LECTUREMIND AI — AI-POWERED LEARNING COMPANION

Presented By:

UTTAM RAWAT

THDC-IHET ,TEHRI , UTTARAKHAND

B. Tech CSE (Ongoing 2023-2027).

edunet foundation

OUTLINE

- Problem Statement
- Proposed System/Solution
- System Development Approach
- Algorithm & Deployment
- Result
- Conclusion
- Future Scope
- References

PROBLEM STATEMENT

- Students struggle to convert long lecture audio into structured study material.
- Manual note-taking is time-consuming and error-prone.
- Existing tools are : Cloud-dependent.
- Costly (paid APIs)Limited to transcription only.
- Lack of interactive learning from lecture content.
- No unified system for notes, flashcards, quizzes, and Q&A.

PROPOSED SOLUTION

- End-to-end local-first web application powered by AI
- Automatic transcription, cleaning, and structured content generation
- RAG-based chatbot for intelligent Q&A with lectures
- No paid API required — fully open-source and self-hosted

SYSTEM DEVELOPMENT APPROACH

- Modular architecture: separate concerns (transcription, cleaning, generation, RAG)
- Streamlit for UI — rapid prototyping and user-friendly dashboard
- Local vector database (FAISS) for efficient retrieval
- Fallback heuristics for LLM functions when API unavailable

