AI for Software Developers and Architects

This 5-day accelerated program equips software developers and architects with practical knowledge to apply Generative AI, LLM APIs, RAG, prompt-driven development, and agentic AI patterns across the modern software delivery lifecycle.

Overview

This 5-day accelerated program is designed for software developers and architects who want to integrate Artificial Intelligence into their software development lifecycle. The course provides a structured journey from understanding Generative AI fundamentals to building AI-powered development environments, working with Large Language Models, and designing agentic AI systems. Participants will learn how to use tools like ChatGPT and Codex effectively, build prompt-driven development workflows, integrate AI APIs into applications, implement Retrieval-Augmented Generation, and explore model fine-tuning techniques such as LoRA. By the end of the program, participants will be able to design AI-assisted development pipelines and understand how to scale them into enterprise-level solutions.

Introduction

Artificial Intelligence is rapidly reshaping how software is designed, built, tested, documented, and operated. For developers and architects, AI is no longer only a research topic or a business feature layer; it is becoming a core part of engineering productivity, system design, and application architecture. This program introduces participants to the essential concepts, tools, and architectural patterns required to work effectively with Generative AI in software development. The course covers modern AI-assisted coding workflows, prompt engineering, LLM integration, embeddings, RAG, model customization, and enterprise AI architecture in a practical way that helps technical teams understand how these capabilities fit into real-world engineering environments.

Topics Covered

- Introduction to AI in Software Development
- Fundamentals of Generative AI
- Machine Learning Foundations
- Understanding LLM Internals (Simplified)

- Mathematical Intuition (High-Level)
- Prompt Engineering Fundamentals
- Vibe Coding (AI-Assisted Development Workflow)
- AI Coding Tools and Platforms
- Building a Prompt-Driven Development Workflow
- Working with LLM APIs
- Embeddings and Vector Databases
- Retrieval-Augmented Generation (RAG)
- Building AI-Powered Applications
- Fine-Tuning and Customization
- Working with Open Source LLMs
- Introduction to Agentic AI
- Multi-Agent Systems for Development
- Enterprise AI Architecture
- AI Security and Risk Management
- Advanced RAG and Knowledge Systems
- AI in Software Architecture
- Final Capstone Project
- Final Presentation and Evaluation

Prerequisites

1. Programming experience (Java, JavaScript, Python, PHP, etc.).
2. Basic understanding of APIs and web applications.
3. Familiarity with software development lifecycle

Audience:

1. Software Developers (Frontend, Backend, Full Stack)
2. Software Engineers and Technical Leads
3. Solution Architects and System Designers
4. Teams adopting AI-assisted development

Duration: 5 Days

Course Outline

Day 1 - Foundations of AI and Generative AI for Developers

1. Introduction to AI in Software Development
    - 1.1. Evolution of AI in software engineering
    - 1.2. Role of AI in modern development lifecycle
    - 1.3. AI as a productivity tool (not a replacement)
    - 1.4. Real-world use cases in coding, testing, and automation

2. Fundamentals of Generative AI
    - 2.1. What is Generative AI
    - 2.2. Types of Generative Models (Text, Image, Code, Audio)
    - 2.3. Overview of models (GPT, LLaMA, Claude, Gemini)
    - 2.4. Business and engineering applications

3. Machine Learning Foundations
    - 3.1. Supervised, Unsupervised, Reinforcement Learning
    - 3.2. Neural Networks Basics
    - 3.3. Introduction to Deep Learning
    - 3.4. Training vs Inference

4. Understanding LLM Internals (Simplified)
    - 4.1. Tokenization
    - 4.2. Embeddings and vector representation
    - 4.3. Attention mechanism (conceptual)
    - 4.4. Transformers vs RNN vs LSTM

5. Mathematical Intuition (High-Level)
    - 5.1. Vectors and similarity
    - 5.2. Probability intuition in prediction
    - 5.3. Loss functions (conceptual)
    - 5.4. Why Transformers outperform RNN/LSTM

6. Lab 1: Exploring AI Tools
    - Use ChatGPT for coding tasks
    - Compare outputs with different prompts
    - Basic debugging with AI assistance

Day 2 - Prompt Engineering and AI-Assisted Development (Vibe Coding)

