Especificación de Requerimientos de Software

Despliegue web de un Modelo de Machine Learning para ELX

Modelo de Recomendación y Customer Journey

Manizales
2023

Contenido

1. INTRODUCCIÓN

Este documento sirve como medio de comunicación entre las partes: cliente ELX y equipo Desarrollo. En la ERS (Especificación de requerimientos de software) deben recogerse tanto las necesidades de clientes y usuarios (necesidades del negocio) como los requisitos que debe cumplir el sistema software a desarrollar para satisfacer dichas necesidades (requisitos del producto, también conocidos como requisitos de sistema o requisitos software).

La ERS es un documento consensuado entre todas las partes, de tal manera que al superar la aprobación de los requerimientos en este documento, se entiende la primera línea base definida, como insumo de las siguientes etapas del desarrollo del proyecto.

La ERS describe a través de una metodología del desarrollo del software los alcances y procesos asociados a la creación de una aplicación web (front-end) y el despliegue del aplicativo en Google Cloud, para disponibilizar el consumo del modelo de Machine Learning (Backend) desarrollado por el área de Analítica de Desarrollo.

1.1. Beneficios relevantes

El sistema software a partir de ciertas características que describen a un cliente (B2B) de ELX, entrega por la ejecución del Modelo de ML de Recomendación las probabilidades de intención de compra de ese cliente para cada producto y destaca el producto de mayor propensión.

El usuario que hará uso del aplicativo podrá consumir el Modelo de ML, usando el procesamiento en la nube de Google y recibir en tiempo real la respuesta a la solicitud para un cliente o una respuesta con una latencia mayor para la ejecución en lote cuando se ejecuta a través de un archivo con múltiples clientes.

1.2. Objetivos

General:

1.2.1. Diseñar y desarrollar una aplicación web que disponibiliza el Modelo de Recomendación y propensión para cliente ELX

Específicos:

1.2.1.1. Identificar los elementos de comunicación entre el Modelo de ML, el informe Customer Journey y el aplicativo front-end

1.2.1.2. Diseñar la experiencia de usuario y definir los elementos de interacción del usuario en el aplicativo

1.2.1.3. Definir los requerimientos de Nube y desplegar el aplicativo en Google Cloud

1.3. Metas y limitaciones

1.3.1. Meta: Proveer a los usuarios un aplicativo web que les permita ejecutar el modelo y consultar los informes de análisis.

1.3.2. Se contemplan dentro de las limitaciones:

1.3.2.1. Errores de sincronización o procesamiento entre el modelo y la aplicación

1.3.2.2. Re-entrenamientos del Modelo que pueden generar cambios en el sistema

1.3.2.3. El proceso extenso que se debe cumplir para que se pueda desplegar el aplicativo en el entorno ELX.

2. DESCRIPCIÓN GENERAL DE LA APLICACIÓN

2.1 Perspectiva de la Aplicación

Se proyecta implementar una aplicación que permite ejecutar el modelo de ML para generar las estimaciones a partir de una serie de criterios, mostrando el resultado que arroja el modelo con la recomendación para 1 cliente o para muchos. También se proyecta incluir un módulo para presentar un informe resumen del Customer Journey del cliente, del Análisis de las variables relacionadas en el proceso.

2.2 Funciones de la Aplicación

Las funciones que se contemplaron para la aplicación son las siguientes:

A. Análisis de variables del Customer Journey
B. Modelo Recomendación para 1 cliente: Ingresa parámetros y ejecuta el Modelo ML
C. Modelo Recomendación para Múltiples clientes: se sube un archivo, se visualiza un resumen de las variables y se genera un archivo con los campos agregados del resultado del modelo, el cual es un Descargable

Y se contempla un módulo de Validación de acceso para garantizar que el usuario que ejecuta el aplicativo esté autorizado

2.3 Características de los Usuarios

Tipo de usuario	Experiencia	Nivel educativo	conocimientos técnicos
Usuario del Modelo ML	Se requiere conocimiento de los productos ELX y en la interpretación de los Datos	Alfabetización tecnológica básica Conocimiento del negocio ELX	Manejo de aplicativos web

2.4 Restricciones

El desarrollador estará limitado por el desarrollo del backend.
Al igual limitado por los protocolos de seguridad de las dependencias, limitaciones de acceso y políticas propias de Desarrollo y ELX para terminar con el despliegue en estos entornos.

2.5 Suposiciones y Dependencias

Los cambios en el Modelo, condiciones de Nube Google, políticas de acceso y seguridad del aplicativo harán necesario que se hagan adecuaciones en la plataforma o arquitectura del sistema, y se informará al cliente de los cambios en las condiciones y bases disponibles.

3. REQUERIMIENTOS ESPECÍFICOS

3.1 Requerimientos

El sistema software será un aplicativo web que tiene las siguiente los siguientes requerimientos funcionales:
Módulo de Validación de acceso para garantizar que el usuario que ejecuta el aplicativo esté autorizado
Ejecución del ML para un cliente: a partir de ciertas características que describen al cliente el Modelo de ML de Recomendación entrega las probabilidades de intención de compra de ese cliente para cada producto y destaca el producto de mayor propensión.
Módulo de ML para múltiples clientes: a partir de un archivo que se carga al aplicativo, el modelo de ML se ejecuta y genera un resultado para cada cliente contenido en el archivo, muestra un resumen de las variables y permite la descarga del informe con el resultado del modelo.
Módulo para generar el informe de Customer Journey a partir de la selección de una variable

Como requerimiento de calidad, se espera un sistema software: funcional, confiable, eficiente y con usabilidad

Y los siguientes son requerimientos de implementación a considerar para el diseño y desarrollo:
a) Diseño cumple con el manual de marca de ELX,
b) Despliegue debe hacerse en Google cloud,
c) Registro log del procesamiento y resultados del Modelo de ML
d) Migración a Entorno ELX

3.1.1 Product Backlog

El Product Backlog es el listado de todas las tareas técnicas que se planean hacer durante el desarrollo del proyecto para lograr el objetivo en el tiempo estimado. A continuación presento el detalle del PB donde se estima un peso para cada tarea y la cantidad en horas estimada que una tarea requiere para estar finalizada.

