 8. Estimar historias de usuario 9. Comprometer historias de usuario 10. Identificar tareas 11. Estimar tareas 12. Crear el Sprint Backlog
IMPLEMENTACIÓN	13. Crear entregables 14. Realizar Daily Standup 15. Refinar el Backlog Priorizado del Producto
REVISIÓN Y RETROSPECTIVA	16. Demostrar y validar el sprint 17. Retrospectiva del sprint
LANZAMIENTO	18. Enviar entregables 19. Retrospectiva del proyecto

Tabla 1 Procesos fundamentales Scrum

Arquitectura Django

Django se basa en la arquitectura MVT (Model-View-Template). MVT es un patrón de diseño de software para desarrollar una aplicación web.

La estructura MVT tiene las siguientes tres partes:

- **Modelo:** el modelo actuará como la interfaz de sus datos. Es responsable de mantener los datos. Es la estructura de datos lógica detrás de toda la aplicación y está representada por una base de datos (generalmente bases de datos relacionales como MySQL, PostgreSQL).
- **Vista:** La vista es la interfaz de usuario, lo que ve en su navegador

cuando representa un sitio web. Está representado por archivos HTML / CSS / JavaScript y Jinja.

- Plantilla: una plantilla consta de partes estáticas de la salida HTML deseada, así como una sintaxis especial que describe cómo se insertará el contenido dinámico [17].

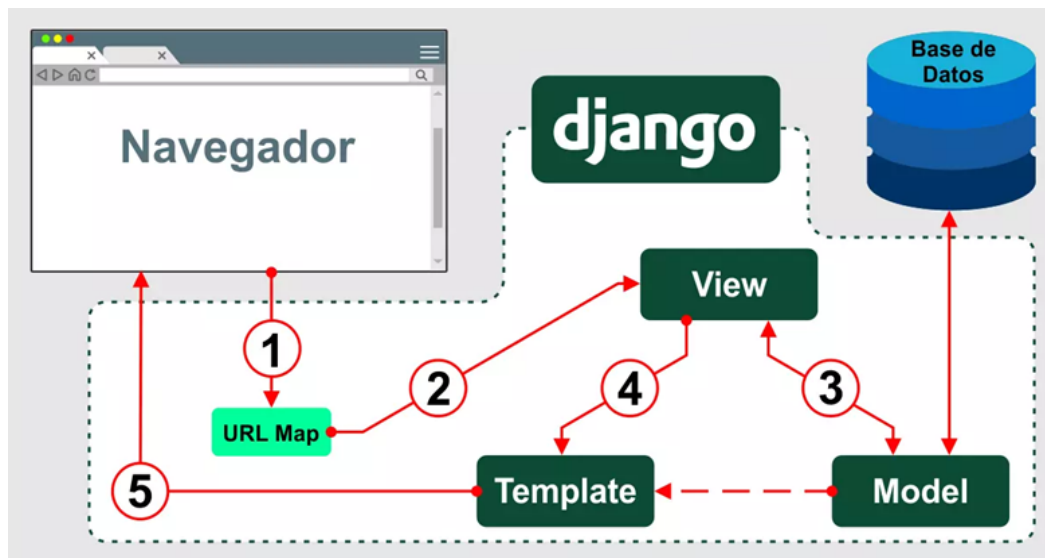


Ilustración 5 Esquema del Modelo - Vista - Template en Django

Fuente: Que es el patrón MTV (Model Template View). Accedido el 16 de enero de 2023. [Imagen]. Disponible: <https://espifreelancer.com/mtv-django.html>

Django es un framework web de alto nivel que permite el desarrollo rápido de sitios web seguros y mantenibles. Desarrollado por programadores experimentados, Django se encarga de gran parte de las complicaciones del desarrollo web, por lo que puedes concentrarte en escribir tu aplicación sin necesidad de reinventar la rueda. Es gratuito y de código abierto, tiene una comunidad próspera y activa, una gran documentación y muchas opciones de soporte gratuito y de pago.

Los sitios de alto nivel que usan Django incluyen: Disqus, Instagram, Knight

Foundation, MacArthur Foundation, Mozilla, National Geographic, Open Knowledge Foundation, Pinterest y Openstack [18].

Por otro lado, para comprender claramente aspectos relacionados con la escala Teherán -Morales es necesario comprender la escala HSV, que es una forma de representar puntos en el espacio de color RGB en un cono invertido. HSV significa tono, saturación, valor y HSB (B significa brillo). El tono es el atributo básico del color, que es el nombre del color habitual, como el rojo y el amarillo. La saturación (S) se refiere a la pureza del color. Cuanto más alto es el color, más puro es el color, y más bajo el color se volverá gradualmente gris. Se utiliza un valor de 0-100%. Brillo (V), tome 0-max (el rango de HSV en la computadora está relacionado con la duración del almacenamiento) [19].

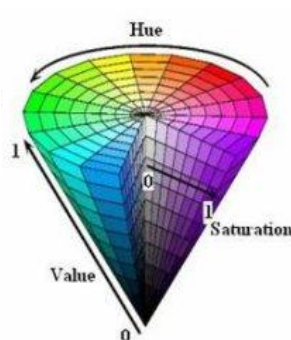


Ilustración 6 Canal HSV

Fuente: Descomponer los colores de una imagen. Accedido el 16 de enero de 2023.

[Imagen].

Disponible:

<https://docs.wplib.org/es/stable/docs/software/vision-processing/wplibpi/image-thresholding.html>

Con las facilidades que brinda la tecnología del siglo XXI para la investigación, se creó en la Universidad del Sinú una escala de graduación del color del fondo de ojo, diseñada para clasificar el color dándole un valor numérico según su tono y saturación. Esta escala es llamada “Escala de graduación del color Teherán-Morales”.

GRADOS TONO	NOMBRE DEL TONO	SATURACIÓN				
		100 - 80%	79,9 - 60%	59,9 - 40%	39,9 - 20%	19,9 - 0%
0 - 9,9°	Rojo oscuro	0 ROJO	1	2	3	4
10 - 19,9°	Rojo	1 NORMAL		2	3	4
20 - 29,9°	Rojo cálido	2 LEVE PALIDEZ			3	4
30 - 39,9°	Naranja	3 PALIDEZ MARCADA				4
40 - 59,9°	Amarillo cálido	4 PALIDEZ CÉREA				

Ilustración 7 Escala de graduación del color Teherán-Morales

1.6.3 Marco legal y consideraciones éticas

Debido a que se usarán diversos datos personales, la app web debe seguir las leyes nacionales sobre la protección de datos.

Cuando se mencionan datos personales se refiere a toda aquella información relacionada a una persona y que permite su identificación. Por ejemplo, su documento de identidad, el lugar de nacimiento, estado civil, edad, lugar de residencia, trayectoria académica, laboral, o profesional.

Ley de Protección de Datos Personales o Ley 1581 de 2012 reconoce y protege el derecho que tienen todas las personas a conocer, actualizar y rectificar las informaciones que se hayan recogido sobre ellas en bases de datos o archivos que sean susceptibles de tratamiento por entidades de naturaleza pública o privada [20].

Artículo 1°. Objeto. La presente ley tiene por objeto desarrollar el derecho constitucional que tienen todas las personas a conocer, actualizar y rectificar las informaciones que se hayan recogido sobre ellas en bases de datos o archivos, y los demás derechos, libertades y garantías constitucionales a que se refiere el artículo 15 de la Constitución Política; así como el derecho a la información consagrado en el artículo 20 de la misma.

Artículo 2°. Ámbito de aplicación. Los principios y disposiciones contenidas en la presente ley serán aplicables a los datos personales registrados en cualquier base de datos que los haga susceptibles de tratamiento por entidades de naturaleza pública o privada.

La presente ley aplicará al tratamiento de datos personales efectuado en territorio colombiano o cuando al responsable del Tratamiento o Encargado del Tratamiento no establecido en territorio nacional le sea aplicable la legislación colombiana en virtud de normas y tratados internacionales.

El régimen de protección de datos personales que se establece en la presente ley no será de aplicación a:

- Bases de datos o archivos mantenidos en un ámbito exclusivamente personal o doméstico.

Cuando estas bases de datos o archivos vayan a ser suministrados a terceros se deberá, de manera previa, informar al Titular y solicitar su autorización. En este caso los responsables y Encargados de las bases de datos y archivos quedarán sujetos a las disposiciones contenidas en la presente ley;

- Archivos y bases de datos que tengan por finalidad la seguridad y defensa nacional, así como la prevención, detección, monitoreo y control del lavado de activos y el financiamiento del terrorismo;
- Bases de datos que tengan como fin y contengan información de inteligencia y contrainteligencia;
- Bases de datos y archivos de información periodística y otros contenidos editoriales.

- Archivos y bases de datos regulados por la Ley 1266 de 2008;
- Archivos y bases de datos regulados por la Ley 79 de 1993.

Parágrafo. Los principios sobre protección de datos serán aplicables a todas las bases de datos, incluidas las exceptuadas en el presente artículo, con los límites dispuestos en la presente ley y sin referir con los datos que tienen características de estar amparados por la reserva legal. En el evento que la normatividad especial que regule las bases de datos exceptuadas prevea principios que tengan en consideración la naturaleza especial de datos, los mismos aplicarán de manera concurrente a los previstos en la presente ley.

2. IMPLEMENTACIÓN

En este capítulo se describe la planeación de un proyecto mediante el uso de cronograma, ya que estos ofrecen al equipo un plan de acción, aumentan la responsabilidad y ayudan a evitar posibles obstáculos. Una programación detallada es necesaria para cualquier proyecto, es por eso que se decidió crear un cronograma en el que se pudiese gestionar el desarrollo de la aplicación web, en este se muestran todas las tareas implicadas y una duración para cada una de ellas para que así todo el equipo pueda ver cuándo se llevará a cabo cada paso y cuándo finalizará el proyecto.

Proyecto de Grado - Automatización vía Unisinú

Ruta del proyecto

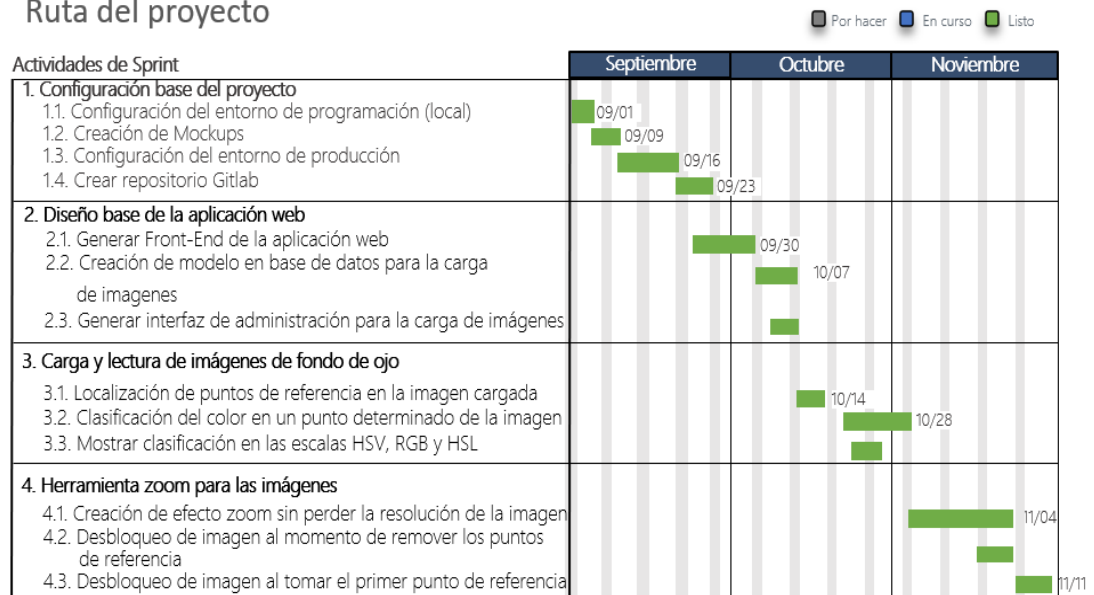


Ilustración 8 Cronograma

2.1 Planificación y estimación

Para el desarrollo del proyecto tecnológico planteado, se selecciona la metodología Scrum que es una metodología ágil para el desarrollo de software. Con la cual, se pudo construir la aplicación bajo un enfoque dinámico, aplicando iteraciones rápidas, complaciendo al cliente con entregas continuas y

en el tiempo estimado, pudiendo así evitar resultados insatisfactorios. Se creó un *backlog* de tareas para trabajar en los diferentes sprint y poder llevarlas a cabo en los tiempos estimados (puntuaciones en la secuencia Fibonacci).

A continuación, se detallan los roles de Scrum para la creación de la aplicación web:

ROL	PERSONA A CARGO	DESCRIPCIÓN
Product Owner	Eugenia Arrieta	Negocia decisiones sobre el producto con los clientes.
Scrum Master	Nellyth Arroyo	Guía al equipo en la metodología Scrum
Equipo de desarrollo	Nellyth Arroyo Jaime Hernández	Encargados de diseñar y desarrollar la aplicación.

Tabla 2 Roles Scrum

Los roles fueron asignados con base a las capacidades de cada persona en el equipo: Eugenia Arrieta se ha desenvuelto como Product Owner debido a su agilidad en la comprensión de los requerimientos de usuarios, asimismo, el rol de Scrum Master ha sido asignado a Nellyth Arroyo, debido a su experiencia como líder, por lo cual dirige y apoya al equipo en el uso de la metodología.

En el equipo de desarrollo hubo la participación de Nellyth Arroyo y Jaime Hernández, encargados para el diseño y desarrollo de la aplicación web, así como la ejecución de las pruebas.

