6-Month Generative AI Curriculum: Track 1 - AI Engineering with LLM Focus

Program Overview

This 6-month intensive curriculum is designed for individuals aiming to specialize in AI Engineering, with a strong emphasis on Large Language Model (LLM) engineering. The program balances foundational AI/ML concepts with deep dives into Transformer architectures, embeddings, Retrieval Augmented Generation (RAG) systems, and intelligent agentic applications using frameworks like LangGraph. Students will gain hands-on experience with the broader Generative AI ecosystem (Hugging Face, OpenAI, Anthropic, Ollama) and learn to operationalize these models. While touching upon essential MLOps practices on AWS for deployment, the core focus remains on building and understanding the AI engines themselves. The curriculum also integrates product management principles to foster an entrepreneurial mindset.

Target Audience

- **Experienced Professionals:** Individuals with prior programming experience (e.g., Python) and a STEM background, particularly those with existing AI/ML knowledge, looking to specialize in LLM-focused AI Engineering.
- **Non-Experienced Individuals:** Beginners with strong analytical skills and a STEM background, eager to enter the AI Engineering field. The curriculum provides a solid foundation for those new to programming or AI.

Learning Format

- **100% In-Person:** Facilitates direct interaction, collaborative learning, and immediate instructor support, fostering a dynamic learning environment.
- **25 Hours/Week:** Dedicated time for lectures, hands-on labs, interactive workshops, and collaborative project work, ensuring deep engagement and skill development.

Key Learning Objectives

Upon completion of this program, students will be able to:

- Master Python programming for AI Engineering and LLM development.
- Understand core Machine Learning and Deep Learning concepts relevant to Generative AI.
- Gain a strong understanding of Transformer architectures, embeddings, and their implementation from foundational principles.
- Design and implement Retrieval Augmented Generation (RAG) systems from foundational principles.
- Develop intelligent agentic applications using frameworks like LangGraph.
- Leverage the Generative AI ecosystem, including Hugging Face, OpenAI, Anthropic, and local models like Ollama.
- Apply advanced Prompt Engineering techniques for effective interaction with LLMs and agents.
- Implement essential MLOps practices on AWS for deploying and monitoring Generative AI models.
- Understand and apply product management principles to AI projects, from ideation to deployment.
- Collaborate effectively using Git and GitHub for version control and team projects.
- Design and execute end-to-end Generative AI projects with a focus on building and understanding the AI engines.
- Prepare for roles such as AI Engineer (LLM Focus), Generative AI Developer, and AI/ML Product Manager, capable of contributing to or founding startups.

Curriculum Structure: Month-by-Month Progression

Month 1: Foundations of Python, ML, and Introduction to LLMs

Goal: Establish a strong foundation in Python programming, essential mathematics for ML, fundamental Machine Learning concepts, and an initial understanding of Large Language Models.

Weekly Breakdown:

- **Week 1: Python for Data Science & ML:**
    - Python basics: syntax, data types, control flow, functions.
    - Data structures: lists, tuples, dictionaries, sets.
    - Object-Oriented Programming (OOP) in Python.

- Introduction to NumPy for numerical computing.
- Introduction to Pandas for data manipulation and analysis.
- **Mini-Project:** Data cleaning and basic analysis using Pandas.

- **Week 2: Essential Mathematics for ML & Core ML Concepts:**
  - Linear Algebra: vectors, matrices, operations (conceptual understanding for ML algorithms).
  - Calculus: derivatives, gradients (conceptual understanding for optimization).
  - Probability & Statistics: basic probability, distributions, descriptive statistics.
  - Introduction to Machine Learning: Supervised vs. Unsupervised Learning.
  - Key ML concepts: features, labels, training, testing, overfitting, underfitting.
  - **Mini-Project:** Implement basic statistical analysis and build a simple regression or classification model.

- **Week 3: Deep Learning Fundamentals & Neural Networks:**
  - Introduction to Deep Learning: differences from traditional ML.
  - Perceptrons and Feedforward Neural Networks (FNNs).
  - Backpropagation algorithm (conceptual understanding).
  - Introduction to TensorFlow/Keras or PyTorch (choose one as primary).
  - **Mini-Project:** Build a simple FNN for classification.

