UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR

POSGRADOS

MAESTRÍA EN

SOFTWARE CON MENCIÓN EN DISEÑO DE ARQUITECTURA DE SISTEMAS

RPC-SO-34-NO.778-2021

OPCIÓN DE TITULACIÓN:

PROYECTO DE TITULACIÓN CON COMPONENTES DE INVESTIGACIÓN APLICADA Y/O DE DESARROLLO

TEMA:

SOFTWARE DE PREDICCIÓN DE VENTAS EN UNA PYME BASADO EN TÉCNICAS DE APRENDIZAJE AUTOMÁTICO

AUTOR

CRISTIAN GEOVANNY CAZARES BALDEÓN

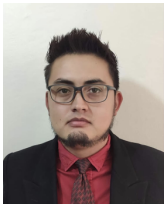
DIRECTOR:

ING. MÁXIMO GIOVANI TANDAZO ESPINOZA

QUITO – ECUADOR

2023

Autor:



Cristian Geovanny Cazares Baldeón
Ingeniero en Sistemas
Candidato a Magíster en Software por la Universidad Politécnica Salesiana – Sede Quito.
ccazares@est.ups.edu.ec

Dirigido por:

Máximo Giovani Tandazo Espinoza
Ingeniero en Sistemas Computacionales.
Maestría en Administración de Empresas con Mención en Sistemas de Información Gerencial, Maestría de Investigación en Ciencias y Tecnologías de la Computación para Smart Cities, Maestría en Ciencia de Datos
mtandazo@ups.edu.ec

Cristian Geovanny Cazares Baldeón
SOFTWARE DE PREDICCIÓN DE VENTAS EN UNA PYME BASADO EN TÉCNICAS DEAPRENDIZAJE AUTOMÁTICO

DEDICATORIA

Dedico este trabajo:

A Dios, fuente inagotable de bendiciones, quien me ha acompañado y guiado en cada etapa de mi formación académica. A pesar de los retos y adversidades, siempre sentí su presencia, iluminando mi camino.

A mi amada esposa, Samantha Mosquera, pilar de mi vida y constante impulso en mi crecimiento profesional. Samantha, gracias por ser mi compañera incondicional, por estar siempre a mi lado y por recordarme la importancia de alcanzar este título de maestría.

A mis hijos, Antonella, Kristen y Matthew, verdaderos motores de mi existencia. Ustedes son la inspiración detrás de cada esfuerzo y la razón de mi superación constante, tanto personal como profesional.

A mis padres, Martha y Stalyn, quienes con su amor, dedicación y ejemplo, no sólo me guiaron hacia mi título de Ingeniero, sino que también me inspiraron a seguir adelante y alcanzar esta maestría.

Finalmente, a mis familiares y amigos, testigos y cómplices de cada lección aprendida durante estos años de estudio. Su apoyo y los inolvidables momentos compartidos enriquecieron mi travesía universitaria.

Cristian Geovanny Cazares Baldeón.

AGRADECIMIENTO

En primer lugar, deseo expresar mi profundo agradecimiento a Dios, quien me ha dotado de la fortaleza y determinación para completar este camino académico.

A mi tutor de tesis, Máximo Tandazo, por su invaluable orientación, paciencia y meticulosidad en cada etapa de este trabajo. Su experiencia y conocimientos fueron esenciales para el desarrollo y culminación de esta investigación.

A la Universidad Politécnica Salesiana, por brindarme todas las herramientas y recursos necesarios, así como un ambiente propicio para el aprendizaje y la investigación.

A mi esposa, Samantha Mosquera, por ser mi inspiración y soporte durante estos años de estudio. Su amor, comprensión y ánimo constante fueron esenciales para mantenerme enfocado y motivado.

A mis hijos, Antonella, Kristen y Matthew, por ser la alegría de mis días y por recordarme que todo esfuerzo tiene su recompensa. Cada página de este trabajo lleva consigo el amor y el deseo de dejarles un legado.

A mis padres, Martha y Stalyn, por inculcarme la importancia de la educación desde temprana edad y por ser un modelo de perseverancia y dedicación.

A todos mis compañeros de maestría, con quienes compartí momentos de esfuerzo, dudas, pero también de logros y satisfacciones. Juntos hemos crecido y aprendido.

Y finalmente, a todos aquellos que, de alguna u otra forma, han contribuido a la realización de este proyecto. Ya sea con una palabra de aliento, un consejo o simplemente con su presencia, cada uno ha dejado una huella en este camino que hoy culmino.

Tabla de Contenido

Indice de Figuras 7
Indice de Tablas 9
Resumen 11
Abstract 12
1. Introducción 13
1.1 Antecedentes 13
1.2 Determinación de problema 14
Descripción del problema 14
Formulación del problema 15
Justificación del problema 16
Delimitación del problema 17
1.3 Objetivos 17
Objetivo general 17
Objetivos específicos 17
2. Marco teórico referencial 18
2.1 Software 18
2.1.2. Ingeniería del software 19
2.2. Desarrollo de software 20
2.2.1. Contenedores de software 21
2.2.2. Docker 21
2.2.3. Microservicios 23
2.3. Patrones de diseño 23
2.3.1. Estructura de los patrones 24
2.3.2. Tipos de Patrones de diseño 25
2.3.3. Herramientas de codificación 26
2.4. Machine learning (aprendizaje automático) 27
2.4.1. Tipos de aprendizaje 27
2.5. Modelos predictivos 29
2.5.1. Tipos de modelos predictivos 29
2.5.2. Regresión lineal múltiple 31
2.5.3. Lenguaje de Programación Python 31

2.6. JSON Web Token 32
2.6.1. Como se estructura los JSON Web Token 32
2.6.2. Funcionamiento de JSON Web Token 33
2.6.3. En qué casos se utilizan JSON Web Token 34
2.7. Definiciones de Arquitectura cliente servidor del software 34
3. Desarrollo del proyecto 36
3.1 Arquitectura 36
3.2 Frontend 37
3.2.1 Módulos Frontend 38
3.3 Backend 40
3.3.1 Gateway 40
3.3.2 Micro servicios 41
3.3.3 Micro de predicción 42
3.3.3.1 Python y lib. Panda 42
3.3.3.2 Regresión Lineal Múltiple 43
3.4 Servicios de la Nube 44
3.4.1 Auth0 44
3.4.2 Mongo Atlas 44
3.5 Infraestructura 45
3.5.1 Docker Files 45
3.5.2 Docker Compose 45
3.5.3 OVCloud VPS 45
3.6 Desarrollo 46
3.6.1 Ingreso Auth0 74
4. Resultados y discusión 78
5. Conclusiones 86
6. Glosario 87
Referencias 88

Índice de Figuras

Figura 1. Arquitectura del Docker 22
Figura 2. Arquitectura Cliente servidor 37
Figura 3. Representación arquitectónica 38
Figura 4. Diseño de los módulos del Frontend 39
Figura 5. Estructura del Gateway 41
Figura 6. *AuthoFront* 46
Figura 7. CI-CD Gitlab 47
Figura 8. Update tab 48
Figura 9. networks 48
Figura 10. Services - Traefik 49
Figura 11. Spa-principal 50
Figura 12. Gateway 51
Figura 13. mc-auth 52
Figura 14. mc-file-update 52
Figura 15. Ms-machine-learning 53
Figura 16. DockerIonic 54
Figura 17. DockerNoder 56
Figura 18. DockerPhyton 58
Figura 19. GuardGuateway 59
Figura 20. libPhyton 61
Figura 21. Makefile 63
Figura 22. Patron estrategy contex 65
Figura 23. Patron estrategy estrategia 66
Figura 24. Patron estrategy estrategia (Continuación) 67
Figura 25. Patron estrategy estrategia (Continuación) 67
Figura 26. Patron estrategy interface 68
Figura 27. Prediction Phyton 69
Figura 28. Prediction Phyton (Continuación) 70
Figura 29. Validación Auth 71
Figura 30. Dashboard del front 73
Figura 31. Ingreso Auth0 74
Figura 32. Ingreso a prediction 75
Figura 33. Historial de archivos subidos a la nube 75
Figura 34. Datos encontrados de la predicción total ventas hombres y mujeres. 76
Figura 35. Ventas futuras representación gráfica en barras 77
Figura 36. Datos en Excel 78
Figura 37. Identificación del mes de los datos 79
Figura 38. Tercer paso colocar el año del cual pertenecen los datos. 79
Figura 39. Importe de archivo 80
Figura 40. Quinto paso, cargar el archivo esto permite 80
Figura 41. Sexto paso, verificación del documento cargado. 81
Figura 42. Confirmación MongoAtlas 81
Figura 43. Resumen datos de Excel 82

Figura 44. Predicción 2020.....83

Índice de Tablas

Tabla 1. Pasos para desarrollar un Software 20
Tabla 2. Patrones de propósito 25
Tabla 3. Tipos de modelos predictivos. 29
Tabla 4. Estructura de los JSON Web Token 32

Software de predicción de ventas en una PYME basado en técnicas de aprendizaje automático

Autor:

Cristian Geovanny Cazares Baldeón

Resumen

El siguiente proyecto investigativo, posee como objetivo implementar un Software para predecir las ventas de una PYME con la ayuda de las técnicas de aprendizaje automático, ya que esta es una herramienta fundamental para los aprendizajes automáticos las cuales podrán ayudar a los manejos eficientes de la empresa en torno a la información que se obtiene, para concretar dichas integraciones se diseñará e implementará las arquitecturas de softwares basadas en microservicios.

La metodología que se usó es la cualitativa, porque se utilizó como herramienta de evaluación la observación y el tipo de investigación fue la descriptiva, ya que se detalló toda la realización del software, en donde se tuvo como resultados que la exploración de los diferentes enfoques, permitió que Mongo Atlas tome la decisión correcta para desarrollar este trabajo, ya que el proceso requería investigar las necesidades y comprender el negocio, lo que otras tecnologías no podían proporcionar. Aunque métodos como SEMMA pueden simplificar el proceso, la estandarización de pasos que ofrece el método utilizado promueve un desarrollo organizado y eficiente.

Concluyendo, que el modelo que se realizó permitió el ingreso de los datos para la automatización de las predicciones ventas, evaluando el desempeño del software en la determinación de las ventas, y así, implementar un Software para predecir las ventas de una PYME con la ayuda de las técnicas de aprendizaje automático.

Palabras clave:

PYME, Software, Predecir, Automático, Microservicios.

Abstract

The following research project aims to implement a software to predict the sales of an SME with the help of machine learning techniques, as this is a fundamental tool for automatic learning which can help the efficient management of the company around the information obtained, to realize these integrations will be designed and implemented software architectures based on microservices.

The methodology used is qualitative, because observation was used as an evaluation tool and the type of research was descriptive, since all the realization of the software was detailed, where we had as results that the exploration of the different approaches, allowed Mongo Atlas to make the right decision to develop this work, since the process required to investigate the needs and understand the business, which other technologies could not provide. Although methods such as SEMMA can simplify the process, the standardization of steps provided by the method used promotes an organized and efficient development.

Concluding that the model that was performed allowed the input of data for the automation of sales predictions, evaluating the performance of the software in determining sales, and thus, implement a Software to predict the sales of an SME with the help of machine learning techniques.

Keywords:

SMEs, Software, Predictive, Automatic, Microservices.

1. Introducción

El mundo de los negocios está cada vez más competitivo, ya que las medianas y pequeñas empresas "PYME" no son las excepciones. Para mantenerse a la vanguardia, es fundamental contar con herramientas innovadoras y eficientes que les permitan predecir su desempeño y tomar decisiones acertadas. En este contexto, el software de predicción de ventas juega un papel clave, por lo que permite a las PYME anticiparse a las tendencias del mercado y ajustar sus estrategias de manera oportuna.

En el marco de esta tesis de maestría en software, se busca desarrollar las aplicaciones del espacio virtual que utilice técnicas de aprendizajes automáticos para predecir las ventas de una PYME. Para ello, se aplicarán conocimientos de API REST, Docker, arquitectura de software y patrones de diseño. El propósito principal de esta investigación es brindar una solución innovadora y accesible para las PYME que les permita maximizar sus ganancias y reducir los riesgos asociados con la toma de decisiones basadas en suposiciones. Al utilizar técnicas de aprendizaje automático, el software de predicción de ventas será capaz de hacer un análisis de grandes cantidades de datos y detectar patrones y tendencias en las ventas, lo que permitirá a las PYME hacer previsiones precisas y ajustar su estrategia en consecuencia.

1.1 Antecedentes

Las ventas son un aspecto clave en el mundo de los negocios, ya sea en una gran empresa o en una pequeña y mediana empresa "PYME". Según el Directorio de Empresas 2019 del Instituto Nacional de Estadística y Censos "INEC", en el "Ecuador existen 882.766 empresas, de las cuales el 99.5 % son consideradas como PYME. Estas empresas son las principales en términos de empleo y su importancia es fundamental" (Álvares, 2019).

La predicción de ventas es un tema relevante en el ámbito empresarial, ya que permite tomar decisiones informadas y mejorar la eficiencia del negocio. La utilidad de un método de predicción de ventas se puede medir comparando métricas de rendimiento y analizando los factores que influyen en la eficiencia de dicho método.

Además, la identificación de sentimientos y emociones de los clientes puede ayudar a mejorar las ventas. Analizando los sentimientos y emociones de los clientes, se pueden identificar oportunidades de mejora y establecer estrategias efectivas para alcanzar un mayor nivel de ventas (Sampedro, 2022).

En el marco de investigaciones sobre técnicas de aprendizaje automático en marketing, se puede identificar tendencias y problemas, lo que ayuda a desarrollar estrategias efectivas y aumentar las ventas de las PYME.

Por otro lado, las técnicas de aprendizaje automático también pueden ser aplicadas para predecir la solvencia financiera de las PYME, lo que permite evaluar la situación financiera de cada empresa y minimizar el riesgo de su desaparición. En la actualidad, la tecnología es un aspecto clave en el mundo de los negocios y muchas empresas medianas y pequeñas han decidido adaptarse y aprovechar las herramientas tecnológicas para seguir compitiendo en un mercado cada vez más competitivo.

Al utilizar software de predicción de ventas basado en técnicas de aprendizaje automático, las PYME pueden mejorar su eficiencia y ofrecer mejores cambios a sus clientes.

1.2 Determinación de problema

Descripción del problema

El éxito de una pequeña y mediana empresa PYME depende de la información que se posea para la creación de las mismas, lo cual, logrará la colocación de los presupuestos de las ventas anuales, por ende, si existe inconsistencia va a surgir problemas en los inventarios, como es la pérdida, la falta de acceso a herramientas

de análisis de datos avanzadas y la escasez de recursos para contratar expertos en el área, puede dificultar esta tarea y poner en riesgo la continuidad de la empresa, teniendo en cuenta que, se quiere ser más eficiente con la tecnología, ya que, al automatizar procesos manuales se tiene 75% de efectividad y con *Machine Learning* un 95% en el mejoramiento de la calidad (Flores, 2020).

En este contexto, el desarrollo de un software de predicción de ventas basado en técnicas de aprendizaje automático puede ser de gran utilidad para las PYME. Este software permitiría a los empresarios analizar datos históricos de ventas y predecir tendencias futuras, permitiéndoles tomar decisiones informadas y mejorar sus estrategias de negocios (Sampedro, 2022).

Formulación del problema

Las pequeñas y medianas empresas (PYME) son la columna vertebral de muchas economías, pero enfrentan una serie de desafíos para mantenerse a flote y crecer. Uno de ellos es la falta de profesionales que sean capaces de entrenar a la herramienta *Machine Learning* para que les permitan un mejoramiento en la toma de decisiones en términos de ventas y finanzas.

Por lo cual, si no se posee una buena predicción de ventas, se tendrá un aspecto crítico en la gestión de una empresa, ya que, permite identificar oportunidades de crecimiento y poder tomar medidas preventivas ante posibles disminuciones. Sin embargo, se evidencia que la mayoría de las PYME, no son utilizadas, por ende, no tienen la capacidad o los recursos para implementar soluciones tecnológicas complejas y costosas (Sampedro, 2022).

El problema que se quiere abordar es la falta de eficiencia de los procesos que se realiza en las predicciones de ventas en una PYME, lo que puede afectar negativamente su crecimiento y estabilidad financiera. La solución propuesta, es implementar un software de predicción de ventas de la manera más adecuada, en donde se tenga como resultado ayudar a los departamentos de ventas, marketing y gerencia a tomar decisiones estratégicas más informadas para preservar el futuro de las empresas.

JUSTIFICACIÓN DEL PROBLEMA

Este proyecto investigativo es una herramienta fundamental para los aprendizajes automáticos las cuales podrán ayudar a los manejos eficientes de la empresa en torno a la información que se obtiene, para concretar dichas integraciones se diseñará e implementará las arquitecturas de softwares basadas en microservicios.

Estas se especificarán en sistemas de gestión de productos, y se utilizarán algoritmos para predecir las cantidades de ventas que las empresas realizarán a futuro, estimando cuáles serán los stocks necesarios para poderlos afrontar, en las ventas futuras de los sistemas, por ende, se recomendará, las compras o movimientos de stock entre los diferentes puntos de ventas, donde se da las posibilidades de los usuarios finales, así poder tomar las decisiones de realizar los movimientos de stock. Los beneficiarios serían los dueños de las empresas y el personal administrativo ya que facilitara su trabajo.

Teniendo en cuenta que, es de gran importancia que las empresas posean procesos adecuados para cada actividad para poder observar la economía de un país y lo que estas representan para poder obtener una gran cantidad de empleo y riqueza. En este sentido, la tecnología y el aprendizaje automático pueden ser herramientas muy valiosas para el mejoramiento de la situación de las empresas y permitirles alcanzar sus objetivos y metas de ventas, logrando posicionarles dentro del mercado laboral este sea internacional o nacional.

Por ende, el desarrollo del software el cual utiliza técnicas de aprendizaje automático para predecir las ventas en una PYME es una herramienta indispensable porque le permitirá a la empresa tener una visión más clara y precisa de sus futuras ventas, lo cual, les consentirá planificar y tomar decisiones más informadas. Esto les aprobará aumentar su productividad y reducir costos, ya que, podrán utilizar sus recursos de manera más eficiente.

Además, la aplicación web estará diseñada con una arquitectura de software escalable y segura, lo que permitirá a las PYME utilizarla en el futuro sin preocuparse por problemas de rendimiento o seguridad. Por otro lado, la integración de Docker

permitirá una implementación rápida y sencilla, lo que significa se podrán empezar a utilizar el software de predicción de ventas en cuestión de minutos.

Delimitación del problema

La delimitación del problema se enfoca en PYME y en su necesidad de tener una ayuda tecnológica para mejorar sus predicciones de ventas y optimizar su manejo financiero, con el objetivo de minimizar las pérdidas y maximizar las ganancias.

1.3 Objetivos

Objetivo general

Diseñar un Software para predecir las ventas de una PYME con la ayuda de las técnicas de aprendizaje automático.

Objetivos específicos

- Levantar los requerimientos del software para su implementación.
- Integrar el software a tecnología usada por los PYME para la abstracción de información.
- Identificar el modelo más eficiente en técnicas de aprendizaje automático para las predicciones ventas.
- Evaluar el desempeño del software en la determinación de las ventas.

2. Marco teórico referencial

Para comenzar con este capítulo se va a realizar una descripción detallada de todos los conceptos que intervienen en el Software de predicción de ventas en una PYME basado en técnicas de aprendizaje automático, definiendo los conceptos y las nociones teóricas necesarias para la comprensión de la creación del software, describiendo al principio todos los conceptos derivados a este tema, teniendo en cuanta diversos criterios para la factibilidad de la investigación.

2.1 Software

Se los definen como los componentes lógicos que se encargan de la relación que existe entre la persona y la computadora, ya que, el usuario es el encargado de enviar diversas órdenes por medio de sistemas periféricos de entrada, como es el teclado y el mouse, y después de este proceso la computadora los traduce a través de los sistemas periféricos de salida, como es la pantalla, altavoces, etc.

Según Quishpe (2019) manifiesta que son programas que se van ejecutando en las maquinas, que a la vez se pueden describir como los medios que ofrecen los servicios, es decir que realizan una función compleja. La elaboración del software va ofreciendo grandes ayudas a diversas personas, por lo que se les consideran como las actividades económica.

Se denominan software a los diversos programas o sistemas operativos, que sirven para la elaboración de los productos que se ponen a la venta. Los cuales, donde existen diferentes tipos de software, como ejemplo se presenta el tipo de software con propósitos generales, los cuales están diseñados para la realización de las tareas específicas, dentro de estos tipos se encuentran a las diversas aplicaciones web.

2.1.2. Ingeniería del software

Para poder comprender sobre las producciones del software, se debe de poner atención en dos aspectos, la primera se basa a la teoría y a las diversas herramientas que se posee con la finalidad de poder brindar soluciones a los problemas que se presenten, como segundo punto se tienen a los técnicos para poder desarrollarlo, en donde intervienen las teorías de apoyo, las gestiones del proyecto y el desarrollo de la herramienta y el método.

Como se puede observar estos aspectos fundamentas intervienen en las producciones del software, las cuales empiezan desde las etapas iniciales hasta los mantenimientos después de haber ejecutado el programa. Se evidencia que para los ingenieros de software estas diciplinas poseen grandes importancias para que se puedan adaptar a los trabajos y enfoques sistemáticos.

Según Olmedo (2019) menciona que la ingeniería de Software, poseen como fin brindar softwares de la calidad, en aspectos como las eficiencias, usabilidades, entre otras. No solo el software debe poseer funcionalidades, sino que deben poseer calidad por lo que se usan una serie de métodos, técnicas y más de tal forma que permita la obtención un software de calidad, por ende hay que conocer que este abarca diversas etapas de los desarrollos del software.

Ingeniera web

Están basados en las tecnologías web de los desarrollos del software, donde se va utilizando diversos métodos que brindan mejores calidades a los softwares web. Esta deriva a las aplicaciones web, que son codificadas por los lenguajes de las programaciones y son construidas para que sean soportadas por diversos navegadores web y así que puedan interactuar entre sí, es decir los usuarios y los servidores.

Según Paredes y Millanes (2019) dicen que es la aplicación que está escrita por medio de los tipos de lenguaje y que pueden ser ejecutados en cualquier tipo de

sistemas operativos, los que se les denominan también a las aplicaciones web a las comunicaciones que utilizan mediante los protocolos HTTP.

2.2. Desarrollo de software

Según Delgado y Díaz (2021) manifiesta que para poder desarrollar un software que posea calidad existen diversos tipos de modelos de desarrollos, en el cual, se encuentran fases o etapas para los desarrollos del Software, los cuales se describen a continuación:

Tabla 1.

Pasos para desarrollar un Software

Pasos	Definición
Analizar	Este se basa a la recopilación, examinación y formulación del requerimiento de los clientes, por lo que se utilizan diversas técnicas que ayudan a las recopilaciones de los requisitos del software.
Diseñar	Esta se divide en dos fases, las que sirven para los diseños generales, que son los encargados del requisito general de las arquitecturas de las aplicaciones y los diseños en detalles que son los subconjuntos de dichas aplicaciones.
Codificar	Después de las funciones que se definen en las fases de los diseños, que se van implementando en los softwares por medio de los lenguajes de programaciones.
Pruebas	Se realiza dos pruebas diferentes en esta fase, fases de las unidades que es prueba individual para cada subconjunto de las aplicaciones y las otras pruebas de integridades que garantizan que el módulo diferente se integre con las aplicaciones.
Mantenimiento	Hacen referencia a la actualización secundaria y procedimiento correctivo del software.

Nota. La tabla anterior muestran los pasos para poder desarrollar un software.

2.2.1. Contenedores de software

Los principales propósitos de los contenedores son la agilización de los procesos de desarrollo que se despliegan del Software, estas facilitan las creaciones de las arquitecturas clientes servidor, sobre las cuales se despliegan una serie de aplicaciones, los contenedores a la vez, trabajan con instancias virtuales de partes de los sistemas operativos completos, ya que, corre sobre Kernel de Linux.

2.2.2. Docker

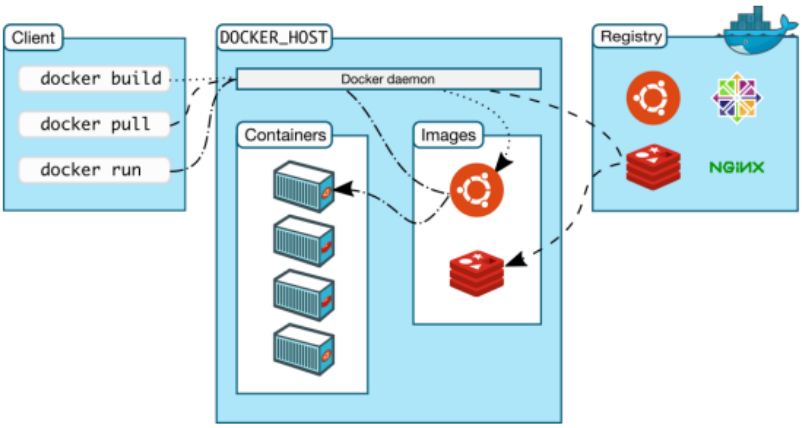
El Docker se trata de los proyectos de los códigos abiertos, los cuales tienen por objetivo, la compilación y ejecución de la aplicación sobre el contenedor de los softwares, esto separa las diversas aplicaciones en las infraestructuras y agilizaciones de los procesos de despliegues de softwares, los cuales van disminuyendo las maneras significativas los tiempos, desde que se escriben los códigos de las aplicaciones hasta que se ponen en las producciones, el ciclo de vida de los contenedores en Docker se manejan de la siguiente manera:

- Desarrollar de la de las aplicaciones utilizando contenedores.
- Disminuir y probar las aplicaciones haciendo utilización de los contenedores
- Despliegue de los contenedores a producciones sin importar la plataforma con la que trabaje.

