Course: CSE299 | **Section**: 05 | **Group**: 06

Project Title: **Unified CLI-based Codebase for Running Multiple Generative AI Models.**

Submitted To: M. Shifat-E-Rabbi(MSRB)

Team Members

Name	ID
2212079042	ARAF TAHSAN PAVEL
2211935042	MD TANVEER HOSSAIN NIHAL
2212108042	ISMOT ARA EMU

1. Project Title

Unified CLI-based Framework for Running Multiple Generative AI Models

2. Objective

The objective of this project is to develop a Python-based command-line interface (CLI) tool that allows users to easily run and experiment with multiple popular generative AI models using a single unified framework. This framework will accept datasets and model specifications as input, run the selected model on the dataset, and output results in an organized manner. The goal is to simplify the usage of complex AI models, making them accessible for students and researchers for quick testing and comparison.

3. Motivation

Generative AI models have become highly influential in areas such as text generation, speech synthesis, and image/audio generation. However, these models often come with distinct

APIs, dependencies, and usage workflows, which can create barriers for learners and researchers who want to experiment with multiple models efficiently.

Our motivation is to streamline the process by providing a unified CLI tool that manages different models under one roof. This will reduce setup time and technical complexity, encouraging wider exploration of generative AI techniques in academic and project environments.

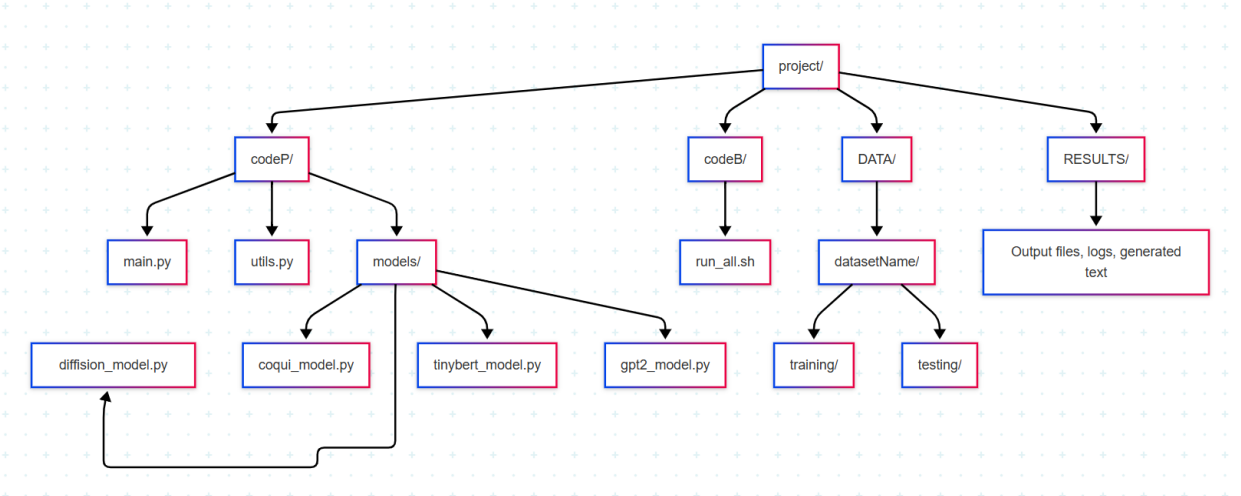
4. Scope

- Support for running **at least four** generative AI models:
  - Diffusion model (for image/audio generation)
  - Coqui TTS (text-to-speech synthesis)
  - TinyBERT (lightweight NLP tasks)
  - GPT-2 (text generation)
- Provide a **unified command-line interface** for users to specify datasets, select models, and set task-specific options.

- Modular system architecture to facilitate **easy addition of new models** in the future.
- Organized folder structure:
    - Input datasets stored in DATA/
    - Model outputs and logs saved in RESULTS/

5.Proposed System Architecture

The system consists of a central command-line program that interprets user commands and dispatches requests to individual model modules. Each model is implemented as a separate Python file or module that adheres to a common interface for input and output handling.



6. Technologies Used

- **Programming Language:** Python 3.x
- **Libraries and Frameworks:**
  - transformers (Hugging Face) for GPT-2 and TinyBERT
  - coqpit and Coqui TTS for speech synthesis
  - torch and torchaudio for model inference and audio handling
  - Custom code for Diffusion model implementation
- **Operating System Compatibility:** Windows and Linux
- **Shell scripting:** Bash scripts to automate running multiple commands and testing

7. Dataset Structure

The dataset folder structure will be standardized as:

bash

CopyEdit

DATA/

└── datasetName/

├── training/   # Training input data for models that require it

└── testing/   # Testing or inference input data

Datasets can be text files, audio files, or image files, depending on the model requirements. The system will be flexible enough to support different data formats, with pre-processing modules added as needed.

8. Model Descriptions

Model	Purpose	Implementation Details
Diffusion	Image/audio generation	Custom or simplified implementation
Coqui TTS	Text-to-speech synthesis	Based on Coqui AI open-source tools
TinyBERT	Lightweight NLP tasks (e.g., classification, embedding)	Hugging Face Transformers library

Model	Purpose	Implementation Details
GPT-2	Text generation	Hugging Face Transformers library

10. Work Distribution

Team Member	Responsibilities
MD Tanveer Hossain Nihal(2211935042)	• Develop the main CLI handler (main.py) that parses user commands and directs model calls. • Implement the Diffusion model and ensure it integrates well with the framework.

	• Manage the overall project folder structure and ensure results are saved properly. • Perform system integration testing and resolve cross-model issues.
Araf Tahsan Pavel (2212079042)	• Implement GPT-2 and TinyBERT models using Hugging Face libraries. • Adapt these models to handle input data and output results within the folder structure. • Write code for data pre-processing and post-processing of model outputs. • Optimize model loading and inference for efficiency.
Ismot Ara Emu(2212108042)	• Implement Coqui TTS for text-to-speech capabilities.

	• Develop Bash scripts (run_all.sh) to automate running different models and datasets. • Format and organize outputs such as generated audio and logs. • Test batch runs and ensure user-friendly CLI experience.

11. Expected Outcome

- A fully functional command-line interface tool that runs four generative AI models from a single codebase.
- Well-organized input/output folder management.
- Modular and extensible codebase for easy addition of future models.
- Documentation and user guide for running the tool.

12. Challenges and Risks

- Different models have varying input/output formats and dependencies which require careful handling and abstraction.
- Some models (e.g., GPT-2) are large and may need significant computational resources.
- Ensuring smooth integration so that the CLI tool can manage all models without errors.
- Handling user errors or invalid inputs gracefully.

13. Conclusion

This project aims to deliver a unified, modular platform that simplifies running and experimenting with multiple generative AI models. It will help learners and researchers save time and effort, promote understanding of different model types, and provide a foundation for further model integration and expansion.