Antes de iniciar con el desarrollo de la aplicación web, el equipo se reunió para llegar a un acuerdo de puntuación de cada trabajo que sería realizado en los Sprint necesarios.

Se utilizó un sistema de puntuación basado en las horas trabajadas en donde:

Puntuación	Duración
0.25	2 horas
0.5	4 horas
1	8 horas

Tabla 3 Puntuaciones usadas en los sprints

Además, estas puntuaciones fueron asignadas a cada requerimiento en la herramienta JIRA, que facilita y otorga más orden al momento de utilizar la metodología Scrum.

The screenshot displays the Jira Software interface for a project named 'GraduationProject'. The main view is a Kanban board titled 'Tablero GRPR'. The board is organized into four columns: 'POR HACER 14 INCIDENCIAS', 'EN CURSO', 'REVISION', and 'LISTO'. The 'POR HACER' column contains five tasks, each with a progress indicator and a status icon (e.g., 'GRPR-11', 'GRPR-1', 'GRPR-10', 'GRPR-8', 'GRPR-9'). The 'EN CURSO' column is currently empty. The 'REVISION' and 'LISTO' columns are also empty. The interface includes a sidebar with navigation options like 'PLANIFICACIÓN', 'DESARROLLO', and 'Tablero'. The top navigation bar shows 'JiraSoftware' and various menu options like 'Proyectos', 'Filtros', 'Paneles', 'Personas', 'Aplicaciones', and 'Crear'. A search bar is visible in the top right corner.

Ilustración 9 Tabla de actividades en JIRA

Requerimientos

Funcionalidades		
ID	Descripción	Puntuación
GRPR-1	Selección del mockup adecuado para el proyecto.	1 punto
GRPR-2	Definir y estructurar el modelo para la carga de imagen.	1.5 puntos
GRPR-3	Funcionalidad que permite clasificar el color en un punto determinado de la imagen, utilizando la escala Teherán-Morales.	3 puntos
GRPR-4	Mostrar la clasificación en las escalas HSV, RGB y HSL de los puntos seleccionados.	1 punto
GRPR-5	Funcionalidad que permita localizar cuáles fueron los puntos de referencia que se tomaron.	1 punto
GRPR-6	Desarrollar e implementar una herramienta que permita el acercamiento de la imagen, respetando su calidad de colores.	3 puntos
GRPR-7	Funcionalidad que desbloquee la imagen, en los aspectos de movimiento y zoom al remover todos los puntos de referencia.	0.5 punto
GRPR-8	Crear el entorno de controlador de versiones del repositorio del proyecto.	0.5 puntos
GRPR-9	Servir los documentos estáticos y media de	2 puntos

	la aplicación.	
GRPR-10	Despliegue de aplicación.	2 puntos
GRPR-11	Configuración del entorno de programación.	0.5 puntos
GRPR-12	Reparar registros duplicados.	1 punto
GRPR-13	Generar interfaz de administrador para el modelo de la base de datos	0.5 puntos
GRPR-14	Aplicar una funcionalidad que se encargará de desactivar el movimiento de la imagen y el zoom al momento de tomar el primer punto de referencia.	1 punto

Tabla 4 Requerimientos del proyecto

2.2 Sprint 1

El primer sprint se utilizó principalmente para que los desarrolladores pudieran trabajar de forma eficiente durante el resto del proyecto, se destinaron puntos para las configuraciones del proyecto tanto en entornos locales como los de producción, a su vez se configuró un repositorio de versiones (gitlab) para llevar un seguimiento de todos los cambios a las funcionalidades que se iban creando.

Algunas de las configuraciones locales que se aplicaron para el proyecto fueron:

- Crear un directorio para todos los archivos del proyecto.
- Instalar Python 3+ en las máquinas locales.

- Crear un entorno virtual para instalar todos los requisitos del proyecto y así se pueda ejecutar de manera correcta.
- Instalar Django en el entorno virtual creado.
- Crear un proyecto Django utilizando el comando: `django-admin startproject`.
- Crear una aplicación Django con el comando: `python manage.py startapp`.
- Configurar la base de datos en el archivo `settings.py` y `local_settings.py` del proyecto.

Cabe destacar que también se destinaron puntos para la creación de mockups y así se pudiera desarrollar el frontend de la aplicación web de manera más organizada y sencilla, teniendo claro lo que se buscaba desarrollar desde el principio.

2.3 Sprint 2

En el segundo sprint se gestiona la parte visual del proyecto usando las plantillas dinámicas que permite construir Django, luego renderizando las plantillas en las vistas (views) correspondientes, luego se pasó a desarrollar una de las funcionalidades principales que es guardar las imágenes de fondo de ojo en la base de datos del proyecto, además también se desarrolló una interfaz de administrador para que sea más fácil gestionar los datos que se guardan en el proyecto. Al utilizar una metodología de desarrollo ágil, a medida que el sprint avanzaba, se le iban mostrando avances a los usuarios para así recibir la retroalimentación y poder corregir en el tiempo estipulado.

2.4 Sprint 3

El tercer sprint se enfoca en las funcionalidades que se aplican a una imagen cargada como lo es la localización de puntos de referencia que básicamente se

trata de elegir 4 puntos en la imagen de fondo de ojo para que estos 4 puntos puedan ser procesados en el proyecto y luego, gracias a la funcionalidad que se encarga de procesar estos puntos de referencia y se pueda asignar una clasificación a la imagen y luego el profesional de la salud pueda dar su diagnóstico más preciso.

2.5 Sprint 4

Luego de varias reuniones con el usuario se planea un cuarto sprint en el que se desarrollan varias funcionalidades para mejorar la experiencia del usuario, permitiendo lo siguiente:

- Zoom a la imagen sin perder resolución.
- Desbloquear la imagen al momento de remover los puntos de referencia.
- Desbloquear la imagen al momento de tomar el primer punto de referencia.

Cabe resaltar que aplicar esta metodología ágil a la perfección es casi imposible por diversos motivos es por ello que se puede notar en el cronograma que los sprints y las actividades se desarrollaron en tiempos un poco extensos.

3. DESARROLLO

Este capítulo muestra en detalle el desarrollo de la aplicación web para la lectura de imágenes de fondo de ojo. Además, se explica en profundidad las funcionalidades y cómo se ejecuta cada proceso. También se detallan las métricas y modelos de desarrollo aplicados para la obtención del producto final. El entorno de desarrollo integrado que se va a mostrar en las imágenes es el llamado Pycharm el cual es una herramienta especialmente diseñada para Python, ofrece una interfaz fácil de usar, soporta diferentes frameworks y ayuda a mejorar la productividad.

3.1 Casos de uso del proyecto

Para llevar a cabo la elaboración del proyecto es necesario tener un análisis y un diseño para lograr visualizar y documentar los casos de uso del sistema. Estos casos de uso son las diferentes formas en las que el usuario puede interactuar con el sistema para realizar tareas específicas.

Los casos de uso frecuentemente se componen de:

- Actores: Representados como figuras humanas o como símbolos que representan a otras entidades externas al sistema.
- Casos de uso: Representados como círculos que contienen el nombre del caso de uso.
- Relaciones: Líneas que conectan a los actores con los casos de uso para indicar cómo interactúan entre sí.

A continuación, se observa una ilustración de un caso de usuario del proyecto.

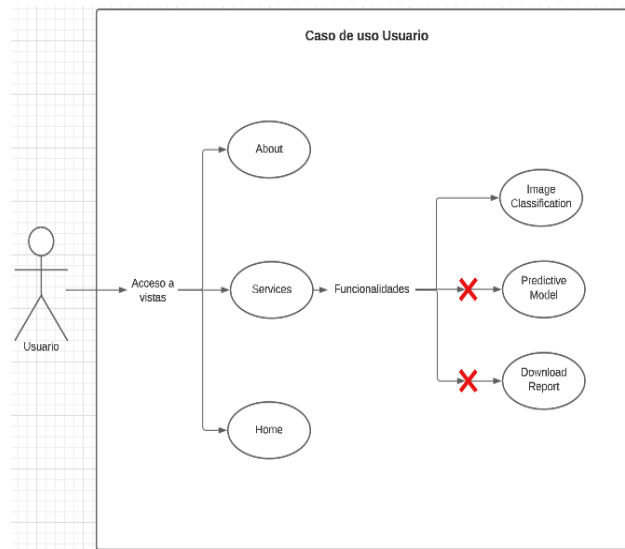


Ilustración 10 Diagrama de casos de uso del proyecto

3.2 Mockups del proyecto

Los mockups se utilizan con frecuencia como una forma de obtener una comprensión clara de la dirección del diseño y de cómo se verá y funcionará el producto final. Es debido a esto que se diseñaron los siguientes mockups para el proyecto.

El primer mockup es para la página inicial, esta página se compone por el logo, una breve descripción del funcionamiento del proyecto y botones para redirigir a las diferentes funcionalidades o secciones del proyecto.

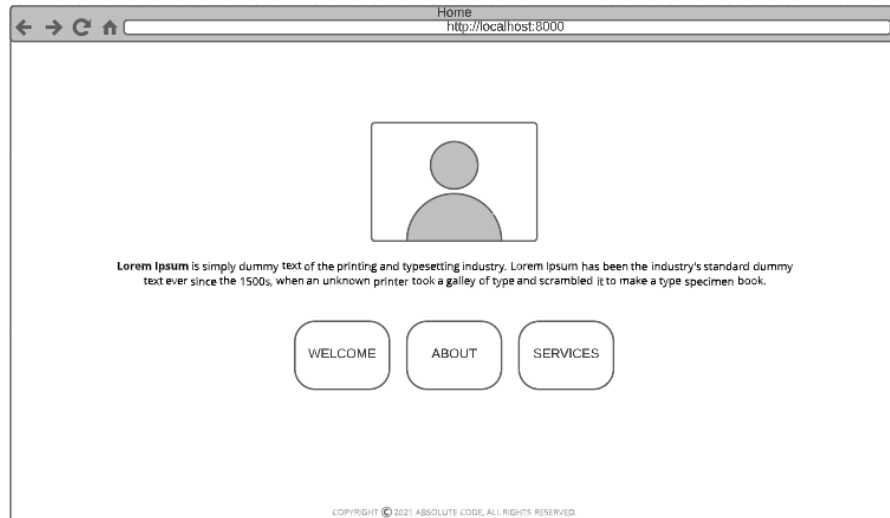


Ilustración 11 Mockup de la página inicial del proyecto

El segundo mockup se construye en base a la página de servicios, en la cual se puede observar el logo del proyecto, una breve descripción de los servicios disponibles y botones para redirigir a los diferentes servicios que se ofrecen en el proyecto.

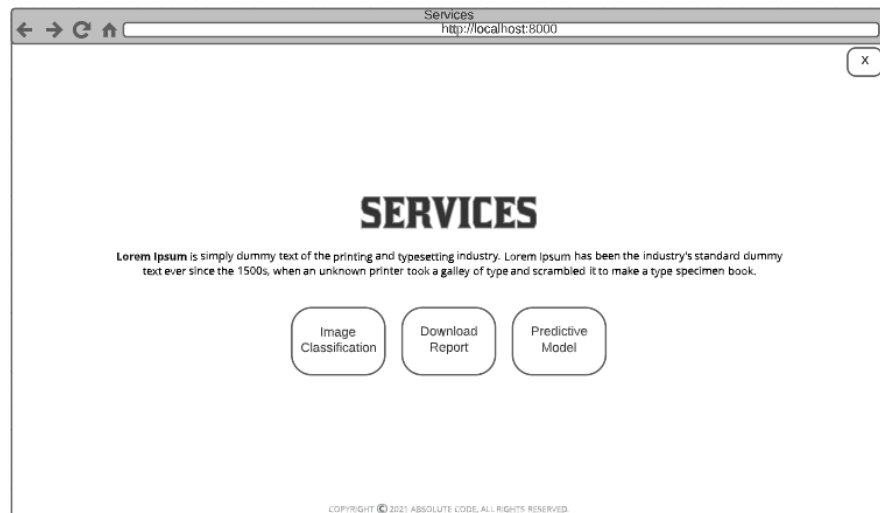


Ilustración 12 Mockup de la página de servicios del proyecto

Por último, el mockup para la página de la clasificación de imagen de fondo de

ojo, este mockup se compone del título de la funcionalidad, una breve descripción, un cuadro para visualizar la imagen con dos botones en la parte inferior del cuadro en la cual se puede cargar la imagen presionando en el botón de la izquierda y ver el color que se está seleccionado en un punto específico de la imagen, además de una tabla en la que se irán cargando los puntos seleccionados e información extra y por último al final de la página se muestra un botón para conseguir una clasificación para la imagen de fondo de ojo.

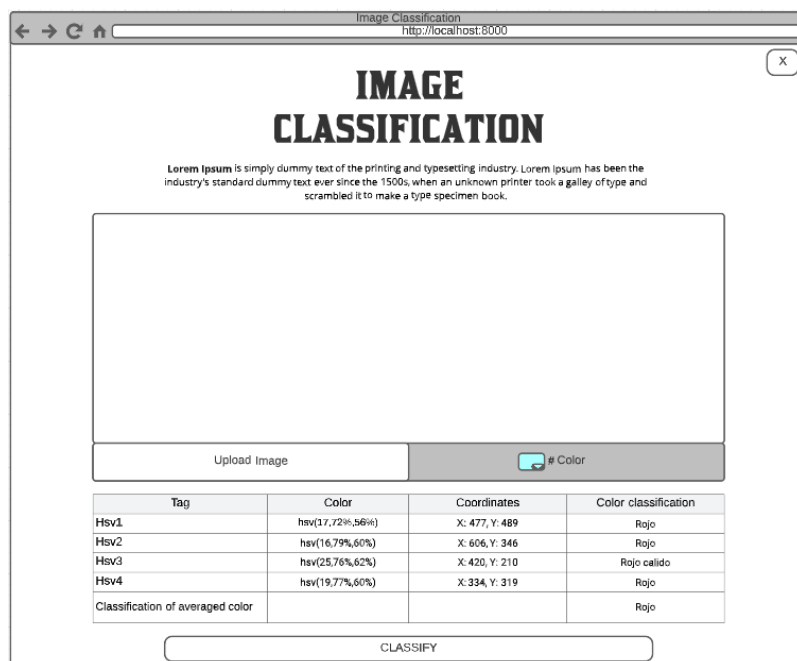


Ilustración 13 Mockup de la página de clasificación de imagen del proyecto

3.3 Arquitectura del proyecto

La arquitectura que se implementó para el desarrollo de este proyecto fue basada en el patrón de diseño MVT, el cual es el que se utiliza frecuentemente para los proyectos hechos con el framework Django y que ayuda a organizar los componentes del sistema.

