# 1.Complete Curriculum

## PHASE 1: Programming Foundations

**Week 1 | Classes 1–2**

### Class 01: Introduction to AI Engineering

Topics:

- Introduction to Artificial Intelligence.
- AI vs Machine Learning vs Deep Learning.
- Introduction to Generative AI.
- What does an AI Engineer do?
- How modern AI applications work.
- Introduction to Large Language Models.
- Understanding the AI engineering ecosystem.
- Setting up Python, VS Code, Google Colab and GitHub.

Practical: Build your first simple AI-powered application.

### Class 02: Python Fundamentals – Part 1

Topics:

- Introduction to Python.
- Variables and data types.
- Operators and expressions.
- Input and output.
- Conditional statements.
- Loops and iterations.
- Writing basic Python programmes.

Practical:

- Build a calculator.

- Build a number-guessing application.

# PHASE 2: Python and Data Handling

**Week 2 | Classes 3–4**

## Class 03: Python Fundamentals – Part 2

Topics:

- Functions and reusable code.
- Lists, tuples, dictionaries and sets.
- String manipulation.
- File handling.
- Exception handling.
- Modules and packages.
- Virtual environments.

Practical: Build a Python-based file-processing application.

## Class 04: Python for Data Science

Topics:

- Introduction to data analysis.
- NumPy fundamentals.
- Pandas fundamentals.
- Reading and processing CSV files.
- Data cleaning.
- Missing values and duplicates.
- Data manipulation.
- Introduction to Matplotlib.
- Basic data visualisation.

Practical: Analyse and visualise a real-world dataset.

# PHASE 3: Machine Learning Foundations

**Week 3 | Classes 5–6**

## Class 05: Introduction to Machine Learning

Topics:

- What is Machine Learning?
- How machines learn from data.
- Supervised and unsupervised learning.
- Features and labels.
- Training and testing datasets.
- Regression and classification.
- Introduction to model evaluation.

## Class 06: Building Machine Learning Models

Topics:

- Introduction to Scikit-learn.
- Linear regression.
- Classification models.
- Decision trees.
- Model training and prediction.
- Accuracy, precision and recall.
- Model evaluation.

Practical project preparation: Build a basic prediction model.

# PHASE 4: Generative AI and LLM Engineering

**Week 4 | Classes 7–8**

## Class 07: Generative AI and Large Language Models

Topics:

- Introduction to Generative AI.
- Understanding Large Language Models.
- How LLMs work.
- Transformers and attention mechanisms (conceptual).
- Tokens and context windows.
- Embeddings and vector representations.
- Model providers and inference.
- Understanding hallucinations and limitations.

### Class 08: Prompt Engineering and LLM APIs

Topics:

- Fundamentals of prompt engineering.
- System and user prompts.
- Zero-shot and few-shot prompting.
- Structured outputs and JSON responses.
- Connecting LLMs through APIs.
- API keys and environment variables.
- Error handling.
- Streaming responses.

**PROJECT 1: Build an AI Chatbot**

Students build an AI-powered chatbot using an LLM API and apply prompt engineering techniques.

## PHASE 5: AI Application Development

**Week 5 | Classes 9–10**

### Class 09: Backend Development with FastAPI

Topics:

- Understanding frontend and backend architecture.
- Introduction to APIs.
- REST API fundamentals.

- HTTP methods and status codes.
- Requests and responses.
- Building APIs using FastAPI.
- Pydantic and data validation.
- Connecting LLMs to backend applications.
- Testing APIs using Swagger UI.

### Class 10: Databases and AI Interfaces

Topics:

- Database fundamentals.
- Introduction to SQL.
- PostgreSQL basics.
- Storing and retrieving application data.
- Introduction to Streamlit.
- Connecting frontend interfaces with backend APIs.
- Building interactive applications.
- Storing conversation history.

**PROJECT 2: Build a Full-Stack AI Application**

Students build an AI application with a user interface, backend API and database integration.

