# DS565 - Generative Al-Driven Intelligent Apps Development

Project: ChatBot Using Streamlit

Course: DS565

Professor: Dr. Henry Chang

Presented by: Karan Shrestha (20087)

#### Introduction to OpenAl API

- What is OpenAl API?
  - Provides access to powerful language models (GPT series)
  - Applications: Text generation, chatbots, translation, etc.
- Why GPT-4o Mini?
  - Efficient, lightweight version of GPT-4, optimized for quick interactions

#### **Streamlit Overview**

- What is Streamlit?
  - Python framework for building web apps quickly
  - Focuses on simplicity and real-time interaction
- Why Use Streamlit?
  - Easy to integrate with OpenAI for chatbot development
  - Great for prototyping interactive apps

#### **Chatbot Architecture**

- Key Components
  - OpenAl API: Powers the chatbot responses
  - Streamlit: Manages the user interface
  - Session State: Maintains the conversation history
- Flow of Interaction
  - User input → OpenAl API → Response → Display in Streamlit

#### **Code Walkthrough (Key Parts)**

- 1. Setting Up
  - Import necessary libraries: openai, streamlit, and doteny
  - Initialize the OpenAl client using OpenAl()
- 2. Capturing User Input
  - Use st.chat\_input("Your Prompt:") for user interaction
  - Store conversation in st.session\_state["messages"]

## **Code Walkthrough (Response Generation & Display)**

- 1. Sending Input to OpenAl
  - Use client.chat.completions.create to send prompts to GPT-4o Mini
  - Stream responses in real time
- 2. Displaying Messages
  - Loop through conversation history and display with st.markdown()
  - Real-time conversation updates

```
CHATBOT STREAMLIT
                     chatbot.py > ...
                           import streamlit as st
env.
                           from openai import OpenAI
chatbot.py
                           from dotenv import load dotenv
                           load dotenv()
                          client = OpenAI()
                           st.title("My GPT-40 Mini Chatbot is")
                          if "messages" not in st.session state:
                               st.session state.messages = []
                           for message in st.session state["messages"]:
                               with st.chat message(message["role"]):
                                   st.markdown(message["content"])
                           if user prompt := st.chat input("Your Prompt:"):
                               st.session state.messages.append({"role": "user", "content": user prompt})
                               with st.chat message("user"):
                                   st.markdown(user prompt)
                               with st.chat message("assistant"):
                                   chatbot msg = st.empty()
                                   full response = ""
                                   stream = client.chat.completions.create(
                                       model="gpt-4o-mini",
```

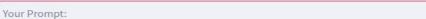
### **Running the Chatbot**

- Run Command: streamlit run chatbot.py
- Local URL: View in browser via local/remote network URL provided by Streamlit

# My GPT-40 Mini Chatbot 🤖

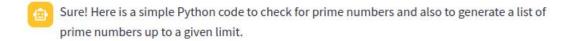


Hello! How can I assist you today?









#### Checking if a single number is prime:

```
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True
# Example usage
number = 29
if is_prime(number):
    print(f"{number} is a prime number.")
else:
    print(f"{number} is not a prime number.")
```

