# LAB-2

## Pair_Programming_Team_25_Lab_2

### PART-1: MultiModal Retrieval-Augmented Generation

We're building a MultiModal Retrieval-Augmented Generation system to answer economic queries from PDF. Our goal is to provide accurate text answers and relevant visuals efficiently.

Below is our system breakdown:

**Step 1: Data Processing & Storage**

We process PDFs to:

- Extract text chunks using PyMuPDF (fitz). We split text into chunks with RecursiveCharacterTextSplitter and tag each with page numbers.
- Extract images (figures and tables) along with captions using PyMuPDF. We store this data as image embeddings with metadata.

We then generate embeddings:

- Text embeddings via sentence-transformers/all-mpnet-base-v2.
- Image embeddings using OpenAI's CLIP (clip-vit-base-patch32). We combine these with caption embeddings (70% visual, 30% textual).

Embeddings are stored in ChromaDB for fast retrieval.

**Step 2: Retrieval with Indexing & Ranking**

For incoming queries:

- Queries are encoded into embeddings.
- We retrieve top matches from ChromaDB using cosine similarity.
- Text relevance is refined through keyword-based reranking.

**Step 3: Response Generation**

Using retrieved data, we:

- Create prompts from text excerpts and image captions.
- Generate answers using either simulated methods or OpenAI's GPT-4 Turbo.
- Reference figures or tables explicitly if relevant.

**Step 4: Evaluation with BERTScore**

We evaluate responses using BERTScore to measure semantic relevance against reference responses.

**Step 5: CSV Submission**

We save our answers for the 11 questions in the required CSV format.

**Problem**

Currently, there's an issue referencing images correctly. The 'Image' column always shows 0. We'll need to debug image retrieval and embedding integration.

## PART-2: Fine-Tuning  Stable Diffusion Model for Image Generation

Goal: To fine-tune a stable diffusion model using a custom dataset and evaluate the image generation quality using Inception Score and CLIP similarity Score.

In our case above the dataset we took is '*Anime Face Dataset*' from Kaggle (https://www.kaggle.com/datasets/splcher/animefacedataset). We have done the fine-tuning based on this dataset.

### Dataset Preparation & Preprocessing

- Resized all images to 512X512 resolution.
- Applied image normalization compatible with the Stable Diffusion base model.
- Used dataset of over 2000 images with paired captions stored in captions_2k.txt.

### Model Fine-Tuning

- Base Model: Stable Diffusion v1.5
- Fine-Tuning Method: LoRA (Low-Rank Adaptation).
- Trained using Diffusers library on an NVIDIA RTX 4070 GPU.
- Training parameters:
    1. Batch Size: 4
    2. Learning Rate: $1e^{-4}$
    3. Epochs: 3
    4. Resolution: 512X512
- Saved output LoRA weights to the folder '*fine_tuned_lora_weights*' as 'pytorch_lora_weights.safetensors'.

### Model Checkpoints and Directory Structure

The fine-tuned Stable Diffusion model is organized as follows:

- Base Model Components (*fine_tuned_stable_diffusion/*)
- *feature_extractor/:* Stores image preprocessing logic.
- *scheduler/:* Manages noise schedule for diffusion timesteps.
- *text_encoder/:* Contains CLIP encoder and config.
- *tokenizer/:* Includes vocabulary files and special tokens.
- *unet/:* Core diffusion architecture responsible for denoising.
- *vae/:* Handles latent space encoding and decoding.
- *model_index.json:* Maps model components for loading with Hugging Face diffusers.

**Text-to-Image Generation**

Generated images using the following prompts:

1. Anime girl with long silver hair and green eyes, standing in a school classroom, wearing a school uniform, holding a notebook, with a chalkboard in the background
2. Anime boy with short black hair and blue eyes, standing in a school hallway, wearing a school uniform, with lockers in the background
3. Anime girl with long brown hair, wearing a traditional school uniform, walking on a school campus, with cherry blossom trees in the background
4. Anime boy with short blonde hair, wearing a school uniform, sitting at a school desk, reading a textbook, with a window showing a sunny day
5. Anime girl with short pink hair, wearing a school uniform, standing in a school library, holding a book, with bookshelves and a window in the background

The output images are stored in the folder '*generated_images*'.

**Evaluation**

| Metric | Score |
|---|---|
| CLIP similarity Score | 0.39 |
| Inception Score | 1.00 |

- CLIP Similarity Score: Score of 0.39 suggests moderate alignment.
- Inception Score: Score of 1.00 indicates low diversity.

**Conclusion**

The fine-tuned model demonstrated excellent performance on creative text prompts. LoRA enabled efficient fine-tuning with limited compute while retaining high fidelity and prompt adherence. This process can be easily replicated across domains by simply swapping the training dataset.

**NOTE:**

We were not able to upload our model weights to github because our files were over 1.5gb in size. So we have provided a weights.txt file with a google drive link with our

## PART-3: Developing an Agentic AI Travel Assistant

Goal: To develop an AI travel assistant by implementing autonomous agents that interact with external APIs to provide comprehensive travel planning experience.