- **Week 4: Introduction to Large Language Models (LLMs) & Prompt Engineering:**
  - What are LLMs? Basic architecture (conceptual overview of Transformers).
  - Pre-trained vs. Fine-tuned LLMs.
  - Introduction to Prompt Engineering: techniques for effective interaction with LLMs.
  - Zero-shot, one-shot, and few-shot prompting.
  - Ethical considerations and biases in LLMs.
  - **Mini-Project:** Experiment with different prompt engineering techniques using a publicly available LLM API (e.g., Hugging Face Inference API or OpenAI API with free tier/credits).

Month 2: LLM Architectures, Embeddings, and Fine-tuning Foundations

Goal: Dive deeper into LLM architectures, understand embeddings, and lay the groundwork for fine-tuning pre-trained LLMs.

Weekly Breakdown:

- **Week 5: Transformer Architecture Deep Dive (from Scratch):**
    - Encoder-Decoder architecture of Transformers.
    - Self-Attention Mechanism and Multi-Head Attention.
    - Positional Encoding, Layer Normalization, Feed-Forward Networks.
    - Understanding the role of Transformers in modern LLMs.
    - **Mini-Project:** Implement a simplified Transformer block from scratch (conceptual or using a library like PyTorch/TensorFlow) to solidify understanding.
- **Week 6: Embeddings and Vector Databases (from Scratch):**
    - Understanding text embeddings: how they are generated, properties, and use cases.
    - Common embedding models (e.g., Word2Vec, GloVe, BERT embeddings, OpenAI embeddings).
    - Implementing a basic word embedding model (e.g., Word2Vec) from scratch.
    - Introduction to Vector Databases: purpose, types, and basic operations (e.g., Pinecone, ChromaDB, FAISS).
    - **Mini-Project:** Generate embeddings for a text corpus and perform semantic search using a local vector store, potentially building a simple in-memory vector store.
- **Week 7: Introduction to LLM Fine-tuning:**
    - Why fine-tune LLMs? Adapting pre-trained models to specific tasks or domains.
    - Types of fine-tuning: full fine-tuning vs. parameter-efficient fine-tuning (PEFT).
    - Data preparation for fine-tuning: dataset formatting, tokenization.
    - Introduction to Hugging Face `transformers` library for fine-tuning.
    - **Mini-Project:** Prepare a small dataset and perform basic fine-tuning of a small pre-trained LLM for a classification or text generation task.
- **Week 8: Advanced LLM Fine-tuning Techniques:**
    - Parameter-Efficient Fine-Tuning (PEFT) methods: LoRA, QLoRA, Prompt Tuning.
    - Understanding the trade-offs between different fine-tuning approaches.
    - Evaluation metrics for fine-tuned LLMs (e.g., perplexity, BLEU, ROUGE, custom metrics).

- **Mini-Project:** Apply a PEFT method to fine-tune an LLM and evaluate its performance against a fully fine-tuned model.

Month 3: Retrieval Augmented Generation (RAG) & LangGraph Fundamentals

Goal: Gain a strong understanding of RAG systems and build intelligent agentic applications using LangGraph.

Weekly Breakdown:

- **Week 9: Retrieval Augmented Generation (RAG) Systems (from Foundational Principles):**
  - Deep dive into RAG architecture and its necessity for grounding LLMs.
  - Document loading and chunking strategies.
  - Implementing a custom retriever: indexing documents into a vector store and performing retrieval.
  - **Mini-Project:** Build a complete RAG system from foundational principles, allowing it to answer questions based on retrieved documents.
- **Week 10: Advanced RAG Techniques & Evaluation:**
  - Advanced RAG techniques: query expansion, re-ranking, HyDE, multi-hop RAG.
  - Evaluation metrics for RAG systems: faithfulness, relevance, coherence, answer correctness.
  - Tools for RAG evaluation (e.g., Ragas, LangChain Evaluation).
  - Addressing RAG limitations: handling complex queries, reducing hallucinations, prompt injection attacks.
  - **Mini-Project:** Implement and evaluate an advanced RAG system, demonstrating improved performance on complex queries.
- **Week 11: Introduction to LangGraph & State Management:**
  - Why LangGraph? Limitations of traditional LLM chains for complex, stateful, and cyclical workflows.
  - Graph concepts: Nodes (actions), Edges (transitions), StateGraph (managing application state).
  - Defining and managing graph state: `TypedDict`, `Annotated` with reducers for persistent context across turns.
  - Building a basic conversational agent with LangGraph that maintains conversation history and can make decisions based on state.