ID	Nombre	Estado	Peso	Horas	Sprint	Prioridad	Tipo	Comentarios
1	Desarrollar Product Backlog	Hecho	1	2	0	1	Documento	
2	Cronograma	Hecho	1	2	1	1	Documento	
3	Especificación de Requerimientos	Hecho	4	8	2	1	Documento	
4	Arquitectura de Software	Hecho	2	4	2	1	Diagrama	
5	Historias de Usuario	Hecho	4	8	3	1	Diseño UX	
6	Casos de Uso	Hecho	3	6	3	1	Diseño Funcional	
7	Construcción Mockups	Hecho	5	10	3	1	Diseño Gráfico	
8	Modelo de Datos	Pendiente	3	6	4	1	Diagrama	
9	Diagrama de secuencia UML	Hecho	4	8	4	1	Diagrama	
10	Repositorio Github	En proceso	3	6	5	1	Configuración	
11	Gestión de configuración	En proceso	4	8	5	1	Instalación	
12	Creación y configuración cuenta Devops	En proceso	4	8	5	1	Configuración	
13	HU-ADM_001	Planeado	6	12	5	1	Código	
14	CU-001	Planeado	6	12	5	1	Código	
15	HU-ADM_002	Planeado	6	12	6	1	Código	
16	CU-002	Planeado	6	12	6	1	Código	
17	HU-ADM_003	Planeado	6	12	6	1	Código	
18	CU-003	Planeado	6	12	7	1	Código	
19	HU-ADM_004	Planeado	6	12	7	1	Código	

20	CU-004	Planeado	6	12	7	1	Código	
21	Pruebas	Planeado	3	6	8	1	Documento	
22	Despliegue	Planeado	8	16	8	1	Configuración	
23	Pruebas Usuario	Planteado	3	6	8	1	Documento	

3.1.2 Historias de usuario (Tareas y Subtareas)

Una historia de usuario es una representación de un requisito visto desde: qué necesita el usuario.

Historia de Usuario HU-ADM_001	
Número: 001	Nombre: Autenticación en el sistema
Prioridad en Negocio: Media	Iteración Asignada:(fecha aceptación)
rol: Usuario principal	
funcionalidad: Validar acceso e Ingresar al aplicativo	
Criterio de Aceptación: ● Acceder al aplicativo con autenticación Google ● Campo de correo electrónico ● Botón "Siguiente". ● Campo de contraseña ● Botón "Siguiente" ● Validación Google y validación si el usuario está autorizado en la aplicación	

Historia de Usuario HU-ADM_002	
Número: 002	Nombre: Visualizar el Informe Customer Journey
Prioridad en Negocio: Alta	Iteración Asignada:
rol: Usuario principal	
Funcionalidad: Interactuar y ver el resumen de las variables	
Criterio de Aceptación: ● Contenedor Menú con logo inicial ● Recuadro Informe Customer Journey desplegado ● Campo selección variable 1 por defecto ● Campo selección variable 2 por defecto ● Campo selección variable 3 por defecto ● Visualización de gráficos en área principal	

Historia de Usuario HU-ADM_003	
Número:003	Nombre: Ejecutar el Modelo de ML para un cliente específico
Prioridad en Negocio: Alta	Iteración Asignada:
rol: Usuario principal	
funcionalidad: Visualizar el resultado del modelo y descargar el informe	
Criterio de Aceptación: ● Contenedor Menú con logo inicial ● Recuadro Informe Customer Journey desplegado ● Visualiza recomendaciones y mensajes de validación para cada parámetro en área principal ● Campo selección categoría del producto ● Campo selección Actividad económica ● Campo selección categoría del producto ● Campo selección tamaño empresa ● Campo selección forma legal empresa ● Campo total ingresos operativos ● Campo activos totales ● Campo total patrimonio ● Campo Ganancias después de impuestos ● Campo Número de empleados ● Botón "Ejecutar modelo" ● Visualización de gráficos en área principal ● Boton "Descargar informe"	

Historia de Usuario HU-ADM_004	
Número:004	Nombre: Ejecutar el Modelo de ML para múltiples clientes
Prioridad en Negocio: Alta	Iteración Asignada:
rol: Usuario principal	
funcionalidad: Visualizar un resumen de las variables y descargar el informe	
Criterio de Aceptación: ● Contenedor Menú con logo inicial ● Recuadro Informe Customer Journey desplegado ● Visualiza un ejemplo de archivo y su estructura en el área principal ● Botón "Cargar archivo". ● Ventana de búsqueda y selección de archivo en pc ● Botón "Ejecutar modelo" ● Visualización de resumen de variables	

● Boton "Descargar informe"

3.2 Modelo de Requerimientos

Con el Modelo de requerimientos como mecanismo para entender lo que el cliente requiere, analizar sus necesidades y consensuar la solución propuesta. A continuación se especifica la solución, para certificar la definición y gestionar los requisitos del proyecto.

Usuario	Proceso	Caso de Uso
Principal	Validar acceso	CU-001
	Visualizar Customer Journey	CU-002
	Modelo de un cliente	CU-003
	Modelo múltiples clientes	CU-004

3.2.1 Modelo de Casos de Uso

Un caso de uso es un artefacto que define una secuencia de acciones que da lugar a un resultado de valor observable. Visto desde el cómo el usuario llega a cumplir su objetivo en el aplicativo. Describir el comportamiento del software de todas las maneras que los actores podrían trabajar con el software o el sistema.

El **actor** en el modelo de casos de uso es cualquier entidad que desempeñe un rol en el sistema. Puede ser una persona, una organización o un sistema externo. Para el proyecto en desarrollo solo tenemos un actor que en adelante será definido como usuario.

Caso de Uso: CU-001 Acceder al aplicativo	
Dependencias	Registro de los correos autorizados en Google
Precondición	El usuario ha sido autorizado para tener acceso a la aplicación mediante registro de correo electrónico en Google cloud El usuario accede a la url establecida del aplicativo en la nube
Descripción	El sistema debe comportarse como se indica a continuación:
Paso 1	El sistema solicita que digite el correo electrónico del usuario
Paso 2	El usuario digita el correo y oprime el botón “Siguiente”
Paso 3	El sistema solicita la contraseña
Paso 4	El usuario digita contraseña y oprime el botón “Siguiente”
Paso 5	El sistema realiza validaciones de google, verifica que esté autorizado en usuario para ver el aplicativo y muestra HU-ADM_001

Postcondición	
Excepción	Paso 5: Si no coinciden los valores, Paso 6 el sistema muestra aviso “Acceso denegado”
Comentarios	

Caso de Uso: CU-002 Consultar Customer Journey	
Dependencias	CU-001 Acceder al aplicativo
Precondición	Usuario verificado
Descripción	El sistema debe comportarse como se indica a continuación:
Paso 1	El sistema muestra por defecto el recuadro del Customer Journey con las variables por defecto y la visual del informe con estas variables
Paso 2	El usuario selecciona los datos de las variables deseadas para observar en el informe
Paso 3	El sistema muestra el informe con los nuevos valores de las variables
Postcondición	
Excepción	
Comentarios	

