Day 1: GenAI Foundations & Vibe Coding

Agentic AI Training Program

Learning Objectives

- ✅ Explain how LLMs work at conceptual level
- ✅ Understand tokens, context windows, limitations
- ✅ Use AI coding assistants effectively
- ✅ Apply "vibe coding" for rapid development
- ✅ Build and deploy first AI-assisted application

What is Generative AI?

AI that creates new content vs analyzing data

Traditional ML → "Is spam?" → Yes/No
Generative AI → "Write email" → Full email

Types:

- **Text**: GPT-4, Claude, Gemini
- **Images**: DALL-E, Midjourney
- **Audio**: Whisper, ElevenLabs
- **Code**: Copilot, Claude Code

Large Language Models (LLMs)

Neural networks trained to **predict next token**

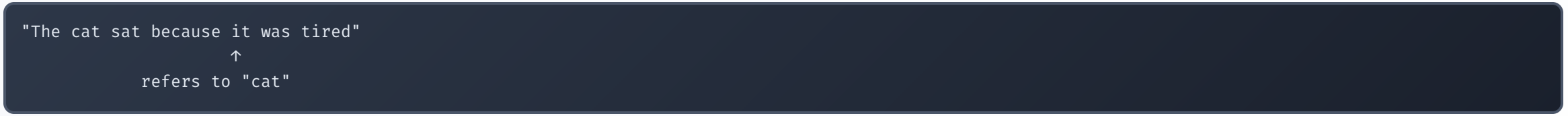
```
"The cat sat on the ___"
        ↓
[mat: 0.3, floor: 0.2, chair: 0.15 ... ]
```

Emergent capabilities:

✅ Following instructions
✅ Reasoning
✅ Code generation
✅ Translation

Transformer Architecture

Self-Attention = Focus on relevant parts



Understanding Tokens

Not characters, not words - tokens!

```
"Hello"   → ["Hello"]      # 1 token
"Hello!"  → ["Hello","!"]  # 2 tokens
"don't"   → ["don","'t"]   # 2 tokens
```

Rule: 1 token ≈ 4 characters

Matters for:

💰 Pricing 📏 Context limits 📝 API usage

Context Windows

Model	Context	~Pages
GPT-3.5	4K	6
GPT-4	8-128K	12-200
Claude 3.5	200K	300
Gemini 1.5	1M	1,500

> *Context = Input + Output (both count!)*

LLM Limitations

Issue	Description
📅 **Cutoff**	No info after training
🎭 **Hallucinations**	Confidently wrong
🚫 **No execution**	Can't run code
📏 **Context limits**	Not unlimited
🎲 **Non-deterministic**	Varies
🧮 **Math**	Unreliable

Hallucinations

```
User: "Airspeed of unladen swallow?"

LLM:  "11 m/s or 24 mph."

      ❌ Sounds right but made up!
```

Mitigate:

- Ask for sources
- Verify facts
- Use RAG
- Chain-of-thought

Temperature Parameter

0.0	0.7	1.0
Deterministic	Balanced	Creative
For: Code, Facts	**For:** Chat, Writing	**For:** Brainstorm

API Basics: Python

```python
from openai import OpenAI
client = OpenAI()

response = client.chat.completions.create(
    model="gpt-4",
    messages=[
        {"role": "system", "content": "Helper"},
        {"role": "user", "content": "What is Python?"}
    ]
)
print(response.choices[0].message.content)
```

Types: `system`, `user`, `assistant`

API Basics: TypeScript

```typescript
import Anthropic from "@anthropic-ai/sdk";
const client = new Anthropic();

const response = await client.messages.create({
  model: "claude-3-5-sonnet-20241022",
  max_tokens: 1024,
  messages: [
    { role: "user", content: "What is TypeScript?" }
  ]
});
console.log(response.content[0].text);
```

AI Coding Assistants

Tool	Type	Best For
Claude Code	CLI	Terminal
Cursor	IDE	Full IDE
Copilot	Extension	Inline
Aider	CLI	Git-integrated

What is Vibe Coding?

Collaborative coding with AI

Traditional:

1. Think solution
2. Type everything
3. Debug syntax
4. Look up docs

Vibe Coding:

1. Describe want
2. Review AI code
3. Refine feedback
4. Integrate & test

Vibe Coding Loop

> *You architect, AI builds*

Effective Prompting

```
❌ "Make better"

✅ "Refactor to:
   1. Add type hints
   2. Handle empty input
   3. Add docstring
   4. Reduce to O(n)"
```

Include: Language, framework, patterns, constraints

When to Use AI

✅ **Great:**

- Boilerplate
- Format conversion
- Tests & docs
- Debugging

⚠️ **Careful:**

- Security code
- Novel algorithms
- Optimized code

Trust but Verify

Always:

1. Read code
2. Understand logic
3. Test thoroughly
4. Check security
5. Verify facts

```
# Dangerous!
os.system(f"rm -rf {user_input}")
```

Free Tier Options

Provider	Free	Best For
Google AI	Generous	General
Groq	Fast	Speed
Ollama	Local	Privacy

> *See* `FREE-TIER-STRATEGY.md`

Model Selection

Simple → Smaller models (GPT-3.5, Haiku)
Complex → Larger models (GPT-4, Opus)

By need:

- Speed → Groq, Haiku
- Quality → Opus, GPT-4
- Cost → Open source
- Privacy → Local (Ollama)

Lab 01: First AI App

Project: URL Shortener + AI categorization

Build:

- REST API (FastAPI/Hono)
- LLM categorization
- Cloud deploy

```
cd labs/lab01-vibe-coding-intro
cd python/  # or typescript/
```

Key Takeaways

1. **LLMs predict tokens** - Understand behavior
2. **Context has limits** - Plan accordingly
3. **Hallucinations happen** - Verify facts
4. **Vibe = collaborative** - You direct
5. **Trust but verify** - Review code

What's Next: Day 2

Advanced Prompting

- RCFG Framework
- Chain-of-Thought
- Few-shot learning
- System prompts
- Code patterns

Questions?

Lab 01 awaits!

```
cd labs/lab01-vibe-coding-intro
```