3.3.1 Modelo de datos

En este proyecto el modelo de datos se refiere a la estructura de las tablas de la base de datos y a la forma en que estas tablas están relacionadas. En el archivo “models.py” se utiliza la clase “Model” de Django para definir un modelo de datos. Cada modelo de Django representa una tabla de la base de datos y cada atributo o campo del modelo representa una columna de la tabla.

Cada tabla (entidad) tiene un rol importante para que el proyecto funcione correctamente, entre esas está la tabla “UploadImage”, la cual se encarga de guardar los datos necesarios para procesar una clasificación cada vez que se carga una imagen de fondo de ojo.

Otra de las tablas de las cuales se hace y que genera Django de forma automática al generar el proyecto es la tabla “user”, la cual almacena información sobre los usuarios que se creen en el proyecto, esta se relaciona con otras tablas a través de claves foráneas para poder guardar información adicional sobre los usuarios.

La tabla “content_type” se utiliza para guardar información sobre los diferentes tipos de contenido que se puedan crear en el proyecto. Al igual que la tabla “user”, cada tipo de contenido tiene su propia entrada en esta tabla, y se relaciona con otras tablas a través de claves foráneas para almacenar información adicional sobre el contenido.

Para resumir las demás tablas que se muestran en el diagrama, la tabla “session” almacena información sobre las sesiones del usuario, la tabla “site” guarda información sobre los sitios que se están ejecutando en el proyecto y la tabla “admin_log” almacena un registro de las acciones realizadas por los usuarios con rango de administrador.

En el diagrama de la siguiente ilustración se pueden observar todas las tablas (entidades) usadas en el proyecto y sus respectivas relaciones entre sí.

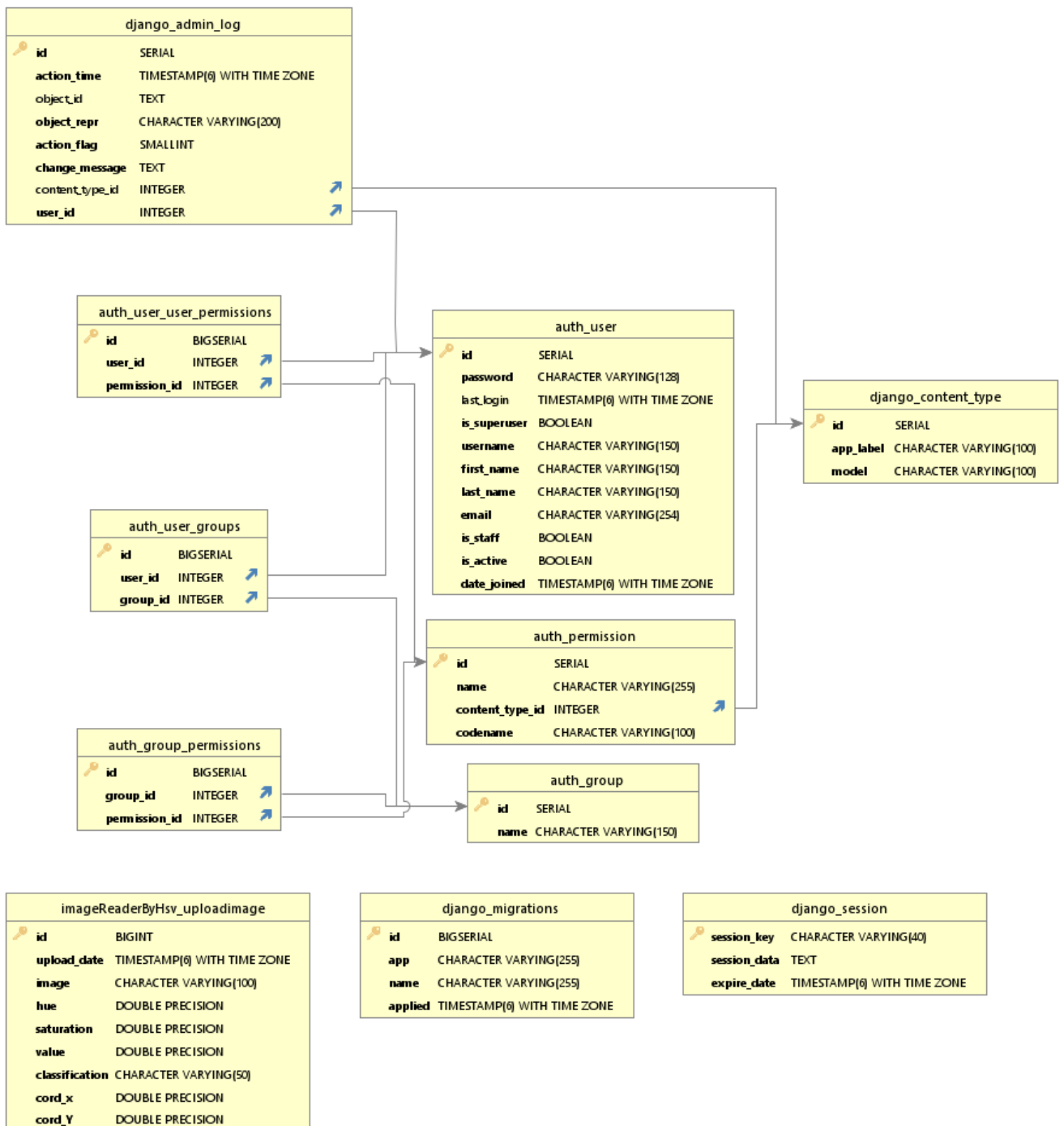


Ilustración 14 Diagrama entidad-relación

3.3.2 Estructura de carpetas del proyecto

A continuación, se podrá observar la carpeta raíz del proyecto, en la cual se encuentran otras carpetas (a las que se les llama “apps”) y archivos que gestionan los parámetros para la configuración del proyecto y así este pueda funcionar de manera estable, se encuentran también carpetas en donde se manejan las imágenes y estilos del proyecto.

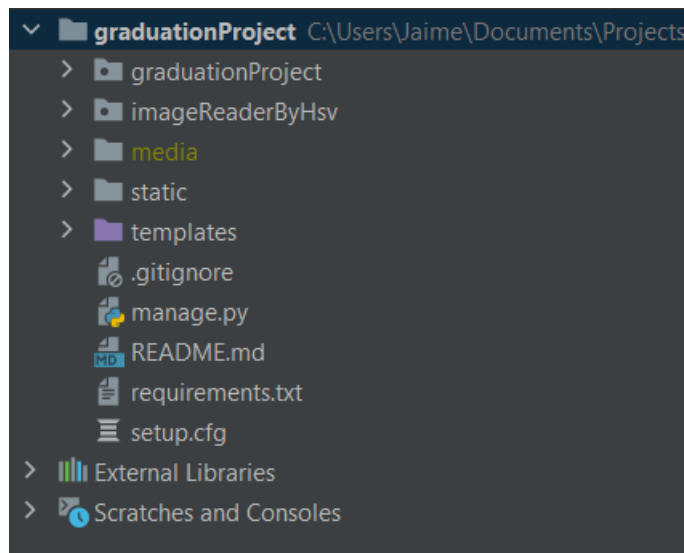


Ilustración 15 Estructura del proyecto

➤ **Carpeta (app) graduationProject**

En la siguiente ilustración se muestra la primera carpeta (app) llamada “graduationProject”, ahí se encuentra la carpeta “static” la cual contiene los archivos para dar estilos al panel de administrador del proyecto, luego están los archivos de configuración del proyecto en general los cuales se explicarán en detalle.

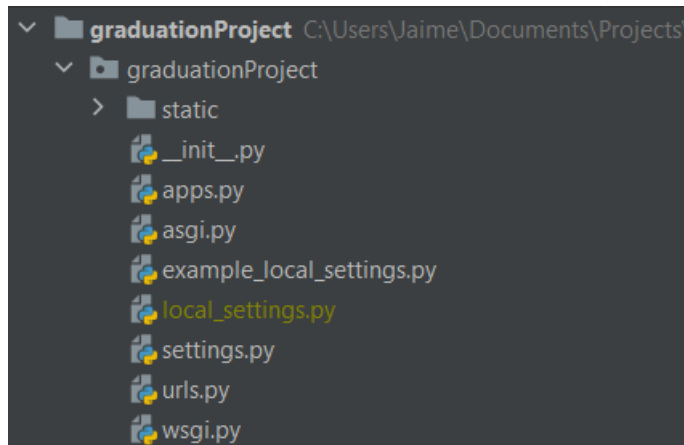


Ilustración 16 Carpeta de configuración general del proyecto

➤ Archivo apps.py

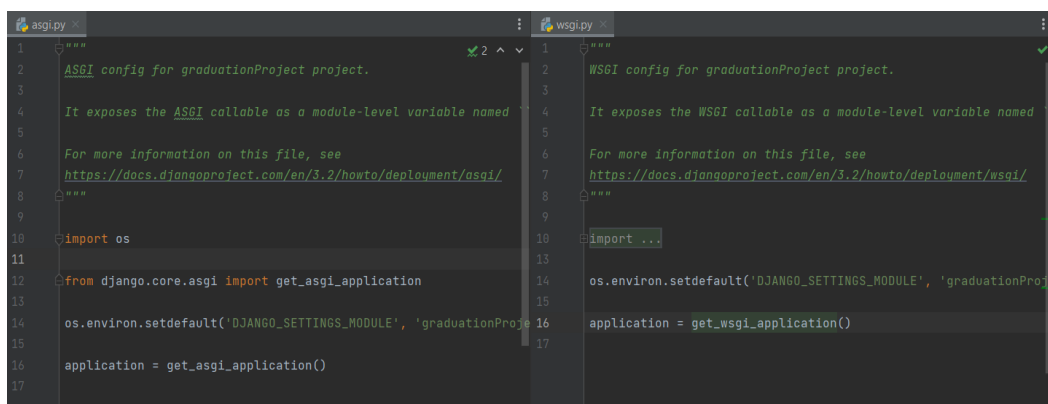
El primer archivo de configuración que se puede observar es el llamado “apps.py”, en este archivo se construyen las diferentes apps que van a existir dentro del proyecto. En la siguiente imagen se muestra que, para generar una app, se debe crear una clase (ImageReaderByHsvConfig) que herede de una clase padre que representa una aplicación Django y su configuración, además sobre escribir los atributos “name” y “verbose_name” con los valores que se elijan.

```
apps.py x
1 from django.apps import AppConfig
2
3
4 class ImageReaderByHsvConfig(AppConfig):
5     name = 'imageReaderByHsv'
6     verbose_name = "Image Reader By Hsv"
7
```

Ilustración 17 Archivo de configuración de la app

➤ Archivos ASGI.py y WSGI.py

Los archivos “asgi.py” y “wsgi.py”, acrónimos para Asynchronous Server Gateway Interface y Web Server Gateway Interface respectivamente; son estándares que se utilizan para conectar aplicaciones de servidor y cliente en el desarrollo de aplicaciones web con Python. En la siguiente imagen podemos ver cómo Django implementa la configuración de estos archivos de manera automática.



```
1 """
2 ASGI config for graduationProject project.
3
4 It exposes the ASGI callable as a module-level variable named
5
6 For more information on this file, see
7 https://docs.djangoproject.com/en/3.2/howto/deployment/asgi/
8 """
9
10 import os
11
12 from django.core.asgi import get_asgi_application
13
14 os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'graduationProject.settings')
15
16 application = get_asgi_application()
17
```

```
1 """
2 WSGI config for graduationProject project.
3
4 It exposes the WSGI callable as a module-level variable named
5
6 For more information on this file, see
7 https://docs.djangoproject.com/en/3.2/howto/deployment/wsgi/
8 """
9
10 import os
11
12 from django.core.wsgi import get_wsgi_application
13
14 os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'graduationProject.settings')
15
16 application = get_wsgi_application()
17
```

Ilustración 18 Configuración de archivos ASGI y WSGI

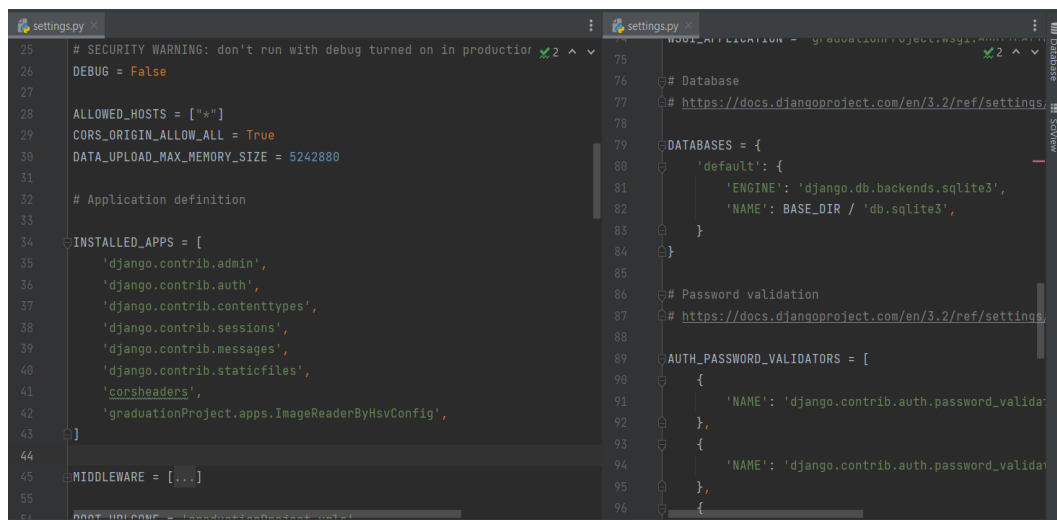
➤ Archivo settings.py

El próximo archivo de configuración es el “settings.py”, este permite personalizar y controlar bastantes aspectos del comportamiento del proyecto. Algunas de las cosas que se configuraron en este archivo fueron:

- La base de datos.
- Los idiomas y localizaciones.
- Las aplicaciones creadas para el proyecto.
- Las distintas opciones de seguridad, como la clave secreta y la configuración de seguridad SSL.
- Las opciones de desarrollo, como la activación de la depuración y

el modo de depuración.