La Arquitectura del Docker se hace por medio de la utilización de estas arquitecturas cuando los clientes y servidores, pueden fusionarse en las mismas máquinas o en máquinas diferentes ya que, estos utilizan las Api Rest para comunicarse entre sí.

Figura 1.

Arquitectura del Docker



Nota. La imagen anterior muestran la arquitectura del Docker.

Como se puede evidenciar en la imagen anterior, se conforma la arquitectura del docker en tres partes, el cliente, son los que ejecutan los comandos necesarios para la administración del contenedor, interactúan con el Daemon de Docker, por otro lado , el Daemon, sirve para escuchar las peticiones de los clientes, por medio de la Api Rest del Docker, en esta interactúan directamente con los contenedores y el repositorio, es el que almacena la imagen necesaria para poderse desplegarse, también se puede almacenar contenedores que sean personalizados, en estos repositorios para así poder descargar las imágenes necesarias para la ejecución de algunos comandos desde los clientes (Montalvo, 2020).

Docker compose

Docker Compose se puede utilizar para crear esta arquitectura de microservicios y vincular los contenedores entre ellos, o se puede utilizar para un solo servicio. Además puede crear imágenes, escalar contenedores y vuelva a ejecutar los contenedores detenidos. Toda esta funcionalidad es parte de Docker, este solo una abstracción de alto nivel de los comandos de ejecución de contenedores. Ustedes pueden hacer todo lo que pueda en un archivo de redacción con comandos simples, excepto que esto requiere más memoria y requiere un esfuerzo adicional para ejecutar todos los comandos adicionales, conexión a la red, etc. Docker compose ayuda a simplificar este proceso (Racero y Som, 2019).

2.2.3. Microservicios

Los micro servicios, se ha constituido por medio de grandes cambios para la manera en las que se van construyendo y se van desplegando de los sistemas informáticos, gracias a los continuos avances de la nueva tecnología que se han ido integrándose en los desarrollos y en los ciclos de vida de los sistemas, lo que va facilitando sus creaciones, manteniendo y puesta en producción.

Según Sanmartín (2021) menciona que de forma tradicional, el sistema informático está constituido sobre la base monolítica del código fuente que a pesar de contar con bases sólidas documentaciones y estructuras modularizadas pueden presentarse inconvenientes a los momentos de ir añadirse de la nueva característica y funcionalidad, ya que, los códigos fuentes aumentan cada vez más en tamaño, haciendo más difícil el mantenimiento de los sistemas y añadiendo nuevas características a los mismos, limitando tamaño y la escalabilidad.

2.3. Patrones de diseño

Según Shvets (2019) los patrones de diseño, que no son códigos, sino que son las ideas abstractas que van proporcionando plantillas de tales formas que al adaptarlos a los desarrollos del software que brinde las soluciones a los problemas determinados de los dises. Para poder considerar que es un patrón deben de desempeñar con dos de los siguientes requerimientos que son indispensables:

- Las primeras soluciones que son resueltas de los problemas deben ofrecer soluciones muy efectivas y las mejores opciones en las comparaciones a otras posibles que se obtengan.
- Las segundas son las reutilizaciones de las soluciones para cuando se tengan los problemas.

Los propósitos de la utilidad de los patrones son los siguientes:

- Al desarrollar el software se debe de poseer grupos de pares, es decir el problema y la solución, para que estos puedan ser usados.

- Se debe de evitar las búsquedas de las soluciones en las que ya se probaron efectividad.
- Cuando se realiza los diseños del software se buscan que se impongan diversos estándares.

Los patrones son las formas para documentar la mejor práctica y lección que se ha aprendido a los momentos de resolver los determinados problemas complejos que se encuentran dentro de los dominios de diseños concretos con los desarrollos del software, las arquitecturas o los diseños, por lo que los patrones que poseen las capacidades de brindar las soluciones reutilizables de tales maneras que ayuden a las personas que no tienen experiencias.

2.3.1. Estructura de los patrones

Su estructura de los patrones de los diseños don los siguientes:

- **Nombres:** Esto va a permitir las identificaciones de los patrones por los que deben ser únicos.
- **Problemas:** Describen los resúmenes de una o fases las metas que tienen patrones.
- **Contextos:** Aplicaciones de los patrones a los problemas recurrentes.
- **Fuerza:** Describen los propósitos, la fuerza y restricción relevante para esos patrones, además de las comunicaciones.
- **Soluciones:** Son los conjuntos de instrucción en los que describen las construcciones de los productos resultantes.
- **Ejemplos:** El ejemplo puede ser visual de tal forma que ayuden a los lectores a entendimiento de sus utilidades y en qué momento se aplican los patrones.
- **Contextos resultantes:** Indican cuales son los estados de los sistemas después de que se aplicaron los patrones, en las que incluyen las consecuencias, estas sean positivas o negativas.
- **Exposiciones razonadas:** Exponen cómo funcionan los patrones y el porqué de sus utilidades, es decir explican sus mecanismos subyacentes.

- **Patrones relacionados:** Son patrones en el cual se pueden agrupar con estos o las posibilidades de aplicarse a partir de los contextos resultantes o también representan la solución alternativa.

2.3.2. Tipos de Patrones de diseño

A continuación, se van a describir los patrones de diseño que son utilizados para la creación de los softwares.

Según (Pavón, 2019) los patrones de diseño están orientados a objetos, son organizados a este patrón por dos criterios que son: los propósitos, en donde intervienen las creaciones, estructurales y de comportamientos en los ámbitos, que son las clases y objeto.

Tabla 2.

Patrones de propósito

Patrón	Definición	Clasificación
Creación	Este tipo de patrón ayuda a que los sistemas sean independientes de cómo están representados, creados, compuestos de los objetos.	*"Abstract Factory"* *"Factory Method"* *"Singleton"* *"Builder"* *"Prototype"*
Estructurales	Este tipo de patrón forma estructura grande por medio de las combinaciones de la clase y objeto.	*"Adapter"* *"Facade"* *"Flyweight"* *"Decorator"* *"Bridge"* *"Composite"* *"Proxy"*
Comportamiento	Este patrón tiene que ver con algoritmo y con las asignaciones de responsabilidad de objetos.	*"Chain of Responsibility"* *"Comand"*

	“Interpreter”
	“Mediator”
	“Memento”
	“Observer”
	“State”
	“Strategy”
	“Template Method”
	“ Visitor”

Nota. La tabla anterior muestra los tipos de patrones de propósito. Adaptado de: (Masa, 2019).

2.3.3. Herramientas de codificación

Para poder lograr una buena codificación para la programación del software es necesario hacer referencia al Lenguaje de programación Python, este es de alto nivel y naturaleza *“open source”,* el cual es usado para los objetivos generales en los desarrollos de softwares, en la actualidad, se utilizan por diversos programadores que se encuentran alrededor del mundo con las diferentes áreas como: la ingeniería de software, Internet, diseño de GUI, automatización de procesos, etc. Los rasgos notables de este programa de lenguaje son los siguientes:

- Soporte de los paradigmas de POO.
- Sintaxis sencillas de manejo y lectura.
- Amplias variedades de módulo, librería e interfaz precodificada.
- Sobresale por sus calidades, productividades, portabilidades e integraciones.

Python cuenta con diversas características técnicas que los diferencian de otros lenguajes de programaciones y lo hacen sencillos para su aprendiza y su manejo (Ríos, 2021).

2.4. Machine learning (aprendizaje automático)

Son las categorías más amplias del algoritmo que pueden tomar los grupos de datos y utilizarlos para la identificación de los patrones, se puede evidenciar que según Hiff (2021) hace referencia a dos distintas definiciones para la contextualización del aprendizaje automático, a continuación se van a dar a conocer:

Arthur Samuel lo describió como: los campos de estudio que brindan a la computadora las capacidades de aprendizaje sin ser programados explícitamente. Estas son la definiciones más antiguas e informales. Por otra parte Tom Mitchell proporcionan las definiciones más modernas: Se dicen que los programas de computadora aprenden de las experiencias E con respecto a algunas clases de tarea T y medidas de desempeños P, si sus desempeños en tareas en T, medidos por P, mejorando las experiencias.

2.4.1. Tipos de aprendizaje

Se puede evidenciar dos diferentes tipos de aprendizaje automático, estos son el aprendizaje no supervisado y el supervisado, a continuación, se da a comer estas contextualizaciones.

Aprendizaje supervisado

Son uno de los tipos de problema de aprendizaje automático más común, ya que, se debe de tener en cuenta que las máquinas aprenden por medio de ejemplos. Los algoritmos se entrenan a través de preguntas, las que se conocen como la característica *"features"* y sus debidas respuestas, que son llamadas etiquetas *"labels"*. Cuando los algoritmos poseen ciertas respuestas a algunas preguntas, que guardan esas informaciones para hacer la previsión futura.

Este se puede subdividirse según Moreno et al. (2020) en:

- **En regresión:** Posee como la predicción del valor continuo. Con los valores numéricos de la etiqueta, que se usa en diversas variables para la obtención de datos que interesen. Estos tipos de aprendizajes sirven para las series de finalidades concretas como pueden predecir los precios de los productos, las propiedades o los valores de las acciones en las bolsas.
- **En clasificaciones:** Las ideas de estos algoritmos son buscar diferentes patrones y clasificarlos por diversos elementos en grupos distintos. Estos modelos buscan sacar conclusiones del valor observado, ya que, una o más entradas pretenden la predicción de los valores de uno o más resultados (p.9).

Aprendizaje no supervisado

Estos enfoques se basan en las ausencias de cualquier supervisor y, por lo ello, de la medida de errores absolutos. Son utilizados cuando es necesario adquirir un aprendizaje de cómo se pueden agruparse los conjuntos de elementos de acuerdo con las similitudes.

Balderas et al. (2020) mencionan que el aprendizaje no supervisado proporcionan análisis descriptivos implícitos porque todas las piezas de informaciones son descubiertas por los algoritmos de agrupamientos se puede usar para la obtención de una visión completa del conjunto de datos. se debe de tener en cuenta que todos estos hechos, poseen objetos que se comparte en los subconjuntos de las diversas características y son las que permiten que se hagan posible que estos conjuntos se puedan agruparse, por medio de otros lados que son diferentes bajo otro punto de vista o característica.

En este tipo de aprendizaje también se puede evidenciar una subdivisión, según (Elfenbaum, 2019) menciona lo siguiente:

- **Análisis clúster:** Extraen la referencia del conjunto de datos como las partes de los datos de entradas sin respuesta etiquetada sin el conocimiento nada

de cómo se clasifican. Se tratan de clasificarlos como los conjuntos de datos en el grupo homogéneos entre sí y los más heterogéneos entre ellos.

- **Reducción de la dimensionalidad:** Se tratan de las reducciones de los números de variable aleatoria por medio de las obtenciones de los conjuntos variables principales, donde se elimina así la variable irrelevante y mejora los rendimientos computacionales (p.14).

2.5. Modelos predictivos

El desarrollo de este tipo de método, se usan para los para los análisis del pronóstico inteligente, donde se necesitan de estudios para las proyecciones de las predicciones de la decisión, las empresa que es comercial, se supeditan en la proyección de la venta en las funciones a los conocimientos de los mercados, en estos sentidos, requieren las predicciones de la tendencia de venta para poderlos proyectarlos de maneras fiables, las fiabilidades en los pronósticos de las ventas y proyección, proveen herramientas vitales para los cumplimientos de metas en los servicios.

Según Cali (2022) señala que el centro de comercialización, específico son las cadenas de supermercados Big Mart, las que desarrolla rastreos a las ventas individualizadas por cada artículos, este dato permite pronosticar los consumos de los usuarios y hacer más efectivas la labor de bodegas con inventarios adecuados (p.17). La fluctuación y los volúmenes de las demandas de cada artículo, se la realiza para poder ejecutar y realizar un inventario.

2.5.1. Tipos de modelos predictivos

Estos modelos predictivos se clasifican de la siguiente manera, según Norambuena et al. (2020) explica a continuación:

Tabla 3.

Tipos de modelos predictivos.

Tipo de modelo	Definición

Técnica de regresión	Son los pilares de los análisis predictivos, en los enfoques se basan en los establecimientos de las ecuaciones matemáticas como modelos para las representaciones de la interacción entre las diversas variables en consideraciones.
Modelo de regresión lineal	Estos modelos analizan las relaciones existentes entre las variables dependientes y los conjuntos de variables independientes. Estas relaciones se expresan como las ecuaciones que predicen las variables de las respuestas.
Análisis de supervivencia o duración	Estas técnicas se desarrollan de manera principal en la ciencia médicas y biológicas, pero a la vez se utilizan ampliamente en las ciencias.
Arboles de clasificaciones y regresiones	El árbol de las clasificaciones y regresiones son técnicas de los aprendizajes de las decisiones no paramétricas que producen el árbol de las clasificaciones o regresiones, dependiendo de si las variables dependientes son categóricas o numéricas respectivamente.
Curvas de regresiones adaptativas multivariables	La curva de regresiones adaptativas multivariables, que son técnicas no paramétricas que se construyen en los modelos flexibles al ajustarse de la regresión lineal por las piezas, son conceptos importantes asociados con la curva de regresiones de los nudos.
Máquinas de vectores de soportes	La máquina de vector de los soportes se utiliza para poder tener detección y exploración de patrones complejos del dato agrupado, ordenado y clasificado de los datos.
Redes neuronales	Es la técnica de los modelados no lineales sofisticadas que es capaz de modelar las funciones complejas.

Nota. La tabla anterior muestra los tipos de modelos predictivos. Adaptado de: (Masa, 2019)

2.5.2. Regresión lineal múltiple

La regresión lineal múltiple, permite la determinación de las relaciones de las causas y efectos que existen entre las variables dependientes e independientes. Este modelo es aplicado cuando se posee razones para entender que hay más de un elemento que afecta a las variables de estudio, su fórmula es $\gamma = 0 + B1 \times X1 + B2 \times X2 + \cdots + Bn \times Xn + \in$ (Arellano, 2020).

2.5.3. Lenguaje de Programación Python

Se puede observar que este tipo de lenguaje sirve para la programación de Python, son lenguajes que son factibles para poder aprender, teniendo en cuenta las estructuras de los altos niveles que permiten dar distintos enfoques de forma simple pero efectivos, hacia las programaciones orientadas al objeto. La característica que presentan en sus sintaxis y sus tipados dinámicos lo convierten en lenguajes ideales para scripting y para los desarrollos de la aplicación en las diversas áreas, estas sean inteligencias artificiales, big data, data science, los desarrollos web, etc.

Este tipo de lenguaje posee diversas ventajas, que son las siguientes:

- Software libre.
- Se instala en cualquier ordenador.
- Posee un lenguaje sencillo.
- Es un lenguaje rápido.
- No requieren declaraciones del tipo de dato.
- Es muy legible.
- Existen gran información.
- Incorpora grandes cantidades de librerías.
- Fórmula de la regresión lineal múltiple.

Librería Pandas

Son paquetes de Python que proporcionan estructuras de datos que actúan de forma rápida, flexible y expresiva, que es diseñada para trabajar con datos "etiquetados" las cuales, facilitaran las manipulaciones y análisis. Pandas son las herramientas utilizadas por un científico de dato, ya que, permiten: poder limpiar, analizar y finalmente organizar los datos de análisis en una manera adecuada para las visualizaciones. Este puede servir para diversos tipos de datos: estos son datos tabulares con las columnas de tipo heterogéneo, datos de series de tiempo ordenado y no ordenado, cualquier otra manera de conjuntos de datos estadísticos (Manobanda, 2019).

2.6. JSON Web Token

Son tokens de accesos estandarizados en el RFC 7519 que permiten los intercambios seguros de datos entre dos partes. Contienen todas las informaciones importantes sobre las entidades, lo que implican que no hacen falta para consultar las bases de datos ni que las sesiones tengan que guardarse en los servidores.

2.6.1. Como se estructura los JSON Web Token

Un JWT firmado consta de tres partes, todas ellas codificadas en Base64 y separadas por un punto:

Tabla 4.

Estructura de los JSON Web Token

Estructura	Definición	Fórmula
Header	Estos están compuestos generalmente de dos diversos valores y proporcionan informaciones importantes sobre los tokens. Contienen los tipos de tokens y los algoritmos de las firmas y cifrados usados.	{"alg":"HS256","typ": "JWT"}

Payload	Contienen las informaciones reales que se transmitirán a las aplicaciones. Aquí se define algún estándar que determina qué los datos se transmite y cómo. Las informaciones que se proporcionan los como pares “key/value (clave-valor)”; la clave denominada “claims en JWT”	{"sub":"123","name": "Alicia","exp": 30}
Firma	Estas se crean usando las codificaciones Base64 del header y del payload, así como los métodos de firmas o cifrados especificados. Las estructuras vienen definidas por JSON Web Signature (JWS), son estándares establecidos en el RFC 7515. Para que las firmas sean eficaces, es necesario usar una clave secreta que solo conozcan las aplicaciones originales.	{7WK5T79u5mIzjIXXi2oI9F glmgivv7RAJ7izyj9tUyQ}

Nota. La tabla anterior muestra la estructura de los JSON Web Token. Adaptado de: (Salas, 2020).

2.6.2. Funcionamiento de JSON Web Token

Al comienzo de las sesiones de los usuarios ejemplifican de manera correcta las funciones del JSON Web Token. Antes de usar el JWT, hay que establecerse una clave que sea secreta. Una vez que la persona que está usándola han introducido de manera adecuada las credenciales, el JWT se devuelven a colocar con las claves y se guardan localmente. Las transmisiones deben realizarse por medio de HTTPS para que el dato esté mejor protegido (Salas, 2020).

De esta forma, cada vez que los usuarios acceden al recurso protegido, como a una API o a las rutas protegidas, el user agent utilizan el JWT como parámetros o como header de autorizaciones. Las otras partes pueden descifrarse el JSON Web Token y ejecutar las solicitudes si las verificaciones se realizan correctamente.

2.6.3. En qué casos se utilizan JSON Web Token

JSON Web Token ofrecen diversas ventajas en las comparaciones con los métodos tradicionales de autentificaciones y autorizaciones con las cookies, por lo que se usan en las situaciones que se mencionan a continuación:

- **Aplicaciones REST:** En la aplicación REST, el JWT garantizan las ausencias de los estados enviados en el dato de autentificaciones directamente con las peticiones.
- **Intercambio de recursos de origen cruzado:** JSON Web Token envían informaciones por medios de los llamados "*cross-origin resource sharing*", los cuales le dan las grandes ventajas sobre las cookies, que no suele enviarse con estos procedimientos.
- **Uso de varios frameworks:** JSON Web Token están estandarizados y pueden usarse una y otra vez. Cuando se utilizan múltiples frameworks, el dato de autentificaciones puede compartirse con fácilidad.

2.7. Definiciones de Arquitectura cliente servidor del software

Auth0: Son soluciones flexibles e inmediatas para añadir los servicios de autenticaciones y autorizaciones a su aplicación. Sus equipos y organizaciones puede evitarse los costos, tiempos y los riesgos que conllevan las creaciones de sus propias soluciones para la autenticación y autorización del usuario (Ben Dalla, 2020).

Cloud Computing o computación: Es un término generalizado que se aplican a los procesos y servicios que están alojados en la nube por medio de Internet. Son sistemas que permiten el ofrecimiento del servicio de computación por medio de las redes (Ben Dalla, 2020).

Cloud Services: Es la infraestructura, plataforma o sistema de software que el proveedor externo aloja y pone a disposición al usuario por medio de Internet.

GitLab: Son servicios de alojamientos de códigos y gestiones de la versión, unida a las completas plataformas DevOps. Descubren todos lo que tiene que saber sobre diversas cuestiones como son: cómo funcionan, diferencia con GitHub, caso de utilización para la Data Science y el Machine Learning, cursos, entre otras.

MongoDB: Se basa en el modelo de almacenamiento BSON (JSON binario). Los documentos se almacenan en colecciones. Al ser un almacén de datos sin esquema, no es necesario que dos documentos sean iguales en cuanto a los campos de un documento BSON (Calapucha y Tarco, 2019).

MongoDB Atlas: Son las bases de los datos en las nubes completamente administradas que manejan todas las complejidades de la implementación, administración y repartición de la implementación en los proveedores de servicios (Calapucha y Tarco, 2019).

MongoDB Charts: Son los gráficos Atlas de MongoDB, que son herramientas modernas de visualizaciones de datos. Que son partes integrales de las aplicaciones web del dato en la nube de MongoDB (Calapucha y Tarco, 2019).

Servidor privado virtual (VPS): Son máquinas que alojan todos los softwares y el dato necesario para las ejecuciones de las aplicaciones o sitios web. Se llaman virtuales porque solo consumen partes del recurso físico subyacente del servidor, que es administrado por los proveedores externos (Navarrete, 2019).

SOAP: Es un estándar de protocolo de nivel de aplicación que se utiliza para transportar mensajes en sistemas distribuidos. El estándar fue definido y es mantenido por el Grupo de Trabajo de Protocolo XML del Consorcio World Wide Web. SOAP se usa comúnmente en el contexto de los servicios web. Los mensajes SOAP se codifican mediante XML y están destinados a transportar datos de aplicación codificados en XML.

3. Desarrollo del proyecto

3.1 Arquitectura

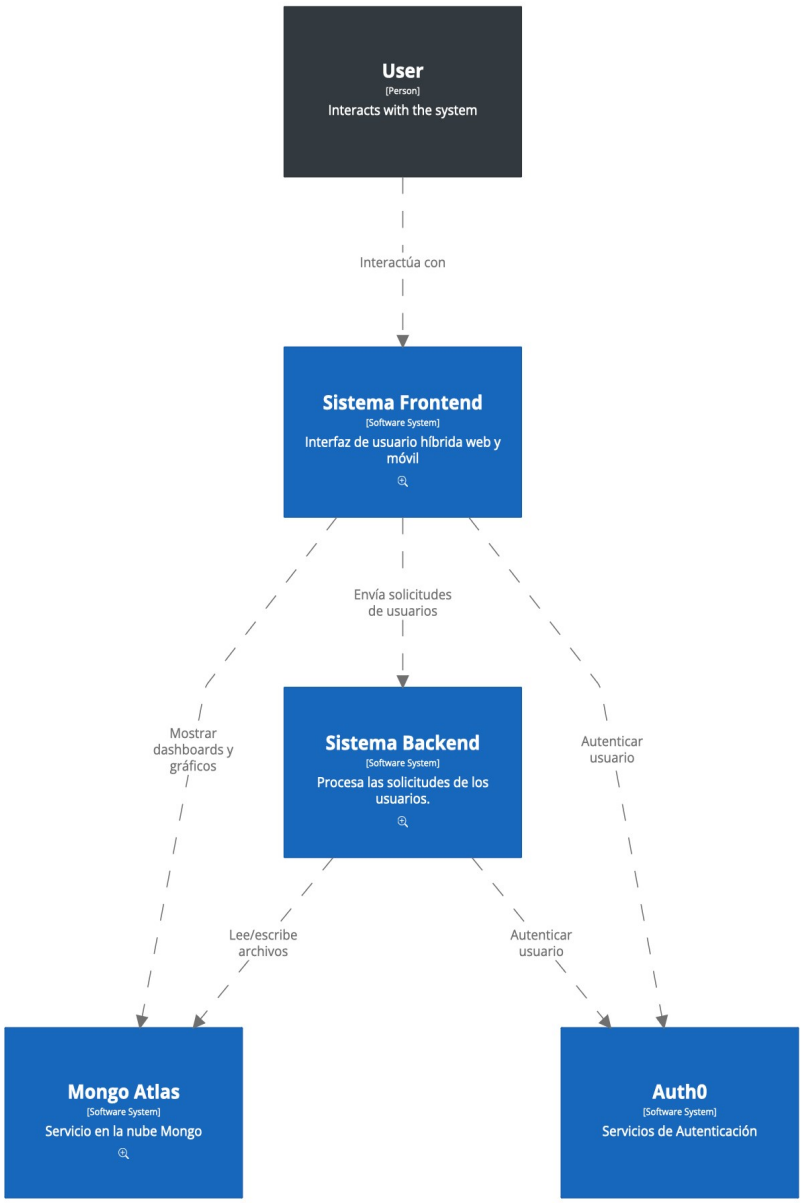
La arquitectura utilizada en el presente proyecto es cliente servidor, en el entorno web, en donde se evidencia que el Frontend se encuentra implementado en IONIC angular, el Backend está con el Auth JS, y el servicio de cloud con Auth0, para el manejo de usuarios y Mongo atlas para la base de datos y Dashboart.

La arquitectura de Frontend es monoltica donde se tiene un solo repositorio para todo el código del Frontend, utilizándose micro servicios para el cambio del Backend, teniendo sus responsabilidades cada uno. Por otro lado, el Gateway es el encargado de recibir todas las peticiones desde el Frontend encargándose de validar si el Frontend está autentificado con un usuario, por medio del micro servicio Out Service, entonces se conecta con Auth0 y verifica que el token de usuario que manda el Frontend sea el correcto y el Gateway permite pasar a los demás micro servicios como el File Update Service que es el encargado del cambio y de la validación del Excel para el aprendizaje de Machine Learning y después de validar lo guarda en la base de datos.

