**Project Report**

**Name: Nirmit Chaurasia**

**Task Chosen: Task 1 (Docker Setup) & Task 4 (AI-powered Customer Support)**

**Approach Summary**

For the first task, I directly followed the steps mentioned in the official Pathway Docker deployment guide. I installed Docker Desktop, enabled WSL2, connected Ubuntu through Docker settings, and then ran a sample Pathway program inside a Docker container. The setup worked successfully, and I was able to verify the output through the container.

For the second task, I was initially deciding between building a stock price predictor or an AI-powered customer support bot, as both topics are closely related to my personal interests and past projects. Since stock prediction is a very common project, I chose to work on the AI-powered customer support chatbot (Task 4) to create something more interactive and different.

To approach this, I revisited one of my earlier projects that used LangChain, as well as our Bajaj Finserv hackathon submission that had a somewhat similar concept. I studied how conversation memory, prompt templates, and model integration were handled there. Then, I applied the same understanding here to design the chatbot using the Hugging Face API and LangChain framework.

I did take help from GPT and other AI tools mainly for coding faster, debugging, and improving formatting, but I made sure to understand each part of the implementation clearly, both for the Docker setup and the chatbot logic. This helped me complete the tasks efficiently while still learning how everything works.

**Task 1 – Docker Setup for Pathway Application**

Since I primarily use Windows, I had to create aLinux-based containerized environment using Docker so that the application could run smoothly across all systems.

**Steps Followed**

1. Installing Docker and Enabling WSL2

I downloaded and installed Docker Desktop from the official Docker website.

Then, I enabled Windows Subsystem for Linux (WSL) and Virtual Machine Platform by running:

dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart

dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart

wsl --set-default-version 2

wsl --install -d Ubuntu

This installed Ubuntu as my default Linux distribution.

1. Connecting Docker with WSL

Opened Docker Desktop → *Settings* → *Resources → WSL Integration*.

Enabled integration for Ubuntu so Docker could use the Linux kernel for running containers.

1. Setting up the Project Folder
   * Created a folder containing three files:

sample\_app.py – a simple Python script using the Pathway library.

requirements.txt – listing dependencies.

Dockerfile – containing build instructions for Docker.

1. Building and Running the Container

Opened Ubuntu shell from PowerShell using:

wsl -d Ubuntu

Navigated to the project folder and built the Docker image:

docker build -t my-pathway-app .

Ran the container:

docker run --rm my-pathway-app

cat output.jsonl

This successfully executed the sample Pathway program inside the container, confirming that Docker was properly set up.

**Task 4 – Financial Support Chatbot using Hugging Face APIs**

I decided to build the chatbot using LangChain and Hugging Face Inference API using Mistral-7B-Instruct model because I had recently learnt Langchain and created a simple personal project on Youtube Videos Summarizer and Questioning Chatbot for the Bajaj Finserv hackathon I participated it.

The chatbot is designed to mimic a real customer support assistant by maintaining accuracy.

**Steps Followed**

1. Hugging Face API Setup

Installed the required libraries:

pip install langchain langchain-huggingface langchain-community huggingface\_hub

Generated an API key from Hugging Face and stored it securely using:

os.environ["HUGGINGFACEHUB\_API\_TOKEN"] = "your\_token"

Integrated the model mistralai/Mistral-7B-Instruct-v0.2 using HuggingFaceEndpoint.

2. UserProfile Class

This class was designed to:

Extract user details using regular expressions (email, phone number, account number, name).

Maintain a structured dictionary with all user information.

Save queries and responses with timestamps.

Save the complete profile to a JSON file (user\_profile.json).

3. Financial Support Bot Class

It manages:

The conversation flow using LangChain’s ConversationBufferMemory.

A prompt template defining the assistant’s behavior and tone.

The process\_input method that: extracts user details, generates responses from the Hugging Face model, stores conversation history.

4. Chat Interface

A simple command-line interface was built where the user can:

Type messages and receive responses.

Type profile to see stored details.

Type save to save user info to a JSON file.

Type quit to exit the chat.

Example Interaction

You: Hello, I'm Binod. You can reach me at binod@gmail.com or 789067854.

Bot: Hello Binod, I'm happy to help answer any questions you have about our loan products...

You: My account number is 9988776655. Can you tell me what documents I need to update my address?

Bot: You’ll need a valid proof of address such as a utility bill or government ID...

You: profile

Stored User Information:

Email: binod@gmail.com

Phone: 9988776655