Tu tarea es hacer una aplicación web de “PROYECTO COMPRA CASAS”

*El mínimo esperado a nivel de la estructura del proyecto son los siguientes pasos:*

1. Utilizando Python, preparar los documentos “readme” y “requirements”.
2. La aplicación se hará con VS Code, para ser commit con GitHub y Render según sea necesario.
3. Al menos un encabezado con texto de ejemplo “Escoge la vivienda a tu comodidad”
4. Poner una caja para entrada de cantidad de Habitaciones
5. Poner una caja para entrada de cantidad de Baños
6. Poner una caja para entrada si desea Ascensor
7. Poner una caja para entrada si desea Garaje
8. Poner una caja para entrada si desea Energía Renovable
9. Al menos un gráfico de dispersión - puede ser de la columna de precios – cambiar el color cada 50.000 EUR
10. Al menos un histograma. Crea un botón que, al hacer clic en él, construya un histograma plotly-express. Considera utilizar las funciones [[st.write()](https://docs.streamlit.io/library/api-reference/write-magic/st.write)] y [[st.plotly\_chart()](https://docs.streamlit.io/library/api-reference/charts/st.plotly_chart)].
11. Al menos un botón para mostrar las selecciones mostrando el filtrado de las cajas. Puede presentarse como una lista tipo Word o una tabla tipo Excel con el encabezado del archivo 1house\_prices\_spain

Puedes explicar los pasos uno por uno

**COPILOT**

**🧱 Paso 1: Preparar README.md y requirements.txt con Python**

✅ requirements.txt

Este archivo define las dependencias mínimas para tu app:

streamlit>=1.30.0

pandas>=2.0.0

numpy>=1.24.0

plotly>=5.20.0

Puedes generarlo automáticamente con:

pip freeze > requirements.txt

✅ README.md

Un ejemplo básico:

# 🏡 Proyecto Compra Casas Aplicación interactiva para explorar viviendas en España según tus preferencias. Construida con Streamlit, pandas y plotly. ## 🚀 Características - Filtros por habitaciones, baños, ascensor, garaje y energía renovable - Gráfico de dispersión de precios con colores por rango - Histograma interactivo con botón - Tabla de resultados filtrados ## ▶️ Ejecutar ```bash streamlit run app.py



📦 Requisitos

Ver requirements.txt

**💻 Paso 2: VS Code + GitHub + Render 1. Crea tu proyecto en VS Code. 2. Inicializa Git:** ```bash git init git add . git commit -m "Inicio del proyecto Compra Casas"

1. Sube a GitHub.
2. En [Render.com](https://render.com/), crea un nuevo servicio web → selecciona tu repo → configura para ejecutar streamlit run app.py.

**🏠 Paso 3: Encabezado**

st.header("🏡 Escoge la vivienda a tu comodidad")

**🔢 Paso 4–8: Entradas de usuario**

habitaciones = st.number\_input("Cantidad de habitaciones", min\_value=1, max\_value=10, value=3) baños = st.number\_input("Cantidad de baños", min\_value=1, max\_value=5, value=2) ascensor = st.checkbox("¿Desea ascensor?") garaje = st.checkbox("¿Desea garaje?") energia\_renovable = st.checkbox("¿Desea energía renovable?")



**📊 Paso 9: Gráfico de dispersión con color por precio**

Supongamos que tienes un DataFrame df con columna precio:

import plotly.express as px df['color'] = pd.cut(df['precio'], bins=[0, 50000, 100000, 150000, 200000, 300000], labels=["🟩", "🟨", "🟧", "🟥", "🟪"]) fig = px.scatter(df, x="metros", y="precio", color="color", title="Precios por tamaño") st.plotly\_chart(fig)



**📈 Paso 10: Histograma con botón**

if st.button("Mostrar histograma de precios"): fig\_hist = px.histogram(df, x="precio", nbins=30, title="Distribución de precios") st.write("Histograma de precios") st.plotly\_chart(fig\_hist)



**📋 Paso 11: Mostrar selecciones filtradas**

Filtra el DataFrame según las entradas:

filtro = ( (df['habitaciones'] == habitaciones) & (df['baños'] == baños) & (df['ascensor'] == ascensor) & (df['garaje'] == garaje) & (df['energia\_renovable'] == energia\_renovable) ) df\_filtrado = df[filtro] st.write("🏷️ Resultados filtrados:") st.dataframe(df\_filtrado)

¿Quieres que te ayude a generar el archivo app.py completo con esta lógica integrada y listo para subir a GitHub?

**Código appcasa.py:**

import streamlit as st

import pandas as pd

import numpy as np

import plotly.express as px

# Título principal

st.header("🏡 Escoge la vivienda a tu comodidad")

# Entradas del usuario

habitaciones = st.number\_input("Cantidad de habitaciones", min\_value=1, max\_value=10, value=3)

baños = st.number\_input("Cantidad de baños", min\_value=1, max\_value=5, value=2)

ascensor = st.checkbox("¿Desea ascensor?")

garaje = st.checkbox("¿Desea garaje?")

energia\_renovable = st.checkbox("¿Desea energía renovable?")

# Cargar datos simulados o reales

# Aquí puedes reemplazar con tu archivo real '1house\_prices\_spain.csv'

# Por ahora, generamos un DataFrame simulado

np.random.seed(42)

df = pd.DataFrame({

'habitaciones': np.random.randint(1, 6, 100),

'baños': np.random.randint(1, 4, 100),

'ascensor': np.random.choice([True, False], 100),

'garaje': np.random.choice([True, False], 100),

'energia\_renovable': np.random.choice([True, False], 100),

'precio': np.random.randint(50000, 300000, 100),

'metros': np.random.randint(50, 200, 100)

})

# Gráfico de dispersión con color por rangos de precio

df['color\_precio'] = pd.cut(df['precio'],

bins=[0, 50000, 100000, 150000, 200000, 300000],

labels=["🟩 <=50k", "🟨 <=100k", "🟧 <=150k", "🟥 <=200k", "🟪 >200k"])

fig\_dispersion = px.scatter(df, x="metros", y="precio", color="color\_precio",

title="📈 Precio vs Tamaño (m²)", labels={"metros": "Metros cuadrados", "precio": "Precio (€)"})

st.plotly\_chart(fig\_dispersion)

# Botón para mostrar histograma

if st.button("Mostrar histograma de precios"):

fig\_hist = px.histogram(df, x="precio", nbins=30, title="📊 Distribución de precios")

st.write("Histograma de precios")

st.plotly\_chart(fig\_hist)

# Botón para mostrar resultados filtrados

if st.button("Mostrar viviendas filtradas"):

filtro = (

(df['habitaciones'] == habitaciones) &

(df['baños'] == baños) &

(df['ascensor'] == ascensor) &

(df['garaje'] == garaje) &

(df['energia\_renovable'] == energia\_renovable)

)

df\_filtrado = df[filtro]

st.write("🏷️ Resultados filtrados según tus preferencias:")

st.dataframe(df\_filtrado)

st.write(f"🔎 Se encontraron {len(df\_filtrado)} viviendas que cumplen con tus criterios.")

else:

st.write("🛠️ La aplicación está en construcción. Ajusta los filtros y haz clic en los botones para explorar.")