Building with LLM APIs

- Week 5 - Putting LLMs to Work in Applications
- Presented by: [Your Name or Team]
- Date: [Insert Date]

Agenda

- Overview of LLM APIs
- Request patterns and use cases
- Popular API providers
- Building a basic LLM-powered app
- Cost, latency, and limitations
- Q&A

Why Use LLM APIs?

- Access cutting-edge models without hosting infrastructure
- Quickly prototype text-based applications
- Support for natural language, code, structured reasoning, etc.
- Flexible integration into internal tools and workflows

Popular LLM API Providers

- OpenAI: ChatGPT, GPT-4, function calling
- Anthropic: Claude models with long context
- Google: Gemini Pro series
- Mistral, Cohere, open models via Hugging Face

Core Request Patterns

- Chat Completion (multi-turn conversations)
- Text Completion (single-shot prompt-response)
- Function Calling (tool use and API integration)
- Embedding generation (for RAG and similarity search)

Sample Chat Completion Request

- POST to /v1/chat/completions
- Payload: model, messages, temperature, tools (optional)
- Messages: system, user, assistant roles
- Response: structured JSON with content and metadata

Building a Simple LLM App

- Frontend (CLI, web form, or chatbot UI)
- Backend server calling LLM API (e.g., Flask, Node.js)
- Handle user input → send prompt → return response
- Log inputs/outputs for monitoring and improvement

Considerations: Cost & Latency

- Token-based pricing (input + output tokens)
- Model choice affects cost (GPT-4 > GPT-3.5)
- Latency: larger models are slower, especially with long context
- Strategies: caching, prompt trimming, fallback models

Practical Use Cases

- Internal knowledge bots
- Email summarization or drafting
- Customer support tools
- Code and data documentation assistants
- Dynamic content generation (e.g., marketing copy)

Best Practices

- Start simple — isolate one task at a time
- Test prompts across different inputs and users
- Handle edge cases and error states
- Use retries, logging, and observability tooling

Recap

- LLM APIs make powerful models easy to use
- Choose the right request pattern for your use case
- Understand costs, rate limits, and limitations
- Next: Agents and function calling for tool-augmented AI apps

Q&A / Discussion

- What use cases are most relevant to your team?
- Any blockers to trying out LLM APIs?
- Would a live demo or code walkthrough be helpful next?