Otro servicio que se tiene es el de Machine Learning que este está hecho con Python, él es el encargado de hacer la predicción y sacar la información desde la base de datos que tiene, después la guarda en una base de datos en Mongo predicción, se trabaja con cuatro micro servicios, teniendo en consideración que Gateway es la puerta de entrada y salida al mundo o al internet, de los otros micro servicios no tienen salida al internet. Por otra parte, el Gateway es el que siempre se encarga de recibir todas las peticiones del Frontend y el que también se encarga de validar el inicio de los usuarios, esto quiere decir que la codificación que se tiene entre Frontend back es con JWT, para lo que es el token de verificación con la ayuda Auth0, conociendo que la arquitectura de en el Backend, micro servicios en el Frontend es un monolito y en general es una arquitectura cliente servidor que es una plataforma web.

Figura 2.

Arquitectura Cliente servidor



Nota. Elaboración propia, con la utilización del programa (System context) Frontend system.

3.2 Frontend

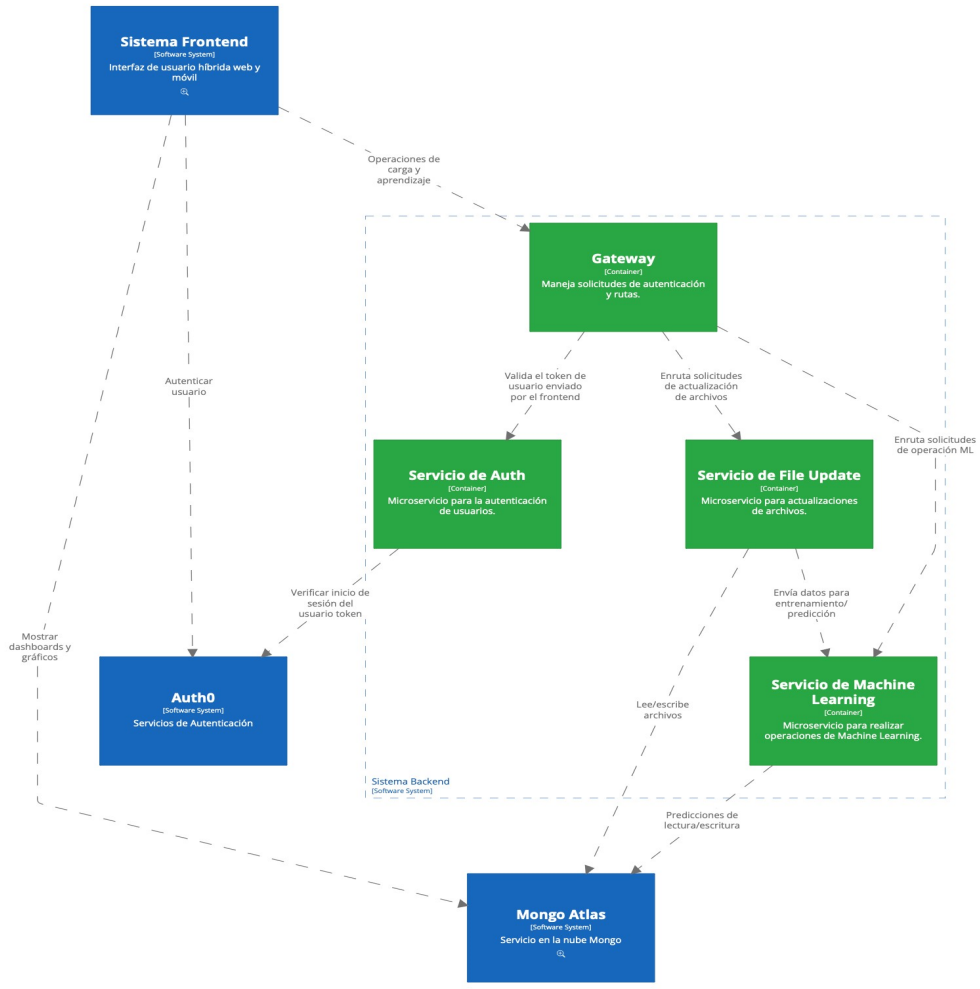
Se encuentra realizado con el framework que es el IONIC, porque permite hacer aplicaciones web y móvil, mejor conocidas como aplicaciones híbridas gracias a esa preferencia que tiene el framework permite que escale la aplicación, ya que se podrá exportarle a una aplicación móvil por eso se utiliza IONIC y Frontend los dos se

manejan con angular como framework, siendo este una motivación para coger el frontend la IONIC.

Cabe mencionar que IONIC tiene componentes ya desarrollados que son reutilizables para menorar los tiempos de desarrollo y cuenta con sus propios iconos los cuales permite hacer una aplicación en corto tiempo.

Figura 3.

Representación arquitectónica



Nota. Elaboración propia, con la utilización del programa (System context) Frontend system.

3.2.1 Módulos Frontend

Se utilizarán cuatro módulos los cuales están descritos a continuación:

Module Login: es conectarse al Auth0 directamente el Frontend Auth0 para autenticarse y leer el token JWT, una vez autenticado tiene acceso a los otros módulos que es el upload, dashboard y el de perfil.

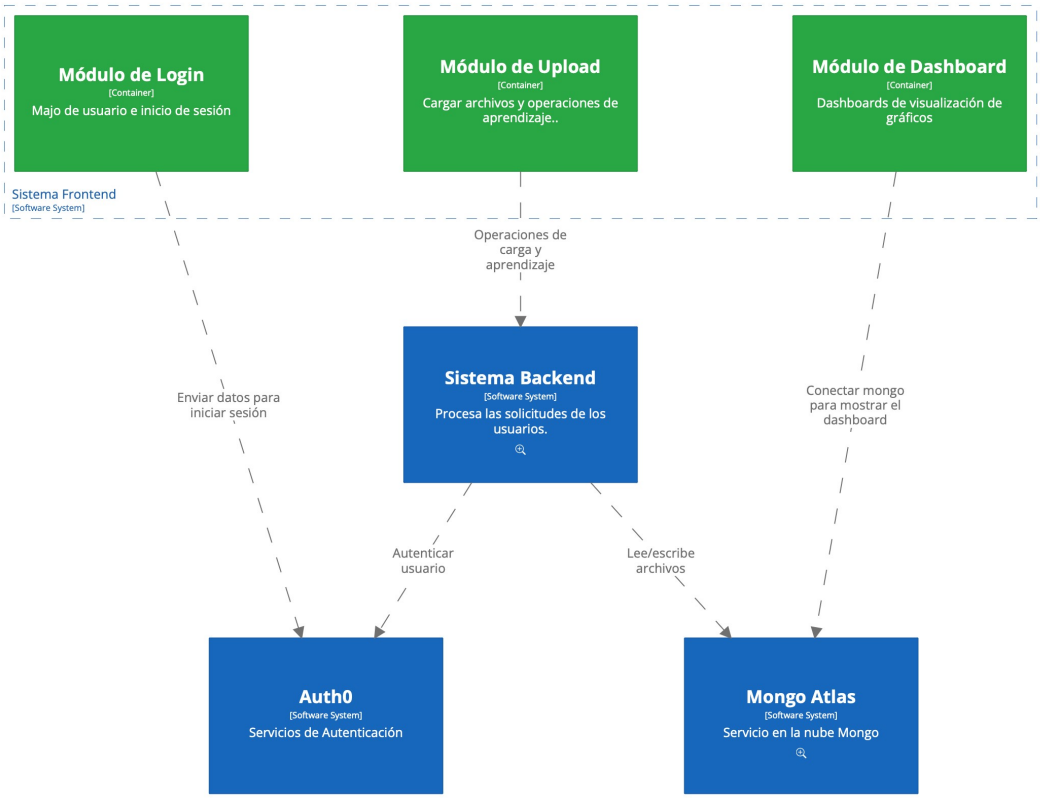
Module Upload: es un formulario donde se sube el Excel con el mes y el año que tiene la información y se manda el Backend, las peticiones que se hacen en el Backend siempre tienen el tocken JWT, esto significa que todas las peticiones que se hacen al Back van de cabecera la autorización con el JWT que otorga el Auth0, en conclusión, se sube el archivo.

Module dashboard: es donde muestra las gráficas con integración con el SDK de MongoAtlasChart, igual se utiliza el Mongochart para menorar los tiempos de desarrollo y reutilizar esta herramienta en la nube que ya está hecha y es el Mongoatlas, con el dashboard es la configuración del SDK de Mongochart y mostrar los gráficos.

Module Perfil: este indica el usuario que este logueado y le indica el correo y cerrar sección.

Figura 4.

Diseño de los módulos del Frontend



Nota. Elaboración propia, con la utilización del programa (System context) Frontend system.

3.3 Backend

Este hecho con microservicios con NestJs, existen responsabilidades separadas como son Gateway, Auth Service, File Update Service y machine learning, estos no tienen salida al internet, por eso siempre es el que se encarga de recibir todas las peticiones desde el fondo y el que también se encarga de validar el inicio de los otros usuarios ya entonces la codificación que se tiene entre Frontend back es con JWT, para lo que es el token con la ayuda de la arquitectura de aquí sería en el back micro servicios y en general es que es una plataforma

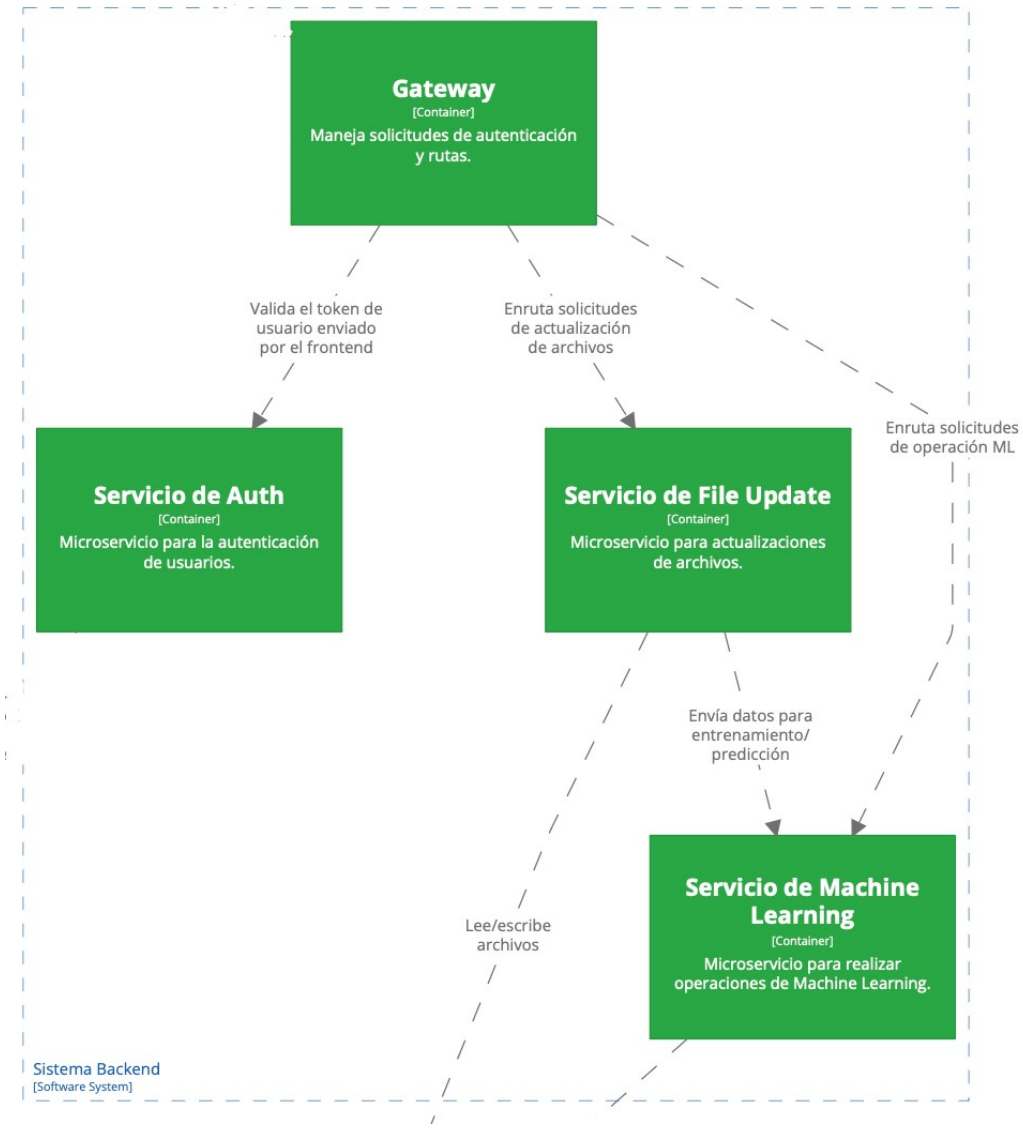
3.3.1 Gateway

Se conecta con el Frontend así sería el flujo entonces la función principal del Gateway es orquestar los micro servicios y la seguridad del Backend entonces él es el encargado de validar el token que manda el Frontend por medio del micro servicio a ount service entonces el Gateway tiene esas dos funcionalidades orquestar los micro servicios conectar entre los micro servicios y la seguridad que brinda en este caso la seguridad que da es la validación del token de autenticación de JWT que lo hace conjuntamente con el out service y el Auth0 entonces el Auth service se conecta con el Auth0 y validan el tocken.

Esta realizado con NestJs es el encargado de manejar todas las peticiones del Front al Back es quien se encarga de validar los tokens JWT con ayuda del microservicio AUTH y luego de validar a los microservicios.

Figura 5.

Estructura del Gateway



Nota. Elaboración propia, con la utilización del programa (System context) Frontend system.

3.3.2 Micro servicios

AUTH service: Es el microservicio encargado de conectarse a AUTH0 para validar los tokens JWT que envía el Front en la cabecera en cada petición al back y es el encargado de codificar el JWT para saber que usuario este logueado, él es el encargado de verificar que el token sea válido y de saber quién es el que está logueado en el sistema.

File Update service: Encargado de validar que un Excel tenga el formato específico aquí podemos acotar que en este micro servicio se utiliza el patrón estrategy se utilizó

porque podemos validar varios Excel, ya que cada uno es una estrategia ahorita la única estrategia que existe es la de ventas porque se va a subir varios Excel de ventas que tienen el mismo flujo que son 4 columnas y después de validar el Excel se guarda en la base de datos mongo, transformando el Excel en un JSON y almacenándolo en mongo.

Machine Learning: Micro servicio echo en Python con Flask Igual se aplica el patrón startegy para que lea la data de la base mongo y se aplique la configuración de la librería panda de python, de acuerdo al tipo de data que se tiene para hacer regresión lineal múltiple, luego de hacer la predicción con la progresión lineal se almacena la data en la base de datos para que utilice Mongo Chart.

3.3.3 Micro de predicción

Es el encargado de realizar la predicción de las ventas, ya que se une la base de datos para leer los archivos transformados y realizar la predicción por medio de la librería panda y Python donde se usa la regresión lineal múltiple.

3.3.3.1 Python y lib. Panda

Python Flask

Según, (Developer Resource Center, 2022), Python es un tipo de lenguaje que se puede programar e interpretar los altos niveles orientados a objetivos con sintaxis fáciles de leer, son muy extendidos en la informática científica, automatización y el desarrollo web, es ideal para los de prototipo y tarea ad hoc. Como lenguaje de programación el propósito general es ser amigable para principiantes, Python cuenta con el respaldo de muchos científicos informáticos y desarrolladores de aplicaciones líderes en todo el mundo.

Flask, un pequeño cuadro de solicitud web en Python, está documentado como un microframework porque no requiere bibliotecas especiales. No tiene una capa de abstracción de base de datos, validación de formularios ni otros componentes cuya funcionalidad común la proporcionen bibliotecas de terceros ya existentes. La biblioteca Mongo Python se utiliza para la conexión de la base de datos.

Pandas

Según, (Chacón, 2021), es una biblioteca que posee códigos abiertos muy populares entre el desarrollador de Python, en donde es especial las ciencias de datos y

aprendizajes automáticos, porque proporcionan un marco muy potente y flexible que facilitarán las manipulaciones y procesamientos de datos.

Pandas nació de la necesidad de combinar todo lo necesario en una sola biblioteca, para que un analista de datos tenga todas las funciones que necesita todos los días en una misma herramienta, tales como: carga de datos, modelado, análisis, manipulación y preparación de datos.

Regresión Lineal Múltiple. Se puede decir que a menudo también se agregan los modelos de los términos de intersecciones constantes. Las regresiones lineales se utilizan para la generación de las informaciones para los gráficos que contienen al menos dos campos continuos, uno de la identificación por los objetivos y el otro como predictores. Además, se puede dar una especificación de los predictores categóricos y de dos diversos campos de forma auxiliar continua en los gráficos usados para la generación de los modelos de las regresiones adecuadas (IBM, 2023).

Los modelos de regresión lineal son estadísticos generales para poseer una estimación generalizada para la estimulación de las relaciones de la medición continua y variable predictiva. Por ende, los predictores pueden ser de forma continua, categórica o derivada, por lo que a la vez admite relaciones no lineales. Los modelos lineales cuentan con variables las cuales son utilizadas y se detallan a continuación:

- **DateBilling:** La variable da a conocer la fecha de facturación de las ventas realizadas. Se utiliza la regresión para analizar patrones temporales y tendencias de ventas durante un período de tiempo. De esta forma, se espera poder predecir las ventas futuras en base al comportamiento histórico asociado a diferentes días del año.
- **Gender:** Esta variable indica el sexo del cliente, hombre o mujer. Al incorporar esto en el modelo, el objetivo es comprender cómo varían las ventas según el género del comprador. Con esta información, las empresas podrán determinar qué género es más probable que compre y diseñar estrategias de marketing y ventas específicas para cada segmento.
- **Age:** Refleja la edad del cliente en el momento de la compra. Incluya esto en una regresión para determinar cómo se distribuyen las ventas entre los grupos de edad. Comprender las tendencias de compra por edad permite a las

empresas segmentar el mercado y adaptar las estrategias y los productos a los diferentes grupos de edad, aumentando así su alcance y eficacia.

Se utiliza las variables mencionadas anteriormente porque ayudan a establecer o predecir las ventas futuras en las fechas seleccionadas durante todo el año, al igual la búsqueda de las ventas por medio del género y verificar quien tiene más fuerza en las ventas si las mujeres o los hombres, al igual que la predicción de las ventas por edad y segmento del público, ayudando a conocer cuáles serán las ventas a futuro y verificar si tiene un impacto positivo en las empresas.

3.4 Servicios de la Nube

La ventaja de este servicio es evidente por su utilización, ya que, no se limita a los ordenadores, y las capacidades de almacenamientos son mayores que las de on ordenador común.

3.4.1 Auth0

Se utiliza en la app tanto en el frontend y el back para el control de usuario y el manejo. El mecanismo de autenticación es responsable de crear credenciales, que son la representación interna del producto de un usuario autenticado con éxito. No todas las credenciales son iguales. La funcionalidad de las credenciales está determinada por el mecanismo de autenticación configurado.

Para iniciar primero se debe crear un usuario con su respectiva contraseña, la cual permitirá el manejo adecuado de la aplicación.

3.4.2 Mongo Atlas

El servicio en la nube de la base de datos mongo y mongomap, también se utilizan para crear gráficos para el panel frontal mediante la lectura de datos de pronóstico. Tenga en cuenta que Mongo Atlas es una base de datos como servicio y puede implementar, utilizar y poder escalar una base de datos Mongo con solo unos pocos clics. Las funciones que ofrece Mongo Atlas incluyen la automatización y también se usa para los almacenamientos de la gran cantidad de datos. A diferencia de la} base de datos relacionales SQL tradicionales, por ende, Mongo no se basan en las columnas o tablas. Los datos se van almacenando en las formas de colección y documentos.

3.5 Infraestructura

Se va a utilizar un servidor VPS, que es un servidor privado virtual el cual trabaja en entornos virtuales aislados en los servidores físicos que pertenecen y son operados por los proveedores de los alojamientos web o de la nube y para el despliegue de la aplicación se va a utilizar 4 CORE de procesamiento y 8 GIGAS en RAM con un disco solido de 512 GIGAS, con la utilización del Docker file IONIC con Angular, Docker file nestJs, y Docker File Python Flask

3.5.1 Docker Files

Es un instrumento de gestión de configuración para automatizar la implementación de software en contenedores ligeros. Estos contenedores ayudan a que las aplicaciones se ejecuten de manera eficiente en diferentes entornos.

Es una ventana acoplable de fácil y rápido de instalación, cuenta con aislamiento de aplicaciones, también con una gestión de seguridad, es productivo, escalabra y tiene una independiente infraestructura.

Por lo cual, un contenedor Docker, son los paquetes de software que tienen todas las dependencias que son necesarias para la ejecución de las aplicaciones.

El Docker con IONIC angular es un marco JavaScript muy popular para diseñar y desarrollar aplicaciones web. Angular es desarrollado y mantenido por el equipo de Google.

Docker file nestJs, añade enviar a la sección file, el cual proporciona una lista de variables las cuales se encuentran en un ambiente que nuestra la aplicación NestJS.

Docker File Python Flask, es donde se archiva como un microframework, ya que no requiere bibliotecas particulares.

3.5.2 Docker Compose

Es una herramienta que le ayuda a compartir y definir aplicaciones de múltiples contenedores. Con Compose, creas un archivo YAML para definir tu servicio y con un solo comando puedes activar o desactivar todo lo que has hecho antes.

3.5.3 OVCloud VPS

Los servidores privados virtuales, a la vez también es conocido como VPS, actúan como entornos virtuales aislados porque los servidores físicos administrados por

proveedores del alojamiento web o en las nubes. El alojamiento VPS utiliza tecnología de virtualización para dividir una máquina física en múltiples entornos de servidores privados que comparten recursos.

3.6 Desarrollo

Para el ingreso a Auth0 y a los demás comandos se describe todo el proceso de ingreso:

Figura 6.

AuthoFront

```
import { isPlatform } from '@ionic/angular';
import config from 'capacitor.config';
import { environment } from 'src/environments/environment';

export const domain = environment.domain;
export const clientId = environment.clientId;
const { appId } = config;

const auth0Domain = domain;
const iosOrAndroid = isPlatform('hybrid');

export const callbackUri = iosOrAndroid
  ? `${appId}://${auth0Domain}/capacitor/${appId}/callback`
  : environment.host;
```

Este código es un módulo de JavaScript que se utiliza para configurar la autenticación en una aplicación que utiliza la plataforma Ionic y la biblioteca Auth0 para la autenticación de los usuarios.

Primero, importa algunas utilidades necesarias:

- **IsPlatform de @ionic/angular**: Una función que determina en qué plataforma se está ejecutando la aplicación (por ejemplo, iOS, Android, web, etc.).
- **Config de capacitor.config**: La configuración de la biblioteca Capacitor, que es un runtime que permite a las aplicaciones Ionic ejecutarse en diferentes plataformas.
- **Environment de src/environments/environment:** Un objeto que contiene información de configuración específica del entorno.

A continuación, se extraen y definen algunas constantes útiles:

- Domain y clientId son extraídos del environment. Estos son valores específicos de la aplicación que se utilizan para la autenticación con Auth0.
- AppId es extraído de la configuración de Capacitor.
- Auth0Domain es simplemente un renombre de domain.
- IosOrAndroid usa la función isPlatform para determinar si la aplicación se está ejecutando en una plataforma híbrida (es decir, iOS o Android).
- Finalmente, callbackUri se define como una URI de devolución de llamada que se utilizará después de que el usuario se autentique con éxito. El valor exacto de callbackUri depende de si la aplicación se está ejecutando en una plataforma híbrida o no. Si se está ejecutando en una plataforma híbrida, la URI de devolución de llamada se genera utilizando el appId y el auth0Domain. Si no, se utiliza el valor host definido en el environment.

Figura 7.

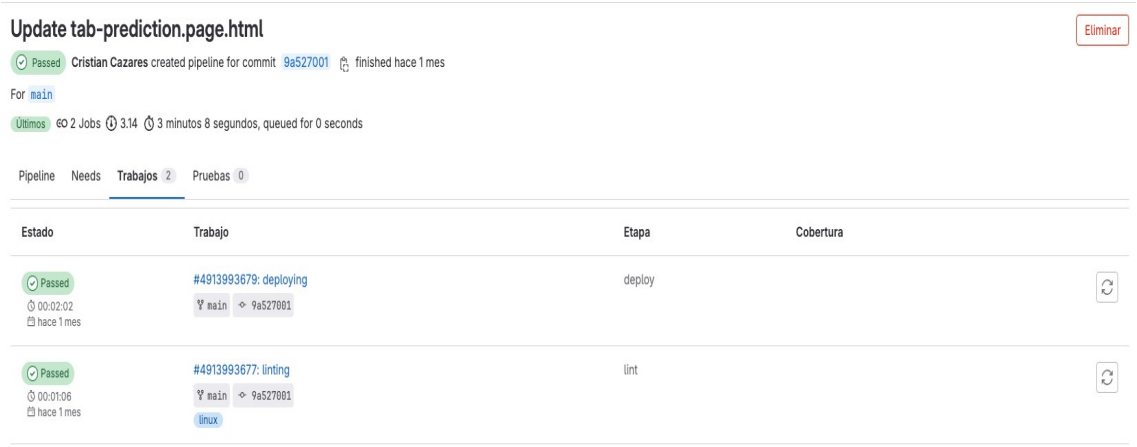
CI-CD Gitlab

```
stages:
  - lint
  - deploy

linting:
  stage: lint
  only:
    - main
  image: node:latest
  tags:
    - linux
  script:
    - npm install
    - npm run lint

before_script:
  - apt-get update -qy
  - apt-get install -y sshpass

deploying:
  stage: deploy
  only:
    - main
  script:
    -
sshpass -p "$SSH_PASSWORD" ssh -o StrictHostKeyChecking=no ubuntu@$SERVER_IP
"cd ~/projects/prediction_infrastructure && make update_mc_file_update"

```

Este es un ejemplo de un archivo. gitlab-ci.yml para la configuración de GitLab CI/CD. En este archivo se definen las etapas y los trabajos de tu pipeline de CI/CD, en la imagen se puede identificar la estabas que se ha utilizado.