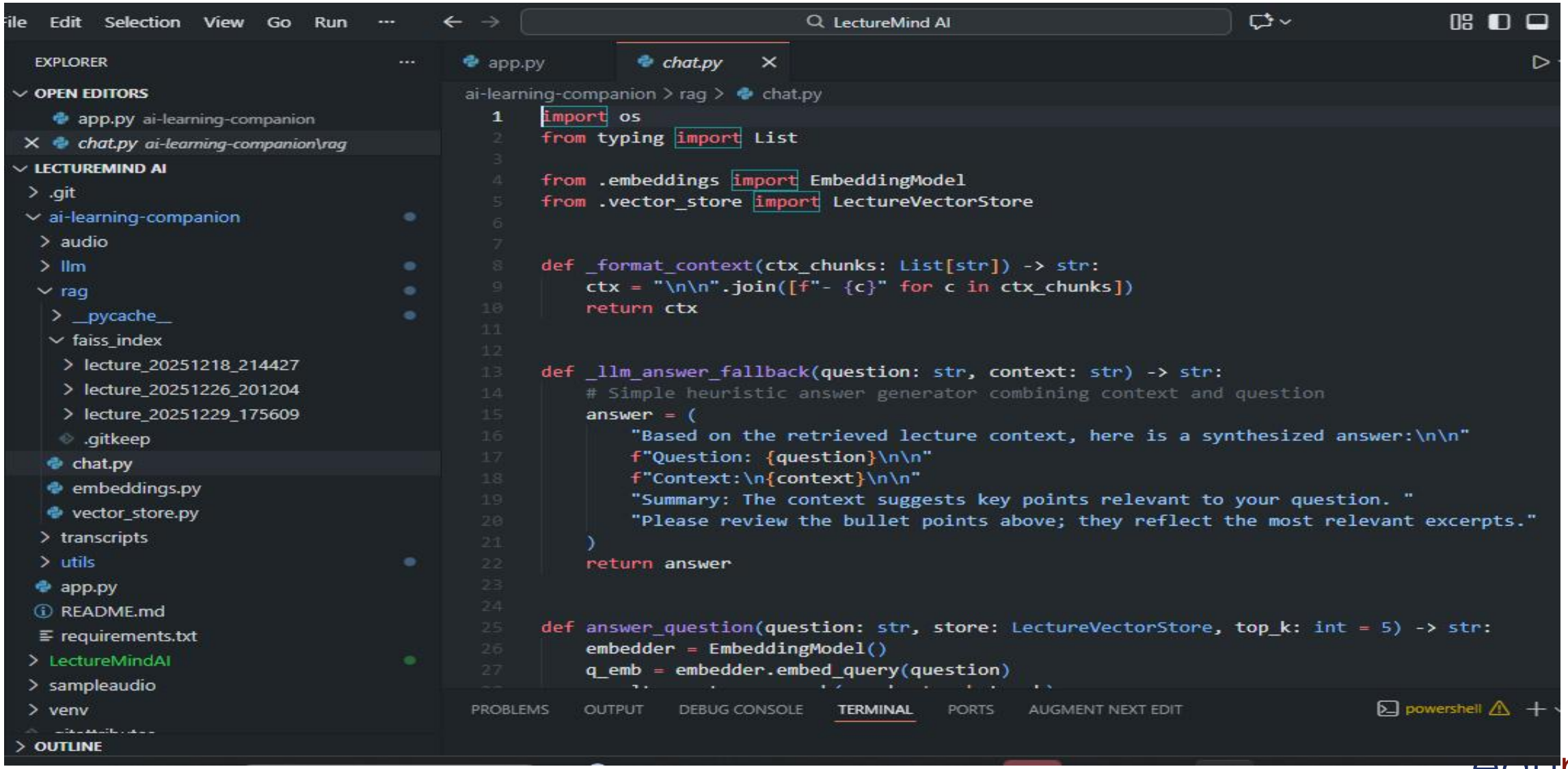
ALGORITHM & DEPLOYMENT

- **Whisper STT:** Convert audio to raw transcript
- **Text Cleaning:** Remove fillers, fix punctuation, split paragraphs
- **Content Generation:** Notes, flashcards, quizzes via LLM or heuristic