En resumen, este archivo es capaz de personalizar muchos aspectos del proyecto, por eso es muy importante tener cuidado al editarlo ya que un mal cambio podría afectar todo el funcionamiento de manera negativa.



```
25 # SECURITY WARNING: don't run with debug turned on in production
26 DEBUG = False
27
28 ALLOWED_HOSTS = ["*"]
29 CORS_ORIGIN_ALLOW_ALL = True
30 DATA_UPLOAD_MAX_MEMORY_SIZE = 5242880
31
32 # Application definition
33
34 INSTALLED_APPS = [
35     'django.contrib.admin',
36     'django.contrib.auth',
37     'django.contrib.contenttypes',
38     'django.contrib.sessions',
39     'django.contrib.messages',
40     'django.contrib.staticfiles',
41     'corsheaders',
42     'graduationProject.apps.ImageReaderByHsvConfig',
43 ]
44
45 MIDDLEWARE = [...]
46
75 # Database
76 # https://docs.djangoproject.com/en/3.2/ref/settings/#databases
77
78
79 DATABASES = {
80     'default': {
81         'ENGINE': 'django.db.backends.sqlite3',
82         'NAME': BASE_DIR / 'db.sqlite3',
83     }
84 }
85
86 # Password validation
87 # https://docs.djangoproject.com/en/3.2/ref/settings/#auth-password-validators
88
89 AUTH_PASSWORD_VALIDATORS = [
90     {
91         'NAME': 'django.contrib.auth.password_validation.
92     },
93     {
94         'NAME': 'django.contrib.auth.password_validation.
95     },
96     {
```

Ilustración 19 Archivo de configuración settings.py

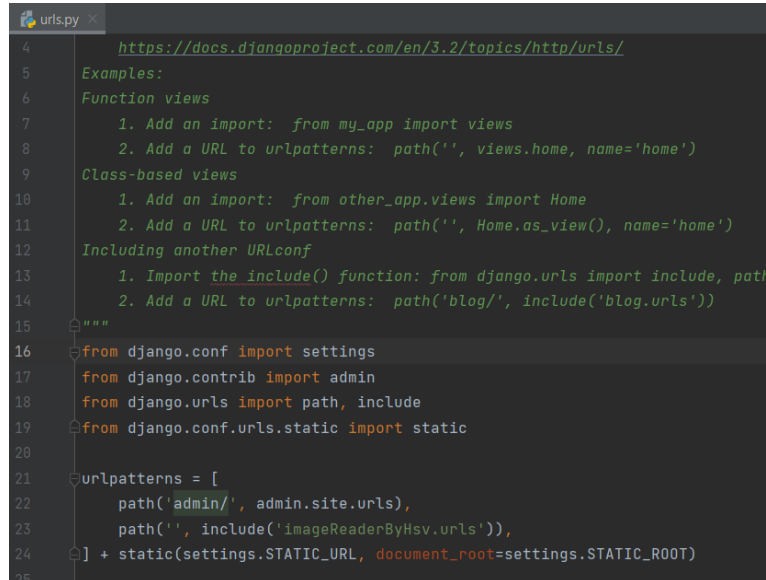
➤ Archivo urls.py

El archivo “urls.py” es otro archivo de configuración en Django utilizado para mapear las URL del proyecto a las vistas (funciones o clases que manejan las solicitudes del usuario y devuelve una respuesta) del mismo.

Este archivo normalmente se compone de dos partes principales:

- Una lista de patrones URL: La cual es una lista de expresiones regulares que se usan para comparar las URL de la solicitud de usuario. Todo patrón URL se asocia con una vista.
- Una lista de vistas: Que se ejecutan cuando se accede a una URL que coincide con un patrón URL.

A continuación, una imagen del archivo “urls.py” del proyecto.



```
4 https://docs.djangoproject.com/en/3.2/topics/http/urls/
5 Examples:
6 Function views
7     1. Add an import: from my_app import views
8     2. Add a URL to urlpatterns: path('', views.home, name='home')
9 Class-based views
10    1. Add an import: from other_app.views import Home
11    2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
12 Including another URLconf
13    1. Import the include() function: from django.urls import include, path
14    2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
15 """
16 from django.conf import settings
17 from django.contrib import admin
18 from django.urls import path, include
19 from django.conf.urls.static import static
20
21 urlpatterns = [
22     path('admin/', admin.site.urls),
23     path('', include('imageReaderByHsv.urls')),
24     + static(settings.STATIC_URL, document_root=settings.STATIC_ROOT)
25 ]
```

Ilustración 20 Archivo de configuración urls.py

➤ Carpeta (app) imageReaderByHsv

La próxima carpeta (app) de la cual se hablará es la de “imageReaderByHsv”, que es la que tiene el comportamiento de la columna vertebral del proyecto debido a que en esta carpeta se implementan las vistas y los modelos.

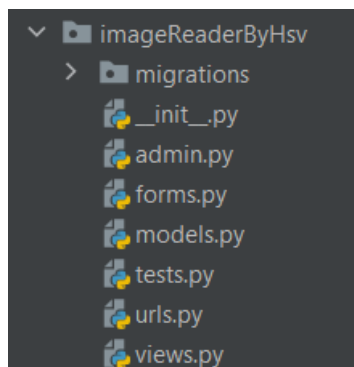
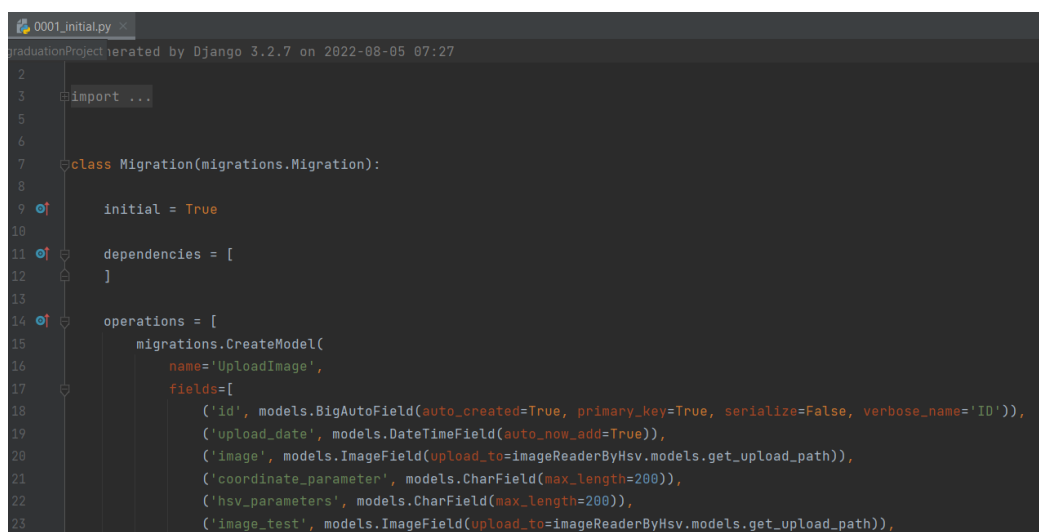


Ilustración 21 Carpeta (app) imageReaderByHsv

➤ Carpeta migrations

Dentro de la app “imageReaderByHsv”, se encuentra la primera carpeta que tiene por nombre “migrations”, esta carpeta almacena los archivos de migración, que son archivos que contienen información sobre cambios realizados en el modelo de datos del proyecto.

Esta carpeta “migrations” se encuentra dentro de cada app que se genere en Django y contiene un archivo de migración para cada cambio realizado en el modelo de datos de la app. Estos archivos de migración tienen un formato específico y contienen instrucciones que Django puede seguir para aplicar el cambio en la base de datos. Se usan estos archivos para mantener la base de datos sincronizada con el modelo de datos de la app y para que sea más fácil aplicar las modificaciones a dicho modelo.



```
0001_initial.py
graduationProject generated by Django 3.2.7 on 2022-08-05 07:27
2
3 import ...
4
5
6
7 class Migration(migrations.Migration):
8
9     initial = True
10
11     dependencies = [
12     ]
13
14     operations = [
15         migrations.CreateModel(
16             name='UploadImage',
17             fields=[
18                 ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
19                 ('upload_date', models.DateTimeField(auto_now_add=True)),
20                 ('image', models.ImageField(upload_to=imageReaderByHsv.models.get_upload_path)),
21                 ('coordinate_parameter', models.CharField(max_length=200)),
22                 ('hsv_parameters', models.CharField(max_length=200)),
23                 ('image_test', models.ImageField(upload_to=imageReaderByHsv.models.get_upload_path)),
```

Ilustración 22 Archivo de migración en Django

➤ Archivo __init__.py

El próximo archivo a detallar es el “__init__.py”, el cual es un archivo especial que usa para indicar que una carpeta o directorio debe tratarse como un módulo Python. El archivo puede estar vacío o puede tener código Python que

se va a ejecutar una vez se importe el módulo.

Este archivo es muy útil en Django cuando se trabaja con aplicaciones, esto debido a que se utiliza para inicializar una aplicación y hacer que esté disponible en todo el proyecto.

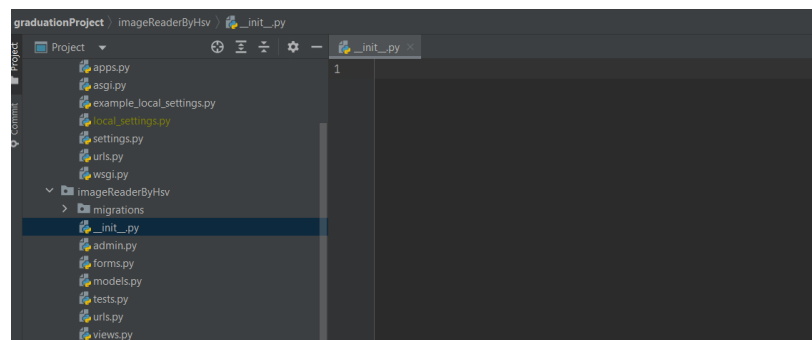


Ilustración 23 Archivo de init de la aplicación imageReaderByHsv

➤ **Archivo models.py**

Para entender los demás archivos en esta carpeta o módulo primero se debe entender el uso del archivo “models.py”, este se utiliza para definir los modelos de la aplicación. Los modelos se generan con clases que representan las entidades o elementos que se almacenan en la base de datos de la aplicación.

Este archivo puede incluir métodos o funciones que se usan para trabajar con los modelos.

```
models.py
15
16 class UploadImage(models.Model):
17     upload_date = models.DateTimeField(auto_now_add=True)
18     image = models.ImageField(upload_to=get_upload_path)
19     coordinate_parameter = models.CharField(max_length=200)
20     hsv_parameters = models.CharField(max_length=200)
21     image_test = models.ImageField(upload_to=get_upload_path)
22     classification = models.CharField(max_length=50, choices=CHOICE_CLASSIFICATION)
23     uuid = models.CharField(max_length=70, null=True, blank=True)
24
25     def __str__(self):
26         return f"Classification: {self.classification} - Parameters: {self.hsv_parameters}"
27
28     def calculate_classification(self, degree: float, saturation: float):...
43
44     def get_parameter_classification(self):
45         classifications_dictionary = {}
46         json_params = self.get_json_from_hsv_parameters()
47         if isinstance(json_params, dict):
48             for key in json_params:
49                 classification = self.calculate_classification(int(json_params.get(key)[0]),
50                                                             int(json_params.get(key)[1]))
51                 classifications_dictionary.setdefault(key, classification)
UploadImage > get_parameter_classification()
```

Ilustración 24 Modelo para la carga de imagen de la aplicación

En la imagen anterior se observa cómo se construyó el modelo UploadImage, definiendo sus campos y algunas de sus funciones o métodos, una vez definido, Django utiliza el modelo para crear la tabla en la base de datos y así poder realizar operaciones de CRUD (Create, Read, Update and Delete) en la base de datos.

➤ **Archivo admin.py**

Una vez comprendido el archivo “models.py”, es posible pasar a detallar el archivo “admin.py” de la aplicación, el cual se utiliza para registrar los modelos de una aplicación en la interfaz de administración, que es la que se utiliza para gestionar y administrar la base de datos de la aplicación.

