import streamlit as st

import matplotlib.pyplot as plt

import pandas as pd

from transformers import pipeline

import torch

# Check if CUDA is available and choose the device

device = 0 if torch.cuda.is\_available() else -1 # Use GPU (device=0) if available, otherwise use CPU

# Initialize the pipeline for question answering using a pre-trained model, with GPU if available

hf\_pipeline = pipeline("question-answering", model="distilbert-base-cased-distilled-squad", device=device)

# Use Streamlit's new caching feature for loading the model

@st.cache\_resource

def load\_model():

return hf\_pipeline

hf\_pipeline = load\_model()

# Title of the app

st.title("My Personal Investment Advisor")

# Add a description or instruction to the app

st.write("Welcome to the Personal Investment Advisor. Please provide your details below.")

# Input fields for the user

income = st.number\_input("Enter your monthly income", min\_value=0)

expenditure = st.number\_input("Enter your monthly expenditure", min\_value=0)

savings = st.number\_input("Enter your monthly savings", min\_value=0)

# Display the values the user entered

st.write(f"Your monthly income is {income}")

st.write(f"Your monthly expenditure is {expenditure}")

st.write(f"Your monthly savings are {savings}")

# Simple calculation: Net savings = Savings - Expenditure

net\_savings = savings - expenditure

st.write(f"Your net savings after expenditure: {net\_savings}")

# Financial Health Calculation (Income - Expenditure)

financial\_health = income - expenditure

st.write(f"Your Financial Health (Income - Expenditure): {financial\_health}")

# Dynamic Chart: Expenditure vs Savings Comparison

plt.bar(["Expenditure", "Savings"], [expenditure, savings], color=["red", "green"])

plt.title("Expenditure vs Savings Comparison")

plt.ylabel("Amount")

st.pyplot(plt)

# Ask for risk tolerance

risk\_tolerance = st.radio("Select your risk tolerance", ["Low", "Medium", "High"])

# Display a suggestion based on risk tolerance

if risk\_tolerance == "Low":

st.write("Suggested Investment: Bond Funds, Fixed Deposits")

elif risk\_tolerance == "Medium":

st.write("Suggested Investment: Balanced Mutual Funds, Stocks")

else:

st.write("Suggested Investment: High-Risk Stocks, Cryptocurrency")

# Investment Growth Prediction based on Risk Tolerance

investment\_growth = []

if risk\_tolerance == "Low":

investment\_growth = [savings \* 1.05, savings \* 1.10, savings \* 1.15, savings \* 1.20] # Low risk grows slowly

elif risk\_tolerance == "Medium":

investment\_growth = [savings \* 1.10, savings \* 1.15, savings \* 1.20, savings \* 1.30] # Medium risk grows moderately

else:

investment\_growth = [savings \* 1.20, savings \* 1.30, savings \* 1.40, savings \* 1.50] # High risk grows quickly

# Display a graph for Risk Tolerance vs Investment Growth

months = ["Jan", "Feb", "Mar", "Apr"]

plt.plot(months, investment\_growth, marker='o', color='blue')

plt.title(f"Investment Growth Based on {risk\_tolerance} Risk Tolerance")

plt.xlabel("Months")

plt.ylabel("Investment Value")

st.pyplot(plt)

# Decision based on the user's net savings

if net\_savings > 10000:

st.write("Based on your savings, we recommend exploring high-risk, high-return investments like stocks.")

elif net\_savings > 5000:

st.write("We suggest balanced investments like mutual funds.")

else:

st.write("You might want to consider safer investments like fixed deposits or bonds.")

# Function to handle user questions with the pre-trained model, customized with rule-based answers

def get\_answer(question, user\_data):

# Rule-based response for financial health or expenditure-related queries

if "reduce expenditure" in question.lower():

return "You can reduce your expenditure by cutting down on unnecessary expenses, creating a budget, and setting saving goals."

if "improve financial health" in question.lower():

return "To improve your financial health, focus on increasing savings, reducing unnecessary expenditures, and managing debt effectively."

# Context for investment modes and financial health

context = f"""

The user has the following financial data:

Income: {user\_data['income']}, Expenditure: {user\_data['expenditure']}, Savings: {user\_data['savings']}

ESG investing refers to an investment strategy that considers Environmental, Social, and Governance (ESG) factors when making investment decisions. It looks at how companies perform on issues like sustainability, social impact, and corporate governance.

Financial health is determined by analyzing an individual's income, expenditure, savings, and investments. Maintaining good financial health requires balancing income, managing debt, and saving for future goals.

Investment modes refer to different strategies for investing money, such as conservative, balanced, and aggressive approaches, each based on risk tolerance and financial goals.

"""

# Use the pre-trained model to get the answer

result = hf\_pipeline(question=question, context=context)

return result['answer']

# Interaction Section: Chatbot for user questions

st.write("## Ask the Investment Advisor")

user\_question = st.text\_input("Enter your question here:")

if user\_question:

user\_data = {'income': income, 'expenditure': expenditure, 'savings': savings}

# Get the answer based on user input

answer = get\_answer(user\_question, user\_data)

st.write(f"\*\*Answer:\*\* {answer}")