7. Prompt Engineering Fundamentals
   - 7.1. What is Prompt Engineering
   - 7.2. Prompt patterns (Zero-shot, Few-shot, Chain-of-Thought)
   - 7.3. Designing structured prompts
   - 7.4. Avoiding hallucinations

8. Vibe Coding (AI-Assisted Development Workflow)
   - 8.1. Concept of Vibe Coding
   - 8.2. AI in coding lifecycle (design to code to test to deploy)
   - 8.3. Using AI for:
   - 8.4. Code generation
   - 8.5. Refactoring
   - 8.6. Documentation
   - 8.7. Test case generation

9. AI Coding Tools and Platforms
   - 9.1. ChatGPT and advanced usage
   - 9.2. Codex and code assistants
   - 9.3. GitHub Copilot and alternatives
   - 9.4. IDE integration strategies

10. Building a Prompt-Driven Development Workflow
    - 10.1. Reusable prompt templates
    - 10.2. Role-based prompting (architect, developer, tester)
    - 10.3. Standardizing prompts in teams

11. Lab 2: AI-Powered Development
    - Generate CRUD application using prompts
    - Refactor existing code using AI
    - Generate unit tests using AI

Day 3 - Building AI Applications with APIs, RAG and Embeddings

12. Working with LLM APIs
    - 12.1. Introduction to AI APIs
    - 12.2. Calling ChatGPT API
    - 12.3. Structuring prompts programmatically
    - 12.4. Handling responses and errors

13. Embeddings and Vector Databases
    - 13.1. What are embeddings
    - 13.2. Similarity search concepts
    - 13.3. Vector databases (FAISS, Pinecone, Weaviate)
    - 13.4. Use cases in development

14. Retrieval-Augmented Generation (RAG)
    - 14.1. What is RAG
    - 14.2. Why RAG is important for enterprise systems
    - 14.3. RAG architecture
    - 14.4. Document indexing and retrieval

15. Building AI-Powered Applications
    - 15.1. Chatbot architecture
    - 15.2. Knowledge assistant design
    - 15.3. Integrating AI into web applications

16. Lab 3: Build a RAG-Based Coding Assistant
    - Connect to LLM API
    - Load documentation into vector DB
    - Build a developer assistant

Day 4 - Fine-Tuning, Open-Source Models and Agentic AI

17. Fine-Tuning and Customization
    - 17.1. Why fine-tuning is needed
    - 17.2. LoRA and QLoRA
    - 17.3. Parameter-efficient tuning (PEFT)
    - 17.4. Instruction tuning vs prompting

18. Working with Open Source LLMs
    - 18.1. Overview of open-source models (LLaMA, Mistral)
    - 18.2. Running models locally
    - 18.3. Customizing models for organization standards

19. Introduction to Agentic AI
    - 19.1. What is Agentic AI
    - 19.2. Single-agent vs multi-agent systems
    - 19.3. AI agents for coding tasks

19.4. Tool usage and orchestration

20. Multi-Agent Systems for Development
    - 20.1. Developer agent
    - 20.2. Tester agent
    - 20.3. Reviewer agent
    - 20.4. Workflow orchestration

21. Lab 4: Build a Multi-Agent Coding Assistant
    - Create agents for coding tasks
    - Chain agents together
    - Automate development workflow

Day 5 - Enterprise AI Architecture, Security and Capstone

22. Enterprise AI Architecture
    - 22.1. AI system design principles
    - 22.2. API orchestration and model routing
    - 22.3. Microservices with AI
    - 22.4. Scaling AI systems

23. AI Security and Risk Management
    - 23.1. Prompt injection attacks
    - 23.2. Data privacy and leakage
    - 23.3. Secure AI API design
    - 23.4. AI governance concepts

24. Advanced RAG and Knowledge Systems
    - 24.1. Hybrid search (dense + sparse)
    - 24.2. Knowledge graphs integration
    - 24.3. Enterprise AI assistants

25. AI in Software Architecture
    - 25.1. AI-driven system design
    - 25.2. AI in DevOps and CI/CD
    - 25.3. AI adoption strategy for organizations

26. Final Capstone Project

Participants will build an AI-Powered Development Assistant that includes a prompt-based code generator, a RAG-enabled documentation assistant, a multi-agent

workflow involving roles such as developer and tester, and a fully API-integrated system; this will culminate in a final presentation and evaluation where participants demonstrate their project, explain the underlying architecture, and discuss the best practices applied throughout the development process.