

# GenForge — Forging the Future of Agentic AI



## Day 01 / Lecture 01 / Student Notes 1 : Foundations, Orientation & Environment Setup

### Scope of This Session

This first session introduces the GenForge course. You'll explore:

- Core distinctions between GenAI and traditional AI
- What makes an AI agentic
- The course structure and weekly rhythm
- Key tools and platforms for GenAI workflows
- How to set up and validate your local development environment
- Prompt engineering styles and delivery methods
- How to deploy your first GenAI app
- Professional habits around reproducibility and documentation

### Learning Objectives

By the end of this session, you should be able to:

- Explain the difference between GenAI and traditional AI
- Describe agentic AI systems and their components
- Identify and install key tools: Python, LangChain, Hugging Face, Ollama, Gemini CLI
- Configure and validate your local environment (Python, Node.js, Git, MySQL)
- Deliver prompts via Web UI, CLI, Python scripting, and Python API
- Apply prompt styles: zero-shot, few-shot, chain-of-thought, instructional, role-based
- Deploy a GenAI app using Streamlit Cloud
- Maintain GitHub hygiene and reproducibility using `requirements.txt`
- Document workflows and share via GitHub Pages

### Key Concepts

- GenAI vs traditional AI
- Agentic AI = LLMs + Goals + Tools + Autonomy
- Prompt engineering styles and delivery methods
- Local-first development and open-source tooling
- Reproducibility and version control hygiene
- Hosting platforms: Streamlit Cloud, GitHub Pages
- CLI vs GUI experimentation strategies
- Folder hygiene and structured workflows

### Tools Introduced & Validated

Tool / Platform	Purpose
Python	Core programming language for GenAI workflows
Node.js	Required for Gemini CLI
LangChain	Agentic workflows and chaining logic

Tool / Platform	Purpose
Hugging Face	Access to models and APIs
Ollama	Local LLM runtime
Gemini CLI	Prompt experimentation with Gemini models
FastAPI	RESTful API wrapper for GenAI chains
Streamlit	Interactive GenAI dashboards
Streamlit Cloud	Hosting GenAI apps with interactive UI
GitHub	Version control and collaboration
GitHub Pages	Hosting markdown documentation and showcases
XAMPP	MySQL-backed prompt workflows

### Prompt Styles

Style	Description
Zero-shot	No examples, just the task
Few-shot	A few examples to guide the model
Chain-of-Thought	Step-by-step reasoning
Instructional	Explicit task directives
Role-based	Assigning persona or context

### Prompt Delivery Methods

Method	Description
Web UI	Paste into Copilot, ChatGPT, etc.
CLI	Pipe via shell or curl
Python Script	Embed prompt in code
Python API	Send via <code>requests</code> or SDK

### Scripts and Workflows

Script	Purpose
<code>test_requirements.py</code>	Validate Python packages
<code>test_node_hello_world.js</code>	Confirm Node.js setup
<code>test_git_config.sh</code>	Verify Git identity and GitHub linking
<code>check_mysql_connection.php</code>   Test MySQL connectivity via XAMPP	
<code>test_ollama.py</code>	Run <code>tinylama</code> locally via Ollama
<code>test_langchain_prompt.py</code>	Chain prompt → LLM using LangChain
<code>test_langchain_fastapi.py</code>	Wrap LangChain in FastAPI

Script	Purpose
test_langchain_streamlit.py	Streamlit frontend for LangChain chain
test_gemini_cli.sh	Run Gemini CLI with sample prompt
test_gemini_fastapi.py	FastAPI wrapper for Gemini CLI
test_gemini_streamlit.py	Streamlit frontend for Gemini CLI
test_streamlit.py	Minimal dashboard launch test

## Exercises to Complete

- Set up and validate your Python, Node.js, and local LLM runtimes
- Run all validation scripts and confirm environment readiness
- Explore prompt delivery via all four methods
- Test LangChain and Gemini CLI workflows via CLI, FastAPI, and Streamlit
- Deploy a sample GenAI app to Streamlit Cloud
- Create a reproducible folder with `requirements.txt` and `README.md`
- Maintain folder hygiene and commit changes to GitHub
- Share your prompt experiments via GitHub Pages
- Submit screenshots via LMS or GitHub issue tracker

## References

- [Streamlit Documentation](#)
- [LangChain Quickstart](#)
- [Ollama](#)
- [Hugging Face Transformers](#)
- [Gemini CLI](#)
- [FastAPI Docs](#)
- [XAMPP](#)
- [GitHub Pages](#)
- [GitHub Student Guide](#)

## Summary

- You've understood the foundations of GenAI and agentic AI
- Your local environment is validated and ready for experimentation
- You've practiced prompt engineering styles and delivery workflows
- You've deployed and documented your first GenAI app
- You've begun cultivating reproducible, professional habits for agentic workflows

## □ License

This repository is licensed under the **Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0)**.

You may share and adapt the materials with proper attribution, but **commercial use is prohibited**.

□ [License Details](#)

Curated by Satya Prakash Nigam

Independent AI Consultant · Fractional CTO · Product Architect · Technical Enablement Strategist

Email: [spnigam25@yahoo.com](mailto:spnigam25@yahoo.com) · [LinkedIn](#) · [GitHub](#) · [Personal](#) · [Platform](#) · [Hugging Face](#)