Perfect — let’s build your **“Crypto Price Tracker”** project step-by-step using requests, the **CoinGecko API**, smtplib, and **Streamlit**. I’ll give you the **codebase structure**, a **Streamlit app**, and supporting files for alerts and logs.

**Project Structure**

crypto-price-tracker/

app.py # Streamlit UI + logic

email\_utils.py # Email alert sender

requirements.txt # Needed libraries

alert\_log.txt # Stores triggered alerts

README.md # Quick start guide

**requirements.txt**

streamlit>=1.37

requests>=2.31

pandas>=2.0

matplotlib>=3.8

**email\_utils.py**

import smtplib

from email.mime.text import MIMEText

def send\_email\_alert(sender\_email, sender\_pass, receiver\_email, subject, body):

msg = MIMEText(body, "plain")

msg["Subject"] = subject

msg["From"] = sender\_email

msg["To"] = receiver\_email

try:

with smtplib.SMTP\_SSL("smtp.gmail.com", 465) as server:

server.login(sender\_email, sender\_pass)

server.send\_message(msg)

return True

except Exception as e:

print("Error sending email:", e)

return False

**app.py (Streamlit App)**

import streamlit as st

import requests

import pandas as pd

import time

import matplotlib.pyplot as plt

from datetime import datetime

from email\_utils import send\_email\_alert

# -----------------------------

# Config

# -----------------------------

API\_URL = "https://api.coingecko.com/api/v3/simple/price"

LOG\_FILE = "alert\_log.txt"

# -----------------------------

# Helpers

# -----------------------------

def fetch\_prices(coin\_list, vs\_currency="usd"):

try:

ids = ",".join(coin\_list)

resp = requests.get(API\_URL, params={"ids": ids, "vs\_currencies": vs\_currency})

if resp.status\_code == 200:

return resp.json()

else:

return {}

except Exception as e:

st.error(f"Error fetching prices: {e}")

return {}

def log\_alert(message):

with open(LOG\_FILE, "a") as f:

f.write(f"{datetime.now()} - {message}\n")

# -----------------------------

# Streamlit UI

# -----------------------------

st.set\_page\_config("Crypto Price Tracker", layout="wide")

st.title("💹 Crypto Price Tracker")

st.sidebar.header("Settings")

coins = st.sidebar.multiselect(

"Select Cryptos",

["bitcoin", "ethereum", "dogecoin", "ripple", "solana"],

default=["bitcoin", "ethereum"]

)

thresholds = {}

for coin in coins:

thresholds[coin] = st.sidebar.number\_input(f"Alert if {coin} below (USD):", min\_value=0.0, step=100.0, value=0.0)

email\_section = st.sidebar.expander("📧 Email Alerts Settings")

sender\_email = email\_section.text\_input("Sender Email (Gmail)")

sender\_pass = email\_section.text\_input("App Password", type="password")

receiver\_email = email\_section.text\_input("Receiver Email")

refresh\_interval = st.sidebar.slider("Auto-refresh interval (seconds)", 10, 300, 60)

# -----------------------------

# Main loop

# -----------------------------

if st.button("Start Tracking"):

placeholder = st.empty()

chart\_placeholder = st.empty()

history = {coin: [] for coin in coins}

timestamps = []

while True:

prices = fetch\_prices(coins)

if prices:

# Table

df = pd.DataFrame(prices).T

df.columns = ["Price (USD)"]

placeholder.dataframe(df.style.format("${:.2f}"))

# Alerts

for coin in coins:

price = prices[coin]["usd"]

th = thresholds[coin]

if th > 0 and price < th:

alert\_msg = f"ALERT: {coin} price ${price:.2f} below threshold ${th:.2f}"

log\_alert(alert\_msg)

st.warning(alert\_msg)

if sender\_email and sender\_pass and receiver\_email:

send\_email\_alert(

sender\_email,

sender\_pass,

receiver\_email,

subject=f"Crypto Alert: {coin}",

body=alert\_msg

)

# Chart

timestamps.append(datetime.now().strftime("%H:%M:%S"))

for coin in coins:

history[coin].append(prices[coin]["usd"])

fig, ax = plt.subplots()

for coin in coins:

ax.plot(timestamps, history[coin], label=coin)

ax.set\_xlabel("Time")

ax.set\_ylabel("Price (USD)")

ax.legend()

chart\_placeholder.pyplot(fig)

time.sleep(refresh\_interval)

**alert\_log.txt (example after run)**

2025-09-08 12:10:33 - ALERT: bitcoin price $24999.00 below threshold $25000.00

**README.md**

# Crypto Price Tracker

## Objective

Track live cryptocurrency prices with alerts.

## Tools

- Python `requests` (CoinGecko API)

- `Streamlit` (UI dashboard)

- `smtplib` (email alerts)

- `matplotlib` (charts)

## Features

1. Fetch live prices (CoinGecko API).

2. Display table of crypto prices.

3. Set custom thresholds for alerts.

4. Send email alerts when conditions are met.

5. Auto-refresh periodically.

6. Visualize historical trends with line charts.

7. Log alerts to `alert\_log.txt`.

## Run

```bash

pip install -r requirements.txt

streamlit run app.py

---

**\*\*Deliverables included\*\*:**

- Script (`app.py`)

- Email utility (`email\_utils.py`)

- Alert log (`alert\_log.txt`)

- Sample email content via `smtplib`