Caso de Uso: CU-003 Ejecutar modelo para 1 cliente	
Dependencias	CU-001 Acceder al aplicativo
Precondición	Usuario verificado
Descripción	El sistema debe comportarse como se indica a continuación:
Paso 1	El usuario selecciona en el menú ésta opción
Paso 2	El sistema extiende el bloque con los campos a diligenciar y muestra en el área principal mensaje de ayuda y validación de cada campo
Paso 3	El usuario selecciona y diligencia los parámetros del cliente a consultar y oprime el botón “Ejecutar modelo”
Paso 4	El sistema Ejecuta el modelo de ML con estos datos, muestra como resultado un informe de Recomendación para este cliente y la opción para descargar el archivo de resultado
Paso 5	El usuario oprime el botón “Descargar archivo”

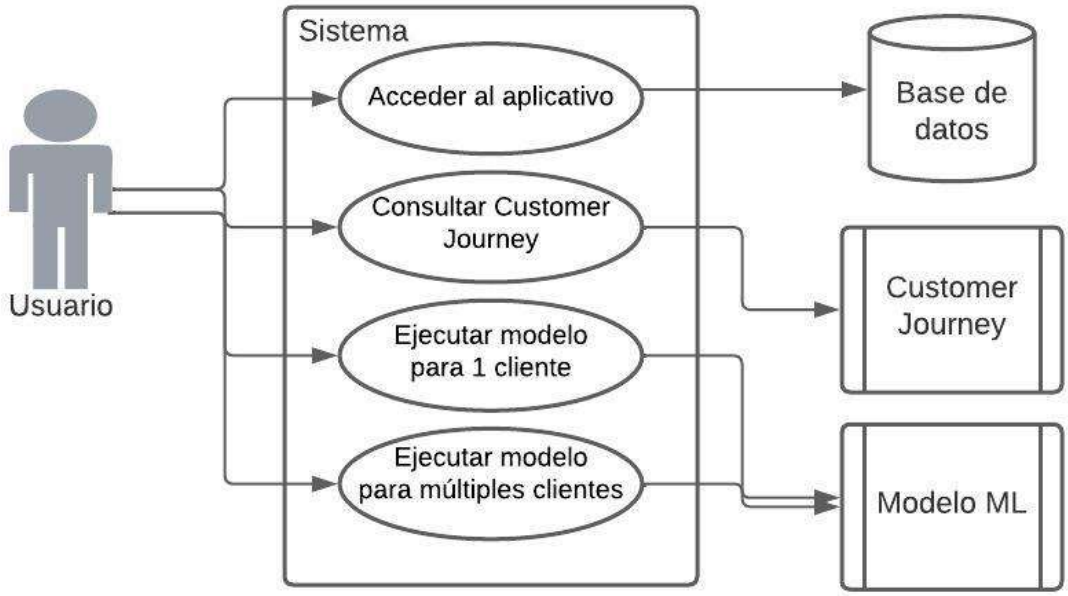
Paso 6	El sistema genera una descarga del archivo en el pc del usuario
Postcondición	El usuario queda con el archivo de resultado que usará en el CRM
Excepción	Paso 3: Si error en algún dato, Paso 2 el sistema muestra mensaje de warning con la recomendación para cumplir con la validación de ingreso de datos
Comentarios	

Caso de Uso: CU-004 Ejecutar modelo para múltiples clientes	
Dependencias	CU-001 Acceder al aplicativo
Precondición	Usuario verificado
Descripción	El sistema debe comportarse como se indica a continuación:
Paso 1	El usuario selecciona en el menú ésta opción
Paso 2	El sistema extiende el bloque con la opción para cargar archivo, muestra en el área principal una imagen ejemplo del archivo y su estructura
Paso 3	El usuario oprime el botón “Cargar archivo”
Paso 4	El sistema despliega una ventana de búsqueda de archivo en el pc del cliente
Paso 5	El usuario selecciona un archivo y enter
Paso 6	El sistema ejecuta el modelo de ML con los parámetros encontrados en el archivo, muestra un informe con el resumen de las variables y un botón para descargar el archivo de resultado
Paso 7	El usuario oprime el botón “Descargar archivo”
Paso 8	El sistema genera una descarga del archivo en el pc del usuario
Postcondición	El usuario queda con el archivo de resultado que usará en el CRM
Excepción	Paso 5: Si el archivo no cumple con la estructura, Paso 6 el sistema muestra un mensaje de error “Archivo no cumple con los parámetros”
Comentarios	

3.2.2 Diagrama de Casos de Uso

El Diagrama de casos de uso se define como una notación gráfica para representar los casos de uso definidos, donde encontramos el usuario como único actor del sistema que va

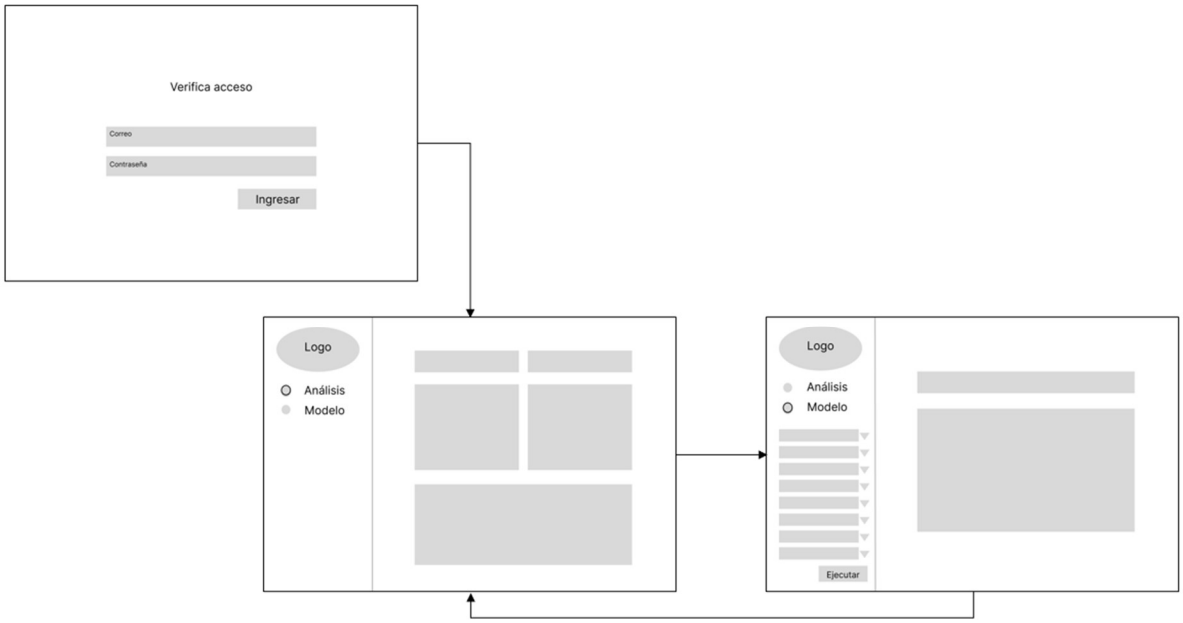
a interactuar con el software a través de 4 casos de uso (figura en elipse). Los casos de uso tienen comunicación con el modelo ML o con el módulo del Customer Journey o con la Base de Datos.