Figura 8.

Update tab



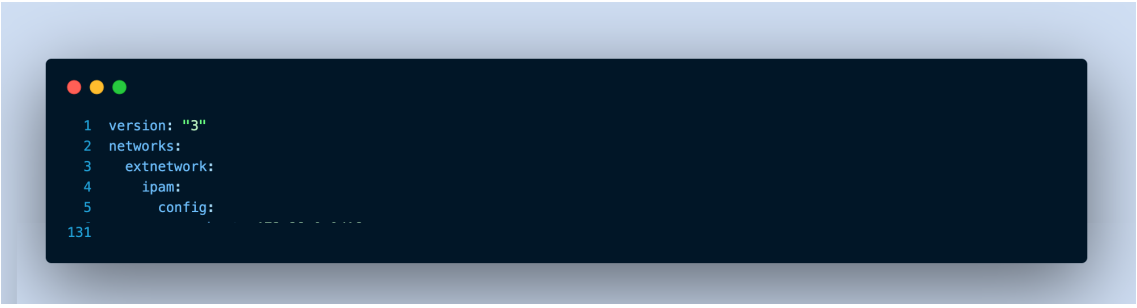
Docker-compose

Este es un archivo de Docker Compose versión 3 que describe una infraestructura de microservicios para una aplicación de predicción. Aquí está lo que hace cada sección:

- **Networks:** Se está definiendo una red con la subnet 172.28.0.0/16. Esta será la red a la que estarán conectados todos los servicios.

Figura 9.

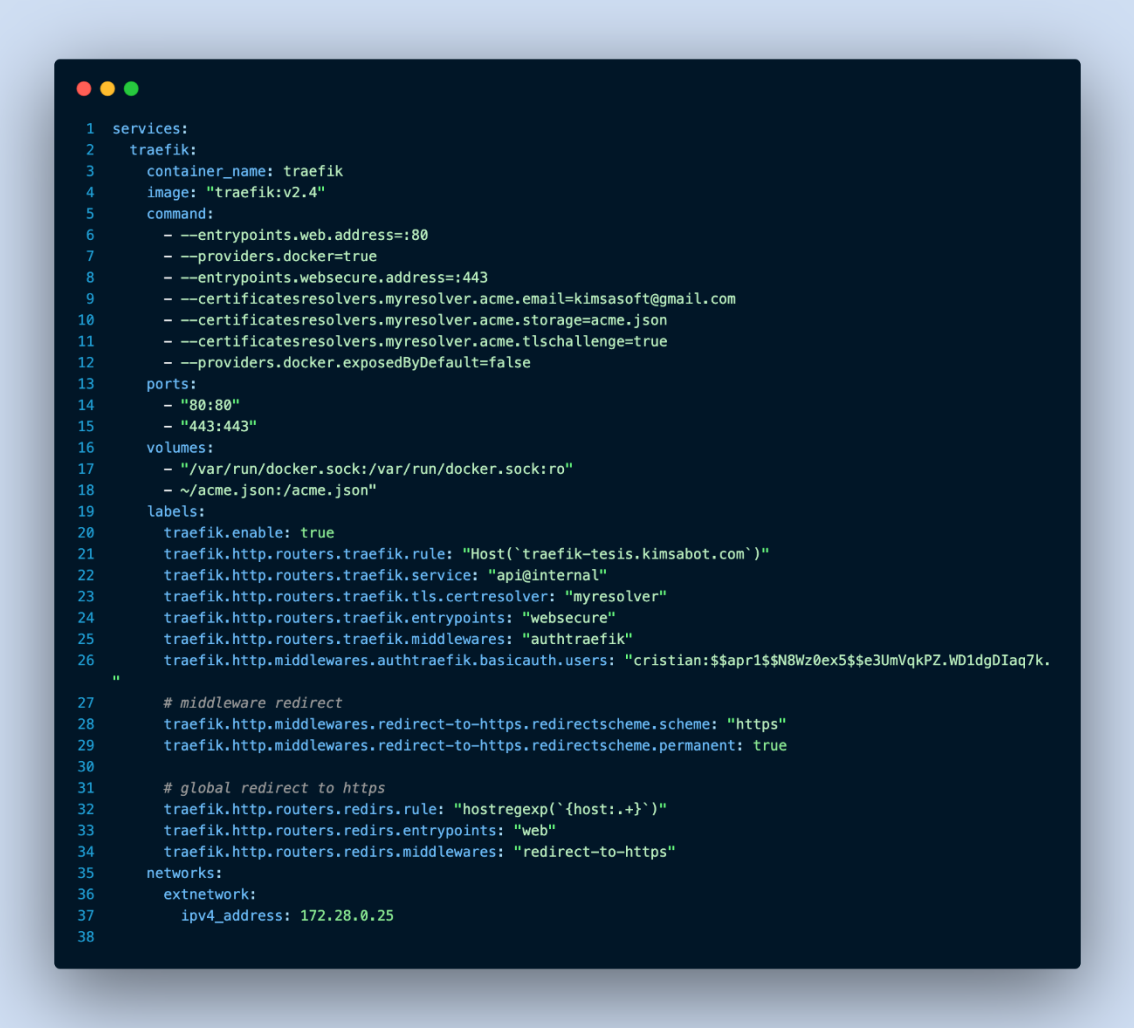
networks



- **Services:** Se definen los servicios que compondrán la aplicación.
- **Traefik:** Traefik es un enrutador de solicitudes HTTP y TCP moderno que se usa aquí para manejar el tráfico entrante. Traefik está configurado para usar certificados HTTPS proporcionados por Let's Encrypt. La autenticación básica está habilitada para el punto final Traefik API con el usuario "cristian". Las solicitudes HTTP son redirigidas permanentemente a HTTPS. Este servicio está conectado a la red extnetwork y tiene asignada la dirección IP 172.28.0.25.

Figura 10.

Services - Traefik

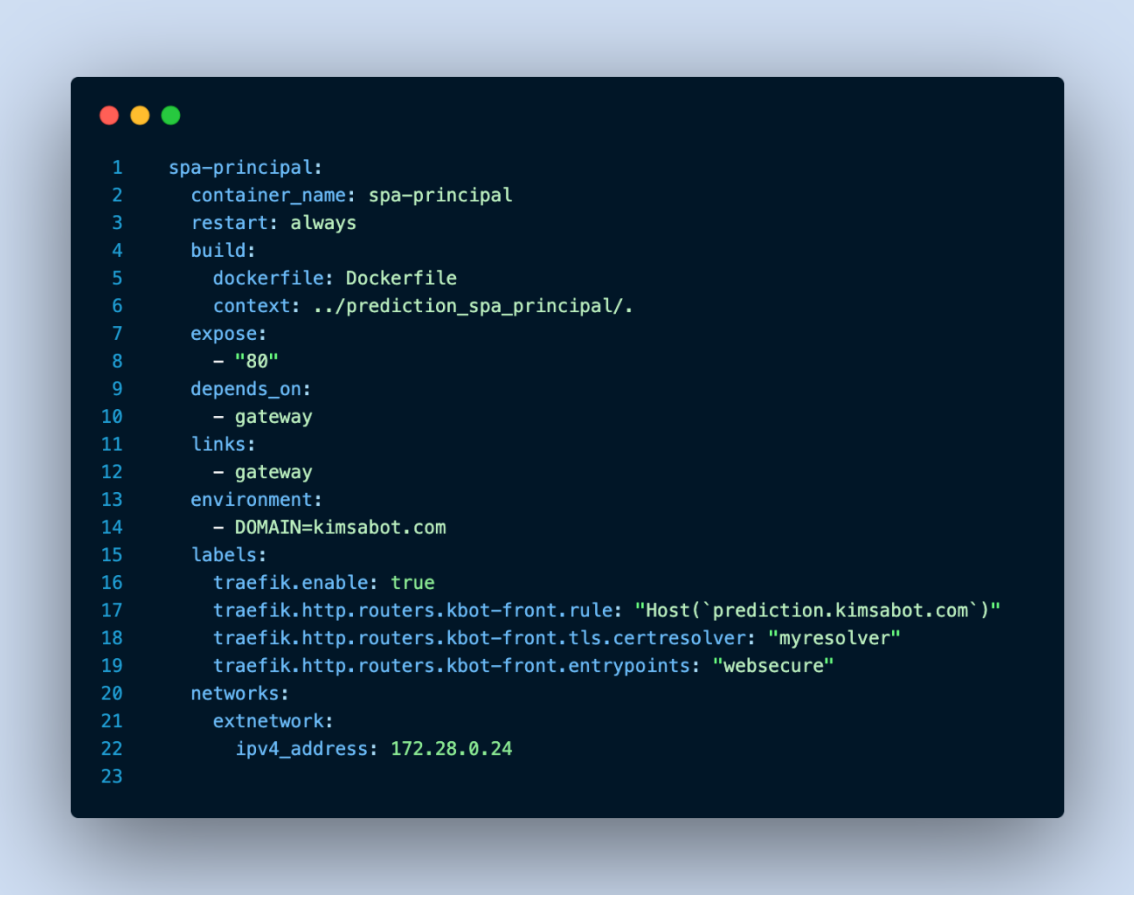


```yaml
services:
  traefik:
    container_name: traefik
    image: "traefik:v2.4"
    command:
      - --entrypoints.web.address=:80
      - --providers.docker=true
      - --entrypoints.websecure.address=:443
      - --certificatesresolvers.myresolver.acme.email=kimsasoft@gmail.com
      - --certificatesresolvers.myresolver.acme.storage=acme.json
      - --certificatesresolvers.myresolver.acme.tlschallenge=true
      - --providers.docker.exposedByDefault=false
    ports:
      - "80:80"
      - "443:443"
    volumes:
      - "/var/run/docker.sock:/var/run/docker.sock:ro"
      - ~/acme.json:/acme.json"
    labels:
      traefik.enable: true
      traefik.http.routers.traefik.rule: "Host(`traefik-tesis.kimsabot.com`)"
      traefik.http.routers.traefik.service: "api@internal"
      traefik.http.routers.traefik.tls.certresolver: "myresolver"
      traefik.http.routers.traefik.entrypoints: "websecure"
      traefik.http.routers.traefik.middlewares: "authtraefik"
      traefik.http.middlewares.authtraefik.basicauth.users: "cristian:$$apr1$$N8Wz0ex5$$e3UmVqkPZ.WD1dgDIaq7k.
"
      # middleware redirect
      traefik.http.middlewares.redirect-to-https.redirectscheme.scheme: "https"
      traefik.http.middlewares.redirect-to-https.redirectscheme.permanent: true

      # global redirect to https
      traefik.http.routers.redirs.rule: "hostregexp(`{host:.+}`)"
      traefik.http.routers.redirs.entrypoints: "web"
      traefik.http.routers.redirs.middlewares: "redirect-to-https"
    networks:
      extnetwork:
        ipv4_address: 172.28.0.25

```

- **Spa-principal:** Este es probablemente el front-end de la aplicación, alojado en un contenedor Docker. Está conectado a la red extnetwork y tiene asignada la dirección IP 172.28.0.24. Dependiendo del servicio "gateway", se expone el puerto 80.

Figura 11.

Spa-principal

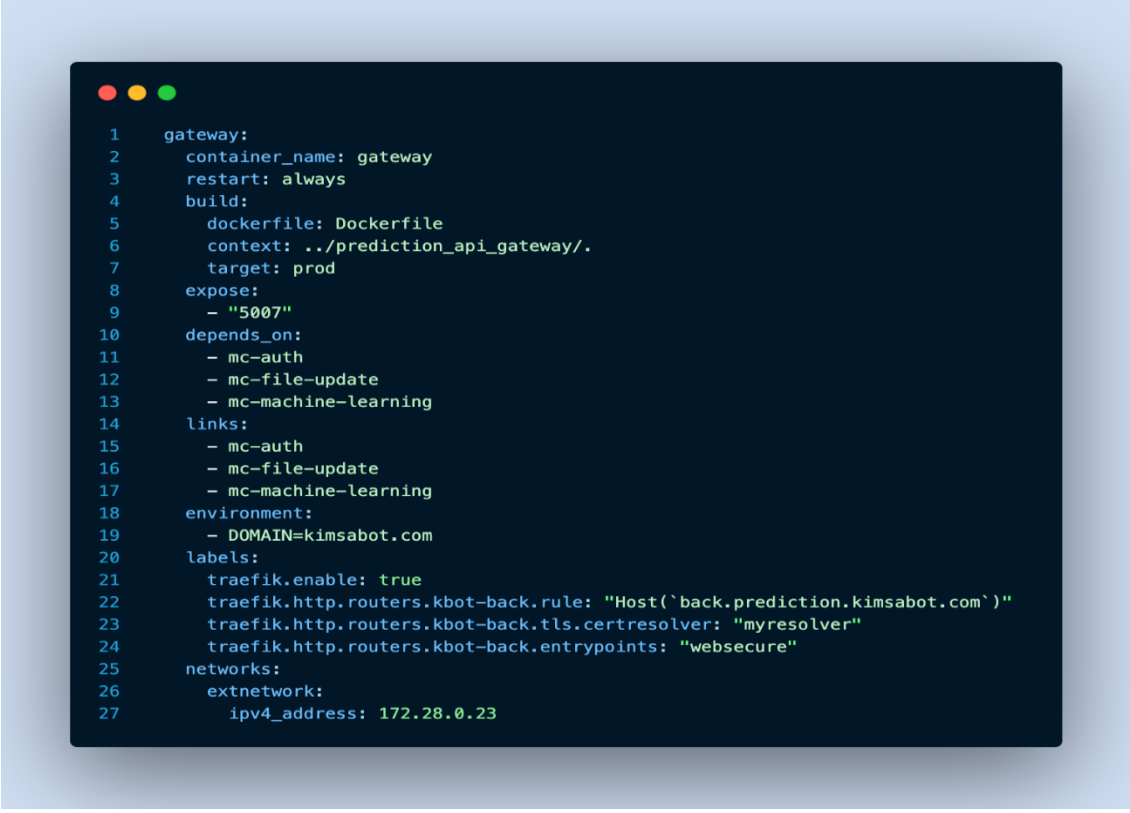


```
spa-principal:
  container_name: spa-principal
  restart: always
  build:
    dockerfile: Dockerfile
    context: ../prediction_spa_principal/.
  expose:
    - "80"
  depends_on:
    - gateway
  links:
    - gateway
  environment:
    - DOMAIN=kimsabot.com
  labels:
    traefik.enable: true
    traefik.http.routers.kbot-front.rule: "Host(`prediction.kimsabot.com`)"
    traefik.http.routers.kbot-front.tls.certresolver: "myresolver"
    traefik.http.routers.kbot-front.entrypoints: "websecure"
  networks:
    extnetwork:
      ipv4_address: 172.28.0.24
```

- **Gateway:** Actúa como un API Gateway, probablemente se encargue de la orquestación de las solicitudes entre los diversos microservicios. Está conectado a la red extnetwork y tiene asignada la dirección IP 172.28.0.23. Dependiendo de varios otros servicios ("mc-auth", "mc-file-update", "mc-machine-learning"), se expone el puerto 5007.

Figura 12.

Gateway

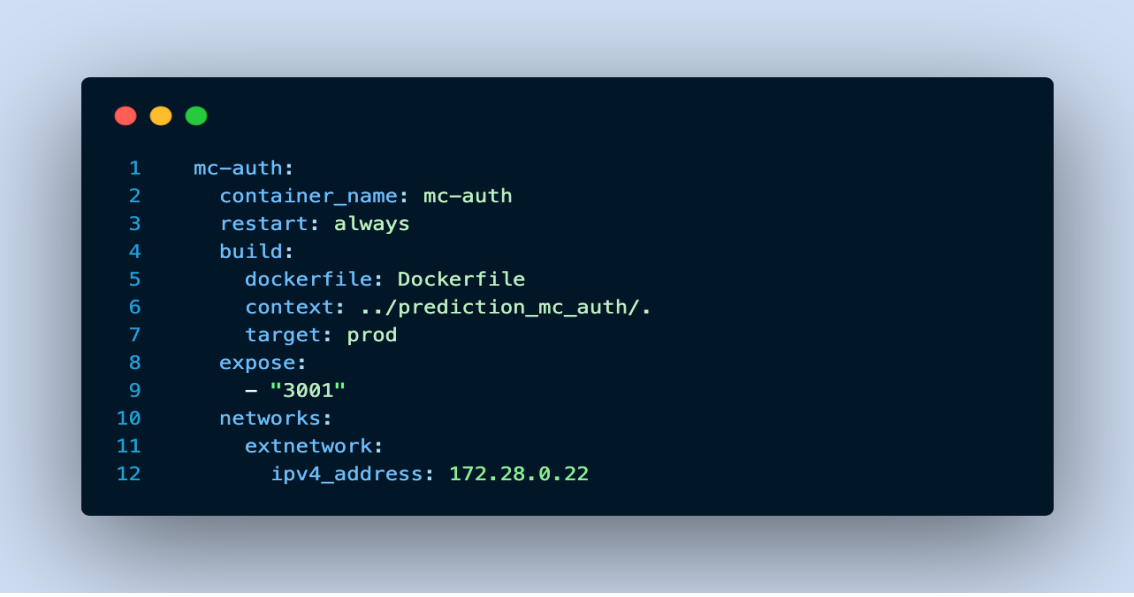


```
gateway:
  container_name: gateway
  restart: always
  build:
    dockerfile: Dockerfile
    context: ../prediction_api_gateway/.
    target: prod
  expose:
    - "5007"
  depends_on:
    - mc-auth
    - mc-file-update
    - mc-machine-learning
  links:
    - mc-auth
    - mc-file-update
    - mc-machine-learning
  environment:
    - DOMAIN=kimsabot.com
  labels:
    traefik.enable: true
    traefik.http.routers.kbot-back.rule: "Host(`back.prediction.kimsabot.com`)"
    traefik.http.routers.kbot-back.tls.certresolver: "myresolver"
    traefik.http.routers.kbot-back.entrypoints: "websecure"
  networks:
    extnetwork:
      ipv4_address: 172.28.0.23
```

- **Mc-auth:** Parece ser el servicio de autenticación de la aplicación. Está conectado a la red extnetwork y tiene asignada la dirección IP 172.28.0.22. Se expone el puerto 3001.

Figura 13.

mc-auth

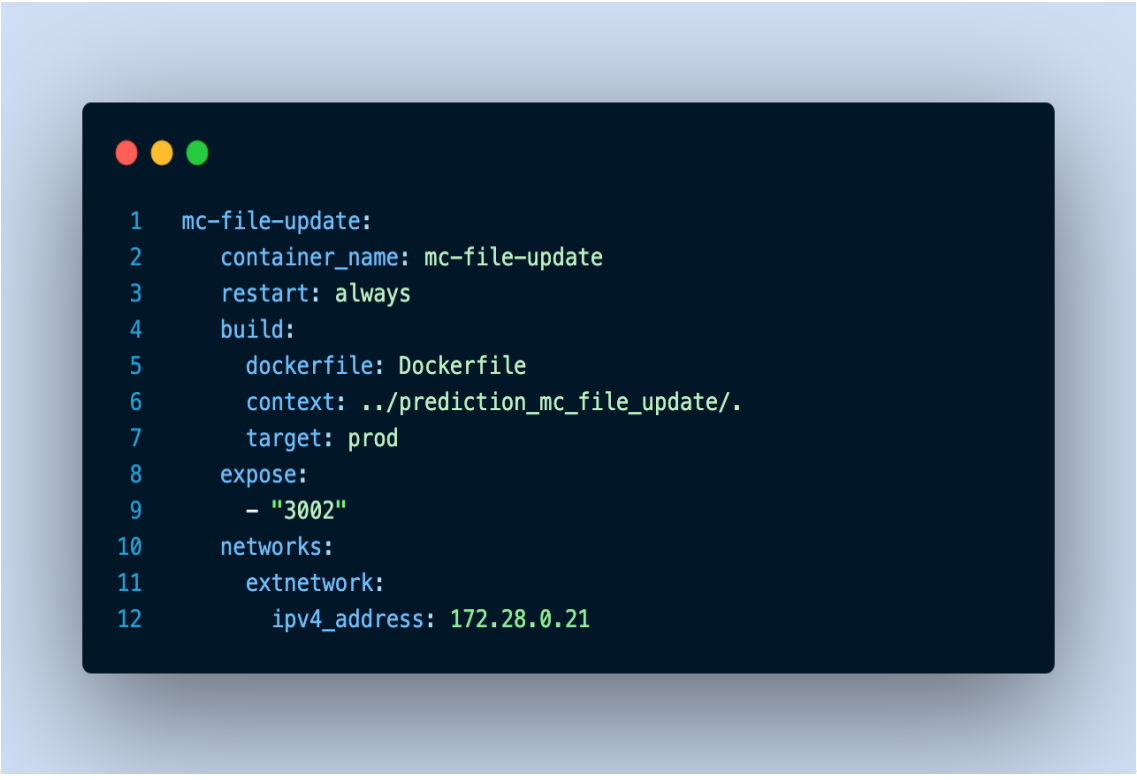


```
mc-auth:
  container_name: mc-auth
  restart: always
  build:
    dockerfile: Dockerfile
    context: ../prediction_mc_auth/.
    target: prod
  expose:
    - "3001"
  networks:
    extnetwork:
      ipv4_address: 172.28.0.22
```

- **Mc-file-update:** Este servicio puede estar relacionado con la actualización de archivos en la aplicación. Está conectado a la red extnetwork y tiene asignada la dirección IP 172.28.0.21. Se expone el puerto 3002.

Figura 14.

} *mc-file-update*

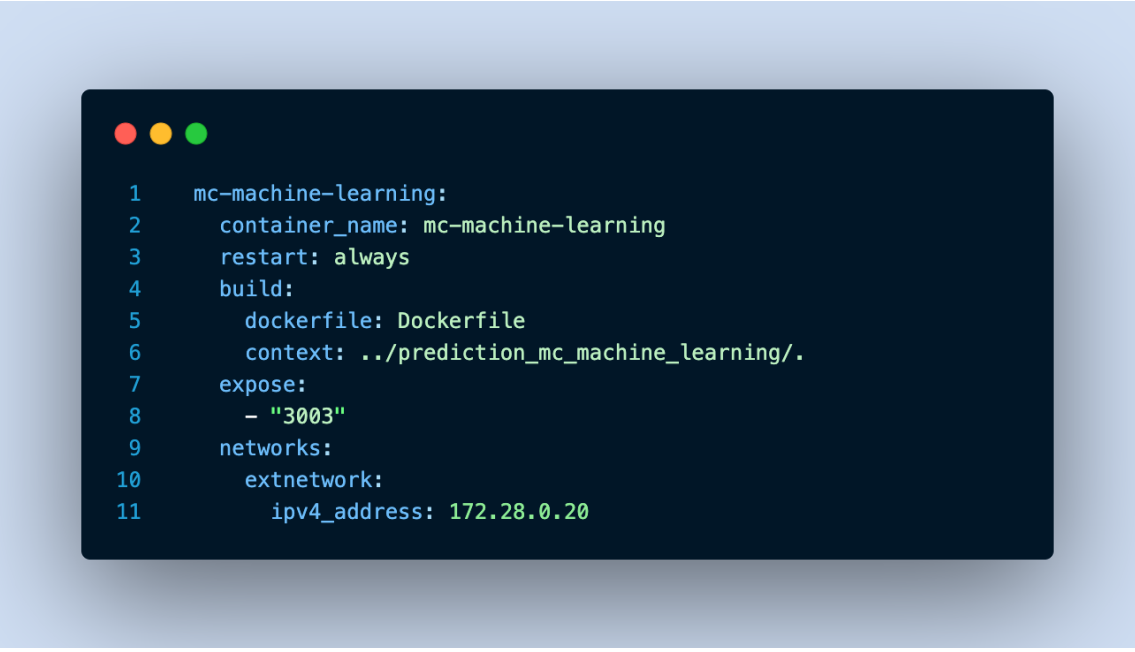


```
mc-file-update:
   container_name: mc-file-update
   restart: always
   build:
     dockerfile: Dockerfile
     context: ../prediction_mc_file_update/.
     target: prod
   expose:
     - "3002"
   networks:
     extnetwork:
       ipv4_address: 172.28.0.21
```

- **Mc-machine-learning:** Este servicio está probablemente encargado de los aspectos de aprendizaje automático de la aplicación. Está conectado a la red extnetwork y tiene asignada la dirección IP 172.28.0.20. Se expone el puerto 3003.