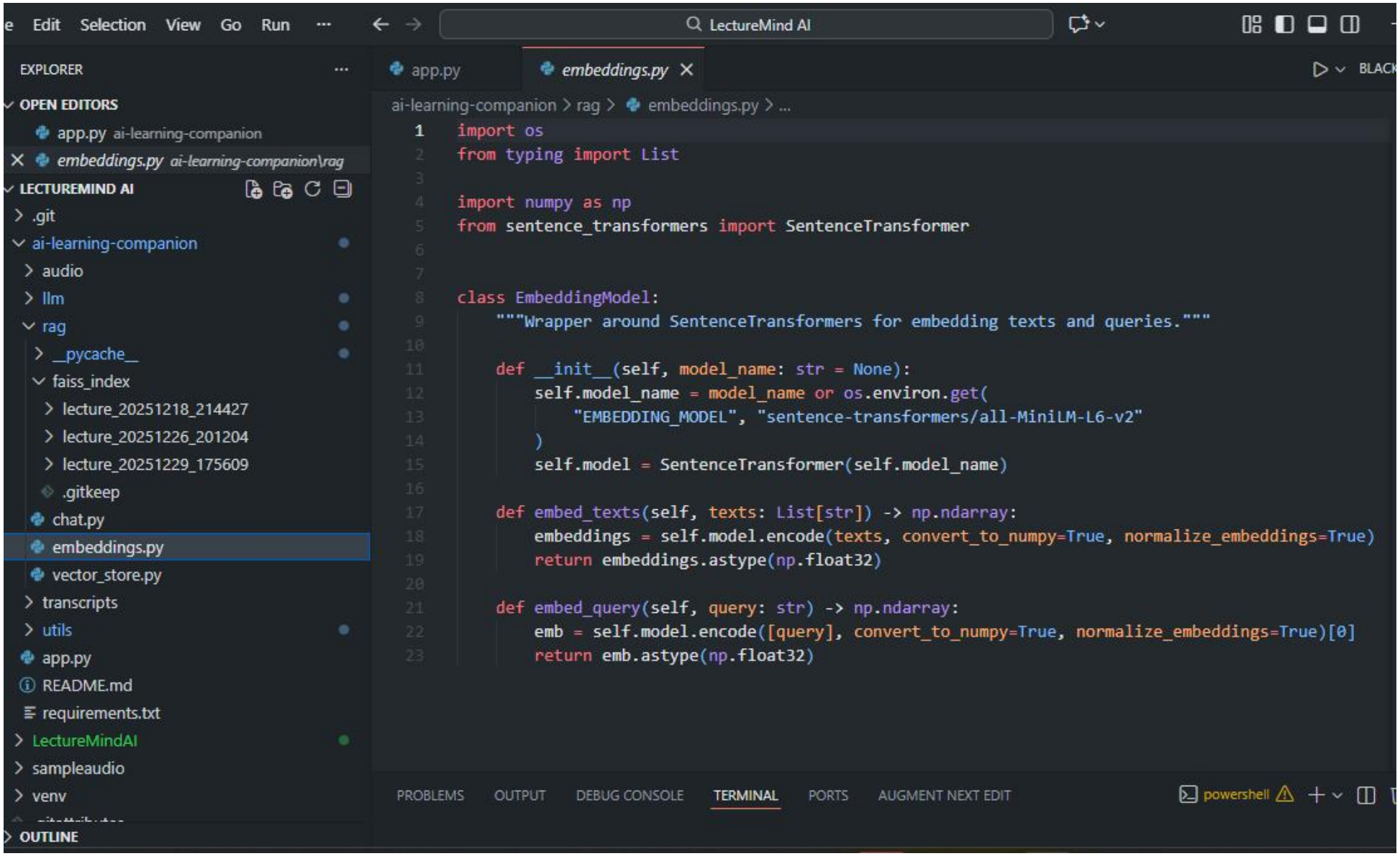
ALGORITHM & DEPLOYMENT

- **Embedding & Indexing:** SentenceTransformers + FAISS vector store
- **RAG Chat:** Retrieve chunks → Generate contextual answers
- **Deployment:** Run via Streamlit; no external service required

