Proposed Approach

The system will consist of:

1. Claim extraction
2. Evidence retrieval (Wikipedia-based)
3. Verification using:
    a. Semantic similarity
    b. Natural Language Inference (NLI)
4. Score aggregation
5. Evaluation and comparative analysis

System Architecture

1. User input (LLM generated text)
2. Claim Extraction Module
3. Evidence Retrieval Module (Wikipedia)
4. Verification Layer
    i. Semantic similarity
    ii. NLI model
5. Score Aggregation
6. Final Hallucination Decision

Core Modules to implement

1. Similarity-based detection
2. NLI-based detection
3. Combined hybrid method

Risk Management

Potential risks:

- Retrieval accuracy may be low
- NLI model may misclassify
- Runtime performance issues
- Integration complexity

Mitigation:

- Start early
- Keep modules independent
- Test weekly
- Keep baseline working version

Evaluation Criteria

Evaluation Metrics:

- Precision
- Recall
- F1-score
- Accuracy

Comparison Strategy

- Similarity-only
- NLI-only
- Hybrid method

Deliverables

Final deliverables:

- Working hallucination detection system
- Evaluation report
- Comparative analysis
- Presentation slides
- Web demo

Proposed Task Distribution

Shariq:

- Design full pipeline architecture
- Implement NLI verification module
- Design score aggregation logic
- Integrate retrieval + similarity + NLI

- Implement hybrid decision method
- Lead evaluation & comparison
- Final backend integration for UI

Fairooz:

- Wikipedia data fetching
- Text chunking & preprocessing
- Embedding generation (SentenceTransformers)
- FAISS indexing
- Top-k document retrieval
- Optimize retrieval performance
- Provide clean function API: retrieve_evidence(claim)

Uday

- Semantic similarity module
- Threshold tuning
- Implement evaluation metrics:
    - Precision
    - Recall
    - F1
    - Confusion matrix
- Results analysis
- Build Streamlit UI
- Connect UI to backend functions
- Format final output display

Weekly Checklist

~~Week 1~~

~~All members:~~

- ~~Install required libraries~~
- ~~Read about hallucination detection basics~~
- ~~Understand FEVER dataset format~~

~~Shariq:~~

- ~~Finalize architecture diagram~~
- ~~Design the pipeline~~

~~Fairooz:~~

- ~~Study FAISS basics~~
- ~~Study sentence embeddings~~

~~Uday:~~

- ~~Study cosine similarity~~
- ~~Study evaluation metrics (Precision, Recall, F1)~~

Week 2

~~Shariq:~~

- ~~Study NLI models~~
- ~~Test facebook/bart-large-mnli~~
- ~~Run sample entailment tests~~

Fairooz:

- Fetch sample Wikipedia pages
- Convert text to embeddings
- Build small FAISS test index

Uday:

- Download FEVER dataset
- Clean and preprocess claims
- Convert dataset into useable format

Week 3

Shariq:

- Define retrieval evaluation criteria
- Review module performance

Fairooz:

- Build complete FAISS index
- Implement top-k retrieval
- Test retrieval quality

Uday:

- Prepare similarity comparison pipeline

Week 4

Shariq:

- Begin integration of retrieval + similarity

Fairooz:

- Optimize retrieval accuracy

Uday:

- Compute cosine similarity
- Define similarity threshold
- Classify supported/hallucinated

Week 5

Shariq:

- Implement NLI verification
- Map entailment/contradiction to labels
- Validate predictions

Fairooz & Uday:

- Test NLI on sample claims
- Compare NLI vs similarity outputs

Week 6

All:

- Design aggregation strategy
- Implement final decision logic
- Run validation experiments

Week 7

Shariq:

- Compare:

    - Similarity only
    - NLI only
    - Combined

Fairooz:

- Measure retrieval coverage

Uday:

- Implement metrics calculation
- Generate confusion matrix

Week 8

All:

- Identify failure cases
- Analyze incorrect predictions
- Document limitations

Week 9

Shariq:

- Finalize hybrid decision logic
- Clean and modularize backend functions
- Test full pipeline locally (without UI)
- Debug any integration issues

Fairooz:

- Optimize retrieval speed
- Cache embeddings if necessary
- Ensure retrieval function returns structured output:
    - Retrieved evidence text
    - Retrieval score
- Fix edge cases

Uday:

- Build simple Streamlit app:

  UI Components:

- Text input box
- “Check Hallucination” button
- Output section:
  - Similarity score
  - NLI result
  - Final decision
  - Retrieved evidence
- Connect backend functions
- Format results clearly
- Add loading indicator
- Test multiple example inputs

Note:

Enter LLM Generated Text: [Text Box] [Check Hallucination]   Similarity Score: 0.72 NLI Result: Entailment Final Decision: Supported Retrieved Evidence: "Wikipedia excerpt here..."

Week 10

All:

- Create slides
- Prepare demo
- Practice presentation