    - **Mini-Project:** Implement a simple chatbot using LangGraph that can remember previous turns and dynamically change its behavior based on the conversation flow.

- **Week 12: Tool Integration in LangGraph Agents:**
    - Integrating external tools with LangGraph agents: defining custom tools (e.g., web search, calculator, database query, custom APIs).
    - Agent decision-making: how agents choose and use tools, interpret tool outputs, and handle tool errors.
    - **Mini-Project:** Develop a LangGraph agent that can use external tools to answer questions or perform actions.

Month 4: Advanced Agentic Applications & Generative AI Ecosystem

Goal: Explore advanced agentic patterns, build multi-agent systems, and leverage the broader Generative AI ecosystem.

Weekly Breakdown:

- **Week 13: Agentic RAG with LangGraph & Multi-Agent Systems:**
    - Deep dive into Agentic RAG architecture: combining RAG with agentic decision-making for improved accuracy and relevance.
    - Implementing advanced retrieval strategies within LangGraph agents (e.g., self-correction, query rewriting, multi-agent RAG).
    - Common agent patterns: sequential agents, hierarchical agents, collaborative agents.
    - **Mini-Project:** Build an Agentic RAG system using LangGraph that can dynamically rephrase queries, perform web searches, and potentially involve a simple multi-agent setup for complex information retrieval.

- **Week 14: Generative AI Ecosystem: Hugging Face & OpenAI:**
    - **Hugging Face Ecosystem:** Transformers library for various models (LLMs, Diffusion Models), Datasets, Accelerate, Spaces for model sharing and deployment.
    - Using Hugging Face models for diverse tasks (text generation, image generation, summarization, translation).
    - **OpenAI Ecosystem:** GPT models, DALL-E, Whisper, API usage, fine-tuning capabilities.
    - Comparison of capabilities, pricing, and use cases between Hugging Face and OpenAI models.

    - **Mini-Project:** Integrate a Hugging Face model and an OpenAI model into separate applications, demonstrating their respective strengths and API interactions.

- **Week 15: Generative AI Ecosystem: Anthropic & Ollama (Local LLMs):**
    - **Anthropic Ecosystem:** Claude models, constitutional AI, safety principles, API usage for conversational AI.
    - **Ollama:** Running large language models locally on personal hardware.
    - Benefits and challenges of local LLM deployment vs. cloud-based APIs (privacy, cost, performance).
    - **Mini-Project:** Experiment with a local LLM using Ollama for a specific task (e.g., text generation, summarization) and compare its performance/ capabilities with cloud-based models.

- **Week 16: Building User Interfaces for GenAI Models (Streamlit/Gradio) & Essential MLOps for GenAI:**
    - Rapid prototyping of user interfaces for Generative AI models using Streamlit or Gradio.
    - Connecting Streamlit/Gradio applications to deployed LLMs, RAG systems, or agentic applications.
    - Essential MLOps concepts for Generative AI: model versioning, basic deployment strategies (e.g., containerization).
    - **Mini-Project:** Build a simple Streamlit/Gradio interface to interact with a deployed RAG or agentic application, and understand basic deployment considerations.

Month 5: AWS Integration for GenAI Deployment, Ethical AI, and Evaluation

Goal: Learn to deploy and manage Generative AI models on AWS, understand ethical considerations, and master evaluation techniques.

Weekly Breakdown:

- **Week 17: AWS Fundamentals for GenAI Deployment:**
    - AWS services for model deployment: SageMaker Endpoints (real-time, batch transform), AWS Lambda with container images, API Gateway.
    - Managing LLM inference costs and latency on AWS.
    - Version control for LLM models and artifacts on AWS.