### Agent Description and Responsibilities

1. Flight API Agent
    - Integrated the Amadeus API to search and retrieve the flight information between the said 2 cities.
    - Converts the city names to valis IATA codes using Amadeus's location search endpoint.
    - Provides the following
      (i) Airline Name
      (ii) Flight Duration
      (iii) Departure and arrival times
      (iv) Total price

2. Weather API Agent
    - OpenWeatherMap API is used to fetch multi-day weather forecasts for the destination.
    - Provides the following
      (i) Minimum and maximum temperature for the day
      (ii) Weather Condition
      (iii) On which specific days

3. Hotel API Agent
    - RapidAPI to retrieve hotel recommendations to at our travel destination.
    - Simplified destination names to improve search results
    - Provides the following
      (i) Hotel name and rating
      (ii) Rating
      (iii) Price
      (iv) Address

4. Itinerary Planner
    - Central orchestrator to call the Flight, Weather and hotel agents.

- Aggregated data and formatted a complete travel itinerary containing:
  (i) Flights overview
  (ii) Daily Weather Forecast
  (iii) Hotel recommendations

**System Flow**

We give a query to our agents and in return we get a proper formatted output.

Input: Query

- Consists of departure and arrival dates and destination.

```
query = f"I want to travel from San Francisco to Hyderabad from {future_date1} to {future_date2} with 2 adults"
```

Output: Formatted result

- Lists both the locations
- Lists Number of Travelers
- Lists Departure and return dates
- Return Flights options (as stated in Flights API)
- Returns Weather of the destination (as stated in Weather API)
- Returns the best hotels (as stated in Hotels API)

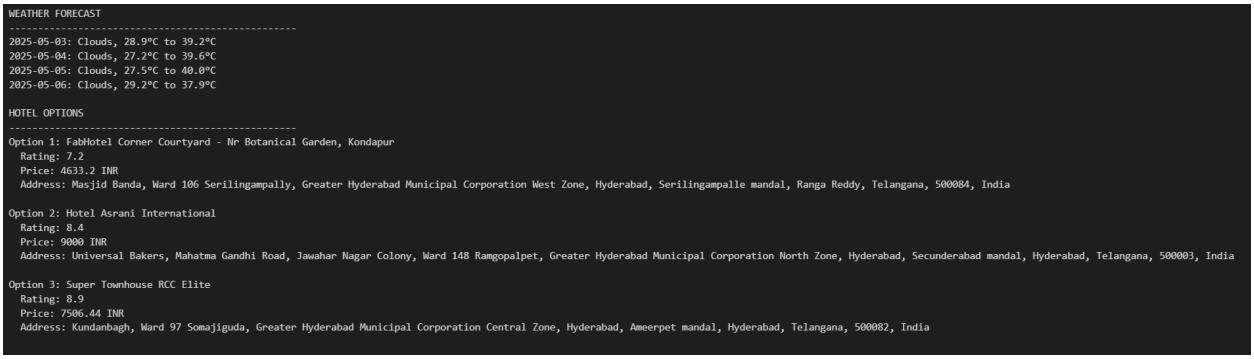
The final output format is as shown in the screenshots below.

```
TRAVEL ITINERARY: San Francisco to Hyderabad
==============================================
Travelers: 2 adults
Departure: 2025-05-03
Return: 2025-05-06

FLIGHT OPTIONS
----------------------------------------------
Option 1: 1936.54 USD
  Segment 1: AI 184
    SFO → DEL
    2025-05-03T22:00:00 → 2025-05-05T02:30:00
    Duration: 16h
  Segment 2: AI 2829
    DEL → HYD
    2025-05-05T06:10:00 → 2025-05-05T08:25:00
    Duration: 2h 15m
  Segment 3: AI 9569
    HYD → BLR
    2025-05-06T23:00:00 → 2025-05-07T00:15:00
    Duration: 1h 15m
  Segment 4: AI 175
    BLR → SFO
    2025-05-07T12:50:00 → 2025-05-07T17:30:00
    Duration: 17h 10m

Option 2: 1936.54 USD
  Segment 1: AI 184
    SFO → DEL
    2025-05-03T22:00:00 → 2025-05-05T02:30:00
    Duration: 16h
  Segment 2: AI 2560
    DEL → HYD
    2025-05-05T07:10:00 → 2025-05-05T09:30:00
    Duration: 2h 20m
  Segment 3: AI 9569
    HYD → BLR
    2025-05-06T23:00:00 → 2025-05-07T00:15:00
    Duration: 1h 15m
  Segment 4: AI 175
    BLR → SFO
    2025-05-07T12:50:00 → 2025-05-07T17:30:00
    Duration: 17h 10m

Option 3: 1936.54 USD
  Segment 1: AI 174
    SFO → DEL
    2025-05-03T10:00:00 → 2025-05-04T15:45:00
    Duration: 17h 15m
  Segment 2: AI 2463
    DEL → HYD
    2025-05-04T19:30:00 → 2025-05-04T21:50:00
    Duration: 2h 20m
  Segment 3: AI 9569
    HYD → BLR
    2025-05-06T23:00:00 → 2025-05-07T00:15:00
    Duration: 1h 15m
  Segment 4: AI 175
    BLR → SFO
    2025-05-07T12:50:00 → 2025-05-07T17:30:00
    Duration: 17h 10m
```




**Limitations**

1. Flights API:
    - We have tried 2 different API's for flights but its extremely hard to find a free one we chose Amadeus because it is free and also because it provides us with the most amount of flight options.
    - Northern Europe (Nordic Countries), Countries in political conflict (such as Ukraine and Russia) and east African countries still don't work as well with this API in our testing but for most part it works just fine.
    - As you can see from our other prompts for some cases the Flights is not being displayed and this is the reason.

2. Weather API:
    - The main issue is the free version can only hold weather information about a particular place for only upto 5 days.
    - For example if today is May 1$^{st}$ then it can only hold weather information from May 1$^{st}$ to May 6$^{th}$.
    - As you can see from our other prompts for some cases the Weather is not being displayed and this is the reason.

**Conclusion**

The AI travel assistant effectively demonstrates how modular autonomous agents can cooperate to deliver useful real-world travel services using live data from external APIs.