4. DESCRIPCIÓN DEL DISEÑO

4.1 Interfaz gráfica (Mockups)

4.1.1 Mockups de bajo nivel



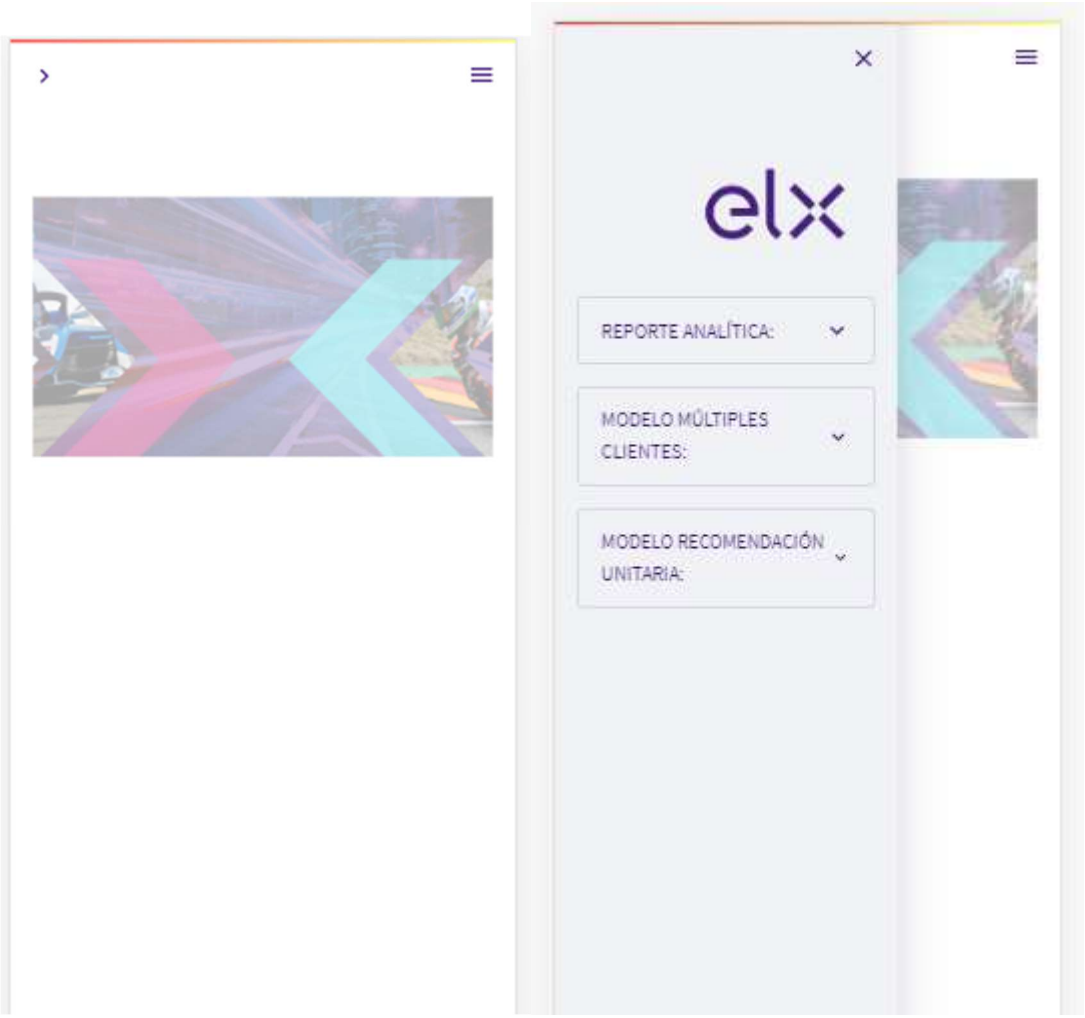
4.1.2 .Mockups de nivel alto

Acceso PC

Móvil

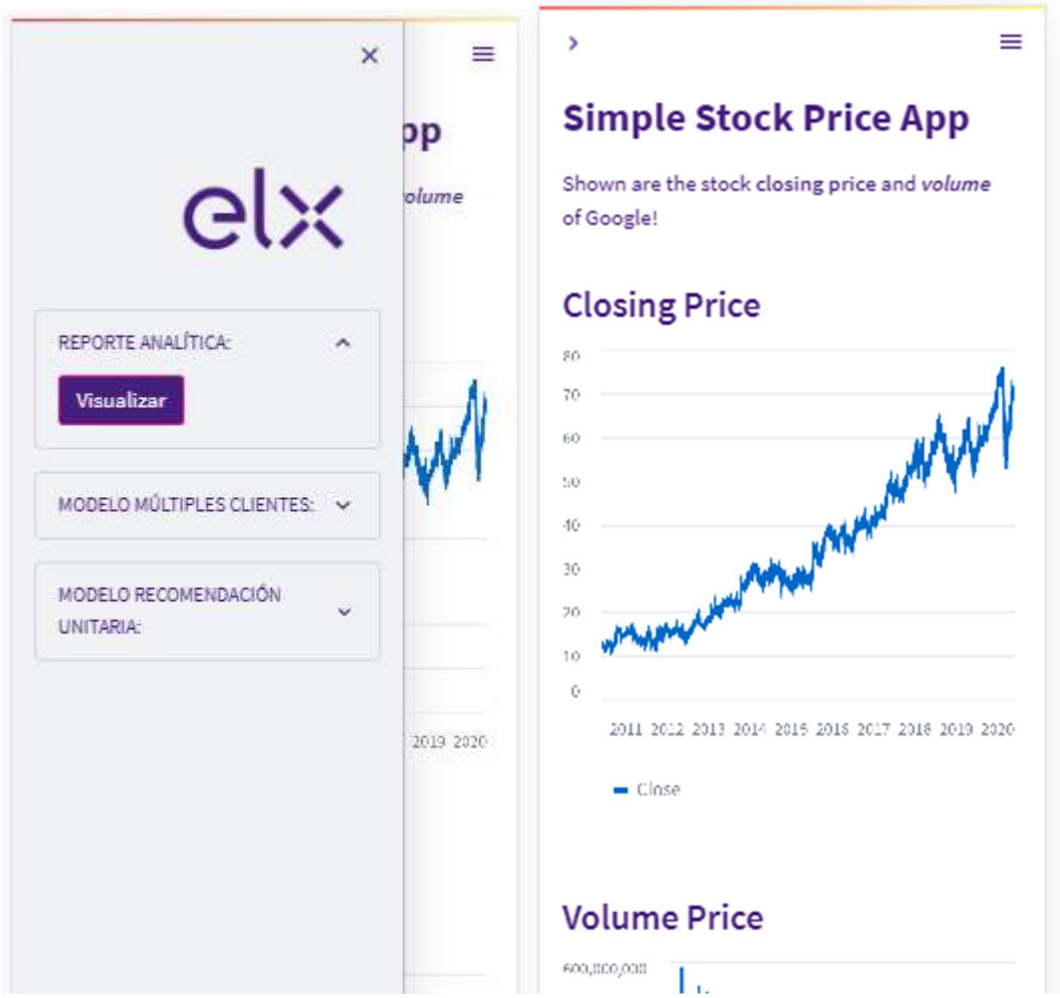
Inicio PC

Móvil



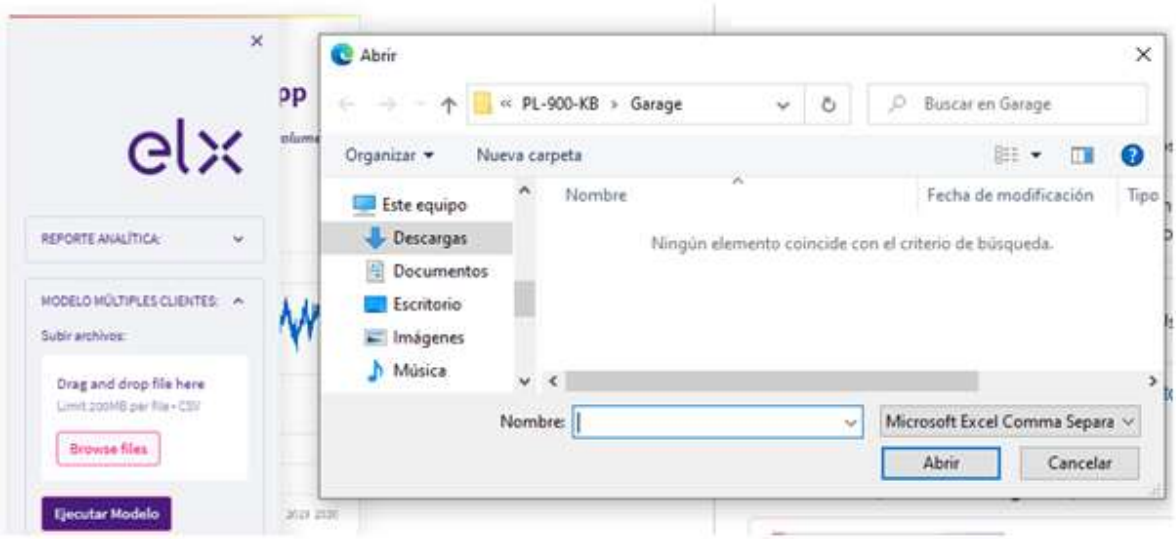
Análisis PC

Móvil



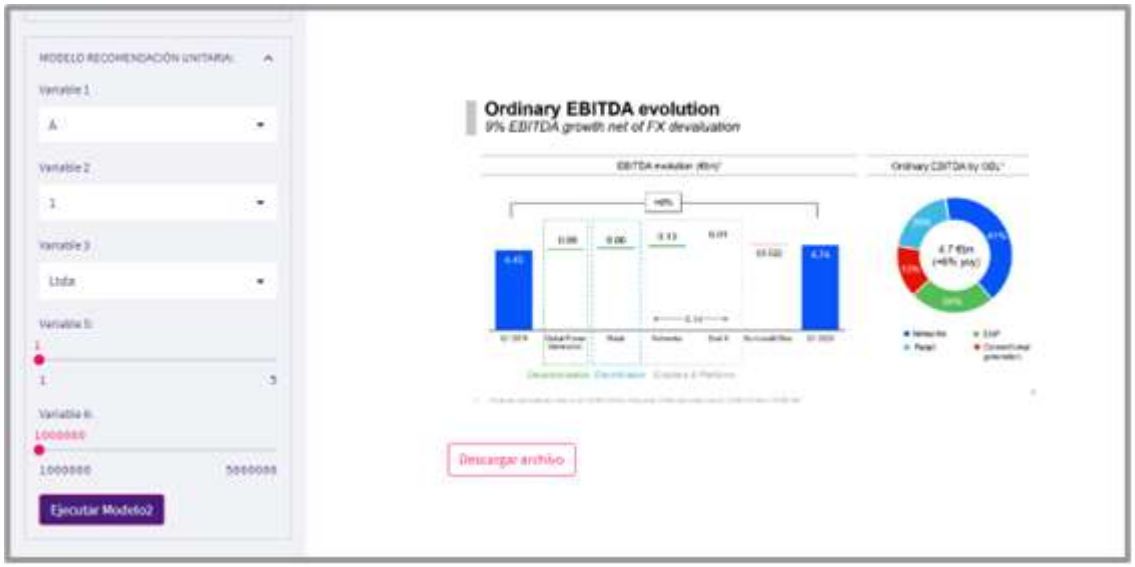
Modelo Múltiple PC

Móvil

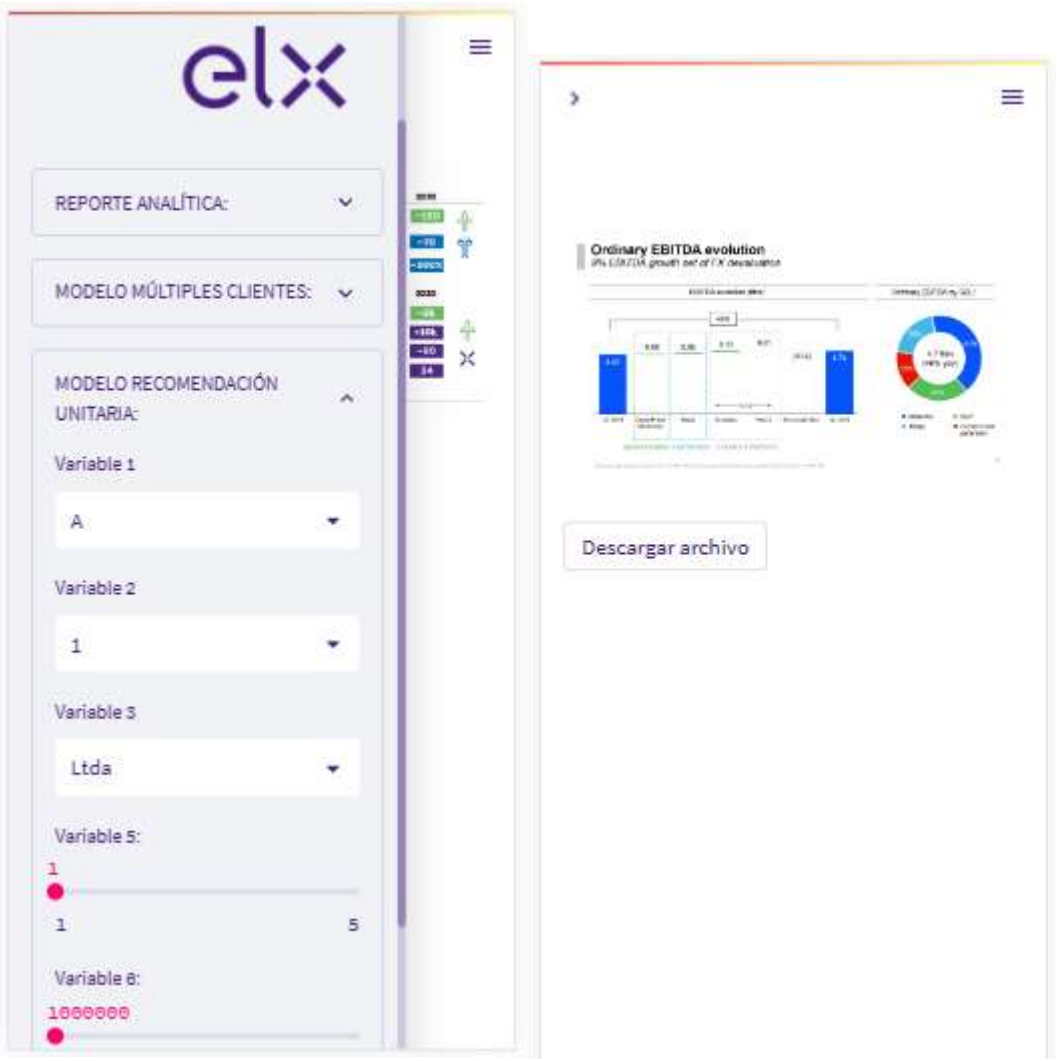

elx
REPORTE ANALÍTICA:
MODELO MÚLTIPLES CLIENTES:
Subir archivos:
Drag and drop file here