Figura 15.

Ms-machine-learning



Todos los servicios utilizan Traefik para enrutamiento y se configuran para utilizar certificados HTTPS proporcionados por Let's Encrypt a través del resolver “myresolver". También, todos los servicios están etiquetados para que se habiliten en Traefik.

Es importante destacar que cada servicio se construye desde un Dockerfile específico y en un contexto definido.

El propósito general de este archivo Docker Compose es definir cómo se orquestan estos servicios para que trabajen juntos en una red definida. El uso de Docker Compose permite manejar toda la infraestructura con un solo comando, lo que simplifica la implementación y el manejo de los servicios.

Figura 16.

DockerIonic

```
FROM node:16 as build-stage

WORKDIR /app

COPY package*.json /app/

RUN npm install -g @ionic/cli

RUN npm install

COPY ./ /app/

RUN ionic build

# Fase de producción
FROM nginx:1.17.1-alpine as production-stage

COPY --from=build-stage /app/www/ /usr/share/nginx/html

# Copia la configuración de nginx personalizada
COPY ./nginx.conf /etc/nginx/conf.d/default.conf

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]
```

Este archivo Dockerfile define una imagen Docker que construye y sirve una aplicación Ionic dentro de un contenedor Nginx.

- **Fase de construcción:** En la primera parte del archivo Dockerfile, se define la fase de construcción.
- **FROM node:**16 as build-stage: Aquí se utiliza la imagen oficial de Node.js versión 16 como base. Se le asigna el nombre build-stage a esta fase de construcción.
- **WORKDIR /app:** Esto establece el directorio de trabajo en el contenedor a /app.

- **COPY package*.json /app/:** Este comando copia los archivos package.json y package-lock.json (si existe) al directorio de trabajo en el contenedor.
- **RUN npm install -g @ionic/cli:** Aquí se instala globalmente el CLI de Ionic en el contenedor.
- **RUN npm install:** Esto instala las dependencias del proyecto que se especifican en package.json.
- **COPY ./ /app/:** Este comando copia todos los archivos y directorios del directorio actual en tu máquina al directorio de trabajo en el contenedor.
- **RUN ionic build:** Este comando construye la aplicación Ionic para la producción.
- **Fase de producción:** En la segunda parte del archivo Dockerfile, se define la fase de producción.
- **FROM nginx:1.17.1-alpine as production-stage:** Aquí se utiliza la imagen oficial de Nginx versión 1.17.1 como base. Se le asigna el nombre production-stage a esta fase de producción.
- **COPY --from=build-stage /app/www/ /usr/share/nginx/html:** Este comando copia los archivos construidos de la aplicación Ionic desde la fase de construcción al directorio por defecto donde Nginx sirve archivos estáticos.
- **COPY ./nginx.conf /etc/nginx/conf.d/default.conf:** Este comando copia una configuración personalizada de Nginx al contenedor. Esta configuración reemplazará la configuración por defecto de Nginx.
- **EXPOSE 80:** Esto expone el puerto 80 del contenedor para que se pueda acceder desde fuera del contenedor.
- **CMD ["nginx", "-g", "daemon off;"]:** Este es el comando que se ejecutará cuando se inicie el contenedor. En este caso, se inicia el servidor Nginx en modo no-daemon.

Figura 17.

DockerNoder

```
1  FROM node:lts  As build
2  RUN mkdir -p /usr/src/app
3  WORKDIR /usr/src/app
4  COPY --chown=node:node package*.json ./
5  RUN npm i
6  COPY --chown=node:node . .
7  RUN npm run build
8  ENV NODE_ENV production
9  USER node
10
11 FROM node:lts  As prod
12 RUN mkdir -p /usr/src/app
13 WORKDIR /usr/src/app
14 COPY --chown=node:node package*.json ./
15 RUN npm i
16 COPY --chown=node:node . .
17 RUN npm run build
18 ENV NODE_ENV production
19 USER node
20 CMD [ "node", "dist/main.js" ]
```

Este código es un Dockerfile de dos etapas para una aplicación Node.js. La construcción de Docker en varias etapas es un método que permite construir una imagen Docker en múltiples etapas, donde cada etapa puede tener un rol específico. En este caso, hay una etapa para la construcción ("build") y otra para la producción ("prod"). Aquí está la explicación de cada parte:

- **FROM node:** lts As build: Esto comienza la primera etapa de la construcción y establece la imagen base para esta etapa como node:lts, que es la última versión estable de Node.js.
- **RUN mkdir -p /usr/src/app:** Crea un nuevo directorio en la ruta especificada dentro del contenedor Docker.

- **WORKDIR /usr/src/app:** Establece el directorio de trabajo en el contenedor Docker. Todos los comandos que se ejecutan después de esta instrucción se ejecutan en este directorio.
- **COPY --chown=node:node package*.json ./:** Copia los archivos package.json y package-lock.json del directorio actual en la máquina host al directorio de trabajo en el contenedor Docker. La bandera --chown=node:node establece el propietario de los archivos copiados al usuario node.
- **RUN npm i:** Ejecuta npm install para instalar todas las dependencias especificadas en package.json.
- **COPY --chown=node:node . .:** Copia todos los archivos restantes del directorio actual en la máquina host al directorio de trabajo en el contenedor Docker.
- **RUN npm run build:** Ejecuta el script de compilación definido en package.json. Normalmente, este script compila la aplicación para la producción.
- **ENV NODE_ENV production:** Establece la variable de entorno NODE_ENV en production.
- **USER node:** Establece el usuario de las instrucciones RUN, CMD y ENTRYPOINT a node.
- **FROM node:lts As prod:** Comienza la segunda etapa de la construcción con la misma imagen base que la primera etapa.

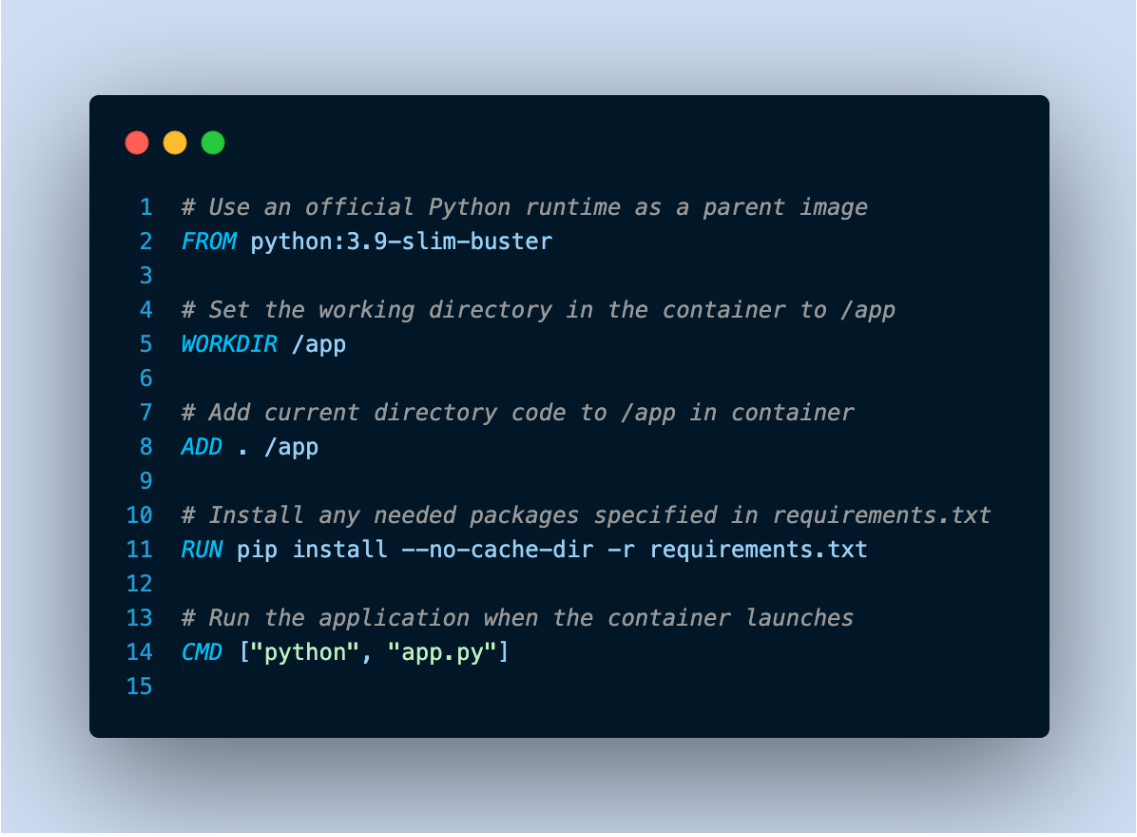
Las instrucciones de la 11 a la 18 son las mismas que en la primera etapa.

- **CMD ["node", "dist/main.js"]:** Especifica el comando que se debe ejecutar cuando se inicie el contenedor. En este caso, se ejecutará el script dist/main.js con Node.js.

Este Dockerfile proporciona una forma reproducible de empaquetar tu aplicación Node.js y sus dependencias en una imagen Docker, la cual luego puede ser ejecutada en cualquier sistema con Docker instalado.

Figura 18.

DockerPhyton



```
# Use an official Python runtime as a parent image
FROM python:3.9-slim-buster

# Set the working directory in the container to /app
WORKDIR /app

# Add current directory code to /app in container
ADD . /app

# Install any needed packages specified in requirements.txt
RUN pip install --no-cache-dir -r requirements.txt

# Run the application when the container launches
CMD ["python", "app.py"]

```

El código presentado es un archivo Dockerfile que se utiliza para crear una imagen Docker para una aplicación Python. Una imagen Docker es esencialmente una instantánea de una aplicación y su entorno, que puede ser ejecutada en cualquier sistema que tenga Docker instalado. Aquí te explico cada línea:

- **FROM python:** 3.9-slim-buster: Esta es la imagen base que se utiliza para construir tu imagen Docker. En este caso, estás utilizando una imagen oficial de Python versión 3.9 con una versión minimalista de Debian Buster como sistema operativo.
- **WORKDIR /app:** Establece el directorio de trabajo en el contenedor Docker.

Todos los comandos que se ejecuten después de esta instrucción se ejecutarán en este directorio.

- **ADD . /app:** Copia todos los archivos de tu directorio actual (en tu máquina host) en el directorio /app en el contenedor Docker.
- **RUN pip install --no-cache-dir -r requirements.txt:** Ejecuta el comando pip install para instalar todas las dependencias Python especificadas en el archivo

requirements.txt. La opción --no-cache-dir se utiliza para que pip no almacene los archivos de caché, lo que puede ayudar a reducir el tamaño de la imagen.

- **CMD ["python", "app.py"]:** Especifica el comando que se debe ejecutar cuando el contenedor se inicie. En este caso, se ejecutará el script app.py con Python.

Este Dockerfile proporciona una forma reproducible de empaquetar tu aplicación Python y sus dependencias en una imagen Docker, que luego puede ser ejecutada en cualquier sistema que tenga Docker instalado. Esta es una forma común de distribuir y desplegar aplicaciones en diversos entornos, desde desarrollo hasta producción.

Figura 19.

GuardGuateway

```
import {
  Injectable,
  ExecutionContext,
  UnauthorizedException,
} from '@nestjs/common';
import { CanActivate } from '@nestjs/common';
import { Observable, from } from 'rxjs';
import { map } from 'rxjs/operators';
import { UserAuth } from 'src/interfaces/auth';
import { Auth } from 'src/providers/auth/auth';

@Injectable()
export class AuthGuard implements CanActivate {
  constructor(private readonly auth: Auth) {}

  canActivate(
    context: ExecutionContext,
  ): boolean | Promise<boolean> | Observable<boolean> {
    const request = context.switchToHttp().getRequest();
    const authorizationHeader = request.headers['authorization'];
    if (!authorizationHeader) {
      throw new UnauthorizedException();
    }
    return from(this.auth.validToken(authorizationHeader)).pipe(
      map((user: UserAuth) => {
        if (!user) {
          throw new UnauthorizedException();
        }
        const userWithoutCircularReferences = {
          email: user['data'].email,
          picture: user['data'].picture,
        };
        request.user = userWithoutCircularReferences;
        return true;
      }),
    );
  }
}
```

Este código define un Guard en NestJS. Los Guards son una característica de NestJS que permite implementar lógica de autorización antes de que se invoque a un manejador de ruta.

Aquí tienes una explicación de cada parte:

- **Importaciones:** Se importan varias clases y funciones necesarias para implementar el Guard, incluyendo el decorador @Injectable, la interfaz CanActivate de NestJS y varias funciones de RxJS.
- **Decorador @Injectable:** Indica que la clase AuthGuard puede ser administrada por el sistema de inyección de dependencias de NestJS, lo que significa que se puede inyectar en otros proveedores.
- **Constructor:** Dentro del constructor, se inyecta una instancia de la clase Auth, que se utilizará para validar el token de autorización.
- **Método canActivate:** Este es el método principal que se debe implementar cuando se crea un Guard en NestJS. Se invoca antes de que se ejecute el manejador de ruta. Recibe un context de tipo ExecutionContext, que es un objeto que encapsula los detalles de la ejecución en curso. El método puede devolver un booleano, una promesa que se resuelva a un booleano, o un observable que emita un booleano. Si el método devuelve false, se rechaza la solicitud.

En el cuerpo del método, se extrae la solicitud HTTP del context y se obtiene el encabezado de autorización. Si no se encuentra el encabezado de autorización, se lanza una excepción UnauthorizedException, lo que resultará en una respuesta HTTP 401 Unauthorized.

Si se encuentra el encabezado de autorización, se utiliza el servicio auth para validar el token. Esto devuelve una promesa, así que se convierte en un observable utilizando la función from de RxJS.

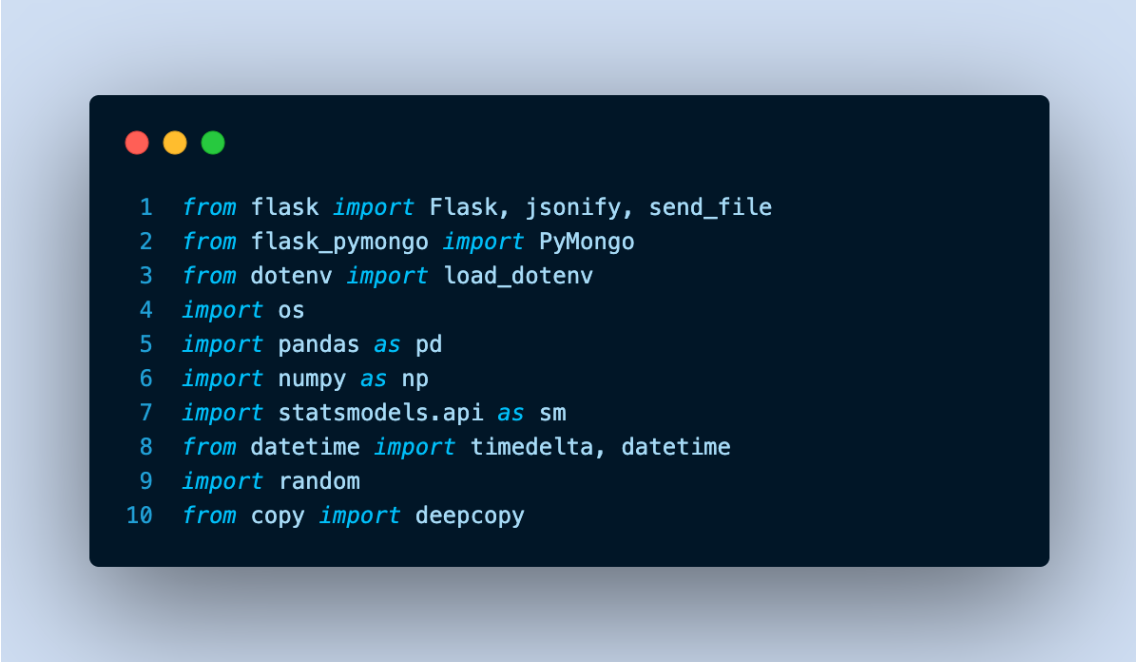
Después, se utiliza el operador map de RxJS para procesar el resultado de la validación del token. Si el usuario es null o undefined, se lanza una excepción UnauthorizedException. De lo contrario, se extraen los datos necesarios del usuario y se agregan al objeto de solicitud antes de devolver true.

En resumen, este código implementa un Guard que valida el token de autorización en el encabezado de las solicitudes HTTP y agrega los datos del usuario a la solicitud si el

token es válido. Es una forma común de manejar la autorización en las aplicaciones de NestJS.

Figura 20.

libPhyton



```python
from flask import Flask, jsonify, send_file
from flask_pymongo import PyMongo
from dotenv import load_dotenv
import os
import pandas as pd
import numpy as np
import statsmodels.api as sm
from datetime import timedelta, datetime
import random
from copy import deepcopy
```

Este código es la sección de importaciones y configuración inicial de un script de Python que utiliza el framework Flask para construir una API web, junto con otras bibliotecas para varias funciones:

- **From flask import Flask, jsonify, send_file:** Esto importa Flask, un marco de microservicios de Python para construir aplicaciones web. jsonify es una función que se usa para convertir datos de Python en un formato JSON que puede ser devuelto por la API. send_file es una función que permite a la API enviar archivos como respuesta.
- **From flask_pymongo import PyMongo:** Importa PyMongo, que es una biblioteca de Python para interactuar con MongoDB. Flask-PyMongo es una extensión de Flask que proporciona una capa de abstracción sobre PyMongo, haciendo que sea más fácil trabajar con MongoDB en una aplicación Flask.
- **From dotenv import load_dotenv:** Importa load_dotenv de la biblioteca python-dotenv. Esto se utiliza para cargar variables de entorno desde un archivo .env, que puede ser útil para mantener seguros ciertos datos, como las claves de la API o las contraseñas de la base de datos.

- **Import os:** Importa el módulo os, que proporciona funciones para interactuar con el sistema operativo, incluyendo la lectura de variables de entorno.
- **Import pandas as pd y import numpy as np:** Importa las bibliotecas pandas y numpy, que proporcionan estructuras de datos y funciones para el análisis de datos.
- **Import statsmodels.api as sm:** Importa statsmodels, una biblioteca de Python para la estimación de modelos estadísticos y la realización de pruebas estadísticas.
- **From datetime import timedelta, datetime:** Importa las clases timedelta y datetime del módulo datetime. Estas se utilizan para trabajar con fechas y horas.
- **Import random:** Importa el módulo random, que proporciona funciones para generar números aleatorios.
- **From copy import deepcopy:** Importa la función deepcopy del módulo copy, que se utiliza para hacer copias completas de objetos complejos, como listas o diccionarios.
- **Load_dotenv():** Llama a la función load_dotenv, que carga las variables de entorno desde un archivo .env. Estas variables de entorno luego se pueden acceder en el código usando os.getenv.

Esencialmente, este script prepara el entorno y las bibliotecas necesarias para construir una aplicación web Flask que interactúa con una base de datos MongoDB y realiza ciertas operaciones de análisis de datos.

Figura 21.

Makefile

```
.PHONY
: update_spa_principal update_gateway update_mc_auth update_mc_file_update update
_infrastructure update_mc_machine_learning

HOME_DIR := $(HOME)

update_spa_principal:
        cd $(HOME_DIR)/projects/prediction_spa_principal && \
        git pull origin main && \
        cd $(HOME_DIR)/projects/prediction_infrastructure && \
        docker-compose build --no-cache spa-principal && \
        docker-compose up -d --force-recreate --no-deps spa-principal

update_gateway:
        cd $(HOME_DIR)/projects/prediction_api_gateway && \
        git pull origin main && \
        cd $(HOME_DIR)/projects/prediction_infrastructure && \
        docker-compose build --no-cache gateway && \
        docker-compose up -d --force-recreate --no-deps gateway

update_mc_auth:
        cd $(HOME_DIR)/projects/prediction_mc_auth && \
        git pull origin main && \
        cd $(HOME_DIR)/projects/prediction_infrastructure && \
        docker-compose build --no-cache mc-auth && \
        docker-compose up -d --force-recreate --no-deps mc-auth

update_mc_file_update:
        cd $(HOME_DIR)/projects/prediction_mc_file_update && \
        git pull origin main && \
        cd $(HOME_DIR)/projects/prediction_infrastructure && \
        docker-compose build --no-cache mc-file-update && \
        docker-compose up -d --force-recreate --no-deps mc-file-update

update_mc_machine_learning:
        cd $(HOME_DIR)/projects/prediction_mc_machine_learning && \
        git pull origin main && \
        cd $(HOME_DIR)/projects/prediction_infrastructure && \
        docker-compose build --no-cache mc-machine-learning && \
        docker-compose up -d --force-recreate --no-deps mc-machine-learning

update_infrastructure:
        cd $(HOME_DIR)/projects/prediction_infrastructure && \
        git pull origin main
```

Este código es un Makefile. Un Makefile es un archivo utilizado por la herramienta make, una herramienta de construcción automatizada utilizada para compilar y construir proyectos. Un Makefile contiene un conjunto de directivas que make utiliza para construir el proyecto.

Un target es el nombre de la regla. Los prerrequisitos son archivos o targets de los cuales depende el target actual. La recipe es una serie de comandos que make ejecutará para construir el target.

En tu caso, tienes varias reglas en tu Makefile, cada una de las cuales se utiliza para actualizar diferentes partes de un proyecto de software.

Todas las reglas siguen un patrón similar, cambian al directorio del proyecto, actualizan el código del proyecto con git pull, vuelven al directorio de infraestructura y luego reconstruyen y reinician el servicio correspondiente con docker-compose.

- **Update_spa_principal, update_gateway, update_mc_auth, update_mc_file_update, update_mc_machine_learning:** Estas reglas se utilizan para actualizar diferentes servicios de tu proyecto. Cada uno de estos targets no tiene prerrequisitos y la receta consta de una serie de comandos que se ejecutan para actualizar el servicio correspondiente.
- **Update_infrastructure:** Esta regla se utiliza para actualizar el propio proyecto de infraestructura. Nuevamente, no hay prerrequisitos y la receta consta de un comando para cambiar al directorio del proyecto de infraestructura y otro para actualizar el código del proyecto.

Finalmente, .PHONY es una regla especial en makefiles que se utiliza para decirle a make que los targets listados no representan archivos. Sin .PHONY, make no ejecutaría el target si existiera un archivo con el mismo nombre. En tu caso, estás diciendo que update_spa_principal, update_gateway, update_mc_auth, update_mc_file_update, update_mc_machine_learning y update_infrastructure no representan archivos, por lo que make siempre ejecutará estas reglas cuando se las llame.

Patron strategy

En el código, se está usando el patrón de diseño Strategy para manejar la validación de archivos en tu aplicación. Esta validación es crítica porque se quiere asegurar de que los datos que se están manejando en tu aplicación son correctos y están bien formateados.

Aquí es donde entra en juego la clase SalesStrategy. Esta clase, es una implementación concreta de la interfaz Strategy, se encarga de validar los archivos de ventas. Durante esta validación, SalesStrategy busca cualquier error en los datos y los recopila.

Para hacer esto, SalesStrategy implementa varios métodos definidos en la interfaz Strategy. El método validateFile es donde ocurre la mayor parte de la validación. Después de que se valida un archivo, puedes obtener los errores y los datos validados utilizando los métodos getFieldsErrors, getFileErrors y getValidatedData.

Ahora bien, uno de los beneficios clave del patrón de diseño Strategy es que te permite cambiar fácilmente la estrategia de validación en tiempo de ejecución. Esto es manejado por la clase Context. Se puede cambiar la estrategia que se está utilizando Context en cualquier momento utilizando el método setStrategy. Esto significa que, si se necesita validar un tipo diferente de archivo o usar una lógica de validación diferente, se puede hacer fácilmente creando una nueva clase que implemente Strategy y configurando Context para usar esa nueva clase.

Figura 22.

