Adaptive Python Learning Companion

Capstone Project Proposal

Project Title: Adaptive Python Learning Companion

Group Name: AI Learning Architects

Participants: Christian Mpabuka, [Other Group Members]

Course Code: ITAI2376

Submission Date: April 7

Adaptive Python Learning Companion

1. Problem Statement

Many students struggle with learning programming due to a lack of personalized guidance. Traditional learning platforms often provide static content that does not adapt to individual needs. This project aims to build an AI-powered interactive learning companion that dynamically adjusts to each learner's progress, provides tailored exercises, and offers real-time feedback to enhance learning outcomes.

2. Project Option

Selected Option: Option 4 - Interactive Learning Companion

3. Agent Design

The agent will follow a modular architecture:
- Input Processing: Natural language understanding to interpret student queries.
- Memory System: Learner profiles stored in a vector database (e.g., FAISS).
- Reasoning Component: Chain-of-Thought reasoning for hint generation and Planning-then-Execution for learning path adjustments.
- Output Generation: Personalized exercises, feedback, and explanations using LLMs.

4. Tool Selection

Tool 1: Python Code Execution Environment
- Used to evaluate student code and provide feedback.

Tool 2: Vector Database (e.g., FAISS or ChromaDB)
- Used to store and retrieve learner history and relevant content.

5. Development Plan

Date Range	Milestone
Mar 24 - Apr 7	Team formation, research, proposal
Apr 8 - Apr 14	Basic architecture, tool integration

Adaptive Python Learning Companion

Apr 15 - Apr 21 | Prototype with feedback loop

Apr 22 - Apr 28 | Testing and refinement

Apr 29 - May 5 | Final testing, report, and video

6. Evaluation Strategy

- Quantitative: Accuracy of code evaluation, hint effectiveness, and learning gains (pre/post quizzes).
- Qualitative: User satisfaction surveys and feedback.
- Adaptability: How well the agent adjusts difficulty based on performance.

7. Resource Requirements

- Platform: Google Colab (free tier)
- Libraries: Transformers, FAISS, OpenAI API, Gradio
- Hardware: GPU access (optional for LLM inference)

8. Risk Assessment

Risk	Mitigation Strategy
Tool integration failures	Use mock tools and fallback strategies
LLM limitations	Break tasks into smaller steps
Time constraints	Focus on MVP before adding features
Colab session timeouts	Use checkpoints and Google Drive backups