- **Mini-Project:** Deploy a fine-tuned LLM (from Month 2) or a RAG system to a SageMaker Endpoint and test its inference capabilities.

- **Week 18: MLOps Practices for Generative AI on AWS:**
  - Continuous Integration/Continuous Delivery (CI/CD) for Generative AI models on AWS (e.g., AWS CodePipeline, CodeBuild).
  - Model monitoring: data drift, model drift, performance monitoring with SageMaker Model Monitor.
  - Alerting and logging with Amazon CloudWatch for GenAI applications.
  - **Mini-Project:** Set up a basic CI/CD pipeline on AWS for a Generative AI model and configure basic monitoring.

- **Week 19: Ethical AI in Generative AI Application Building:**
  - Bias and Fairness in Generative AI: identification, mitigation strategies (e.g., data augmentation, model debiasing, fairness-aware design).
  - Privacy concerns: data leakage, memorization, secure data handling in application development.
  - Transparency and Explainability (XAI) in Generative AI: understanding model decisions.
  - Responsible deployment and governance of Generative AI systems in production environments.
  - **Discussion & Workshop:** Analyze ethical dilemmas in Generative AI and propose solutions, focusing on real-world scenarios.

- **Week 20: Evaluation Techniques for Generative AI Models & Applications:**
  - Quantitative evaluation metrics for text generation (BLEU, ROUGE, METEOR), image generation (FID, Inception Score), and RAG systems (faithfulness, relevance, answer correctness, context adherence).
  - Qualitative evaluation methods: human judgment, user studies, A/B testing for application performance and user experience.
  - Tools and frameworks for automated evaluation (e.g., LangChain Evaluation, Ragas, custom evaluation scripts).
  - **Mini-Project:** Implement an evaluation pipeline for a previously built RAG or agentic application, comparing different models or configurations based on application-specific metrics.

Month 6: Shared Capstone Project and Career Preparation

Goal: Apply all learned concepts to a real-world, industry-relevant Generative AI project, collaborate with Track 2 students, and prepare for job readiness and startup founding.

Weekly Breakdown:

- **Week 21-24: Capstone Project (Group-based, Cross-Track Collaboration):**
    - **Project Selection:** Students from both tracks will form cross-functional teams. Projects will be industry-relevant Generative AI applications, requiring both strong AI Engineering/LLM engineering and robust application development.
    - **Project Planning:** Define scope, objectives, deliverables, and timelines. Formulate project teams. Utilize GitHub for collaborative version control from the start.
    - **Design & Development:** Implement the Generative AI solution. Track 1 students will focus on LLM engineering, RAG, agentic applications, and model deployment/MLOps. Track 2 students will focus on frontend/backend development and integration. Emphasis on best practices for coding, application architecture, MLOps, and ethical considerations.
    - **Testing & Evaluation:** Rigorous testing of the application and its components using learned evaluation techniques. Utilize GitHub for code reviews and issue tracking.
    - **Documentation:** Create comprehensive project documentation, including technical specifications, user manuals, and deployment guides, all version-controlled on GitHub.
    - **Presentation & Demo:** Prepare and deliver a professional presentation and live demonstration of the deployed project to instructors, industry mentors, and potential investors.
- **Week 25-26: Career Preparation & Startup Fundamentals:**
    - **Resume Building & Interview Skills:** Tailoring resumes for AI Engineer (LLM Focus), Generative AI Developer, and AI/ML Product Manager roles. Behavioral and technical interview preparation, mock interviews, emphasizing project contributions on GitHub.
    - **Portfolio Development:** Curating and showcasing capstone and mini-projects, with an emphasis on deployed systems and GitHub repositories.
    - **Networking & Job Search Strategies:** LinkedIn optimization, industry events, job boards specific to AI Engineering and Generative AI.
    - **Startup Fundamentals:** Introduction to business model canvas, value proposition design, market validation, fundraising basics, legal considerations for startups. (Shared sessions with Track 2).
    - **Guest Speakers:** Industry professionals and startup founders sharing insights and career advice on AI Engineering, Generative AI, and entrepreneurship.