Patron estrategy contex

```
import { Strategy, StrategyResponse } from '
../strategy.interface';

export class Context {
  private strategy: Strategy;

  constructor(strategy: Strategy) {
    this.strategy = strategy;
  }

  public setStrategy(strategy: Strategy) {
    this.strategy = strategy;
  }

  public async validateFile(
    file: Express.Multer.File,
  ): Promise<StrategyResponse> {
    await this.strategy.validateFile(file);
    return {
      success:
        this.strategy.getFieldsErrors().length === 0 &&
        this.strategy.getFileErrors() === null,
      listObjects: this.strategy.getValidatedData(),
      errorMapping: this.strategy.getFieldsErrors(),
      errorMessage: this.strategy.getFileErrors(),
    };
  }
}
```

1. Context

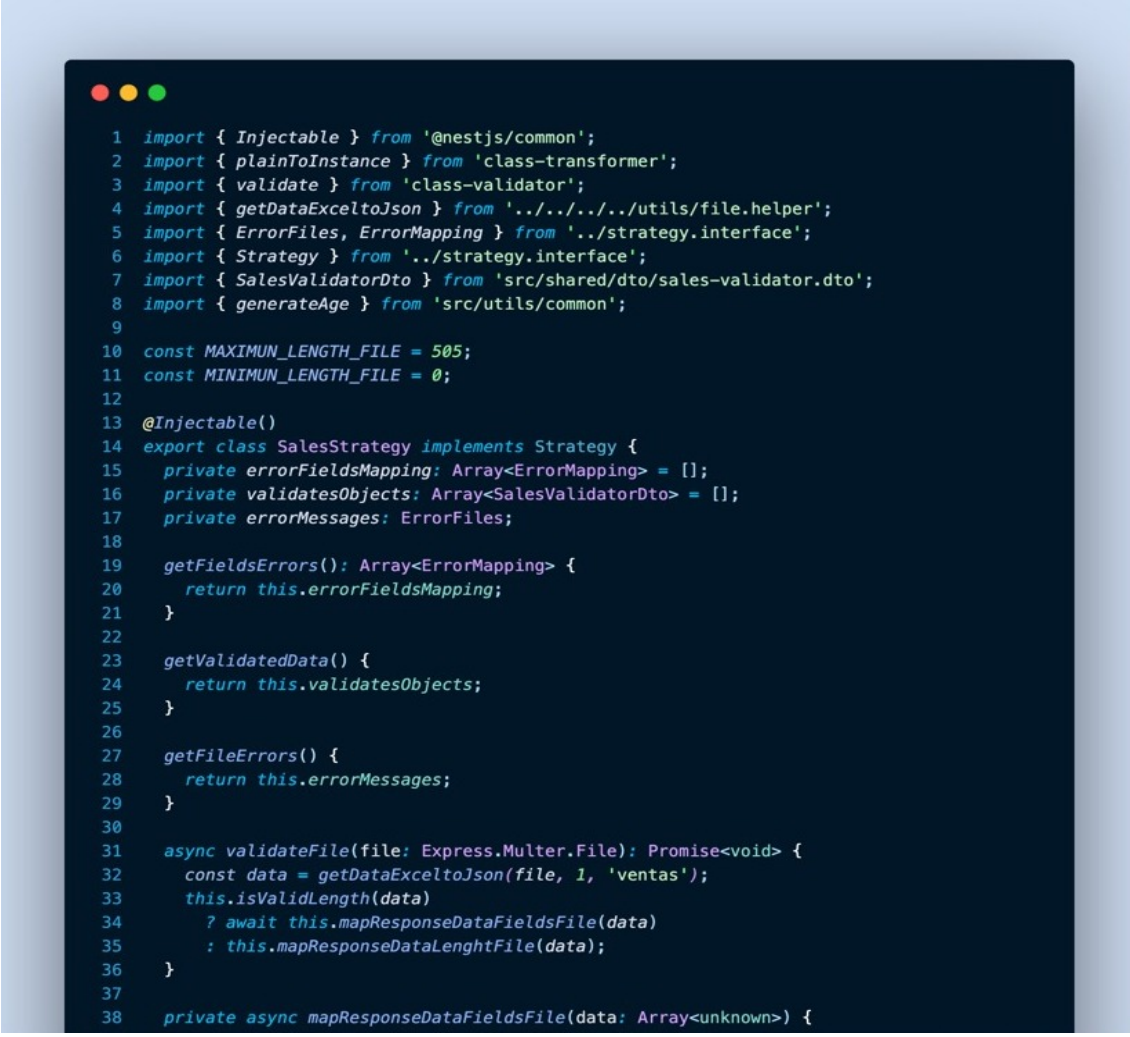
Este código define una clase Context, que es parte integral del patrón de diseño Strategy. Esta clase mantiene una referencia a un objeto de la interfaz Strategy (o una de sus subclases), que define el método validateFile.

La idea aquí es que se puede cambiar la estrategia de validación de archivos en tiempo de ejecución cambiando la referencia de la estrategia en el objeto Context. Esto se logra a través del método setStrategy.

El método validateFile en el Context simplemente delega la validación del archivo a la estrategia actual. Después de la validación, recopila los resultados y los empaqueta en un objeto que luego se devuelve.

Figura 23.

Patron estrategy estrategia



```
import { Injectable } from '@nestjs/common';
import { plainToInstance } from 'class-transformer';
import { validate } from 'class-validator';
import { getDataExceltoJson } from '../../../../utils/file.helper';
import { ErrorFiles, ErrorMapping } from '../strategy.interface';
import { Strategy } from '../strategy.interface';
import { SalesValidatorDto } from 'src/shared/dto/sales-validator.dto';
import { generateAge } from 'src/utils/common';

const MAXIMUN_LENGTH_FILE = 505;
const MINIMUN_LENGTH_FILE = 0;

@Injectable()
export class SalesStrategy implements Strategy {
  private errorFieldsMapping: Array<ErrorMapping> = [];
  private validatesObjects: Array<SalesValidatorDto> = [];
  private errorMessages: ErrorFiles;

  getFieldsErrors(): Array<ErrorMapping> {
    return this.errorFieldsMapping;
  }

  getValidatedData() {
    return this.validatesObjects;
  }

  getFileErrors() {
    return this.errorMessages;
  }

  async validateFile(file: Express.Multer.File): Promise<void> {
    const data = getDataExceltoJson(file, 1, 'ventas');
    this.isValidLength(data)
      ? await this.mapResponseDataFieldsFile(data)
      : this.mapResponseDataLenghtFile(data);
  }

  private async mapResponseDataFieldsFile(data: Array<unknown>) {
```

Figura 24.

Patron estrategy estrategia (Continuación)

```
  private async mapResponseDataFieldsFile(data: Array<unknown>) {
    const dataTransform = this.getFomatedData(data);
    const validations: SalesValidatorDto[] = plainToInstance(
      SalesValidatorDto,
      dataTransform,
      {
        excludeExtraneousValues: false,
      },
    );
    for (const valid of validations) {
      const errors = await validate(valid);
      const errorFlatten = errors.flatMap((el) =>
        Object.values(el.constraints),
      );
      if (errorFlatten.length > 0)
        this.errorFieldsMapping.push({ reason: errorFlatten, item: valid });
    }
    if (this.errorFieldsMapping.length === 0)
      this.validatesObjects = validations;

    this.errorMessages =
      this.errorFieldsMapping.length > 0 || dataTransform.length === 0
        ? {
            title: 'El archivo presenta errores',
            message: 'Corrige el formato',
          }
        : null;
  }

  private getFomatedData(
    dataPayouts: Array<unknown>,
  ): Array<SalesValidatorDto> {
    return dataPayouts.reduce((acc: Array<SalesValidatorDto>, item) => {
      const values = Object.values(item);
      values.shift();
      if (values.length > 0 && values.some((val) => val != null)) {
        acc.push({
          dateBilling: item['A'],
          birthdate: item['B'],
          age: generateAge(new Date(item['B'])),
          gender: item['C'],
          amount: +item['D'],
```

Figura 25.

Patron estrategy estrategia (Continuación)

```
          gender: item['C'],
          amount: +item['D'],
        });
      }
      return acc;
    }, []) as Array<SalesValidatorDto>;
  }

  private isValidLength(data: Array<unknown>): boolean {
    return (
      data.length < MAXIMUN_LENGTH_FILE && data.length > MINIMUN_LENGTH_FILE
    );
  }

  private mapResponseDataLenghtFile(data: Array<unknown>) {
    this.errorMessages = {
      title:
        data.length > MAXIMUN_LENGTH_FILE
          ? 'Se superó el número de registros'
          : 'El archivo no tiene datos',
      message:
        data.length > MAXIMUN_LENGTH_FILE
          ? `Verifica que tu archivo tenga menos de ${MAXIMUN_LENGTH_FILE} registros.`
          : 'Ingresa la información necesaria para crear registros.',
    };
  }
}
```

2. Strategy Interface y SalesStrategy

La interfaz Strategy define los métodos que deben implementar todas las estrategias concretas. En este caso, esos métodos son validateFile, getFieldsErrors, getFileErrors y getValidatedData.

SalesStrategy es una implementación concreta de la interfaz Strategy que se usa para validar archivos de ventas. Implementa el método validateFile para validar un archivo y recopilar datos validados y errores de campo.

Los datos validados y los errores se pueden recuperar utilizando los métodos getFieldsErrors, getFileErrors y getValidatedData.

El patrón Strategy permite que SalesStrategy se use de manera intercambiable con cualquier otra estrategia que implemente la interfaz Strategy, lo que permite cambiar fácilmente la forma en que se validan los archivos.

Figura 26.

Patron estrategy interface

```typescript
import { SalesValidatorDto } from '
src/shared/dto/sales-validator.dto';

export interface Strategy {
  validateFile(file: Express.Multer.File): Promise<void>;
  getFieldsErrors(): ErrorMapping[];
  getFileErrors();
  getValidatedData();
}

export interface StrategyResponse {
  listObjects: Array<SalesValidatorDto>;
  errorMapping?: ErrorMapping[];
  errorMessage?: ErrorFiles;
  success: boolean;
}

export interface ErrorMapping {
  reason: string[];
  item: SalesValidatorDto;
}

export interface ErrorFiles {
  title: string;
  message: string;
}
```

3. StrategyResponse y otras interfaces

Finalmente, las otras interfaces proporcionan tipos para los datos que maneja el patrón Strategy.

StrategyResponse es el tipo del objeto devuelto por validateFile en el Context. Contiene los datos validados, cualquier error de mapeo de campo, cualquier mensaje de error general y un indicador de éxito.

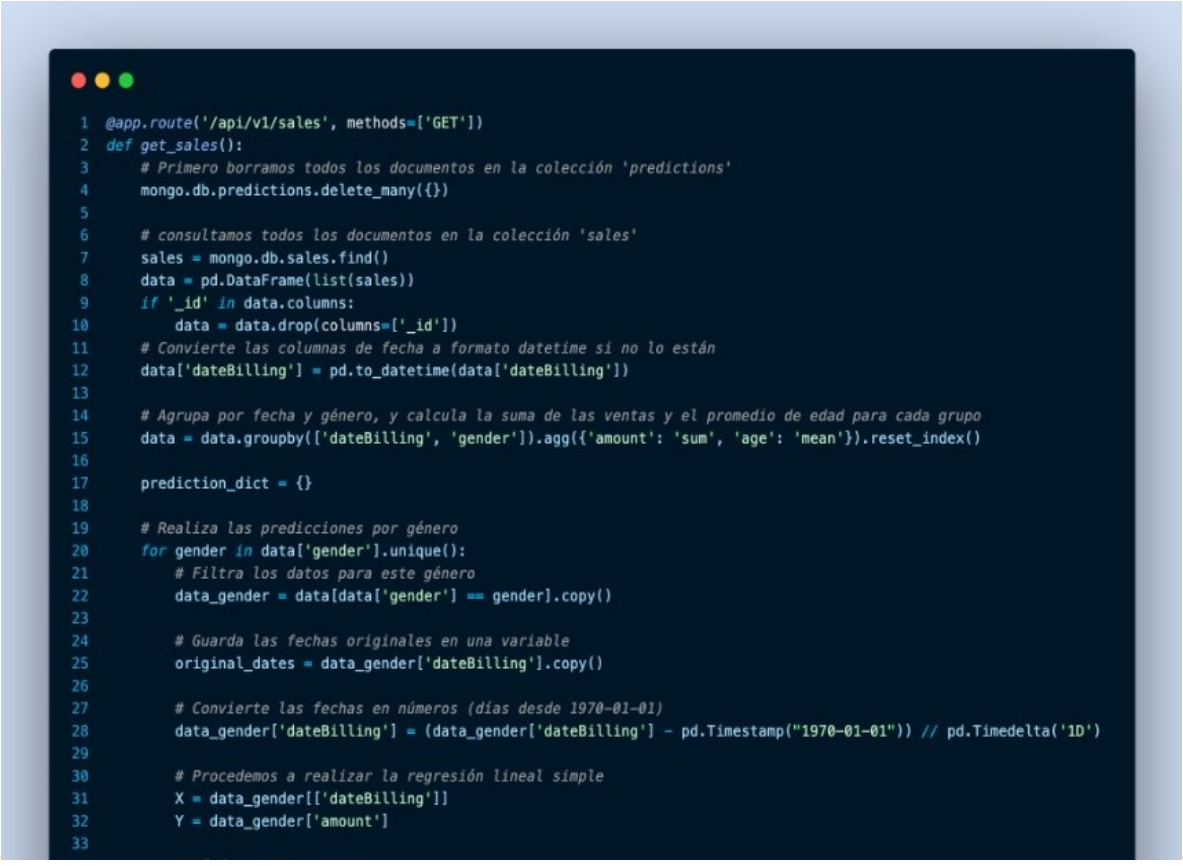
ErrorMapping es el tipo de los errores de mapeo de campos que SalesStrategy recopila durante la validación del archivo.

ErrorFiles es el tipo de los mensajes de error generales que SalesStrategy puede generar durante la validación del archivo.

En resumen, este código implementa el patrón de diseño Strategy para permitir una validación de archivos flexible y extensible. Puedes agregar nuevas estrategias para validar diferentes tipos de archivos simplemente creando nuevas clases que implementen la interfaz Strategy. Estas nuevas estrategias se pueden utilizar en el Context sin cambiar ninguna de su lógica.

Figura 27.

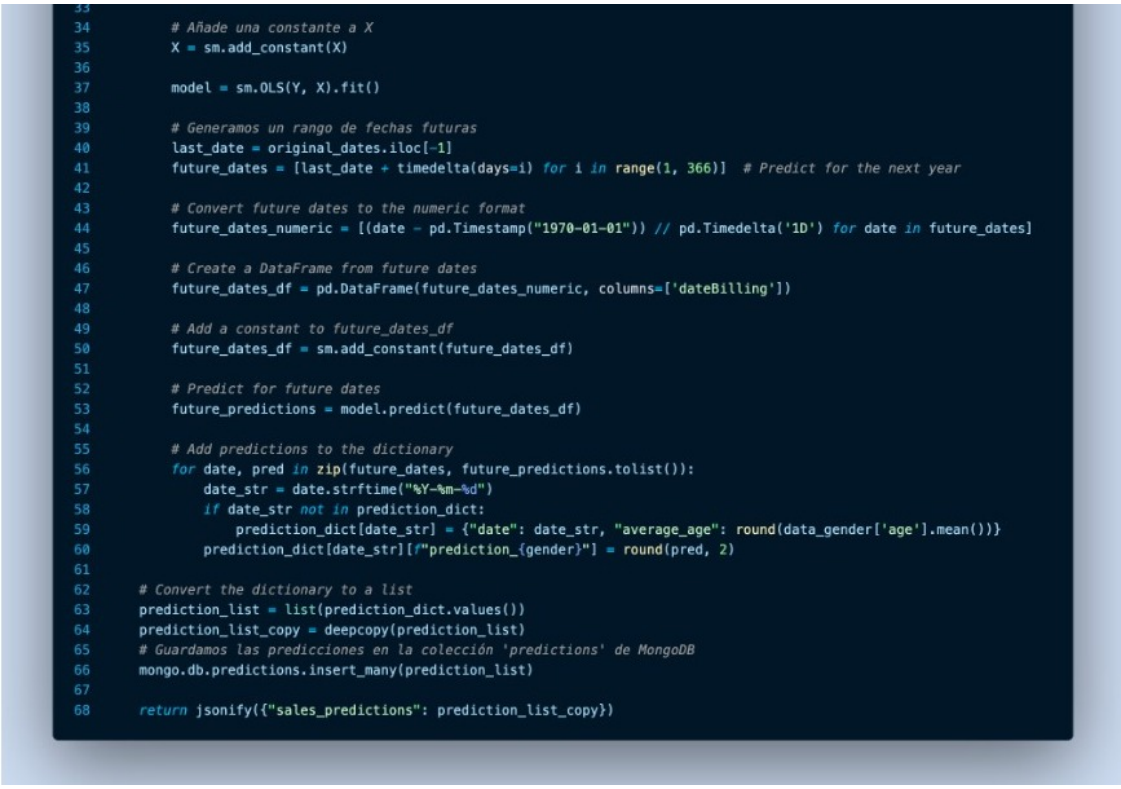
Prediction Phyton



```python
@app.route('/api/v1/sales', methods=['GET'])
def get_sales():
    # Primero borramos todos los documentos en la colección 'predictions'
    mongo.db.predictions.delete_many({})

    # consultamos todos los documentos en la colección 'sales'
    sales = mongo.db.sales.find()
    data = pd.DataFrame(list(sales))
    if '_id' in data.columns:
        data = data.drop(columns=['_id'])
    # Convierte las columnas de fecha a formato datetime si no lo están
    data['dateBilling'] = pd.to_datetime(data['dateBilling'])

    # Agrupa por fecha y género, y calcula la suma de las ventas y el promedio de edad para cada grupo
    data = data.groupby(['dateBilling', 'gender']).agg({'amount': 'sum', 'age': 'mean'}).reset_index()

    prediction_dict = {}

    # Realiza las predicciones por género
    for gender in data['gender'].unique():
        # Filtra los datos para este género
        data_gender = data[data['gender'] == gender].copy()

        # Guarda las fechas originales en una variable
        original_dates = data_gender['dateBilling'].copy()

        # Convierte las fechas en números (días desde 1970-01-01)
        data_gender['dateBilling'] = (data_gender['dateBilling'] - pd.Timestamp("1970-01-01")) // pd.Timedelta('1D')

        # Procedemos a realizar la regresión lineal simple
        X = data_gender[['dateBilling']]
        Y = data_gender['amount']

```

Figura 28.

Prediction Phyton (Continuación)



```
        # Añade una constante a X
        X = sm.add_constant(X)

        model = sm.OLS(Y, X).fit()

        # Generamos un rango de fechas futuras
        last_date = original_dates.iloc[-1]
        future_dates = [last_date + timedelta(days=i) for i in range(1, 366)]  # Predict for the next year

        # Convert future dates to the numeric format
        future_dates_numeric = [(date - pd.Timestamp("1970-01-01")) // pd.Timedelta('1D') for date in future_dates]

        # Create a DataFrame from future dates
        future_dates_df = pd.DataFrame(future_dates_numeric, columns=['dateBilling'])

        # Add a constant to future_dates_df
        future_dates_df = sm.add_constant(future_dates_df)

        # Predict for future dates
        future_predictions = model.predict(future_dates_df)

        # Add predictions to the dictionary
        for date, pred in zip(future_dates, future_predictions.tolist()):
            date_str = date.strftime("%Y-%m-%d")
            if date_str not in prediction_dict:
                prediction_dict[date_str] = {"date": date_str, "average_age": round(data_gender['age'].mean())}
            prediction_dict[date_str][f"prediction_{gender}"] = round(pred, 2)

    # Convert the dictionary to a list
    prediction_list = list(prediction_dict.values())
    prediction_list_copy = deepcopy(prediction_list)
    # Guardamos las predicciones en la colección 'predictions' de MongoDB
    mongo.db.predictions.insert_many(prediction_list)

    return jsonify({"sales_predictions": prediction_list_copy})
```

Este es un código de Python para una API usando Flask y que realiza predicciones de ventas utilizando la regresión lineal simple y guarda los resultados en MongoDB. Aquí está lo que hace cada parte del código:

@app.route('/api/v1/sales', methods=['GET']) - Define un punto de entrada de la API que se activa con solicitudes GET a '/api/v1/sales'.

mongo.db.predictions.delete_many({}) - Borra todos los documentos en la colección 'predictions' en la base de datos MongoDB.

sales = mongo.db.sales.find() - Consulta todos los documentos en la colección 'sales' en la base de datos MongoDB y los carga en un DataFrame de pandas.

data.groupby(['dateBilling', 'gender']).agg({'amount': 'sum', 'age': 'mean'}) - Agrupa los datos por fecha y género, calculando la suma de las ventas y el promedio de edad para cada grupo.

Luego, el código realiza un ciclo para cada género único en los datos y realiza las siguientes operaciones para cada género:

- Filtra los datos para este género.

- Convierte las fechas en números (días desde 1970-01-01).
- Realiza una regresión lineal simple en las ventas por fecha.
- Genera un rango de fechas futuras.
- Realiza predicciones para estas fechas futuras.
- Agrega las predicciones al diccionario de predicciones.

Mongo.db.predictions.insert_many(prediction_list) - Guarda la lista de predicciones en la colección 'predictions' en la base de datos MongoDB.

Return jsonify({"sales_predictions": prediction_list_copy}) - Devuelve un objeto JSON que contiene las predicciones de ventas.

Por lo tanto, esta API permite realizar predicciones de ventas para el próximo año por género, basándose en la información de ventas y edades existentes. Luego guarda las predicciones en la base de datos MongoDB para su posterior uso. Al llamar a la API, se devuelve un objeto JSON con las predicciones de ventas

Figura 29.

Validación Auth

```
import { Injectable, UnauthorizedException } from '@nestjs/common';
import { PassportStrategy } from '@nestjs/passport';
import { ExtractJwt, Strategy } from 'passport-jwt';
import * as jwksRsa from 'jwks-rsa';
@Injectable()
export class Auth0Strategy extends PassportStrategy(Strategy) {
  constructor() {
    super({
      jwtFromRequest: ExtractJwt.fromAuthHeaderAsBearerToken(),
      ignoreExpiration: false,
      secretOrKeyProvider: jwksRsa.passportJwtSecret({
        cache: true,
        rateLimit: true,
        jwksRequestsPerMinute: 5,
        jwksUri: `${process.env.AUTH0_ISSUER_URL}.well-known/jwks.json`,
      }),
      algorithms: ['RS256'],
      audience: `${process.env.AUTH0_AUDIENCE}`,
      issuer: `${process.env.AUTH0_ISSUER_URL}`,
    });
  }

  validate(payload: any) {
    if (!payload) {
      throw new UnauthorizedException();
    }
    return payload;
  }
}
```

Este fragmento de código define una estrategia de autenticación utilizando JSON Web Tokens (JWT) con Auth0, dentro del marco de NestJS.

Aquí se tiene una explicación de cada parte:

- **Importación de Dependencias:** Se importan varias clases y funciones necesarias para la autenticación, incluyendo la estrategia de Passport para JWT y una función para extraer la clave pública de Auth0.
- **Decorador @Injectable():** Indica que la clase Auth0Strategy puede ser administrada por el sistema de inyección de dependencias de NestJS. Esto significa que se puede inyectar en otros proveedores.
- **Herencia de PassportStrategy(Strategy):** Auth0Strategy hereda de PassportStrategy, lo cual significa que implementa una estrategia de autenticación utilizando el módulo de Passport. Passport es un middleware popular para manejar la autenticación en aplicaciones de Node.js.
- **Constructor:** Dentro del constructor, se llama al constructor de la clase base con una configuración específica:
- **JwtFromRequest:** Define cómo se extrae el JWT de la solicitud. En este caso, se espera que el token se encuentre en la cabecera de autorización como un Bearer Token.
- **IgnoreExpiration:** Si está configurado en false, se verificará la fecha de expiración del JWT.
- **SecretOrKeyProvider:** Utiliza jwksRsa.passportJwtSecret para manejar la descarga de la clave de autenticación desde Auth0.
- **Algorithms:** Define el algoritmo utilizado para verificar el JWT (en este caso, RS256).
- **Audience e issuer:** Estos son utilizados para validar la audiencia y el emisor del token, respectivamente, que deben coincidir con los valores proporcionados en las variables de entorno.
- **Método validate:** Este método se invoca después de que el token JWT ha sido decodificado y verificado. Si el payload del token está presente, el método devuelve el payload, lo cual puede incluir información sobre el usuario autenticado. Si el payload no está presente, se lanza una excepción

UnauthorizedException, lo cual puede desencadenar un error 401 Unauthorized en respuesta a la solicitud.

En resumen, este código implementa una estrategia de autenticación utilizando JWT y Auth0 dentro de una aplicación NestJS. Permite a la aplicación aceptar tokens JWT, verificar su validez y autenticar a los usuarios en función de estos tokens.

Figura 30.

Dashboard del front

Este es un componente de Angular que usa el SDK de MongoDB Charts para incrustar gráficos en una aplicación web. Aquí está la explicación del código:

La declaración import ChartsEmbedSDK, { Chart } from '@mongodb-js/charts-embed-dom'; es para importar el SDK de MongoDB Charts en el componente.

En la declaración de la clase TabDashboardPage, se inicializa una nueva instancia del SDK en la línea:

- typescript
- Copy code
- sdk: ChartsEmbedSDK = new ChartsEmbedSDK({
- baseUrl: environment.chartsClientSDK,});

Aquí environment.chartsClientSDK es probablemente la URL de tu instancia de MongoDB Charts.

Luego, en la función ngOnInit(), este componente obtiene una lista de los gráficos para mostrar utilizando el servicio chartsService. Para cada gráfico, se llama a la función createChart() y luego se renderiza el gráfico en el elemento DOM correspondiente.

- **La función createChart(chartId:** string) utiliza el SDK para crear un gráfico dado su ID.

- **La función clickChart(payload:** any, key: any) es un controlador de eventos que se activa cuando se hace clic en un gráfico. Establece un filtro en el gráfico basado en la selección actual del usuario.

Finalmente, getDashboardService() es una función que recupera la información del gráfico desde un servicio remoto, probablemente un backend que se comunica con MongoDB.

Es importante mencionar que el uso del método setFilter permite que el componente de gráfico refleje solo un subconjunto de los datos disponibles, lo que es útil para enfocar el gráfico en aspectos específicos de los datos.

