Final Project Handout

Project Title: AI Tutor for Introductory Python Programming using LLM

This project guides you through designing and implementing an AI-powered tutoring system for introductory Python programming. You may use either an open-source model (e.g., LLaMA, Mistral, Falcon) or an API-based model (e.g., ChatGPT) to answer questions, explain concepts, and provide feedback to beginner students.

Objective

To build a conversational AI tutor capable of teaching Python basics, helping debug simple code, and providing adaptive feedback. Students will demonstrate understanding of LLM-based generation, tokenization, prompt design, and evaluation.

Project Description

Your AI Tutor should be able to:

- Explain Python fundamentals (variables, loops, conditionals, etc.)
- Debug small code snippets
- Provide example code with explanations
- Generate practice exercises and evaluate answers
- Encourage learning through interactive feedback

Deliverables

Component	Description
Implementation (Weeks 2–3)	A functional AI tutor using an LLM API or open-source model, submit with your GitHub Repository.
Report (Final Week)	A 5–10 page report explaining design, tools, goals, and evaluation.

Technical Requirements

- Use either ChatGPT API or an open-source LLM (e.g., via Hugging Face Transformers).
- Demonstrate tokenization and cost analysis if using API models.
- Include clear prompt engineering and example interactions.
- Evaluate system correctness, clarity, and educational usefulness.

Evaluation Rubric (100 pts)

Criteria	Points
Problem Definition & Creativity	10

Technical Implementation	60
Tokenization & Model Understanding	10
User Interaction Design	10
Report Clarity & GitHub repository	10

Step-by-Step Project Guidance

Step 1 — Define Tutor Scope

Determine the audience level, topics (e.g., variables, loops, functions), and the type of help the tutor will provide (explanations, debugging, exercises).

Step 2 — Choose an LLM

Decide whether to use a hosted API model (ChatGPT) or open-source model (LLaMA, Falcon, Mistral). Start with API models for simplicity.

Step 3 — Prototype Interaction

Create a simple chat loop to send user input to the LLM and return responses. Focus on establishing communication before adding logic.

Step 4 — Add Tutoring Features

This step transforms your AI Tutor into an interactive, structured learning assistant. It should now teach Python concepts, generate examples, offer exercises, and provide adaptive feedback.

Core Tutoring Modules

Module	Function	Example Interaction
Concept Explainer	Explains Python concepts simply.	"Explain what a function does."
Code Example Generator	Creates annotated Python examples.	"Show an example of a for loop."
Error Debugger	Identifies and explains errors in code.	"Why doesn't my code work?"
Exercise Creator	Generates short coding exercises.	"Give me a problem about while loops."
Feedback Provider	Gives constructive, motivating feedback.	"That's close! Try adjusting your return statement."

Structured Response Design

Ask the LLM to format responses consistently. Example format:

1. Concept Explanation
2. Code Example
3. Practice Exercise
4. Feedback (if code provided)

Handling Different Input Types

Detect the user's intent and choose a tutoring mode based on input. Example:

```
if 'def' in user_input: mode = 'debug'
elif 'exercise' in user_input: mode = 'exercise'
elif 'explain' in user_input: mode = 'explain'
else: mode = 'feedback'
```

Implementation Example

Sample ChatGPT API logic:

```
from openai import OpenAI
client = OpenAI(api_key='YOUR_KEY')
messages = [{'role':'system','content':'You are a Python tutor.'},
{'role':'user','content':'Explain loops.'}]
response = client.chat.completions.create(model='gpt-3.5-turbo', messages=messages)
print(response.choices[0].message.content)
```

Optional Enhancements

- Add memory to maintain conversation context.
- Track student progress.
- Integrate a web interface using Streamlit.
- Implement an auto-evaluator to test student code safely.

Example Interaction

User: What is a Python list?

Tutor:
Concept Explanation: Lists store multiple items in one variable.
Code Example: my_list = [1,2,3]; print(my_list[0])
Practice Exercise: Create a list of your favorite fruits.

Key Takeaway

Step 4 transforms your system into a structured learning companion that promotes understanding and engagement.

Reference

https://www.youtube.com/watch?v=q5HiD5PNuck (Chatbot)

https://learn.deeplearning.ai/courses/chatgpt-prompt-eng/lesson/dfbds/introduction (prompt engineering)