Option 2: Take-Home Challenge
Creating GenAI-Powered Claim Approval Agent

Objective

Build a prototype that:

- predicts insurance claim approval status from structured claim data,

and extensively uses Generative AI for:

- multi-persona explanations for claim decisions.
- [optional] targeted synthetic claim scenarios for model improvement.

Note: It is **NOT required** for you to build a highly accurate ML model or GenAI solution but consider demonstrating the thought process around hyperparameter tuning and prompt/context engineering or fine-tuning. The solution should be designed for **cloud deployment** with **MLOps/LLMOps** considerations.
Any justification or writeup within a notebook or script is more than welcomed. State any assumptions or alternative tools used in their README.md and design document.

Problem Statement

Insurance companies need efficient, consistent, and transparent claim approval. This challenge focuses on building an AI-driven solution that predicts claim approval, provides detailed, context-aware explanations using GenAI, and (optionally) generates targeted synthetic data to enhance model robustness.

Candidate Deliverables

1. **Codebase:** A runnable Python application demonstrating the core ML and GenAI functionalities, and a basic API.
2. **Design Document:** A concise document covering your architecture, implementation, deployment, and evaluation plans, with clear justifications.
3. **On the day:** Present a demo, the design document or findings to demonstrate your understanding and learning.

Technical Requirements

1. Claim Approval Prediction (ML Modelling)

Task: Using a provided mock dataset of structured insurance claim data, build a machine learning model to predict claim approval.

- Perform essential data preprocessing and feature engineering.
- Train and evaluate an ML model.

2. Advanced Generative AI (GenAI)

Task: Implement the following two GenAI functionalities:

- **Multi-Persona Claim Decision Explanation:** For a given claim and the ML model's prediction, use an LLM to generate plain-language explanations tailored for *at least two distinct personas* (e.g., customer, claims adjuster). The explanation should highlight key contributing factors and provide actionable insights. Guide: What is your prompt engineering strategy for achieving persona-specific outputs?

- **[optional] Targeted Synthetic Claim Scenario Generation:** Use an LLM to generate new, realistic synthetic claim scenarios (structured data + narrative). Focus on generating data for *underrepresented denial patterns* or *borderline cases* that challenge the ML model. Guide: how you would prompt the LLM to generate these specific types of scenarios?

Assumptions for LLM: You may use a publicly available LLM API (e.g., Hugging Face, OpenAI). Clearly state any services used.

Note: for section 3 and 4, if cloud access is not available, a **robust local implementation** demonstrating these concepts **is acceptable** (e.g., using Flask/FastAPI for API, local logging, and version control for models/prompts using MLFlow or similar).

3. Deployment & MLOps/LLMOps
Task: Design and implement a prototype of a high-level cloud deployment strategy (AWS preferred) for your integrated solution.

- Develop a basic RESTful API that exposes the ML prediction and GenAI functionalities.
- Outline the choice of AWS compute, storage, and networking services. **Justify your choices considering cost, scalability, and MLOps/LLMOps requirements (e.g., managing model versions, LLM inference costs, data pipelines).**
- Briefly describe how MLOps/LLMOps principles (e.g., IaC, CI/CD for model/prompt updates, monitoring model drift and LLM output quality) would be applied to ensure robust and maintainable operations, **and if possible, demonstrate a basic implementation (e.g., using a simple CI/CD pipeline for model/prompt updates, or a basic monitoring setup).**

4. Evaluation & Monitoring
Task: Design and implement a concise plan for evaluating and monitoring the system in production.

- Define key performance metrics for both the ML approval model and the GenAI outputs (explanations, synthetic data).
- Briefly discuss how you would monitor the system's health, performance, and output quality (e.g., for drift, hallucinations) and ensure responsible AI practices (e.g., fairness, transparency) are maintained.

Submission Guidelines

- **Code:** Provide a link to a GitHub repository with a clear README.md.
- **Design Document:** A concise PDF or Markdown document covering the above points with clear justifications.