Limit 200MB per file • CSV
Browse files
Ejecutar Modelo
Abrir
PL-900-KB › Garage
Buscar en Garage
Organizar
Nueva carpeta
Este equipo
Descargas
Documentos
Escritorio
Imágenes
Música
Nombre
Fecha de modificación
Tipo
Ningún elemento coincide con el criterio de búsqueda.
Nombre:
Microsoft Excel Comma Separa
Abrir
Cancelar


Descargar archivo

Módulo Unitario PC



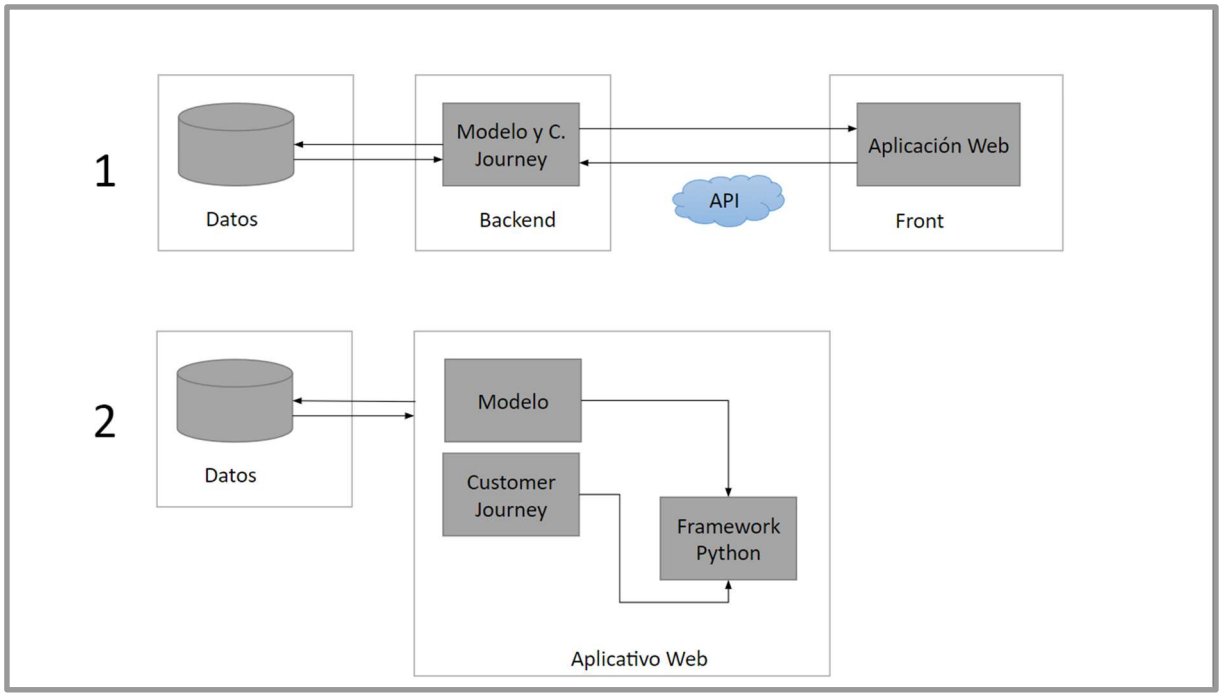
Móvil



5. GESTIÓN DE LA CONFIGURACIÓN

5.1 Arquitectura del software

La arquitectura del software hace referencia a la estructura y la relación entre las diferentes partes de un software y sus componentes independientes. A continuación se muestran las dos opciones contempladas para el desarrollo del proyecto:



OPCIÓN 1:

En el desarrollo se usarán herramientas como:

- Backend estructurado para API
- API rest o API Fast (Backend)
- VUE
- Vuetify: se usó para el desarrollo de las vistas.
- Nodejs: para dar respuesta a las acciones solicitadas
- Despliegue en Google Cloud

OPCIÓN 2:

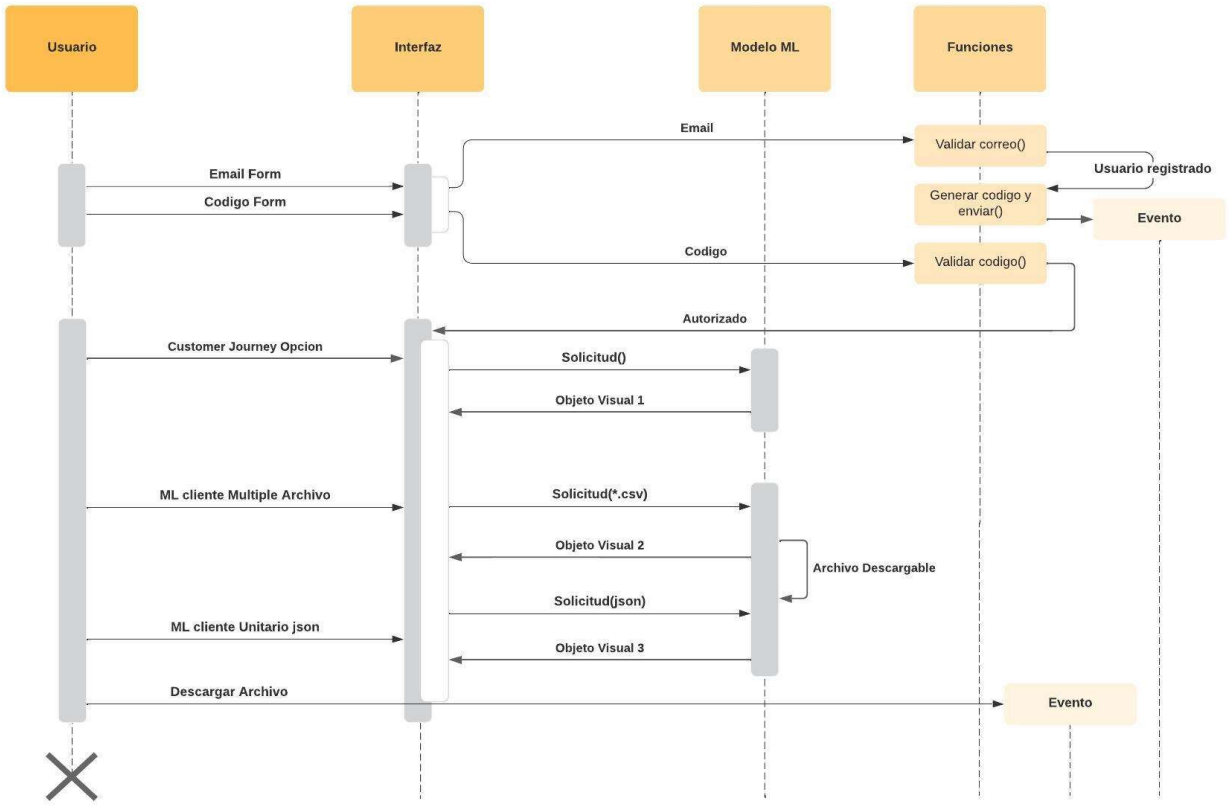
En el desarrollo se usarán herramientas como:

- Python
- Estructura desarrollo modular
- Framework Streamlit embebido en el código
- Despliegue en Google Cloud

La opción seleccionada es la 2 porque permite una integración más ágil y consistente entre el modelo, los objetos resultantes y el front-end.

5.2 Diagrama de Secuencia

El diagrama de secuencia está estructurado de tal manera que representa una línea de tiempo que comienza en la parte superior y desciende gradualmente para marcar la secuencia de las interacciones. Cada objeto tiene una columna y los mensajes intercambiados entre ellos están representados por flechas



5.3 Modelo de Datos

El modelo de datos muestra la estructura de datos lógica, atributos, tipo, identificación y otros parámetros necesarios por cada línea de comunicación entre los módulos o componentes del sistema software.

Customer Journey
Entrada
Salida

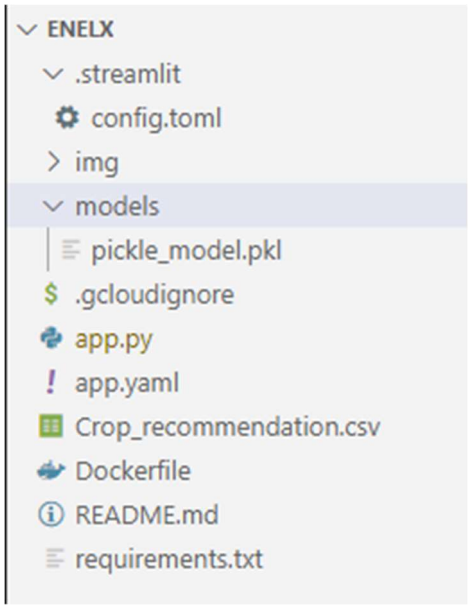
Modelo ML cliente unitario
Entrada
Salida

Modelo ML cliente múltiple
Entrada (Estructura archivo)
Salida resumen informe
Salida descargable (Estructura archivo resultado)

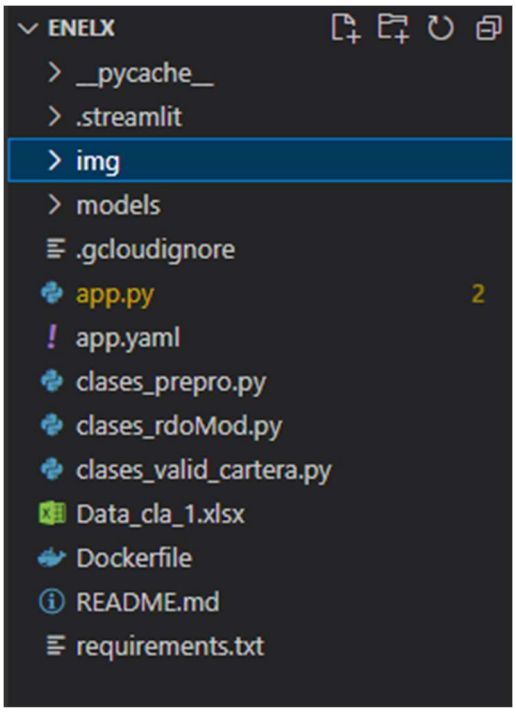
5.4 Estructura del Repositorio

El control de versiones del proyecto se ha realizado por medio de Github. El código del sistema se encuentra en privado en el siguiente repositorio:
https://github.com/krito20

El proyecto sigue la siguiente estructura:



Versión 28 marzo 2023



Carpeta ELX: Contiene todos los archivos del proyecto

Carpeta **.Streamlit**: Contiene el archivo **config.toml** que contiene la configuración del tema para este proyecto
Carpeta img: Contiene las imágenes que se consumen en el desarrollo de la aplicación

Carpeta Models: Contiene el modelo de Machine Learning

.gcloudignore: Archivo de texto que le dice a Gcloud qué archivos o carpetas ignorar en el proyecto, es decir, qué archivos no debe subir al repositorio

app.py: Archivo en python que contiene el código de la interfaz web

app.yaml: Archivo que especifica la configuración para el entorno de la aplicación en App Engine en GCloud

Crop_recomendation.csv: Es el archivo base sobre el que se construye el modelo ML

Dockerfile: Archivo que contiene la lista de comandos necesarios para construir la imagen del contenedor.