En la siguiente imagen se puede observar cómo se importa el modelo que se construyó y cómo se registra para poder visualizarlo en la interfaz de administración.

```
admin.py x
1 from django.contrib import admin
2 from imageReaderByHsv.models import UploadImage
3
4
5 admin.site.register(UploadImage)
6
```

Ilustración 25 Archivo admin.py de la aplicación

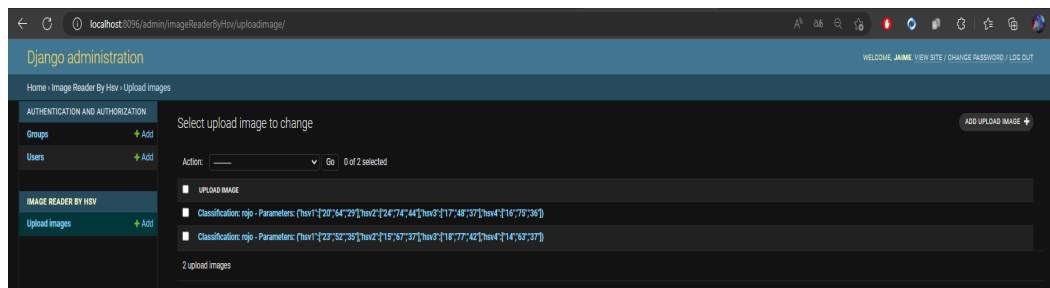


Ilustración 26 Interfaz de administración de la aplicación

➤ **Archivo forms.py**

Otro archivo importante en este proyecto es el “forms.py”, se usa para crear formularios, que en Django son una herramienta útil para recopilar y validar la entrada de datos de un usuario a través de una página web.

En este archivo se definen las clases de formulario y sus campos o atributos, también puede incluir métodos o funciones que se usan para integrarse con los formularios. Una vez definidas las clases en el archivo forms.py, se pueden utilizar en las views (vistas) que se hayan creado en el proyecto para así procesar y validar la entrada de datos del usuario.

En la siguiente imagen se puede observar cómo se implementó el archivo forms.py en el proyecto creando una clase “ReaderImageForm” junto con sus

respectivas funciones las cuales se encargan de procesar todos los datos que se toman una vez se carga una imagen de fondo de ojo.

```
forms.py x
10 class ReaderImageForm(forms.ModelForm):
11     hsv_parameters = forms.CharField(required=True, max_length=200)
12     coordinate_parameter = forms.CharField(required=True, max_length=200)
13     image_test = forms.CharField(required=True)
14     image = forms.ImageField(
15         required=True,
16         widget=forms.FileInput(attrs={'class': 'form-control'})
17     )
18     classification = forms.ChoiceField(
19         required=False,
20         choices=UploadImage().CHOICE_CLASSIFICATION
21     )
22
23 class Meta:
24     model = UploadImage
25     fields = ['image', 'hsv_parameters', 'classification', 'uuid', 'image_test']
26
27 def clean_hsv_parameters(self):...
```

Ilustración 27 Archivo forms.py

➤ **Archivo views.py**

Una de las funcionalidades más importantes la lleva el archivo “views.py”, este se usa para dar definición a las vistas de la aplicación, a su vez estas vistas son métodos que especifican cómo manejar las solicitudes HTTP para así dar una respuesta de forma correcta al usuario.

Además, este archivo se usa para funcionalidades CRUD en la base de datos y para utilizar formularios y modelos definidos en otros archivos como forms.py y models.py tal como se puede observar en la siguiente imagen.

```
views.py
1 from json import loads
2 from django.views import View
3 from django.shortcuts import render
4 from django.views.generic import TemplateView
5 from imageReaderByHsv.models import UploadImage
6 from imageReaderByHsv.forms import ReaderImageForm
7
8
9 class HomeView(TemplateView):
10     template_name = "image_reader_by_hsv/index.html"
11
12
13 class ServicesView(TemplateView):
14     template_name = "image_reader_by_hsv/services.html"
15
16
17 class ReaderImageView(View):
18     form_class = ReaderImageForm
19     template_name = "image_reader_by_hsv/reader_image.html"
20
21     def get(self, request, *args, **kwargs):...
24
25     def post(self, request, *args, **kwargs):...
```

Ilustración 28 Archivo views.py

➤ **Archivo tests.py**

Cuando se habla de un código limpio también se hace referencia a los casos de pruebas por eso también es relevante incluir el archivo “tests.py”, en el cual se añaden las pruebas automatizadas para verificar el comportamiento de la aplicación. Con las pruebas automatizadas se puede asegurar de que la aplicación funcione correctamente, de manera consistente y ahorra tiempo y esfuerzo al momento de la depuración de errores.

A continuación, se podrá observar el caso de prueba para la creación de un usuario que se aplicó en la aplicación. Se usó la función “assertEqual” y “assertTrue” para verificar que se creó el usuario correctamente y que la contraseña se logró guardar de forma segura respectivamente.

```
tests.py x
1 from django.test import TestCase
2 from django.contrib.auth.models import User
3
4 # Create your tests here.
5
6
7 class UserModelTest(TestCase):
8     def test_user_creation(self):
9         # Create user
10        user = User.objects.create_user(
11            username='testuser',
12            password='testpass'
13        )
14
15        # Check the user was created successfully
16        self.assertEqual(user.username, 'testuser')
17        self.assertTrue(user.check_password('testpass'))
18
```

Ilustración 29 Archivo tests.py

➤ **Carpeta media**

La carpeta llamada “media” tiene la función de almacenar los archivos cargados por el usuario del proyecto. Esta carpeta puede incluir imágenes, archivos de audio o vídeo, documentos y cualquier otro tipo de archivo que se necesite guardar y acceder desde el proyecto.

Para poder usar esta carpeta en el proyecto, se configura en el archivo settings.py estableciendo las variables “MEDIA_ROOT” y “MEDIA_URL”. La primera variable hace referencia a la ruta en donde se guardan los archivos cargados, mientras que la variable “MEDIA_URL” se usa para acceder a tales archivos.



Ilustración 30 Carpeta media del proyecto

➤ Carpeta static

“static” es el nombre de otra carpeta importante a detallar, se utiliza para guardar archivos estáticos que se usan en el proyecto, como hojas de estilo (CSS), scripts JavaScript y archivos de imagen. Los archivos dentro de esta carpeta se utilizan para darle estilo y funcionalidad al proyecto, además, no cambian con el tiempo y no dependen de la entrada del usuario.

Para usar esta carpeta, se configura en el archivo settings.py las variables “STATIC_ROOT” y “STATIC_URL”, las cuales representan en donde se almacenan los archivos estáticos y la URL que se usa para acceder a ellos, respectivamente.

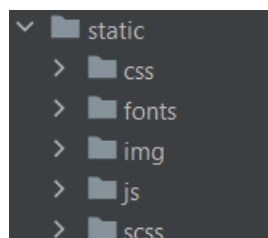


Ilustración 31 Carpeta static del proyecto

➤ Carpeta templates

Siguiendo el patrón de diseño “MVT”, la carpeta “templates” se usa en el directorio raíz del proyecto, se utiliza para guardar las plantillas HTML que se usan para generar las páginas del proyecto. Las plantillas son archivos que contienen código HTML estático y marcadores que indican dónde se deben insertar datos dinámicos en tiempo de ejecución.

Cuando se habla de datos dinámicos en tiempos de ejecución, se hace referencia al uso de Jinja2, el cual es un motor de plantillas para Python que se usa para crear plantillas HTML dinámicas. Estas plantillas permiten incluir variables, bucles y condicionales en el código HTML estático para poder generar las páginas webs dinámicas.

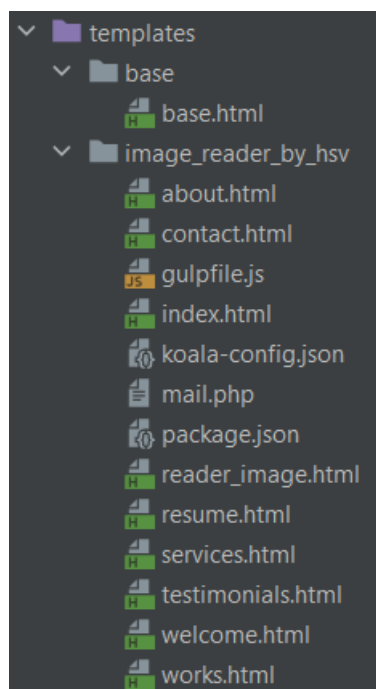


Ilustración 32 Carpeta templates del proyecto

```

64 <th COLSPAN="2" class="primary" scope="row">Coordinates</th>
65 <th COLSPAN="2" class="primary" scope="row">Color classification</th>
66 </tr>
67 <tr>
68 <!-- Tag #1 -->
69 <td COLSPAN="1" class="text-white text-left" scope="row">Hsv1</td>
70
71 <!-- Color #1 -->
72 {% if hsv1_color %}
73 <td COLSPAN="2" class="text-left" id="hsv1">{{ hsv1_color }}</td>
74 {% else %}
75 <td COLSPAN="2" class="text-left" id="hsv1">N/A</td>
76 {% endif %}
77
78 <!-- Coordinates #1 -->
79 {% if coordinate1 %}
80 <td COLSPAN="2" class="text-left" id="coordinate1">{{ coordinate1 }}</td>
81 {% else %}
82 <td COLSPAN="2" class="text-left" id="coordinate1">N/A</td>
83 {% endif %}
84
85 <!-- Classification #1 -->
86 {% if hsv1 %}

```

Ilustración 33 Plantilla dinámica reader_image.html

➤ Archivo .gitignore

Al utilizar un controlador de versiones como gitlab, es importante evitar subir archivos innecesarios o confidenciales al repositorio, es por esto que en el proyecto se hace uso del archivo “.gitignore”, este contiene una lista de nombres de archivos o directorios que Git debe ignorar al momento de subir cambios.

```

1 # Created by https://www.toptal.com/developers/gitignore/api/python,virtualenv,django,pycharm+all_git
2 # Edit at https://www.toptal.com/developers/gitignore?templates=python,virtualenv,django,pycharm+all_git
3
4 ### Django ###
5 *.log
6 *.pot
7 *.pyc
8 __pycache__/
9 local_settings.py
10 db.sqlite3
11 db.sqlite3-journal
12 media
13
14 # If your build process includes running collectstatic, then you probably don't need or want to include staticfiles/
15 # in your Git repository. Update and uncomment the following line accordingly.
16 # <django-project-name>/staticfiles/
17
18 ### Django.Python Stack ###
19 # Byte-compiled / optimized / DLL files
20 *.py[co]d
21 *$py.class
22
23 # C extensions
24 *.so

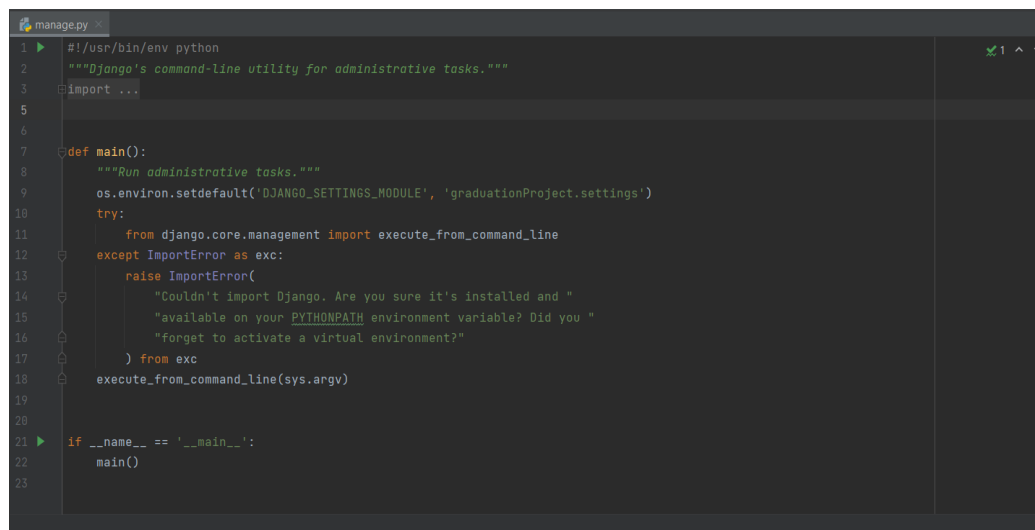
```

Ilustración 34 Archivo .gitignore

➤ Archivo manage.py

Uno de los archivos más relevantes en el proyecto es el archivo “manage.py”, este es un script de Python que se incluye cada vez que se genera un proyecto en Django, se utiliza para ejecutar tareas administrativas y de desarrollo. Algunas de estas tareas y sus comandos son:

- Iniciar el servidor: `python manage.py runserver`
- Crear una aplicación: `python manage.py startapp app_name`
- Ejecutar migraciones de base de datos: `python manage.py migrate`
- Crear un superusuario: `python manage.py createsuperuser`
- Recopilar archivos estaticos: `python manage.py collectstatic`



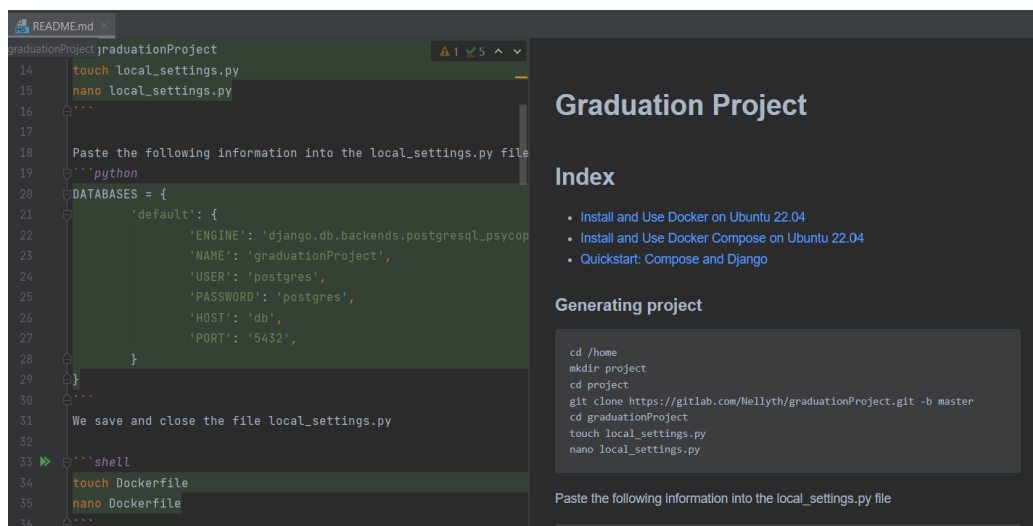
```
1 #!/usr/bin/env python
2 """Django's command-line utility for administrative tasks."""
3 import ...
4
5
6
7 def main():
8     """Run administrative tasks."""
9     os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'graduationProject.settings')
10    try:
11        from django.core.management import execute_from_command_line
12    except ImportError as exc:
13        raise ImportError(
14            "Couldn't import Django. Are you sure it's installed and "
15            "available on your PYTHONPATH environment variable? Did you "
16            "forget to activate a virtual environment?"
17        ) from exc
18    execute_from_command_line(sys.argv)
19
20
21 if __name__ == '__main__':
22     main()
23
```

Ilustración 35 Archivo manage.py

➤ Archivo README.md

Documentar el proyecto también es una parte vital para que futuros desarrolladores puedan implementar mejores de manera más sencilla, es por eso que se crea el archivo “README.md”, que se utiliza para proporcionar información sobre el propósito y el funcionamiento del proyecto, así como también para dar instrucciones sobre cómo instalar y usar el proyecto.