RESULTS

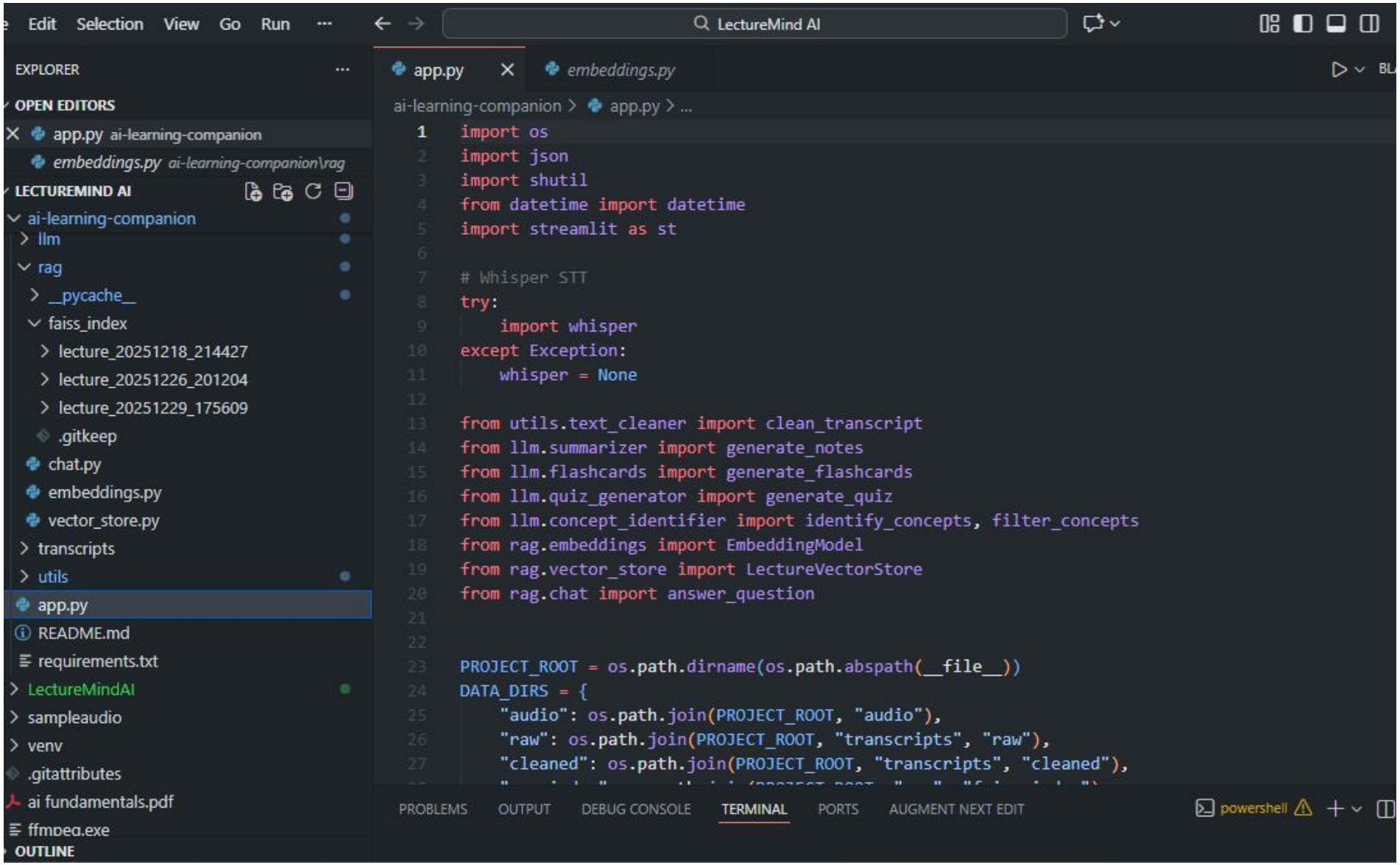

File Edit Selection View Go Run ···
LectureMind AI
EXPLORER
OPEN EDITORS
app.py ai-learning-companion
chat.py ai-learning-companion\rag
LECTUREMIND AI
.git
ai-learning-companion
audio
llm
rag
__pycache__
faiss_index
lecture_20251218_214427
lecture_20251226_201204
lecture_20251229_175609
.gitkeep
chat.py
embeddings.py
vector_store.py
transcripts
utils
app.py
README.md
requirements.txt
LectureMindAI
sampleaudio
venv
OUTLINE
app.py
chat.py
ai-learning-companion > rag > chat.py
import os
from typing import List
from .embeddings import EmbeddingModel
from .vector_store import LectureVectorStore
def _format_context(ctx_chunks: List[str]) -> str:
ctx = "\n\n".join([f"- {c}" for c in ctx_chunks])
return ctx
def _llm_answer_fallback(question: str, context: str) -> str:
# Simple heuristic answer generator combining context and question
answer = (
"Based on the retrieved lecture context, here is a synthesized answer:\n\n"
f"Question: {question}\n\n"
f"Context:\n{context}\n\n"
"Summary: The context suggests key points relevant to your question. "
"Please review the bullet points above; they reflect the most relevant excerpts."
)
return answer
def answer_question(question: str, store: LectureVectorStore, top_k: int = 5) -> str:
embedder = EmbeddingModel()
q_emb = embedder.embed_query(question)
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AUGMENT NEXT EDIT
powershell