Readme.md: Archivo con información del repositorio y una guía de uso

Requirements.txt: Archivo de texto con las librerías de Python requeridas

6. DESARROLLO DEL SISTEMA

La aplicación Streamlit dispone del servicio web, está contenida en el archivo app.py. Para correrlo, usaremos el siguiente comando:

streamlit run ./app.py

6.1 Archivo Principal

Primero importamos las librerías

app.py > ...

```
1   import streamlit as st
2   import yfinance as yf
3   import numpy as np
4   import pandas as pd
5   from flask import Flask, request, jsonify, render_template, url_for
6   import pickle
7   from sklearn import svm
8   import sys
9
```

Inicializamos el objeto del archivo principal y con el IAP (Identity-Aware Proxy) se establece una capa de autorización central para la aplicación a la que se accede mediante HTTPS, por lo que se usa un modelo de control de acceso para que solo puedan acceder usuarios autorizados.

app.py > ...

```python
app = Flask(__name__)

CERTS = None
AUDIENCE = None

def certs():
    """Returns a dictionary of current Google public key certificates for
    validating Google-signed JWTs. Since these change rarely, the result
    is cached on first request for faster subsequent responses.
    """
    import requests

    global CERTS
    if CERTS is None:
        response = requests.get(
            'https://www.gstatic.com/iap/verify/public_key'
        )
        CERTS = response.json()
    return CERTS

def get_metadata(item_name):
    """Returns a string with the project metadata value for the item_name.
    See https://cloud.google.com/compute/docs/storing-retrieving-metadata for
    possible item_name values.
    """
    import requests

    endpoint = 'http://metadata.google.internal'
    path = '/computeMetadata/v1/project/'
    path += item_name
    response = requests.get(
        '{}{}'.format(endpoint, path),
```

Se identifica el path del modelo y la función que entrega el resultado del mismo

app.py > main

```python
# Path del modelo preentrenado
MODEL_PATH = 'models/pickle_model.pkl'

# Se recibe la imagen y el modelo, devuelve la predicción
def model_prediction(x_in, model):

    x = np.asarray(x_in).reshape(1,-1)
    preds=model.predict(x)

    return preds
```

6.2 Función Principal

En la función **main**:
Se carga el modelo y definimos los atributos de la página web con el método set_page_config, establecemos el título y el ícono que se mostrará en la pestaña del navegador.
Se dispone el contenedor del menú con el método st.siderbar y se encabeza con el logo
En formato Markdown con CSS ajusta el estilo de algunos elementos

```python
app.py > main
def main():

    model=''

    # Se carga el modelo
    if model=='':
        with open(MODEL_PATH, 'rb') as file:
            model = pickle.load(file)

    # Configura titulo e icon de pagina
    st.set_page_config(page_title="Modelo Analítica       ", page_icon="img/Icono.ico", layout="wide")

    # Menú y logo
    st.sidebar.image("img/logo3.png")
    st.sidebar.write("")


    st.markdown("""
            <style>
            div.stButton > button:first-child {
                background-color:#461e7d;
                color:#ffffff
            }
            div.stButton > button:hover {
                background-color:#f0f2f6;
                color:#461e7d
            }
            </style>""", unsafe_allow_html=True)
```

En el menú principal se disponen 3 contenedores expandibles con el método st.expander para ejecutar: el Reporte Analítica o Customer Journey, el modelo con múltiples clientes o el modelo unitario. Para cada caso se disponen los elementos de las variables que se requieren como criterios de ejecución.

app.py > main

```
130
131     a, b, c = False, False, False
132     datos =''
133     # Title
134     #st.title("Customer Analytics")
135         
136
137     with st.sidebar.expander("REPORTE ANALÍTICA: ", expanded = False):
138         #if st.sidebar.button("Mi primer opción Modelo"):
139         a = st.button("Visualizar",type="primary")
140
141
142     with st.sidebar.expander("MODELO MÚLTIPLES CLIENTES: ", expanded = False):
143         #if st.sidebar.button("Mi primer opción Modelo"):
144         # uploading files
145         datos = st.file_uploader("Subir archivos: ", type = ["csv"])
146         print(type(datos))
147         #st.image(data)
148         b = st.button("Ejecutar Modelo",type="primary")
149
150
151     with st.sidebar.expander("MODELO RECOMENDACIÓN UNITARIA: ", expanded = False):
```

De acuerdo a la opción del menú seleccionada invoca a la función para cada caso y dispone el resultado con sus elementos en el área principal de la página organizada por contenedores y columnas con los métodos st.container y st.columns

app.py > main

```
175     if a == False and b == False and c == False:
176         st.image("img/Img_presentacion2.jpg",use_column_width="always")
177     elif a == True:
178         with st.container():
179             st.write("""
180             # Simple Stock Price App
181             Shown are the stock **closing price** and ***volume*** of Google!
182             """)
183 
184             # https://towardsdatascience.com/how-to-get-stock-data-using-python-c0de1df17e75
185             #define the ticker symbol
186             tickerSymbol = 'GOOGL'
187             #get data on this ticker
188             tickerData = yf.Ticker(tickerSymbol)
189             #get the historical prices for this ticker
190             tickerDf = tickerData.history(period='1d', start='2010-5-31', end='2020-5-31')
191             # Open  High    Low Close   Volume  Dividends   Stock Splits
192 
193             col1, col2 = st.columns(2)
194             col1.write("""
195             ## Closing Price
196             """)
197             col1.line_chart(tickerDf.Close)
198             col2.write("""
199             ## Volume Price
200             """)
201             col2.line_chart(tickerDf.Volume)
202     elif b == True:
203         with st.container():
204             st.write("CONSUMO")
205             col7,col8 = st.columns(2)
206             col7.image("img/Modelo2.png",use_column_width="always")
```

Finalmente se invoca la función main

app.py > ...

```
277             #st.download_button( Desc
278 
279 if __name__ == '__main__':
280     main()
```