AI Meme Generator

1. Principal Investigator

Name: *Sonal Kamble*

Email: ***kambles2@newpaltz.edu***

2. Introduction

2.1 Project Motivation

- *Memes and GIFs have become a primary form of communication on social media platforms. They allow users to express emotions, reactions, and humor more effectively than plain text. However, finding a suitable meme often requires manual searching and browsing, which can be time-consuming and inconsistent.*
- *The motivation behind this project was to design an AI-based system that can automatically recommend an appropriate meme based on a user's textual input. The goal was not to build a complex machine-learning model, but rather to focus on explainable AI techniques that demonstrate how decision-making can be modelled using rules, weighted logic, and contextual reasoning.*
- *This project has real-world relevance in applications such as social media platforms, chat applications, and content recommendation systems where fast and context-aware responses are valuable.*

2.2 Aims and Objectives

- The primary objectives of this project are:
    - To design an AI system that maps user input text to a relevant meme or GIF.
    - To use explainable, rule-based AI logic instead of black-box models.
    - To demonstrate contextual reasoning using keyword extraction and weighted scoring.
    - To create a simple and interactive user interface.
- The expected deliverables include:
    - A fully functional Java-based application with a graphical user interface.
    - A curated dataset of memes with associated weighted keywords.
    - A working inference mechanism that selects memes based on input context.
- The impact of this project lies in demonstrating how basic AI reasoning techniques can be applied effectively without relying on complex machine-learning frameworks.

3. Approach and Implementation

3.1 System Architecture Overview

- The system follows a modular pipeline architecture:

  1. **User Input Module:** Accepts a sentence from the user.
  2. **Keyword Processing Module:** Extracts relevant keywords from the input.
  3. **Inference Engine:** Compares extracted keywords against a dataset of memes using weighted matching.
  4. **Decision Module:** Selects the meme with the highest cumulative score.
  5. **Output Module:** Displays the selected GIF in the application window.

This architecture ensures clarity, modularity, and ease of extension.

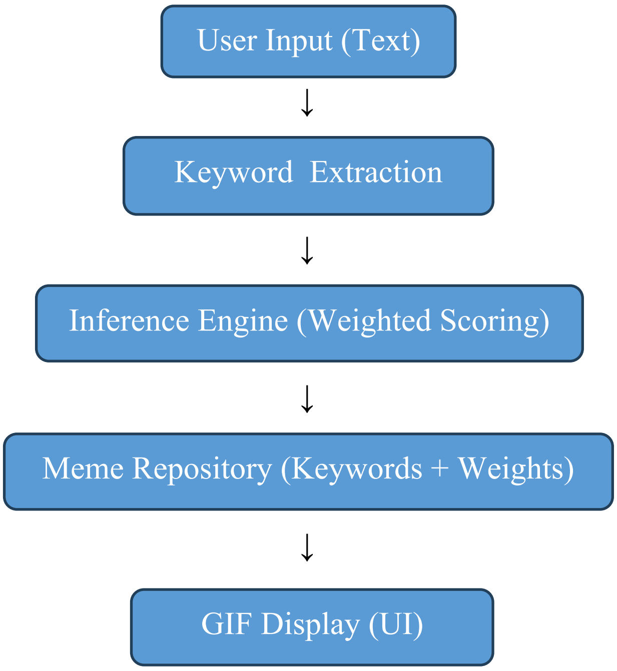
User Input (Text)
Keyword Extraction
Inference Engine (Weighted Scoring)
Meme Repository (Keywords + Weights)
GIF Display (UI)

Figure 1

Figure 1 illustrates the overall architecture of the AI Meme Generator system.
The system consists of five main components:
(1) User Interface,
(2) Input Processing Module,
(3) Keyword Matching and Inference Engine,
(4) Meme Repository, and
(5) Output Display Module.

User input flows from the interface into the inference engine, where it is analyzed and compared against the meme dataset. The meme with the highest weighted score is then selected and displayed as a GIF in the interface.

3.2 Tools and Technologies

List and justify the tools used for development.

- **Programming Languages:** Java
- **Frameworks:** Java Swing
- **Data Structures:** HashMap, Map (for efficient keyword lookup)
- **Development Tools:** IntelliJ IDEA, GitHub

Java was chosen due to familiarity and strong support for UI development and data structures. Java Swing was used to provide a lightweight graphical interface.

3.3 Implementation Strategy

1. Setup and Initial Design

The project began with requirement analysis and planning. The scope was intentionally limited to ensure a complete and functional system within the course timeline.