RESULTS

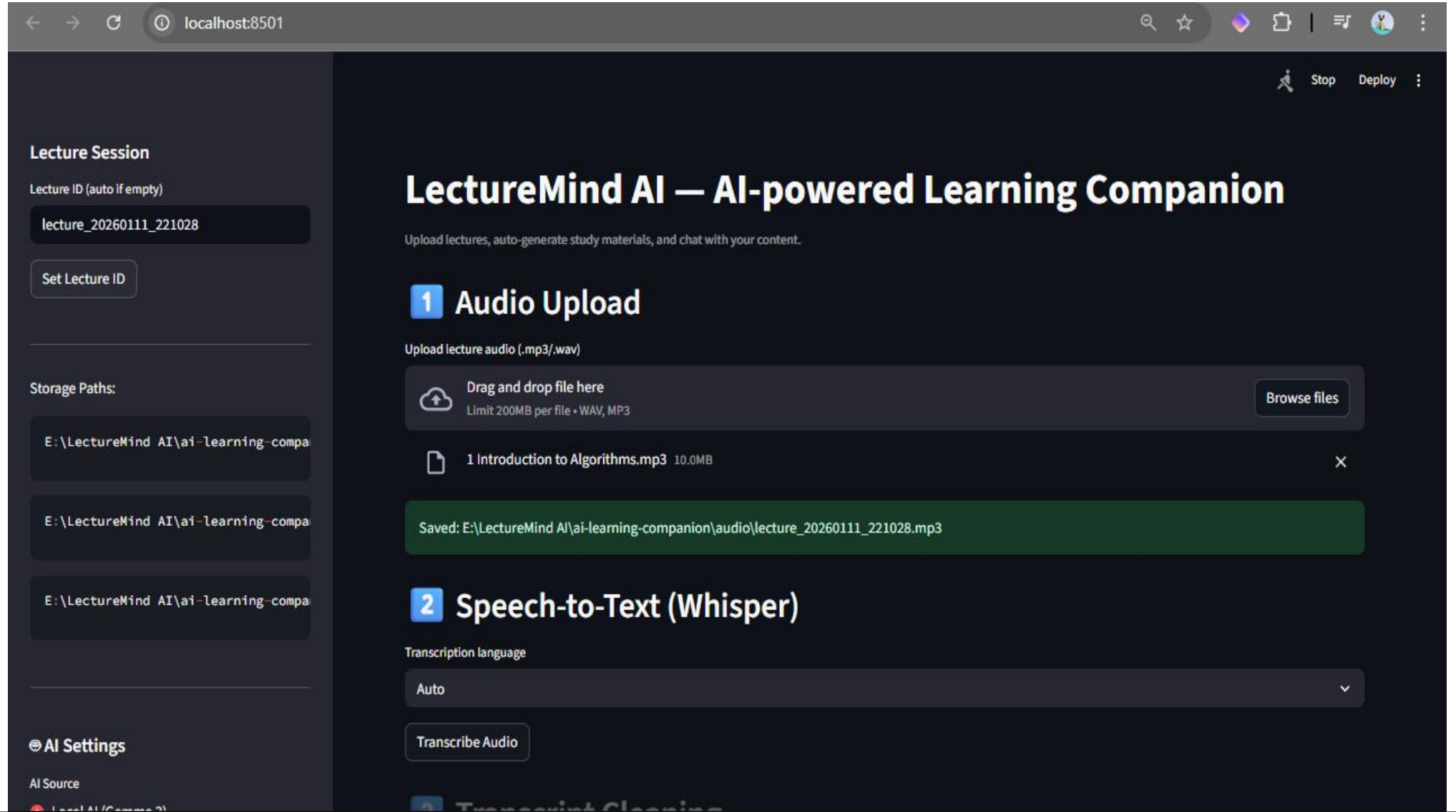


edunet foundation

RESULTS

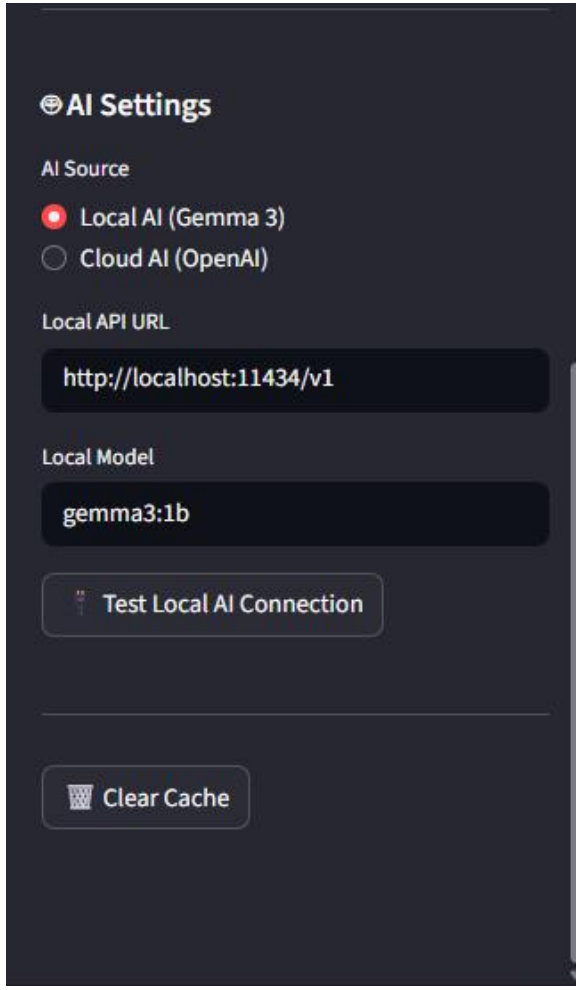


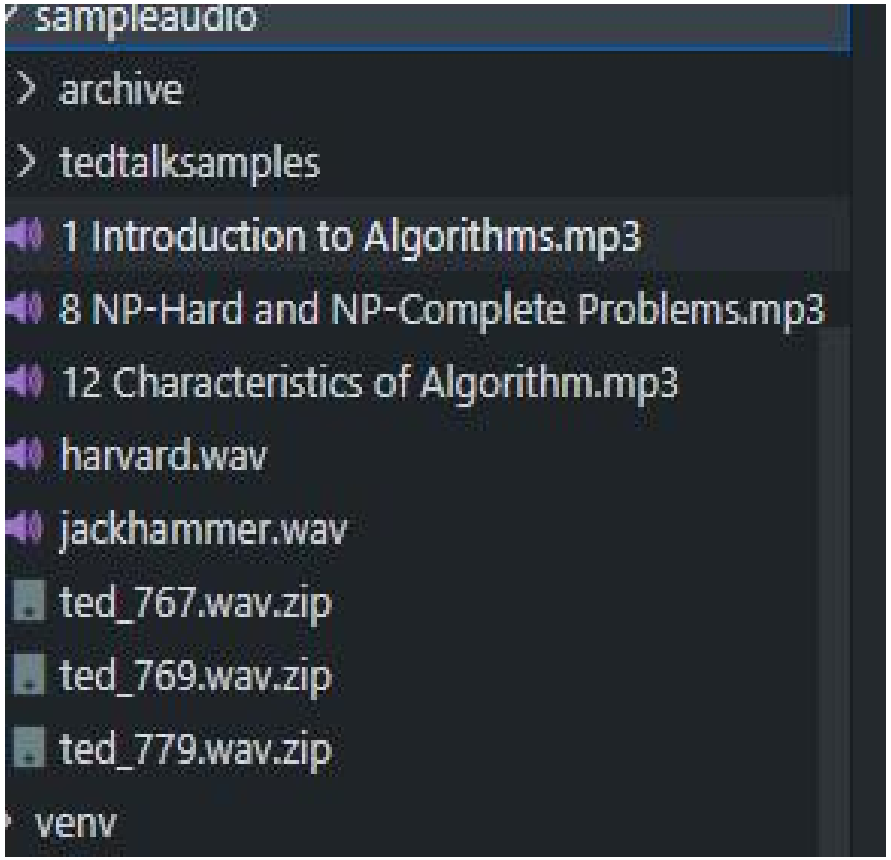
RESULTS



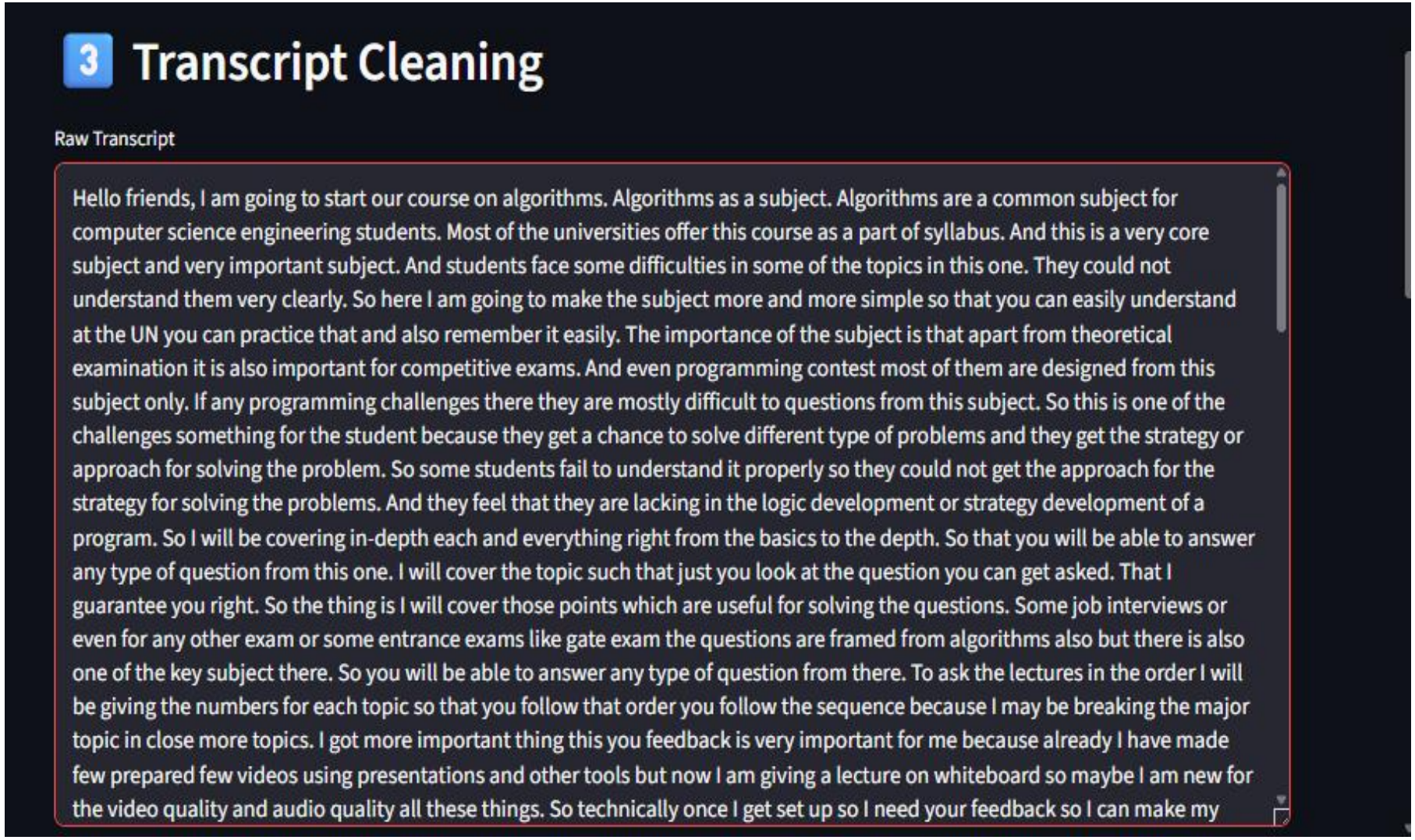
edunet foundation

RESULTS


AI Settings
AI Source
Local AI (Gemma 3)
Cloud AI (OpenAI)
Local API URL
http://localhost:11434/v1
Local Model
gemma3:1b
Test Local AI Connection
Clear Cache


sampleaudio
archive
tedtalksamples
1 Introduction to Algorithms.mp3
8 NP-Hard and NP-Complete Problems.mp3
12 Characteristics of Algorithm.mp3
harvard.wav
jackhammer.wav
ted_767.wav.zip
ted_769.wav.zip
ted_779.wav.zip
venv

RESULTS


3 Transcript Cleaning
Raw Transcript
Hello friends, I am going to start our course on algorithms. Algorithms as a subject. Algorithms are a common subject for computer science engineering students. Most of the universities offer this course as a part of syllabus. And this is a very core subject and very important subject. And students face some difficulties in some of the topics in