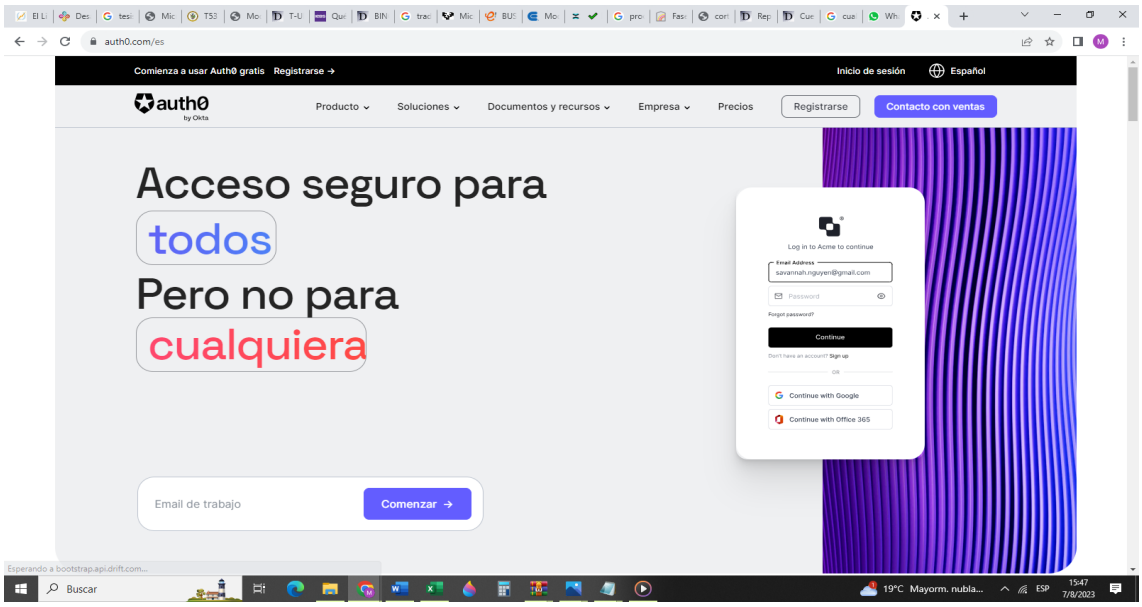
En resumen, este componente se encarga de recuperar los gráficos desde un backend, renderizarlos en la interfaz de usuario, y actualizar el gráfico basado en la interacción del usuario.

3.6.1 Ingreso Auth0

Descripción para el ingreso a Aut0 se procede a ingresar al link https://auth0.com/, con el inicio se procede a registrar el correo el cual se solicita crear una contraseña.

Figura 31.

Ingreso Auth0



Después se procede a colocar en aplicaciones, y elegir SoftPrediction, en el cual se ingresan los datos solicitados, al mismo tiempo se elige si se desea compartir la información en redes sociales los cuales cuentan con una opción para hacerlo, es importarte tomar en consideración si se hace o no publica la información.

Después de todos los datos colocados se realiza la creación del nombre del usuario, contraseña y autenticación. A continuación, se procede a ingresar al link https://prediction.kimsabot.com/, para proseguir con el desarrollo.

Con la creación del usuario se procede a ingresar a Prediction. Al momento del ingreso se llenan los datos del usuario y contraseña el cual permite la realización de la predicción con la subida de los datos adecuadamente.

Figura 32.

Ingreso a prediction

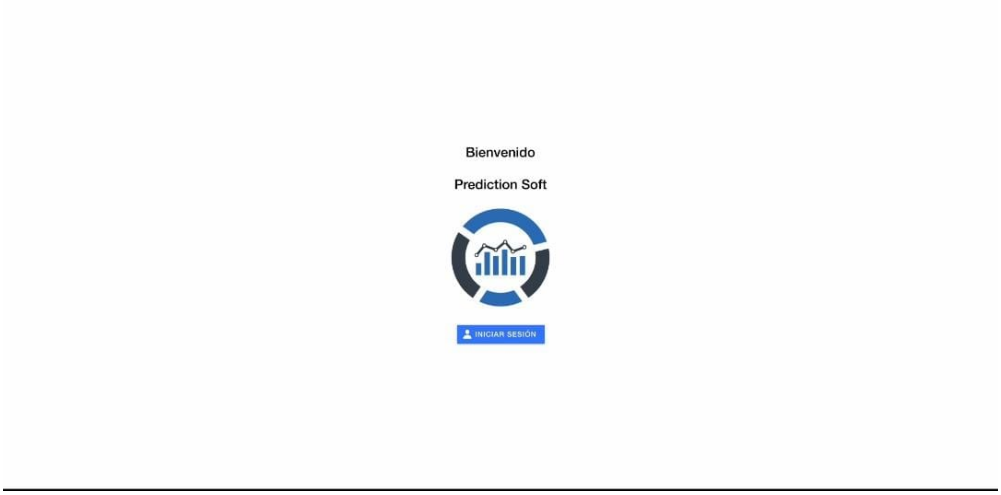
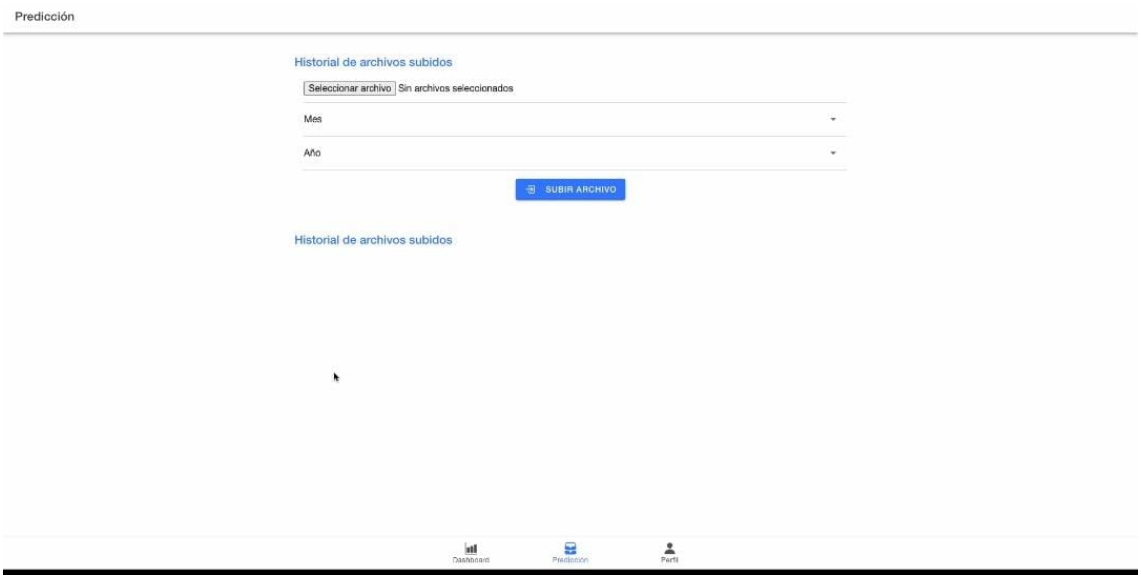


Figura 33.

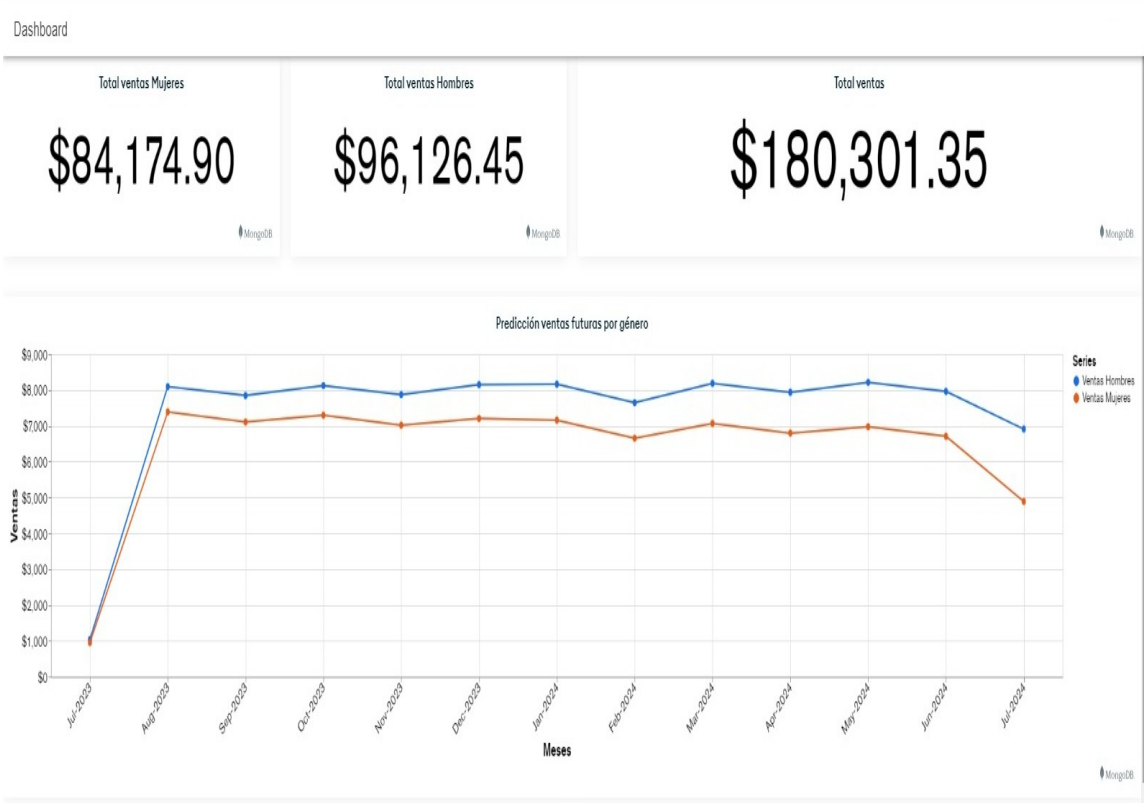
Historial de archivos subidos a la nube



Para proceder a subir los archivos a la nube deben estar correctamente revidado para que no exista error al momento de revisar los datos en el sistema y mas aun pronosticar las ventas.

Figura 34.

Datos encontrados de la predicción total ventas hombres y mujeres.



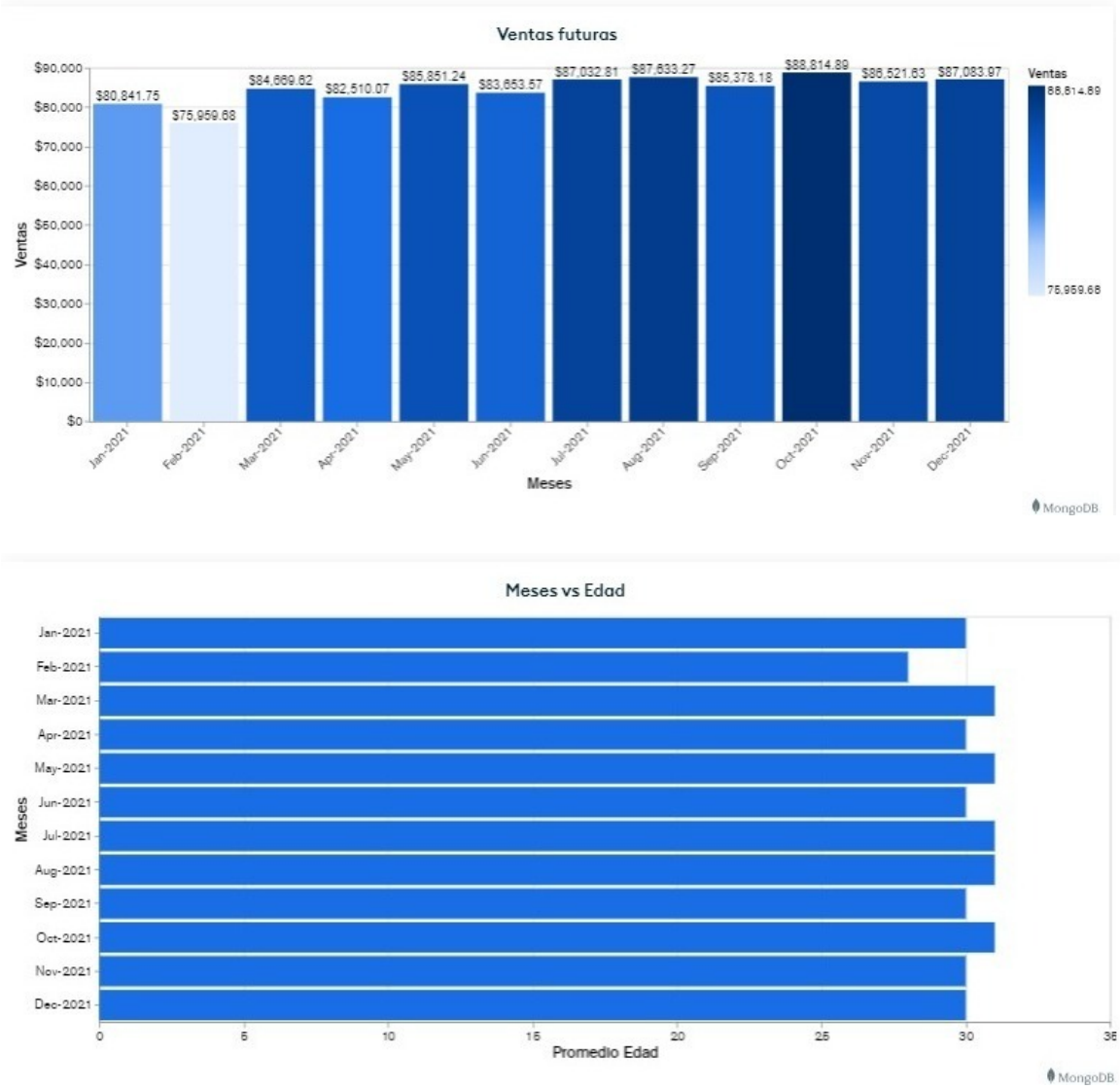
Después del ingreso de los datos de las ventas del año en curso se procede a verificar los datos pronosticados en los años posteriores para conocer si el negocio va desarrollándose adecuadamente en sus ventas.

En este programa permite detallar las ventas clasificadas por hombre y mujeres, al igual que por meses o las fechas que se desee verificar el total de ventas realizadas.

Por todo lo antes descrito se puede decir que con la utilización de prediction soft con los datos existentes en la fuente central se analiza buscando patrones únicos, el cual con la utilización de algoritmos se pueden predecir los resultados de una o más variables de interés estratégico en este caso de las ventas que se desean conocer.

Figura 35.

Ventas futuras representación gráfica en barras



Este sistema ayuda a incrementar la productividad de las empresas para pronosticar los procesos de ventas, con estas herramientas les permiten una exactitud al momento de trabajar con una gran cantidad de datos dando a conocer la representación grafica de los datos que se encuentran en el sistema , se puede interpretar de diferentes maneras como por edades, por meses y por genero.

4. Resultados y discusión

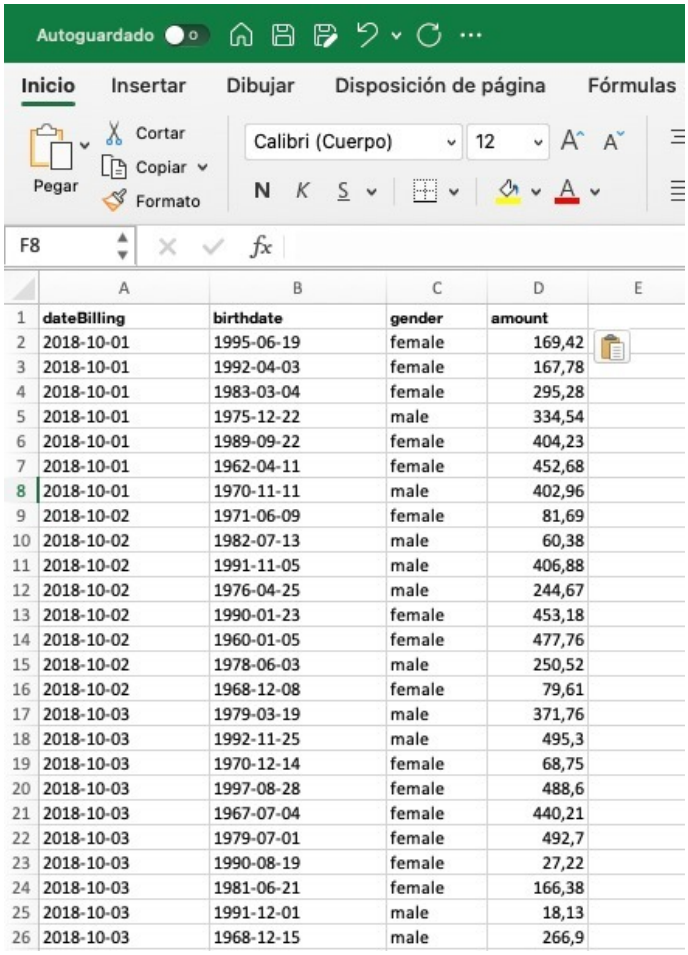
4.1 Resultados

Los resultados que se obtuvieron en la elaboración de la presente investigación son:

- Primer paso, es verificar que los datos del Excel se encuentren correctamente colocados sin existir error alguno, estos datos serán los recopilados durante todo el año los cuales se encuentran diferenciados entre las ventas, género, día, mes y año los cuales se identificaran uno a uno en la hoja de cálculo de Excel, esto permite que al momento de ingresar y cargar la hoja de Excel el programa vaya teniendo sentido de acuerdo a lo que se necesite conocer.

Figura 36.

Datos en Excel



	A	B	C	D	E
1	dateBilling	birthdate	gender	amount	
2	2018-10-01	1995-06-19	female	169,42	
3	2018-10-01	1992-04-03	female	167,78	
4	2018-10-01	1983-03-04	female	295,28	
5	2018-10-01	1975-12-22	male	334,54	
6	2018-10-01	1989-09-22	female	404,23	
7	2018-10-01	1962-04-11	female	452,68	
8	2018-10-01	1970-11-11	male	402,96	
9	2018-10-02	1971-06-09	female	81,69	
10	2018-10-02	1982-07-13	male	60,38	
11	2018-10-02	1991-11-05	male	406,88	
12	2018-10-02	1976-04-25	male	244,67	
13	2018-10-02	1990-01-23	female	453,18	
14	2018-10-02	1960-01-05	female	477,76	
15	2018-10-02	1978-06-03	male	250,52	
16	2018-10-02	1968-12-08	female	79,61	
17	2018-10-03	1979-03-19	male	371,76	
18	2018-10-03	1992-11-25	male	495,3	
19	2018-10-03	1970-12-14	female	68,75	
20	2018-10-03	1997-08-28	female	488,6	
21	2018-10-03	1967-07-04	female	440,21	
22	2018-10-03	1979-07-01	female	492,7	
23	2018-10-03	1990-08-19	female	27,22	
24	2018-10-03	1981-06-21	female	166,38	
25	2018-10-03	1991-12-01	male	18,13	
26	2018-10-03	1968-12-15	male	266,9	

- Segundo paso, cargar el documento dando a conocer el mes al cual corresponde, al momento que se identifica el archivo el cual va a ser cargado al programa se debe tener en consideración el mes y año del cual pertenece el archivo, es fundamental colocar estos datos ya que son esenciales para el programa y para la predicción a futuro.

Figura 37.

Identificación del mes de los datos

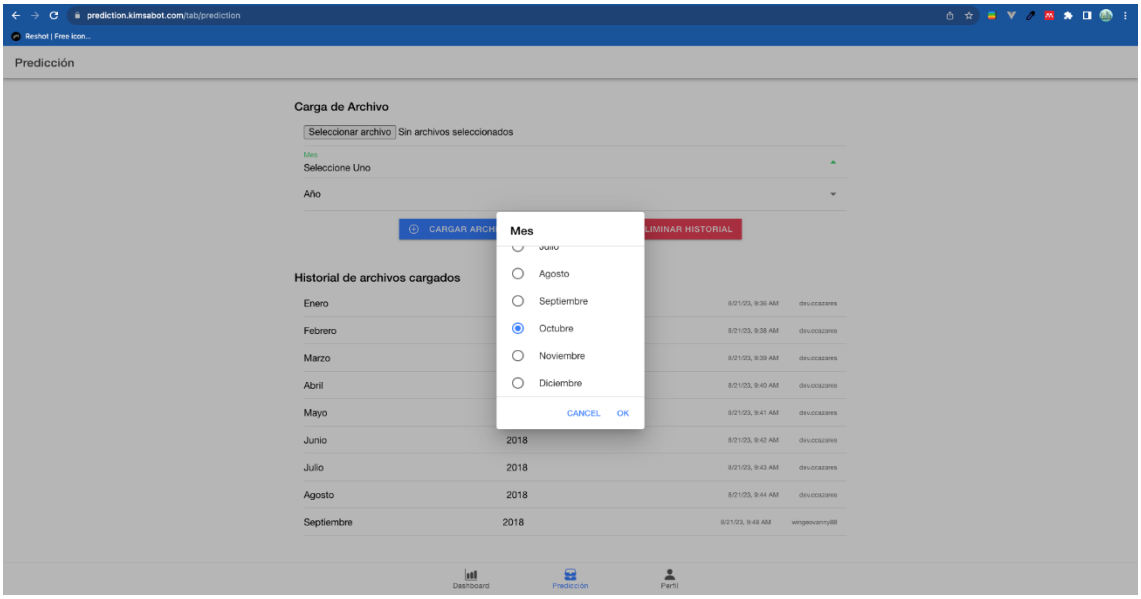
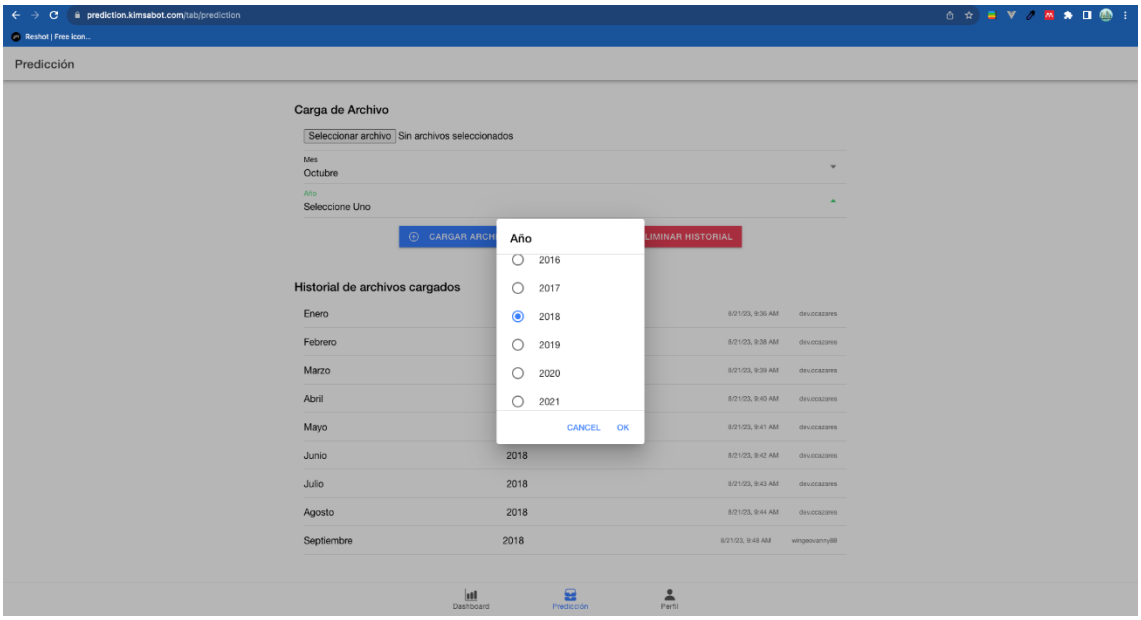


Figura 38.

Tercer paso colocar el año del cual pertenecen los datos.



- Cuarto paso, seleccionar el archivo a ser cargado al sistema. Primero se debe seleccionar la cuenta que se va a importar, después se procede a seleccionar el archivo que se va a cargas, se deben revisar correctamente los encabezados de cada una de las columnas y se hace clic en Importar, revisando la lista de cambios importados, se hace clic y se revisa los cambios importados.

Figura 39.

Importe de archivo

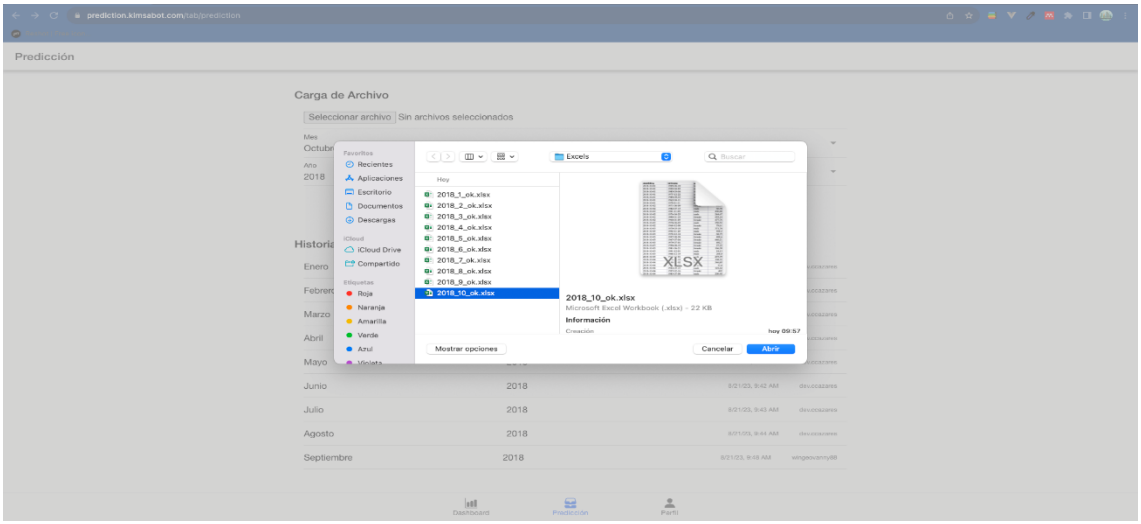


Figura 40.

