UG PROJECT: LLM-Based Geotechnical Report & Boring Log Interpreter

1. Project Title

Automated Interpretation of Geotechnical Reports and Boring Logs using Large Language Models

2. Abstract

This project proposes a deployable Large Language Model (LLM) system that automatically interprets geotechnical reports, boring logs, and laboratory test documents to extract structured engineering parameters. The system reduces manual effort, improves consistency, and supports rapid decision-making in geotechnical design workflows.

3. Problem Statement

Geotechnical reports are largely unstructured documents containing critical soil parameters. Manual interpretation is time-consuming, error-prone, and not scalable. There is currently no automated, deployable AI system dedicated to understanding geotechnical documents.

4. Inspiration

From the recent India AI Summit, where a senior developer explained that India needs a greater variety of LLM applications in fields like civil, mechanical, and biotech. He addresses the necessity of LLM and RAG applications, which have huge potential and a lack of variety.

4. Objectives

- Automatically extract soil layers, SPT/CPT values, groundwater levels, and recommendations
- Use LLMs with domain-specific prompts
- Generate structured JSON/Excel outputs
- Deploy as a web-based tool

5. Literature Review

This section reviews:

- Machine learning in geotechnical engineering
- NLP and document intelligence systems
- Retrieval-Augmented Generation (RAG)

Gap: No geotechnical-domain-specific LLM systems exist for report interpretation.

6. Dataset Sources

-(**[Research Designs & Standards Organisation]** **(RDSO)**

-**National Institute of Rock Mechanics** **(NIRM)**

-**National Geoscience Data Repository** **(NGDR)**

- Academic case-study reports
- Consultant-style sample boring logs

Synthetic Data:
- AI-generated geotechnical reports for augmentation

7. System Architecture

PDF →

OCR/Text Extraction →

Chunking →

Embeddings →

Vector Database →

LLM →

Structured Output

8. Model Architecture

LLM Stack:
- Embeddings: Sentence-BERT / OpenAI embeddings
- Vector DB: FAISS
- LLM: GPT-style or LLaMA-based model

Prompting:
- Domain-specific extraction prompts
- JSON schema enforcement

9. Example Output Schema

```
{
  soil_layers: [{depth, soil_type, N_value}],
  groundwater_level,
  foundation_recommendations,
  allowable_bearing_capacity
}
```

10. Code Template (Core Logic)

Pseudo-code:

1. Load PDF
2. Extract text
3. Chunk text
4. Generate embeddings
5. Query LLM with prompt
6. Parse JSON output

11. Deployment Plan

Backend: FastAPI
Frontend: Streamlit
Cloud: AWS/GCP
User uploads report → receives structured output

12. Evaluation Metrics

- Parameter extraction accuracy
- Engineer validation score
- Processing time per report

13. Conclusion & Future Scope

Future enhancements include multilingual reports, integration with BIM software, and physics-informed validation of extracted parameters.