2. Module-wise Development

- Meme data stored as objects containing GIF URLs and keyword-weight mappings.
- Keyword matching implemented using Map-based scoring.
- UI developed using Swing components.

3. Integration of Components

User input flows through the keyword extractor, inference engine, and output display seamlessly. All components communicate through well-defined interfaces.

4. Testing and Deployment

The application is tested locally with multiple inputs to verify correct meme selection. Deployment is local, focusing on correctness and functionality rather than scalability.

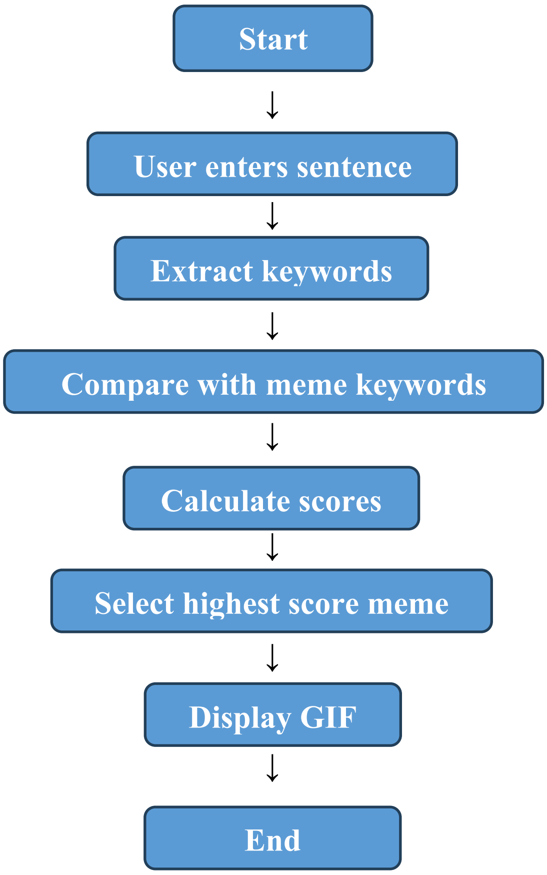

Figure 2

Figure 2 shows the execution flow of the system during runtime.

When a user submits a sentence, the system tokenizes the input, extracts meaningful keywords, and compares them with keyword-weight mappings stored in the meme repository. Scores are accumulated for each meme, and the meme with the highest score is selected as output.

4. Evaluation and Test Cases

4.1 Test Cases

Representative test cases include:

- **Input:** “I’m stressed about deadlines”
  Output: Michael Scott “No God Please No” GIF
- **Input:** “My experiment finally worked”
  Output: Jesse Pinkman “Yeah Science” GIF
- **Input:** “My classmates reacting to my presentation”
  Output: Jim Halpert awkward look GIF

These cases confirm correct contextual matching.

4.2 Performance Evaluation

- **Execution Time:** Meme selection occurs in constant time relative to dataset size due to map-based lookup.
- **Accuracy:** Accuracy is evaluated qualitatively based on relevance of output to input.
- **Load Handling:** Not applicable, as the system is single-user and locally deployed.

Input Keywords

↓

For each Meme:

→ Check keyword match

→ Add weights

→ Store total score

↓

Select Meme with Max Score

Figure 3

Figure 3 illustrates the meme selection logic used by the inference engine. Each meme is associated with a set of keywords and corresponding weights. If a keyword from the user input matches a meme keyword, its weight is added to the meme’s total score. The meme with the maximum score is selected as the final recommendation.

5. Critical Analysis of the Project Outcome

5.1 Successes

- Fully working AI pipeline from input to output.
- Clear and explainable inference mechanism.
- Successful live demonstration during presentation.
- High user engagement due to humor and relatability.

5.2 Challenges and Solutions

- **Challenge:** Overlapping keywords between memes.
  Solution: Use weighted keyword scoring to prioritize stronger matches.
- **Challenge:** Displaying GIFs correctly in Java Swing.
  Solution: Implement proper scaling and URL-based loading.

5.3 Limitations

- Keyword-based approach lacks deep semantic understanding.
- Dataset size is limited and manually curated.
- No advanced NLP or grammar parsing implemented.

5.4 Future Work

- Expand the meme dataset significantly.
- Introduce grammar-based parsing to identify subject and sentiment.
- Integrate NLP libraries for better context understanding.
- Explore scalable storage and retrieval mechanisms.

6. References

- Russell, S., & Norvig, P. (2021). *Artificial Intelligence: A Modern Approach*. Pearson.
- Jurafsky, D., & Martin, J. (2023). *Speech and Language Processing*. Stanford University.
- Oracle Java Documentation. **https://docs.oracle.com**