Quinto paso, cargar el archivo esto permite

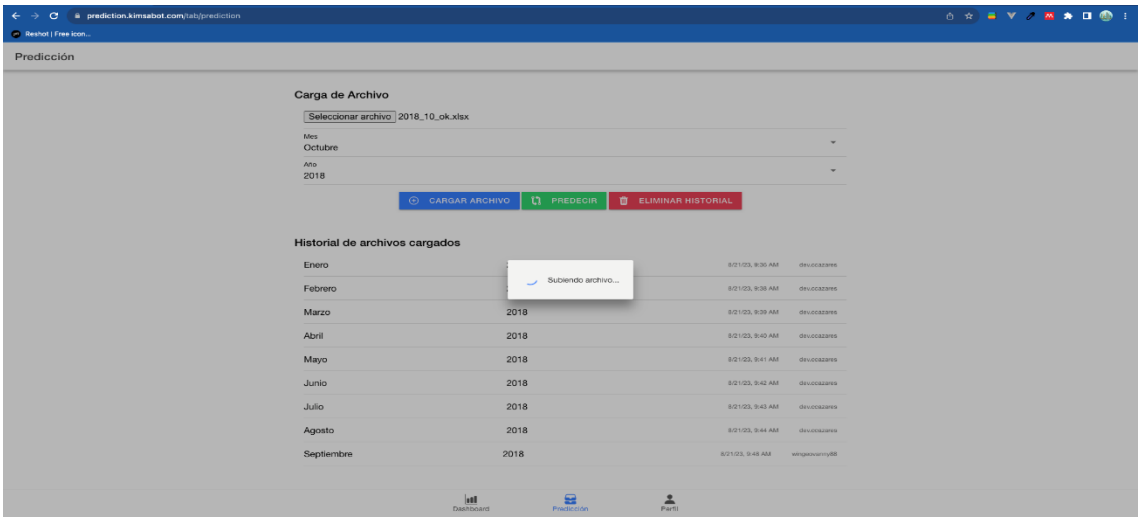
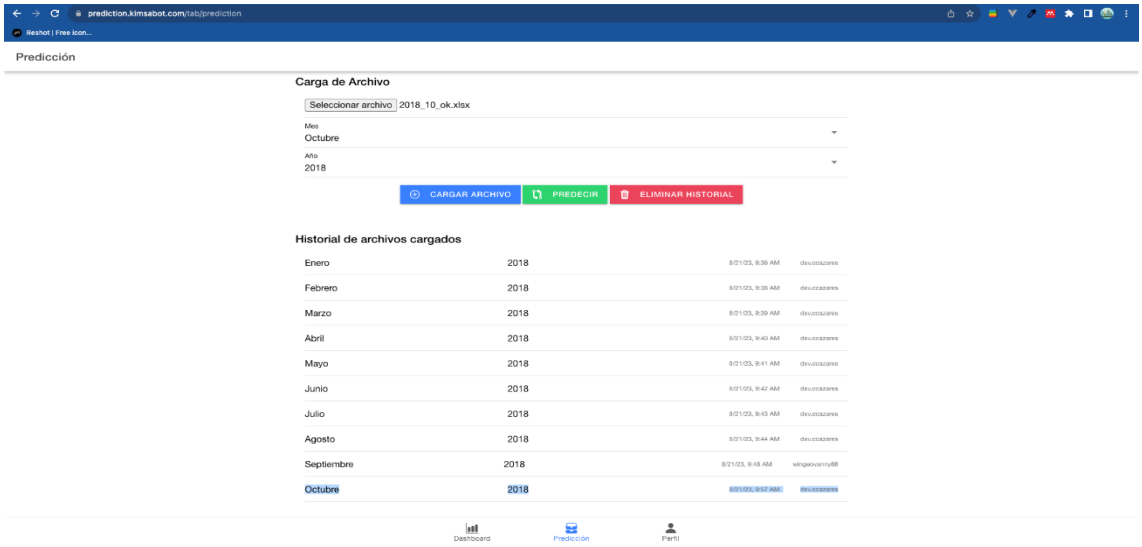


Figura 41.

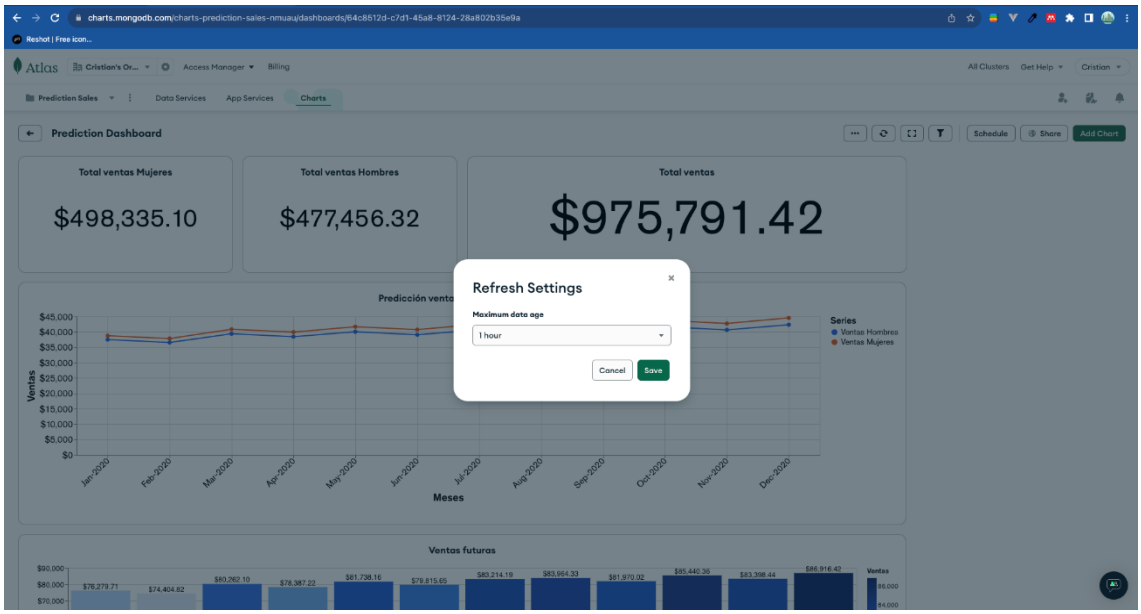
Sexto paso, verificación del documento cargado.



- Séptimo paso, confirmación en mongo atlas. Se ingresa al programa con el ingreso del usuario y contraseña correspondiente, esto permitirá a verificar que el archivo se haya cargado adecuadamente, constando cada uno de los datos directos en el programa.

Figura 42.

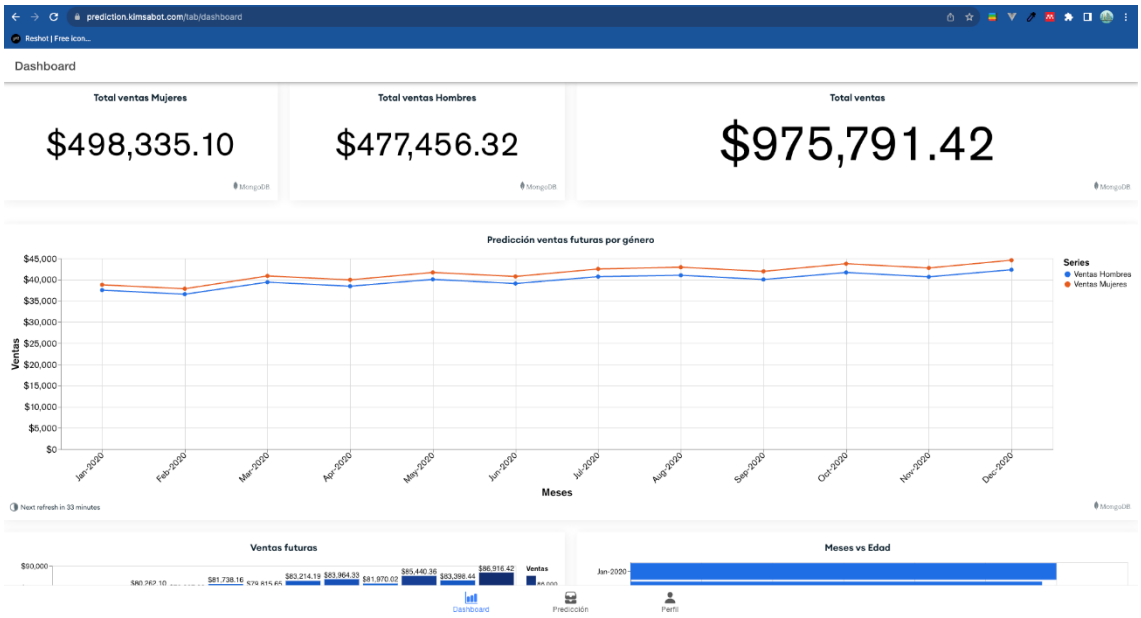
Confirmación MongoAtlas



- Octavo paso, visualización de las estadísticas, con la hoja de cálculo ingresada al sistema se puede verificar mediante un gráfico general las ventas en el cual se identifica en resumen el valor que se ha vendido a mujeres, a hombres y su total y un gráfico en el cual se identifica esas ventas realizadas.

Figura 43.

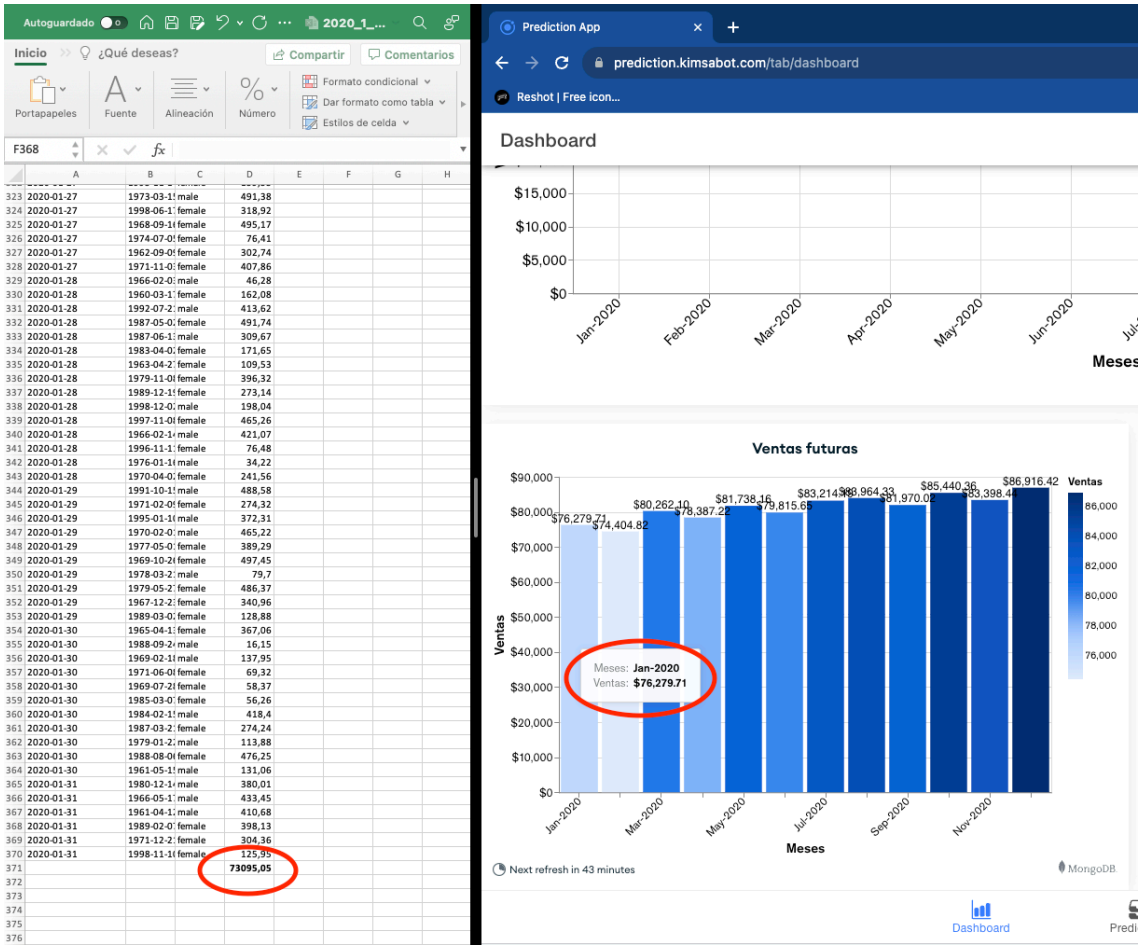
Resumen datos de Excel



- Noveno paso, predicción 2020, con los datos ingresaos del año 2018 y con la aplicación del software para predicciones se procede a predecir las ventas del año 2018 de inicia y cual será su valor de ventas para el año 2020 en el cual se obtuvieron los siguientes resultados los cuales se reflejan en la figura, esto quiere decir que se obtendrán ganancias a futuro.

Figura 44.

Predicción 2020



4.2 Discusión

Después de explorar diferentes enfoques, se decide utilizar Mongo Atlas la decisión correcta para desarrollar este trabajo, ya que el proceso requería investigar las necesidades y comprender el negocio, lo que otras tecnologías no podían proporcionaren la sección 3.4.1., donde se habla sobre el Mongo Atlas explica su funcionalidad y lo relevante que es para su utilización. Aunque métodos como SEMMA pueden simplificar el proceso, la estandarización de pasos que ofrece el método utilizado promueve un desarrollo organizado y eficiente. La principal ventaja, es la capacidad de combinar la parte comercial con la parte técnica, ya que el uso de un plan de trabajo dedicado al procesamiento de datos proporciona un plan muy útil en términos de detección y evaluación de errores. Como es una herramienta relativamente

nueva, se puede encontrar suficiente ayuda en línea, por lo que el aprendizaje fue lento, lo que ralentizó un poco el proyecto.

Afortunadamente, el funcionamiento de este marco se puede realizar satisfactoriamente, resolviendo así el primer problema, es decir, el manejo de volúmenes de datos. Dado que la infraestructura de big data ya está implementada, se pueden realizar todos los procesos descritos en los capítulos anteriores. El centro de datos creado para almacenar información del modelo está ubicado físicamente en el servidor de prueba.

De esta forma se cumplirá uno de los deseos de las PYME que es iniciar la generación de contenedores de información y, por tanto, el desarrollo de análisis. De esta manera, se produce un cambio cultural en el procesamiento de datos, permitiendo una utilidad que difiere del carácter administrativo del pasado; de esta manera se pueden maximizar sus recursos a favor de los beneficios comerciales de esos recursos. La dificultad para subir un archivo de Excel esta si la información se encuentra mal colocada las columnas no concuerdan con los datos establecidos en cada una.

Al momento que se sube un archivo Excel el microservicio que lo valida es el fileUpdate service el mismo debe de contar de cuatro colunas y si ya es validado se guarda en mongo, el cual transforma el Excel en un JSON y lo almacena en Mongo obteniendo los mejores resultados para este modelo, obteniendo la decisión y algoritmos, ya que son los más apropiados para una predicción. Por otro lado, los peores resultados se encontraron para los algoritmos de regresión lineal, que debido a su naturaleza lineal no se adaptan bien al comportamiento oscilatorio recurrente. En el primer enfoque para analizar los resultados, resultó que todos los algoritmos daban resultados muy pobres porque los datos no estaban segmentados por tipo de cliente. Luego de realizar la segmentación se pudo observar que los resultados mejoran paulatinamente a medida que se van ajustando los parámetros.

Finalmente, al especificar parámetros, los regresores de decisión y impulsados por gradiente exceden significativamente los niveles de aceptación preestablecidos, mientras que los bosques aleatorios funcionan inadecuadamente. Es por esa razón que

si el Excel no se encuentra bien elaborado como lo indica en los microservicios puede existir errores. Pero con el trabajo de investigación realizados se pudo verificar las predicciones realizadas como se indican en las imágenes anteriores.

Es importante mencionar que se ha investigado sobre el problema que se enfoca en PYME y en su necesidad de tener una ayuda tecnológica para mejorar sus predicciones de ventas y optimizar su manejo financiero, con el objetivo de minimizar las pérdidas y maximizar las ganancias, y todo esto se cumplió ya que con las predicciones de ventas favorecerá a las empresas a conocer si es pertinente o no seguir con el negocio sea cual sea.

5. Conclusiones

Se pudo verificar la base de datos para obtener el levantamiento de datos, para esto se utilizó un Software adecuado para su implementación, estos archivos en las cuales se encuentran las hojas de Excel con los datos estadísticos de la empresa.

Con la identificación del modelo se realizó el ingreso de los datos para la automatización de las predicciones ventas, evaluando el desempeño del software en la determinación de las ventas, y así, implementar un Software para predecir las ventas de una PYME con la ayuda de las técnicas de aprendizaje automático.

Para finalizar, se manifiesta que, este trabajo ayudará a las PYME primero con herramientas de aprendizaje automático y la gestión de datos, las empresas deberán continuar creando mejores soluciones para la proyección de ventas. Y seguir creciendo como organización, al mismo tiempo, tanto la empresa como las próximas memorias han reforzado la idea de seguir aportando a este informe para mejorar los resultados con nuevas herramientas que puedan surgir en el futuro. Se espera que inspire a la próxima generación de estudiantes en el aprendizaje automático, que tiene un campo enorme por explorar pero que aún no ha recibido toda la atención que merece, ya que cada día aparecen nuevas e interesantes herramientas que pueden ayudar a resolver más problemas identificados.

Mediante la aplicación del software se pudo evaluar el desempeño del sistema a ser utilizado y así se obtuvo el pronostico de las ventas, los cuales dan a conocer cada uno de las especificaciones de las ventas sean estas determinando cuantos hombres y mujeres han realizado las compras en la empresa es así que se reflejara las ventas realizadas tomando en consideración un porcentaje adecuado para el pronóstico.

6. GLOSARIO

Algoritmo: método que describe cómo resolver un problema realizando pasos y especifica el orden en que se realizan estos pasos. Los algoritmos ayudan a los programadores a planificar programas antes de escribirlos en un lenguaje de programación (Java, 2019).

Aplicación: Un programa Java independiente, como cualquier otro programación escrita en los lenguajes altos de los niveles. Las aplicaciones pueden ejecutarse en cualquier computadora con un intérprete de Java (Java, 2019)..

Compilación: el nombre del proceso de traducción del código fuente a código de bytes (Java, 2019).

Framework: Proporciona formas estándar de la creación de la aplicación. Consta de muchas características comunes a las soluciones predefinidas y se utiliza para ayudar a crear aplicaciones de una manera más rápida y estandarizada (Olvera, 2020).

Red: La infraestructura que permite que las computadoras se comuniquen entre sí (Java, 2019).

Servicio en la nube: Es el servicio que se consume por medio del Internet. Esto significa que en realidad no están instalados en su computadora (Olvera, 2020).

Referencias

Álvares, V. (2019). *Manual de pago de nómina*. 1–85. https://dspace.ucuenca.edu.ec/bitstream/123456789/1080/1/tad1078.pdf

Arellano, J. (2020). *Análisis de regresión Múltiple*. http://tecnicasavanzadas.sociales.uba.ar/wp-content/uploads/sites/156/2019/06/Olmo-y-Frías.-Cap.-6.-Regresión-Lineal.pdf

Balderas, R., Chaparro, O., Maldonado, A., & Mart, M. (2020). *Aprendizaje automático*. https://indico.nucleares.unam.mx/event/1594/contribution/84/material/slides/0.pdf

Ben Dalla, L. O. (2020). Lean Software Development Practices and Principles in Terms of Observations and Evolution Methods to increase work environment productivity. *International Journal of Engineering and Modern Technology*, *6*(1). https://doi.org/10.13140/RG.2.2.27514.72648

Calapucha, P., & Tarco, M. (2019). *MongoDB Atlas es una base de datos en la nube completamente administrada que maneja toda la complejidad de implementar, administrar y reparar sus implementaciones en el proveedor de servicios en la nube de su elección (AWS, Azure y GCP). MongoDB Atlas es*. http://190.15.135.60/handle/51000/5569

Cali, C. (2022). *Modelo predictivos de las ventas de productos de primera necesidad en el sector comercial*. file:///C:/Users/USER/Videos/Aprendizaje automático (Machine Learning) .MARCO TEORIOCO.pdf

Chacón, J. (2021). Introducción a Pandas. *Profile*.

Delgado, L., & Díaz, L. (2021). *Modelos de Desarrollo de Software*. *15*(1), 37–51. http://scielo.sld.cu/pdf/rcci/v15n1/2227-1899-rcci-15-01-37.pdf

Developer Resource Center. (2022). ¿Qué es Python? *ORACLE*.

Elfenbaum, M. (2019). *Mchine Learning used in organitation innovation*. 14–24.

Flores, D. (2020). La Logistica como ventaja competitiva en las PYME de servicios: ESTUDIO DE CASO EN ESTACIONAMIENTOS AVINAV. *REVISTA DE LA AGRUPACIÓN JOVEN IBEROAMERICANA*.

Hiff, K. (2021). *Capacitación Data4Now Temas*. https://unstats.un.org/capacity-

development/data-for-now/training-materials/Aprendizaje-Automatico.pdf

IBM. (2023). Regresión lineal múltiple. *Cognos Analytics*.

Java. (2019). Glosario de términos de Programación. *Libro Java*, 1–12.

Manobanda, A. (2019). *Predicción de la demanda de energía eléctrica en la producción de petróleo*. https://bibdigital.epn.edu.ec/bitstream/15000/20936/1/CD 10454.pdf

Masa, J. (2019). *Introducción a los Patrones de Diseño de Software*. https://www.fdi.ucm.es/profesor/jpavon/poo/2.14pdoo.pdf

Montalvo, F. (2020). *Análisis, diseño y desarrollo de un sistema para la gestion del seguimiento de pasantias y practicas*. https://dspace.ups.edu.ec/bitstream/123456789/19483/1/UPS-CT008881.pdf

Moreno, A., Armengol, E., Béjar, J., Belanche, L., Cortés, U., & Gavaldà, R. (2020). *Aprendizaje automático*. https://unstats.un.org/capacity-development/data-for-now/training-materials/Aprendizaje-Automatico.pdf

Navarrete, J. (2019). *Procesos , fortalecimiento de servidores y actualizaciones de seguridad de la empresa International Bussines Machines (IBM) Juan Felipe Navarrete Macias UNIVERSIDAD SANTO TOMAS FACULTAD DE INGENIERÍA ELECTRÓNICA*. https://repository.usta.edu.co/bitstream/handle/11634/19495/2019juannavarrete.pdf?sequence=4&isAllowed=y

Norambuena, J., Badilla, M., & López, Y. (2020). *Modelos predictivos basados en uso de analíticas de aprendizaje en educación superior : una revisión sistemática Modelos preditivos baseados no uso de analítica da aprendizagem no ensino superior : uma revisão sistemática a systematic review*. 1–22. https://doi.org/10.35699/1983-3652.2022.36310

Olmedo, J. (2019). *Introducción al Software*. 19–30. https://www.cerasa.es/media/areces/files/book-attachment-51995.pdf

Olvera, E. (2020). Terminos de programación. *Hireline*.

Paredes, M., & Millanes, J. (2019). *Aplicaciones web*. https://www.sintesis.com/data/indices/9788491714729.pdf

Pavón, J. (2019). *Patrones de diseño orientado a objetos Hacer software no es fácil Diseñar para el cambio Patrones Cómo llegar a ser un maestro de ajedrez*.

https://www.fdi.ucm.es/profesor/jpavon/poo/2.14pdoo.pdf

Quishpe, F. (2019). *Concepto y definición de Software, historia y evolución , características de los Software libre*. https://repositorio.une.edu.pe/bitstream/handle/20.500.14039/4616/Software libre.pdf?sequence=1&isAllowed=y

Racero, A., & Som, A. (2019). *Contenedores: Iniciación a Docker y casos de uso prácticos*. https://osl.ugr.es/wp-content/uploads/2022/05/Contenedores-Iniciación-a-Docker-y-casos-de-uso-prácticos-Mayo2022.pptx.pdf

Ríos, E. (2021). *Desarrollo de un sistema web prototipo para generar y gestionar*. https://bibdigital.epn.edu.ec/bitstream/15000/22859/1/CD 123271.pdf

Salas, E. (2020). *Aplicando seguridad a una API REST con JSON Web Tokens*. https://bibdigital.epn.edu.ec/bitstream/15000/20936/1/CD 10454.pdf

Sampedro, R. (2022). *Automatizar procesos de producción repetitivos en las PYME con robots colaborativos*. http://scielo.sld.cu/scielo.php?pid=S2218-36202022000202102 20&script=sci_arttext&tlng=pt

Sanmartín, A. (2021). *Diseño e implementación de un proxy inverso para una arquitectura de microservicios*. https://riunet.upv.es/handle/10251/174002

Shvets, A. (2019). *Patrones de diseño*. https://refactoring.guru/files/design-patterns-es-demo.pdf