Project-Based Learning Approach

This curriculum heavily emphasizes a project-based learning approach to ensure students gain practical, hands-on experience and develop a strong portfolio. Each week includes a dedicated mini-project designed to reinforce the concepts learned and apply them in a practical context. These mini-projects will vary in scope, from individual coding assignments to small group collaborations, fostering both independent problem-solving and teamwork.

Mini-Projects

Mini-projects are integrated throughout the weekly breakdown of each month. They are designed to be short, focused exercises that allow students to immediately apply newly acquired knowledge. Examples include:

- Implementing a simplified version of a core algorithm (e.g., Transformer block, Word2Vec).
- Building a small, functional component of a larger system (e.g., a FastAPI endpoint, a Streamlit UI).
- Experimenting with different models or techniques and evaluating their performance.
- Solving specific, well-defined problems using the tools and concepts learned.

Capstone Project

The capstone project is the culmination of the 6-month program, providing students with an opportunity to apply all their accumulated knowledge and skills to a real-world, industry-relevant problem. This project is designed to be a collaborative effort between students from both Track 1 (AI Engineering with LLM Focus) and Track 2 (Frontend/ Backend Dev & GenAI), simulating a cross-functional team environment typical in industry.

Capstone Project Goals:

- **Problem Solving:** Address a significant, real-world problem using Generative AI technologies.
- **End-to-End Development:** Cover the entire lifecycle of an AI application, from ideation and design to development, deployment, and monitoring.
- **Cross-Functional Collaboration:** Foster effective teamwork between students with different specializations (AI Engineering/LLM Focus and Full-stack Application Development).

- **Industry Relevance:** Ensure the project is relevant to current industry needs and demonstrates job-ready skills.
- **Entrepreneurial Mindset:** Encourage students to think about the product\"s value proposition, target users, and potential business models.
- **Portfolio Building:** Provide a substantial, demonstrable project for students\" portfolios, showcasing their capabilities to potential employers or investors.

Capstone Project Structure (Weeks 21-24):

1. **Project Selection & Team Formation:** Students will be presented with a curated list of project ideas, or they can propose their own. Teams will be formed with a mix of students from both tracks to ensure diverse skill sets. Project ideas will be vetted for scope, feasibility, and alignment with learning objectives.
2. **Project Planning & Design:** Teams will define the project scope, objectives, key performance indicators (KPIs), and deliverables. They will create a detailed project plan, including architecture design, technology stack selection, and task breakdown. This phase will involve applying product management principles to define the Minimum Viable Product (MVP) and subsequent iterations.
3. **Development & Implementation:** This is the core development phase. Track 1 students will focus on:
    - Data collection, preprocessing, and management.
    - Training or fine-tuning LLMs and other Generative AI models.
    - Implementing MLOps pipelines for model versioning, training, and deployment on AWS.
    - Ensuring model performance, scalability, and reliability. Track 2 students will focus on:
    - Designing and implementing the frontend user interface (HTML, CSS, JavaScript, React).
    - Developing the backend API (FastAPI) to serve the Generative AI models.
    - Integrating databases, authentication, and other backend services.
    - Connecting the frontend and backend components. Both tracks will collaborate closely, utilizing Git and GitHub for version control, code reviews, and issue tracking. Emphasis will be placed on writing clean, modular, and well-documented code.
4. **Testing, Evaluation & Iteration:** Teams will rigorously test their application, including unit tests, integration tests, and end-to-end tests. They will apply learned evaluation techniques to assess the performance of their Generative AI models and the overall application. Based on testing and evaluation results, teams will iterate on their design and implementation to improve functionality, performance, and user experience.

5. **Documentation & Presentation:** Teams will create comprehensive project documentation, including a technical design document, API documentation, user manual, and deployment guide. The culmination of the project will be a professional presentation and live demonstration of the deployed project to instructors, industry mentors, and a panel of potential investors or hiring managers. This presentation will highlight the problem solved, the solution implemented, the technologies used, the team\"s contributions, and the potential business impact.

Through this project-based learning approach, students will not only solidify their technical skills but also develop critical soft skills such as teamwork, communication, problem-solving, and project management, essential for success in both employment and entrepreneurial ventures.