DAY – 5

BUILDING A WEBPAGE SUMMARIZER USING PYTHON

On Day 5, we transitioned from prompt writing and AI platforms into practical implementation using Python. The session was focused on how to integrate Generative AI capabilities into coding projects using APIs and external libraries.

The first task was to build a **Website Summarizer Tool** that takes a webpage URL as input and generates a markdown-formatted summary. The core steps included:

1. **Generating API Key from Google AI Studio:**
   We began by generating a working API key to access generative capabilities from Google's AI services.
2. **Coding the Summarizer:**
   - We wrote a Python program that accepts a dynamic user input (i.e., the URL of any website).
   - Using the Beautiful Soup 4 library, we extracted all the internal and external links from the page.
   - The webpage content was then summarized using the API, and the output was structured in Markdown format.
   - The summarized result was saved as a file to the system for future access and documentation.

This hands-on project helped solidify our understanding of how to integrate AI-generated summaries into real-world applications, especially useful for content researchers, bloggers, and analysts.

IMAGE GENERATION USING API – HUGGING FACE & STABILITY AI

The second task involved dynamic image generation using external APIs:

- We generated API keys for Hugging Face and Stability AI.
- The Python code was written in such a way that users can input prompts dynamically, and based on the prompt, the AI will generate an image.
- The image response was decoded using the Base64 library to retrieve and display the generated visual content.

This task demonstrated how AI can respond not only with text but also visual content, opening doors to creative applications such as content design, marketing automation, and digital art tools.

HOMEWORK TASK

For the day's homework, we were asked to enhance the website summarizer in the following way:

- Reframe the summarized content into a new format using typewriter-style output.
- Store the reframed output in a separate file, different from the original summary.
- Make sure the content appears as if it was typed manually—possibly adding visual structure or delays to simulate the behaviour of a typewriter.

This extension challenged us to combine text formatting, file handling, and creative output simulation in Python.

CONCLUSION

Day 5 was a highly technical and productive day that focused on the real-world application of AI APIs through Python. We not only learned to build functional tools such as a web summarizer and dynamic image generator, but also gained confidence in working with libraries like Beautiful Soup and Base64, and integrating external AI services through API keys.

This hands-on coding experience provided a strong foundation in how AI can be embedded in custom software projects, making it a valuable learning milestone in our training journey.