Este archivo se escribe en el formato de marcado de texto plano Markdown, el cual es un lenguaje de etiquetado sencillo que se utiliza para dar formato al texto, así como en la siguiente ilustración.



The screenshot shows a terminal window with a dark theme. The terminal output is as follows:

```
graduationProject:graduationProject
14 touch local_settings.py
15 nano local_settings.py
16
17
18 Paste the following information into the local_settings.py file
19
20 python
21 DATABASES = {
22     'default': {
23         'ENGINE': 'django.db.backends.postgresql_psycopg2',
24         'NAME': 'graduationProject',
25         'USER': 'postgres',
26         'PASSWORD': 'postgres',
27         'HOST': 'db',
28         'PORT': '5432',
29     }
30 }
31 We save and close the file local_settings.py
32
33 shell
34 touch Dockerfile
35 nano Dockerfile
36
```

On the right side of the terminal, there is a preview of the README.md file content:

Graduation Project

Index

- Install and Use Docker on Ubuntu 22.04
- Install and Use Docker Compose on Ubuntu 22.04
- Quickstart: Compose and Django

Generating project

```
cd /home
mkdir project
cd project
git clone https://gitlab.com/Mellyth/graduationProject.git -b master
cd graduationProject
touch local_settings.py
nano local_settings.py
```

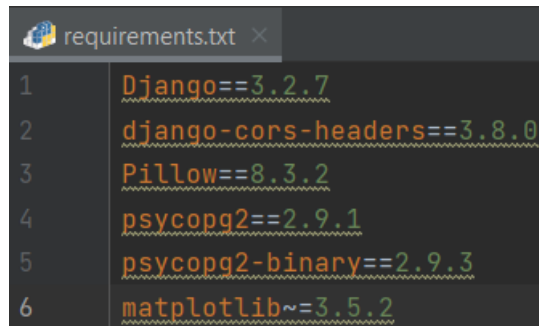
Paste the following information into the local_settings.py file

Ilustración 36 Archivo README.md

➤ Archivo requirements.txt

Como parte de la documentación también se tiene el archivo “requirements.txt” que es un archivo de texto que se usa para guardar una lista de las dependencias del proyecto. Se le conoce como dependencias a los paquetes de Python que se necesitan para que el proyecto funcione de manera correcta.

Para instalar todas las dependencias del archivo “requirements.txt”, se usa el comando “pip install -r requirements.txt”. Este comando instalará en el servidor o en el equipo que se esté corriendo el proyecto todas las dependencias especificadas en el archivo.



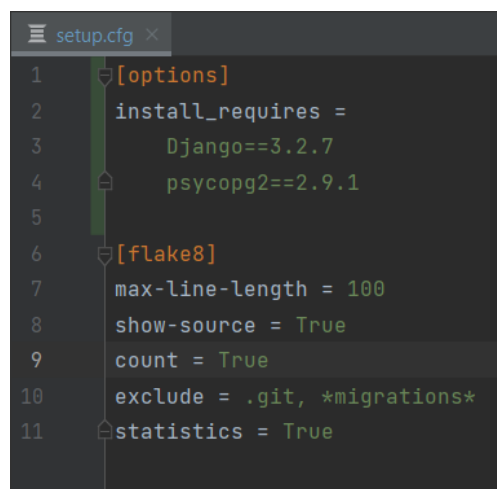
```
requirements.txt
1 Django==3.2.7
2 django-cors-headers==3.8.0
3 Pillow==8.3.2
4 pycopg2==2.9.1
5 pycopg2-binary==2.9.3
6 matplotlib~=3.5.2
```

Ilustración 37 Archivo requirements.txt

➤ Archivo setup.cfg

Por último, pero no menos importante, se tiene el archivo “setup.cfg”, este es un archivo de configuración que se utiliza frecuentemente con el módulo “setuptools” de Python para especificar cómo se deben construir y distribuir paquetes de Python.

Este archivo también se utiliza para especificar opciones de configuración para la instalación del proyecto. En la siguiente imagen se puede observar cómo se especifica que el proyecto necesita Django 3.2.7 y pycopg2 2.9.1, entre otras configuraciones.



```
setup.cfg
1 [options]
2   install_requires =
3     Django==3.2.7
4     pycopg2==2.9.1
5
6 [flake8]
7   max-line-length = 100
8   show-source = True
9   count = True
10  exclude = .git, *migrations*
11  statistics = True
```

Ilustración 38 Archivo setup.cfg

4. RESULTADOS

En este capítulo se detalla los resultados de aplicar la metodología Scrum y el cumplimiento de los sprint definidos en la planeación. Adicionalmente, se presenta la aplicación final, junto con los resultados de las reuniones con los usuarios de la aplicación web recibiendo la retroalimentación.

4.1 Página home del proyecto

Esta es la página principal del proyecto, se desarrolla con el motivo de que sea atractiva y fácil de usar para el usuario. Se puede observar el logo del equipo de desarrollo, una breve descripción de la función del proyecto y diferentes botones para redirigir al usuario al servicio que prefiera utilizar.

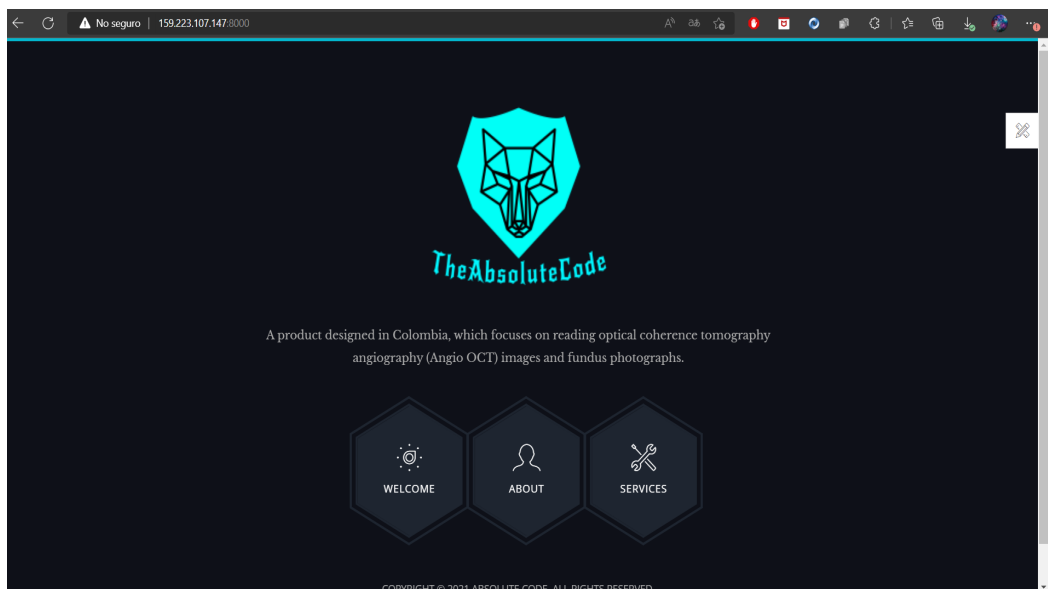


Ilustración 39 Página home del proyecto

5.2 Página de servicios

En la página de servicios se muestra también el logo y una breve descripción de su funcionalidad, pero en esta página de los botones que se encargan de redirigir al usuario, solo funcionan el botón de “clasificación de imagen” debido a que el botón para “descargar reporte” y para entrenar el “modelo predictivo” aún está en proceso de desarrollo.

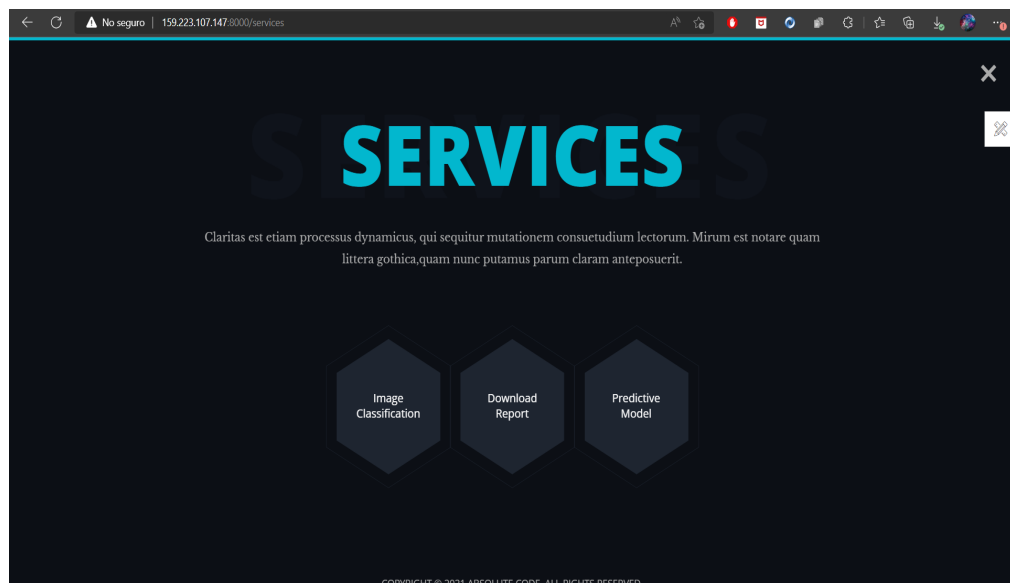


Ilustración 40 Página de servicios

5.3 Página para la clasificación de imagen de fondo de ojo y su resultado

En la página de clasificación de imagen se encuentran varias funcionalidades para facilitarle todo el proceso al usuario y sea un poco más intuitiva, entre las funcionalidades se puede encontrar:

- Carga de imagen.
- Procesamiento de la imagen para obtener una clasificación en la escala Teherán-Morales.
- Tabla con la información de los puntos seleccionados en la

imagen cargada.

El flujo de trabajo para obtener la clasificación de una imagen de fondo de ojo en la aplicación web es bastante sencillo como se pudo observar después de todo, alcanzando así el objetivo de facilitar el trabajo del usuario.

A continuación, ilustraciones de la página de clasificación de imagen y su resultado:

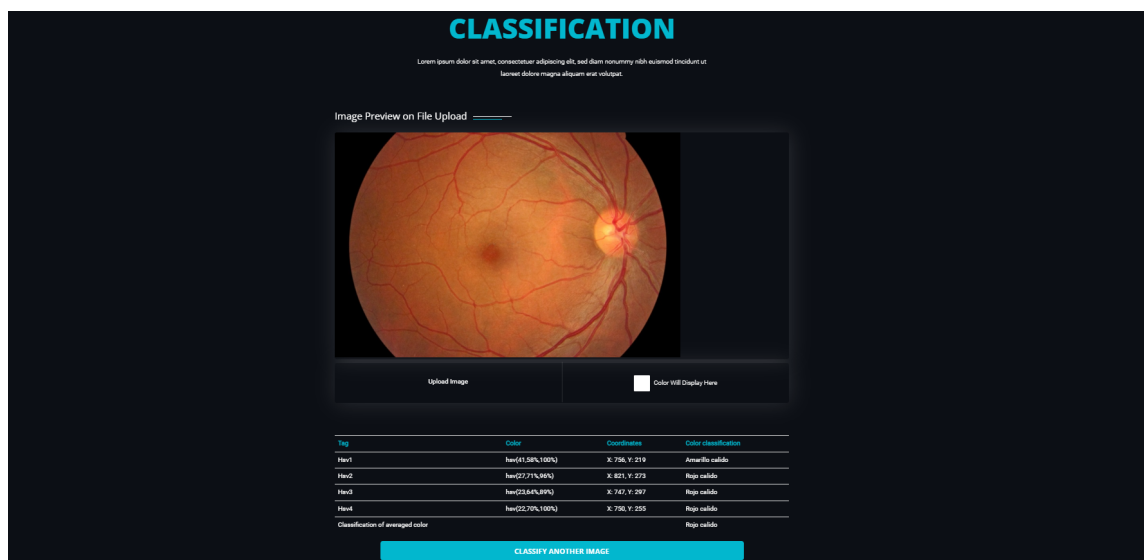


Ilustración 41 Página para la clasificación de imagen de fondo de ojo

Para tener mayor contexto sobre los resultados de la clasificación se explicará el significado de cada columna.