this one. They could not understand them very clearly. So here I am going to make the subject more and more simple so that you can easily understand at the UN you can practice that and also remember it easily. The importance of the subject is that apart from theoretical examination it is also important for competitive exams. And even programming contest most of them are designed from this subject only. If any programming challenges there they are mostly difficult to questions from this subject. So this is one of the challenges something for the student because they get a chance to solve different type of problems and they get the strategy or approach for solving the problem. So some students fail to understand it properly so they could not get the approach for the strategy for solving the problems. And they feel that they are lacking in the logic development or strategy development of a program. So I will be covering in-depth each and everything right from the basics to the depth. So that you will be able to answer any type of question from this one. I will cover the topic such that just you look at the question you can get asked. That I guarantee you right. So the thing is I will cover those points which are useful for solving the questions. Some job interviews or even for any other exam or some entrance exams like gate exam the questions are framed from algorithms also but there is also one of the key subject there. So you will be able to answer any type of question from there. To ask the lectures in the order I will be giving the numbers for each topic so that you follow that order you follow the sequence because I may be breaking the major topic in close more topics. I got more important thing this you feedback is very important for me because already I have made few prepared few videos using presentations and other tools but now I am giving a lecture on whiteboard so maybe I am new for the video quality and audio quality all these things. So technically once I get set up so I need your feedback so I can make my

