Please build the UI using **Streamlit**, and ensure everything works using **only free-tier services and libraries**. The final app should be easily deployable to **Streamlit Community Cloud** or **Replit (free plan)**.

**✅ Settings Panel**

* Dropdown to select **any LLM provider**
  + Prepopulate with: OpenAI, Claude, Gemini, Mistral, Cohere
  + Allow manual input of other/custom providers if needed
* Text input for LLM **API key**
* ✅ Show a **dynamic "Create API Key" link** below the input field for known LLMs (open in a new tab):
  + OpenAI → <https://platform.openai.com/account/api-keys>
  + Claude → https://console.anthropic.com/settings/keys
  + Gemini → https://makersuite.google.com/app/apikey
  + Mistral → https://console.mistral.ai/api-keys
  + Cohere → https://dashboard.cohere.com/api-keys
  + If unknown LLM selected, hide the link or show a generic placeholder
* Input for **Google Analytics Property ID**
* Button to **Save settings** using streamlit.session\_state
* Button to **Test API Key** (optional — basic call to validate)

**✅ Chat Interface**

* Text area for user queries
* “Submit” button to trigger enhancement + data fetching
* Optional toggle: **Enable/Disable Prompt Enhancement**
* Show both:
  + The original user query
  + The enhanced prompt (if enabled)

**✅ Prompt Enhancer**

* Rewrite vague queries into clear analytics prompts
* Example:
  + Input: “users by device”
  + Enhanced: “Generate a report comparing total users by device category over the last 30 days using Google Analytics. Display the result in a pie chart and table.”

**✅ LLM Integration**

* Use the selected provider’s API to send the enhanced prompt
* Receive a structured response or command that can be used to query GA4
* Use requests or similar for API calls

**✅ Google Analytics Integration**

* Use the **Google Analytics Data API (v1)**
* Authenticate via service account or OAuth2
* Extract metrics like: users, sessions, bounce rate, device, country, traffic source, etc.
* Return data as a pandas.DataFrame

**✅ Data Output**

* Display:
  + Table of results
  + Appropriate chart (line, bar, pie, etc.)
* Render results inside the app
* Provide download buttons for:
  + CSV (to\_csv)
  + Excel (to\_excel via openpyxl)
  + PDF (fpdf, plotly image, or reportlab)
  + Word (python-docx)

**✅ Prompt Templates & Query History**

* Show common queries as one-click buttons:
  + “Compare desktop vs mobile traffic”
  + “Generate a PDF report for last month’s traffic sources”
* Store query history in session
* Let users click a previous query to reload it

**📁 Project File Structure**

bash

CopyEdit

/llm-analytics-assistant/

├── app.py # Streamlit app

├── llm\_handler.py # LLM logic and prompt enhancement

├── ga\_query.py # Google Analytics API handler

├── export\_utils.py # Download/export functionality

├── .streamlit/secrets.toml # For secure keys on Streamlit Cloud

├── requirements.txt # Required Python packages

└── README.md # Instructions

**📦 Free-Only Dependencies**

* streamlit
* pandas
* openpyxl
* python-docx
* fpdf or reportlab
* plotly or matplotlib
* google-analytics-data
* requests

**✅ Hosting Platform: Streamlit Community Cloud**

Make sure this app can run entirely on **Streamlit Community Cloud**, with optional development/testing in **Replit (free plan)**. No paid services or libraries should be required.

Please begin by generating the full project scaffold with a minimal working prototype (app.py showing hardcoded output and dynamic API key link). Include placeholder files for future modules and a requirements.txt.