- Tag: Columna que hace referencia al nombre de los puntos seleccionados en la imagen de fondo de ojo.
- Color: Muestra el valor en el sistema HSV para cada punto seleccionado procesando el click dado en la imagen de fondo de ojo empezando con

una función en JavaScript llamada “getColorMouseClicked” esta función maneja el evento de clic del mouse en la imagen, realiza una serie de acciones relacionadas con la obtención y almacenamiento del valor hexadecimal del color seleccionado y luego llamando a otras acciones que convierten ese valor hexadecimal a HSV.

- **Coordinates:** En esta columna están los valores de las coordenadas en X y Y que fueron seleccionadas en la imagen de fondo de ojo y que serán guardadas en la base de datos. También funciona internamente con la función en JavaScript “getColorMouseClicked” para en el proceso obtener las coordenadas y mostrarlas en esta columna.
- **Color classification:** Columna en la se muestra la clasificación del color del punto seleccionado en la imagen de fondo de ojo, la cual se calcula internamente en base a la escala Teherán – Morales.

Los colores para esta clasificación son: Rojo oscuro, Rojo, Rojo cálido, Naranja y Amarillo cálido. Para calcular los valores de esta columna se utiliza una función llamada “calculate_classification” que toma dos argumentos: degree y saturation. La función devuelve una clasificación de un color basada en los valores de matiz (tono) y saturación dados.

5.4 Pruebas automatizadas aplicadas a la aplicación web

Para garantizar el correcto funcionamiento de la aplicación web, se generaron varios conjuntos de pruebas automatizadas.

- **ReaderImageFormTest**

Esta clase de prueba tiene varios métodos de prueba que verifican la funcionalidad del formulario ReaderImageForm, que se utiliza para

cargar imágenes y sus parámetros HSV y coordenadas. El método setUp() se ejecuta antes de cada método de prueba y crea una imagen de 1x1 píxel blanca y establece los datos del formulario. Los métodos de prueba a continuación verifican que el formulario sea válido, que se detecten los parámetros HSV y coordenadas no válidas y que se detecte una clasificación no válida.

```
class ReaderImageFormTest(TestCase):
    def setUp(self):
        # Crear una imagen de 1x1 píxel blanca
        img = Image.new('RGB', (1, 1), color='white')
        img_bytes = BytesIO()
        img.save(img_bytes, format='png')
        img_bytes.seek(0)

        self.image_file = SimpleUploadedFile('test_image.png',
img_bytes.read(), content_type='image/png')
        self.form_data = {
            'hsv_parameters': json.dumps(
                {"hsv1": ["41", "58", "100"], "hsv2": ["27", "71",
"96"], "hsv3": ["23", "64", "89"],
                "hsv4": ["22", "70", "100"]}),
            'coordinate_parameter': json.dumps(
                [{"x": 756, "y": 219}, {"x": 821, "y": 273}, {"x":
747, "y": 297}, {"x": 750, "y": 255}]),
            'image_test':
'R01GOD1hAQABAIAAAP//wAAACwAAAAAQABAAACakQBADs=',
            'image': self.image_file,
            'classification': UploadImage.ROJO
        }

    def test_valid_form(self):
        form = ReaderImageForm(data=self.form_data,
files={'image': self.form_data['image']})
        self.assertTrue(form.is_valid())
```

```

def test_invalid_hsv_parameters(self):
    self.form_data['hsv_parameters'] = json.dumps({})
    form = ReaderImageForm(data=self.form_data,
files={'image': self.form_data['image']})
    self.assertFalse(form.is_valid())
    self.assertEqual(
        form.errors['hsv_parameters'][0],
        'Invalid data, it is necessary to select 4 colored
inputs.'
    )

def test_invalid_coordinate_parameter(self):
    self.form_data['coordinate_parameter'] = json.dumps({})
    form = ReaderImageForm(data=self.form_data,
files={'image': self.form_data['image']})
    self.assertFalse(form.is_valid())
    self.assertEqual(
        form.errors['coordinate_parameter'][0],
        'Invalid data, it is necessary to receive 4
coordinates.'
    )

def test_invalid_classification(self):
    self.form_data['hsv_parameters'] = json.dumps({'color1':
[0, 0, 0], 'color2': [255, 255, 255]})
    self.form_data['classification'] =
'invalid_classification'
    form = ReaderImageForm(data=self.form_data,
files={'image': self.form_data['image']})
    self.assertFalse(form.is_valid())
    self.assertEqual(
        form.errors['classification'][0],
        'Select a valid choice. invalid_classification is not
one of the available choices.'
    )

def test_clean_method(self):

```

```

        form = ReaderImageForm(data=self.form_data,
files={'image': self.form_data['image']})
        self.assertTrue(form.is_valid())
        cleaned_data = form.clean()
        self.assertEqual(cleaned_data['classification'],
UploadImage.ROJO)

    def test_clean_hsv_parameters(self):
        form = ReaderImageForm(data=self.form_data,
files={'image': self.form_data['image']})
        self.assertTrue(form.is_valid())
        cleaned_data = form.clean_hsv_parameters()
        self.assertEqual(cleaned_data,
self.form_data['hsv_parameters'])

    def test_clean_coordinate_parameter(self):
        form = ReaderImageForm(data=self.form_data,
files={'image': self.form_data['image']})
        self.assertTrue(form.is_valid())
        cleaned_data = form.clean_coordinate_parameter()
        self.assertEqual(cleaned_data,
self.form_data['coordinate_parameter'])

    def test_image_field_required(self):
        form = ReaderImageForm(data=self.form_data, files={})
        self.assertFalse(form.is_valid())
        self.assertIn('image', form.errors)

```

- HomeViewTestCase

Solo tiene un método de prueba que verifica que la vista de inicio (HomeView) se cargue correctamente. El método de prueba verifica si la respuesta tiene un código de estado 200 y si se usa la plantilla index.html.

```

class HomeViewTestCase(TestCase):
    def test_home_view(self):

```

```

url = reverse('Image Reader By Hsv:HomeView')
response = self.client.get(url)

self.assertEqual(response.status_code, 200)
self.assertTemplateUsed(response,
'image_reader_by_hsv/index.html')

```

- **ServicesViewTestCase**

Clase de prueba parecida a la anterior solo que esta verifica que la vista de servicios (ServicesView) se cargue correctamente. El método de prueba verifica si la respuesta tiene un código de estado 200 y si se usa la plantilla services.html.

```

class ServicesViewTestCase(TestCase):
    def test_services_view(self):
        url = reverse('Image Reader By Hsv:ServicesView')
        response = self.client.get(url)

        self.assertEqual(response.status_code, 200)
        self.assertTemplateUsed(response,
'image_reader_by_hsv/services.html')

```

- **ReaderImageViewTestCase**

Define varias pruebas unitarias que comprueban diferentes aspectos del comportamiento de la vista ReaderImageView, utilizando diferentes casos de prueba y verificando que las respuestas HTTP y las plantillas de renderizado sean las correctas.

El método setUp se llama antes de cada prueba y se utiliza para configurar los objetos necesarios para las pruebas. En este caso, se crea una imagen de un solo píxel en blanco y se guarda en un objeto

SimpleUploadedFile. También se definen algunos datos de prueba que se utilizan en las pruebas.

El primer método de prueba, “test_get_reader_image_view”, prueba que la vista ReaderImageView se puede cargar correctamente utilizando el método GET y que devuelve una respuesta HTTP con un código de estado 200 y utiliza la plantilla image_reader_by_hsv/reader_image.html.

El método “test_post_reader_image_view_with_invalid_data”, prueba que la vista ReaderImageView maneja correctamente los datos de entrada inválidos. En este caso, se envían datos inválidos (JSON inválido y un UUID), y se comprueba que la vista devuelve una respuesta HTTP con un código de estado 400 y que no se ha creado un objeto UploadImage en la base de datos con el uuid proporcionado.

“test_post_reader_image_view_with_invalid_image”, comprueba que la vista ReaderImageView maneja correctamente las imágenes no válidas. En este caso, se envía un archivo de texto en lugar de una imagen, y se comprueba que la vista devuelve una respuesta HTTP con un código de estado 400.

“test_post_reader_image_view_with_missing_image”, prueba que la vista ReaderImageView maneja correctamente los casos en los que faltan los datos de la imagen. En este caso, se envían todos los datos necesarios excepto la imagen, y se comprueba que la vista devuelve una respuesta HTTP con un código de estado 400.

```
class ReaderImageViewTestCase(TestCase):
    def setUp(self):
        self.url = reverse('Image Reader By Hsv:ReaderImageView')
        # Crear una imagen de 1x1 píxel blanca
        img = Image.new('RGB', (1, 1), color='white')
```

```

img_bytes = BytesIO()
img.save(img_bytes, format='png')
img_bytes.seek(0)

self.image_file = SimpleUploadedFile('test_image.png',
img_bytes.read(), content_type='image/png')
self.form_data = {
    'hsv_parameters': json.dumps(
        {"hsv1": ["41", "58", "100"], "hsv2": ["27", "71",
"96"], "hsv3": ["23", "64", "89"],
        "hsv4": ["22", "70", "100"]}),
    'coordinate_parameter': json.dumps(
        [{"x": 756, "y": 219}, {"x": 821, "y": 273}, {"x":
747, "y": 297}, {"x": 750, "y": 255}]),
    'image_test':
'R01GODlhAQABAIAAAP//wAAACwAAAAAQABAAACAkQBADs=',
    'image': self.image_file,
    'classification': UploadImage.ROJO
}

def test_get_reader_image_view(self):
    response = self.client.get(self.url)

    self.assertEqual(response.status_code, 200)
    self.assertTemplateUsed(response,
'image_reader_by_hsv/reader_image.html')

def test_post_reader_image_view_with_invalid_data(self):
    data = {
        'hsv_parameters': 'invalid_json',
        'coordinate_parameter': 'invalid_json',
        'image': self.image_file,
        'uuid': 'abcd-1234'
    }

    response = self.client.post(self.url, data=data,
format='multipart')

```

```

        self.assertEqual(response.status_code, 400)
        self.assertTemplateUsed(response,
            'image_reader_by_hsv/reader_image.html')

self.assertFalse(UploadImage.objects.filter(uuid='abcd-1234').exists())

    def test_post_reader_image_view_with_invalid_image(self):
        data = {
            'hsv_parameters': '{"0": [10, 20, 30], "1": [40, 50, 60], "2": [70, 80, 90], "3": [100, 110, 120]}',
            'coordinate_parameter': '[{"x": 10, "y": 20}, {"x": 30, "y": 40}, {"x": 50, "y": 60}]',
            'image': SimpleUploadedFile('test.txt', b'This is not an image.'),
            'uuid': 'abcd-1234'
        }

        response = self.client.post(self.url, data=data, format='multipart')

        self.assertEqual(response.status_code, 400)

    def test_post_reader_image_view_with_missing_image(self):
        data = {
            'hsv_parameters': '{"0": [10, 20, 30], "1": [40, 50, 60], "2": [70, 80, 90], "3": [100, 110, 120]}',
            'coordinate_parameter': '[{"x": 10, "y": 20}, {"x": 30, "y": 40}, {"x": 50, "y": 60}]',
            'uuid': 'abcd-1234'
        }

        response = self.client.post(self.url, data=data)

        self.assertEqual(response.status_code, 400)

```


DISCUSIÓN

La metodología Scrum ha surgido como una alternativa ágil a las metodologías tradicionales de gestión de proyectos. En la literatura se ha demostrado que Scrum es una metodología efectiva para mejorar la eficiencia y la calidad de los proyectos de software. Sin embargo, aún existen dudas sobre si Scrum puede ser implementado de manera efectiva en otros tipos de proyectos y si realmente es una alternativa viable a las metodologías tradicionales.

En este capítulo se discutirán los resultados de la implementación de ambas metodologías en diferentes proyectos, se analizarán las similitudes y diferencias entre ellas. Además se realizará una discusión comparativa entre artículos relevantes sobre las metodologías y también sobre las tecnologías para la lectura de imágenes de fondo de ojo.

En "Traditional Project Management vs. Agile Project Management" por Saba Javed y Rizwan Ghani (2019) se compara la metodología tradicional de gestión de proyectos con la metodología ágil de gestión de proyectos. Los autores discuten cómo la metodología tradicional se enfoca en la planificación y ejecución del proyecto en una sola fase, mientras que la metodología ágil se enfoca en la entrega de incrementos de software en ciclos cortos y frecuentes. El artículo destaca las ventajas de la metodología ágil, como la capacidad de adaptarse a cambios y de responder a las necesidades del cliente de manera más efectiva, mientras que también reconoce las limitaciones de la metodología tradicional en términos de flexibilidad y adaptación.

Mientras tanto en "An empirical comparison of traditional and agile software development models" por Shaukat Ali, Umair Qureshi, Arif Raza y Saeed Ahmed (2012) se realiza una comparación empírica de la metodología tradicional y la metodología ágil en el desarrollo de software. Los autores utilizaron un estudio de caso para evaluar la eficacia de cada metodología en

términos de calidad del software, productividad y satisfacción del cliente. El artículo concluye que la metodología ágil es superior en términos de calidad del software y satisfacción del cliente, pero que la metodología tradicional puede ser más efectiva en términos de productividad en ciertos casos.