RESULTS

Notes (Markdown)

Okay, here's a revised and expanded study note based on the provided lecture analysis, aiming for clarity, conciseness, and a professional tone. It builds upon the existing framework while adding more detail and structure.

Understanding Algorithms: A Comprehensive Overview

I. Introduction - What is an Algorithm?

- **Definition:** An algorithm is a precise, step-by-step procedure for solving a computational problem. It's a sequence of instructions designed to achieve a specific goal.
- **Why is it important?**
  - **Core Subject:** Fundamental to computer science, engineering, and programming students.
  - **Competitive Exams:** Crucial for success in algorithm-based exams (e.g., Algorithm Design & Analysis).
  - **Programming Contests:** Essential for problem-solving in programming contests and challenges.
- **The Challenge:** Students often struggle to understand algorithms because they are complex and require careful thought.
- **The Goal:** To provide a clear, step-by-step approach to solving problems.

II. Key Definitions

- **Program:** A set of instructions for a computer to execute. It's the complete, executable piece of code.
- **Difference between Program and Algorithm:**
  - **Program:** A complete, executable piece of code. Think of it as the *result* of the process.
  - **Algorithm:** A step-by-step procedure for solving a problem. It's the *process* itself.
- **Algorithm's Role:**
  - **Design Phase:** The initial stage where the algorithm is conceived and designed.
  - **Implementation Phase:** The stage where the algorithm is translated into code.
  - **Design Time:** The time spent on designing the algorithm.

III. The Algorithm Development Process

1. **Design Phase:**
   - **Purpose:** Define the problem and the solution.
   - **Steps:**
     - Understand the problem thoroughly.
     - Create a design that is perfect and thorough.
     - Ensure the design is clear and easy to understand.
2. **Implementation Phase:**
   - **Purpose:** Translate the design into code.
   - **Steps:**
     - Write the code.
     - Change the mind and delete the program.
     - Start writing the program.
3. **Why is it important to understand the design?**
   - **Waste of time:** If you don't understand the problem, you can waste a lot of time.
   - **Understanding the problem:** You need to understand the problem to create a solution.

IV. The Role of the Programmer

- **Programmers are crucial:**
  - **Domain Knowledge:** Programmers need to have domain knowledge.
  - **Problem Knowledge:** They need to know the problem and its solution.
  - **Knowledge of the Problem:** They can help the student who understands the problem.
  - **Knowledge of the Solution:** They can write the solution.
- **Programmer's Role:**
  - **Write Programs:** Programmers write programs.

REFERENCES

- OpenAI Whisper: Speech-to-text models
- FAISS: Efficient similarity search and clustering
- Streamlit: Rapid web application development
- SentenceTransformers: Semantic text embeddings
- LangChain: RAG pipeline inspiration and chunking strategies

FUTURE SCOPE

- Multilingual transcription and content generation
- .Real-time speaker diarization for multi-speaker lectures
- Collaborative RAG for shared lecture repositories
- Mobile app for on-the-go access to study materials

CONCLUSION

LectureMind AI democratizes learning by transforming raw lecture content into actionable study materials. With a modular, local-first architecture, it empowers students to learn smarter without infrastructure or cost barriers. The project is production-ready and scalable for classrooms and institutions.

THANK YOU

edunet
foundation

GITHUB LINK

- Github Link : https://github.com/the-introvert20/LectureMind.AI.git
- Deployment link : https://lecturemindai-najuzbv6topc4zgedyafxh.streamlit.app/