Phase 1 – Design Document: SEQATO LLM Awareness Program

# General Info

Participant: Anju Prasannan  
Program: SEQATO LLM Awareness and Portfolio Development Program  
Phase: Phase 1 – Mini Projects  
Period: June 2025  
Projects Completed:  
1. Local LLM Chat App using Streamlit + FastAPI + Ollama  
2. PDF Document Summarization using Streamlit + Hugging Face Transformers

# Project 1: Local LLM Chat App

## Objective

Build a local chatbot using an open-source LLM via Ollama with a frontend (Streamlit) and backend (FastAPI).

## Architecture Overview

[User Input in Streamlit]  
 ↓  
 [FastAPI Server]  
 ↓  
[Ollama API (localhost:11434)]  
 ↓  
 [Streamlit displays LLM reply]

## Tech Stack

Frontend: Streamlit  
Backend: FastAPI  
Model Engine: Ollama  
LLM Model: LLaMA 3 (run locally)  
Language: Python 3.10+

## Functional Flow

1. User enters a message in Streamlit  
2. Streamlit sends POST request to FastAPI backend  
3. Backend sends prompt to Ollama (/api/generate)  
4. Ollama generates LLM output  
5. Backend returns the response to Streamlit  
6. UI displays the conversation

## Design Decisions

- Used llama3 model via Ollama to avoid OpenAI keys and ensure offline usage.  
- Used FastAPI to separate logic cleanly from UI.  
- Streamlit chosen for rapid prototyping.

## Known Limitations

- Works in CPU mode (slow responses)  
- Not suitable for large-scale deployment  
- No conversation memory yet

## Possible Improvements

- Add memory (conversation context)  
- Add support for multiple models (model switcher)  
- Dockerize for easy deployment

# Project 2: PDF Document Summarizer

## Objective

Allow users to upload a PDF file, extract the text, and generate a concise summary using Hugging Face LLMs.

## Architecture Overview

[User uploads PDF in Streamlit]  
 ↓  
 [PyMuPDF extracts full text]  
 ↓  
[Text is chunked if needed]  
 ↓  
[Summarizer (Hugging Face BART)]  
 ↓  
 [Streamlit displays summary]

## Tech Stack

Frontend: Streamlit  
Text Extraction: PyMuPDF  
Model Engine: Hugging Face Transformers  
Summarization Model: facebook/bart-large-cnn  
Language: Python 3.10+

## Functional Flow

1. User uploads a .pdf file in Streamlit  
2. App uses fitz (PyMuPDF) to extract text  
3. Long text is broken into chunks (1,000 chars max)  
4. Each chunk is sent to the summarizer pipeline  
5. Summaries are combined and displayed

## Design Decisions

- Used facebook/bart-large-cnn for higher-quality summaries  
- Chunked input to avoid token limit errors  
- Kept the interface simple for demo purposes

## Known Limitations

- Summarization might be inconsistent across long documents  
- BART may truncate or hallucinate content  
- No download/export of the summary

## Possible Improvements

- Add export to .txt or .pdf  
- Allow model switching (e.g., T5, GPT-2)  
- Add multi-language support

# Summary

Feature | Chat App | PDF Summarizer   
-----------------------------|----------------|------------------  
Model Used | LLaMA 3 (local) | BART (Hugging Face)  
User Interface | Streamlit | Streamlit   
Model Backend | Ollama (FastAPI)| Hugging Face Pipeline  
Input Type | User prompt | PDF file   
Output | Text reply | Document summary   
Deployment Mode | Local | Local

# Files & Repo

GitHub Repo: https://github.com/anjuprasannan96/llm-awareness-portfolio  
Folder structure:  
llm-awareness-portfolio/  
├── llm-local-chat-app/  
├── document-summarizer/  
├── learning\_log.md  
└── Phase1-Design-Document.md