Ambos artículos presentan diferentes perspectivas sobre la gestión de proyectos de software. El primer artículo compara la metodología tradicional con la metodología ágil y destaca las ventajas de la metodología ágil, como la capacidad de adaptarse a cambios y de responder a las necesidades del cliente de manera más efectiva. El segundo artículo proporciona una comparación empírica de las dos metodologías y concluye que la metodología ágil es superior en términos de calidad del software y satisfacción del cliente, pero que la metodología tradicional puede ser más efectiva en términos de productividad en ciertos casos. Ambos artículos sugieren que la metodología ágil es una mejor opción para proyectos de software, el primer artículo destaca las ventajas de la metodología ágil en general, mientras que el segundo artículo proporciona una evaluación más específica de las fortalezas y debilidades de cada metodología en función de factores específicos.

Metodología Tradicional	Metodología Scrum
Ventajas	Ventajas
Planificación exhaustiva y detallada que permite una visión clara del proyecto y de los objetivos a alcanzar.	Mayor flexibilidad para adaptarse a cambios en los requisitos del proyecto.
Enfoque riguroso y estructurado que garantiza la calidad y reduce el riesgo de errores.	Mayor involucramiento y colaboración del equipo en la gestión del proyecto, lo que fomenta la comunicación y el trabajo en equipo.
Desventajas	Desventajas
Puede ser demasiado rígida y no	Requiere un mayor compromiso y

permitir la adaptación a cambios en los requisitos del proyecto.	responsabilidad del equipo en la toma de decisiones y el cumplimiento de los objetivos.
Puede ser muy burocrática y llevar a una pérdida de tiempo y recursos en la documentación y los procesos.	Puede haber una falta de claridad en los roles y responsabilidades del equipo, lo que puede llevar a una falta de eficiencia.

Tabla 5 Metodología tradicional vs Metodología ágil

Por lo tanto, el proyecto se adhiere a la metodología Scrum al contar con un equipo de desarrollo multifuncional, un plan de trabajo estructurado en iteraciones (Sprints), reuniones diarias de seguimiento de avances, la creación y mantenimiento de un listado de funcionalidades (Product Backlog), selección de funcionalidades a abordar en cada Sprint (Sprint Planning), la creación de tareas y actividades necesarias para abordar las funcionalidades seleccionadas (Sprint Backlog), la presentación de los resultados al Product Owner y al cliente al final de cada Sprint (Sprint Review), la realización de una revisión de los procesos y mejoras en equipo al final de cada Sprint (Sprint Retrospective), y la entrega de un incremento funcional del producto al final de cada Sprint.

Con respecto a la lectura de imágenes de fondo de ojo, el artículo "Automated Detection and Classification of Retinal Lesions for Diabetic Retinopathy Screening" (D. Wong, et al., 2018), utiliza algoritmos de aprendizaje automático y características específicas extraídas de las imágenes de fondo de ojo para la detección y clasificación automatizada de lesiones retinianas asociadas con la retinopatía diabética. Su enfoque se basa en la extracción manual de características relevantes, como exudados, microaneurismas y hemorragias, para entrenar un modelo de clasificación.

En contraste, el segundo artículo "Deep Learning-Based Automated Diagnosis

of Diabetic Retinopathy Using Fundus Images" (V. Gulshan, et al., 2016)., se basa en técnicas de aprendizaje profundo, específicamente redes neuronales convolucionales, para lograr un diagnóstico automatizado de la retinopatía diabética. En lugar de depender de características manuales, su enfoque permite que la red neuronal aprenda automáticamente las características relevantes directamente de las imágenes de fondo de ojo, lo que puede aumentar la capacidad de detección y clasificación.

Ambos artículos han obtenido resultados prometedores en la detección y clasificación de lesiones retinianas relacionadas con la retinopatía diabética. Sin embargo, el segundo artículo, basado en técnicas de aprendizaje profundo, muestra una mayor capacidad para detectar y clasificar eficazmente las lesiones retinianas en comparación con el primer artículo. Esto puede atribuirse a la capacidad de las redes neuronales convolucionales para aprender características complejas y sutiles de las imágenes de fondo de ojo, lo que puede mejorar la precisión del diagnóstico.

La utilización de algoritmos de aprendizaje automático y aprendizaje profundo en el diagnóstico automatizado de la retinopatía diabética tiene implicaciones significativas en la práctica clínica. Estas tecnologías pueden ayudar a los profesionales de la salud a realizar un diagnóstico más rápido y preciso, lo que a su vez puede mejorar la detección temprana de la enfermedad y facilitar el tratamiento oportuno.

Además, estas técnicas automatizadas pueden ser especialmente útiles en entornos con recursos limitados, donde la disponibilidad de especialistas en oftalmología puede ser escasa. La implementación de sistemas de telemedicina que utilicen estas tecnologías puede permitir la detección remota y el seguimiento de pacientes con retinopatía diabética, brindando acceso a la atención ocular incluso en áreas remotas.

CONCLUSIONES Y TRABAJOS FUTUROS

En este trabajo se desarrolló una aplicación web para la lectura de imágenes de fondo de ojo usando una metodología de desarrollo de software Scrum, apoyado en el lenguaje python con framework Django. Para lograr este propósito se dió cumplimiento a los objetivos planteados y orientados por la metodología mencionada.

Se logró cumplir con el proceso de análisis de requerimientos el cual fue plasmado en historias de usuario en la plataforma JIRA. Este proceso fue cíclico, debido a la naturaleza y dinámica de SCRUM. Para esto, se contó con un experto en optometría “Óscar Teheran”, y mediante reuniones se dialogaron los requisitos y se logró crear un backlog o tabla de actividades que fueron repartidas en diferentes sprints.

El siguiente paso en el proyecto fue la elaboración del diseño de la aplicación web que cumpliera con los requerimientos previamente establecidos. Para ello, se crearon los mockups necesarios para entender las necesidades de los usuarios y, posteriormente, se procedió a desarrollar tanto el frontend como el backend de la aplicación web.

En cuanto al desarrollo de la aplicación web, se estableció la utilización de Python y Django como lenguaje y framework respectivamente, aplicando el patrón de diseño Modelo-Vista-Template para garantizar una estructura y organización adecuada del proyecto. Se aprovecharon al máximo los beneficios que estas herramientas ofrecen, lo que permitió llevar a cabo con éxito el desarrollo completo de la aplicación web.

Por último, se estableció el objetivo de validar la aplicación con el experto, evaluando el cumplimiento de los requisitos funcionales y no funcionales. Luego de cumplir con el desarrollo, se entregó el proyecto finalizado a los

usuarios y se hicieron las respectivas validaciones de todas las funcionalidades requeridas.

En conclusión, la realización del proyecto permitió cumplir con todos los objetivos planteados y garantizar el éxito del proyecto. Cada uno de los objetivos específicos fue alcanzado y con esto se cumplió también con el objetivo general, gracias a la correcta planificación y ejecución del proyecto, además de la utilización de herramientas y metodologías efectivas que permitieron cumplir con los requerimientos del cliente.

Con respecto a trabajos futuros, se sugiere la implementación de una forma más sencilla y efectiva de mostrar los resultados de la clasificación de la palidez del disco ocular, de acuerdo con la escala Teherán-Morales. Actualmente, es importante optimizar la presentación de estos resultados para facilitar su comprensión por parte de los profesionales de la salud, permitiéndoles tomar decisiones más rápidas y precisas. Además, es fundamental continuar con la fase de implementación de técnicas de machine learning e inteligencia artificial en el proyecto. Al aprovechar estas tecnologías, se puede mejorar aún más la experiencia del usuario, brindando diagnósticos más precisos y confiables. La aplicación de algoritmos de machine learning permite el análisis de grandes volúmenes de datos de manera eficiente y puede ayudar a descubrir patrones y relaciones que pueden no ser evidentes para los ojos humanos. Esto no solo beneficiará a los profesionales médicos, sino también a los pacientes, al brindarles un diagnóstico más rápido y certero, y en última instancia, una mejor atención médica.

Referencias

- [1] MedlinePlus, "MedlinePlus - Health Information from the National Library of Medicine," MedlinePlus, 10 07 2020. [Online]. Available: <https://medlineplus.gov/spanish/glaucoma.html>. [Accessed 29 11 2022].
- [2] J. M. Domínguez, "Repositorio Dspace," unisinucartagena, 12 06 2021. [Online]. Available: <http://repositorio.unisinucartagena.edu.co:8080/xmlui/handle/123456789/450>. [Accessed 17 01 2023].
- [3] C. R. J. Daniela, "Repositorio institucional," Universidad Peruana de los Andes, 25 08 2022. [Online]. Available: <http://www.repositorio.upla.edu.pe/handle/20.500.12848/4260>. [Accessed 01 18 2023].
- [4] N. Congdon, "Causes and prevalence of visual impairment among adults in the United States.," Europe PMC, 01 04 2004. [Online]. Available: <https://europepmc.org/article/med/15078664/reload=0>. [Accessed 18 01 2023].
- [5] OMS, "La OMS presenta el primer Informe mundial sobre la visión," OMS, 08 10 2019. [Online]. Available: <https://www.who.int/es/news/item/08-10-2019-who-launches-first-world-report-on-vision>. [Accessed 18 01 2023].
- [6] H. Öhnel, "Detection of glaucoma progression by perimetry and optic disc photography at different stages of the disease: results from the Early Manifest Glaucoma Trial," Acta Ophthalmologica, 25 10 2016. [Online]. Available: <https://onlinelibrary.wiley.com/doi/full/10.1111/aos.13290>. [Accessed 18 01 2023].
- [7] E. B. Catalán, "DESARROLLO DE UN SISTEMA PARA LA OBTENCIÓN Y CLASIFICACIÓN DE RETINOGRAFÍAS DIGITALES COMO APOYO AL DIAGNÓSTICO OPORTUNO DE RETINOPATÍA DIABÉTICA," 12 2020. [Online]. Available:

- <https://rinacional.tecnm.mx/bitstream/TecNM/1470/1/TESIS-Eduardo-Bernal.pdf>. [Accessed 18 01 2023].
- [8] J. C. Mello Román, "Retinal Image Enhancement via a Multiscale Morphological Approach with OCCO Filter," Springer Link, 31 01 2021. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-68285-9_18. [Accessed 18 01 2023].
- [9] Jorge José Rami Rodríguez, "Comparación de la angiografía mediante tomografía de coherencia óptica (OCT-A) y de la angiografía fluoresceínica (AGF) en la identificación de cambios vasculares retinianos en ojos con edema macular diabético," Universidad Zaragoza, 2018. [Online]. Available: <https://zagan.unizar.es/record/76353/files/TAZ-TFG-2018-4751.pdf>. [Accessed 21 06 2023].
- [10] U. Americana, "Diagnóstico automático de coriorretinitis por toxoplasmosis en niños utilizando técnicas de inteligencia artificial," Universidad Americana, 10 2021. [Online]. Available: <http://proyecto.uamericana.edu.py/pinv18-1293/>. [Accessed 24 01 2023].
- [11] J. Choi, "Effect of Nocturnal Blood Pressure Reduction on Circadian Fluctuation of Mean Ocular Perfusion Pressure: A Risk Factor for Normal Tension Glaucoma," Investigative Ophthalmology & Visual Science, 03 2006. [Online]. Available: <https://iovs.arvojournals.org/article.aspx?articleid=2183362>. [Accessed 24 01 2023].
- [12] T. Tokunaga, "Association between nocturnal blood pressure reduction and progression of visual field defect in patients with primary open-angle glaucoma or normal-tension glaucoma," Springer Link, 01 07 2004. [Online]. Available: <https://link.springer.com/article/10.1007/s10384-003-0071-6>. [Accessed 25 01 2023].
- [13] C. Gutiérrez-Ortiz, "Patología del nervio óptico," Hospital Universitario Príncipe de Asturias, 06 2010. [Online]. Available:

- https://www.med-informatica.net/TERAPEUTICA-STAR/PatologiaNervioOptico_Actualizacion00590068_LR.pdf. [Accessed 25 01 2023].
- [14] Ayming, "¿Qué son los TRL (Technology Readiness Levels) o Niveles de Madurez Tecnológica?," Ayming, 02 08 2021. [Online]. Available: <https://www.ayming.es/insights-y-noticias/noticias/trl-technology-readiness-levels/>. [Accessed 21 06 2023].
- [15] J. F. Huambachano, "¿Qué es Scrum?," Scrum.org, 25 09 2017. [Online]. Available: <https://www.scrum.org/resources/blog/que-es-scrum>. [Accessed 25 01 2023].
- [16] Proyectum, "Los tres roles principales en Scrum," Proyectum, 19 10 2016. [Online]. Available: <https://www.proyectum.com/sistema/blog/los-tres-principales-roles-en-scrum/>. [Accessed 25 01 2023].
- [17] N. Arora, "Django Project MVT Structure," geeksforgeeks, 16 08 2021. [Online]. Available: <https://www.geeksforgeeks.org/django-project-mvt-structure/>. [Accessed 25 01 2023].
- [18] J. d. P. Sanchez, "Introducción a Django," Mozilla, 05 12 2022. [Online]. Available: <https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Introduction>. [Accessed 25 01 2023].
- [19] programmerclick, "Modelos y códigos de espacio de color RGB, YUV y HSV," programmerclick, [Online]. Available: <https://programmerclick.com/article/24061863706/>. [Accessed 25 01 2023].
- [20] Ministerio de ambiente y desarrollo sostenible, "Protección de Datos Personales," Ministerio de ambiente y desarrollo sostenible, 25 07 2022. [Online]. Available: <https://www.minambiente.gov.co/politica-de-proteccion-de-datos-personales/>. [Accessed 25 01 2023].