## PHASE 6: RAG and Knowledge-Based AI

**Week 6 | Classes 11–12**

### Class 11: Embeddings and Vector Databases

Topics:

- What are embeddings?
- Semantic search vs keyword search.
- Vector similarity.
- Cosine similarity.
- Document chunking.

- Embedding generation.
- Introduction to FAISS and ChromaDB.
- Storing and retrieving vector representations.

### Class 12: Retrieval-Augmented Generation (RAG)

Topics:

- What is RAG?
- Why do AI applications need RAG?
- RAG architecture.
- Document ingestion.
- Retrieval and context construction.
- Connecting retrieved information to LLMs.
- Introduction to LangChain.
- Source references and reducing hallucinations.

**PROJECT 3: Build an AI Knowledge Assistant**

Students build a document-based AI assistant that can process uploaded documents and answer questions using RAG.

## PHASE 7: AI Agents and Automation

**Week 7 | Classes 13–14**

### Class 13: AI Agents and Tool Calling

Topics:

- What are AI agents?
- Chatbots vs AI agents.
- Function calling and tool calling.
- Connecting external APIs.
- Giving AI access to tools.
- Multi-step task execution.
- Agent memory and state.
- Understanding agent limitations.

### Class 14: AI Workflows and Automation

Topics:

- AI-powered workflows.
- Sequential and conditional workflows.
- Multi-step automation.
- Introduction to LangGraph.
- Human-in-the-loop systems.
- Workflow reliability.
- Error handling and execution.

**CAPSTONE PROJECT DEVELOPMENT BEGINS**

Students identify a real-world problem and begin designing their own AI application.

## PHASE 8: Deployment and Capstone Execution

**Week 8 | Classes 15–16**

### Class 15: Deployment and Production Fundamentals

Topics:

- Local development vs production.
- Git and GitHub.
- Version control.
- Environment variables.
- API security fundamentals.
- Deploying AI applications.
- Cloud hosting fundamentals.
- Testing and debugging.
- Logging and monitoring.
- Token and cost optimisation.

### Class 16: Capstone Presentation and Evaluation

Topics:

- Final application demonstration.
- Application architecture explanation.
- Project documentation.
- GitHub repository presentation.
- Deployment demonstration.
- Technical feedback.
- Portfolio development.
- Showcasing projects on Trafy.

**FINAL DELIVERABLE: An individual, working AI application presented by the student.**

# 2. Project-Based Learning

The programme follows a progressive project-based learning approach.

Students don't wait until the end of the programme to start building.

They begin implementing what they learn from Week 4 onwards.

| Week | Project | Core skills |
|---|---|---|
| Week 4 | AI Chatbot | LLMs, prompt engineering, APIs |
| Week 5 | Full-Stack AI Application | FastAPI, databases, Streamlit |
| Week 6 | AI Knowledge Assistant | Embeddings, vector databases, RAG |
| Weeks 7–8 | Individual Capstone | Independent AI product development |

## Individual Capstone Project

Every student selects a problem they want to solve and develops an AI-powered application.

Possible project ideas include:

- AI Resume Analyzer.
- AI Study Companion.
- AI Customer Support Assistant.

- AI Personal Finance Assistant.
- AI Research Assistant.
- AI Document Intelligence System.
- AI Workflow Automation Application.

Students can propose their own ideas, subject to mentor approval and realistic project scope.

The capstone should demonstrate their ability to identify a problem, design a solution, implement AI functionality and present a working application.

# 3. Technology Stack

Students will gain practical exposure to the following technologies.

| Category | Technologies |
|---|---|
| Programming | Python |
| Development Environment | VS Code, Google Colab |
| Data Analysis | NumPy, Pandas, Matplotlib |
| Machine Learning | Scikit-learn |
| LLM Integration | LLM APIs |
| Backend Development | FastAPI |
| Frontend Development | Streamlit |
| Databases | SQL, PostgreSQL |
| Vector Databases | FAISS, ChromaDB |
| AI Frameworks | LangChain |
| Agent Workflows | LangGraph |
| Version Control | Git, GitHub |
| Deployment | Cloud hosting platforms |

The emphasis is on understanding and implementing these technologies rather than simply learning their definitions.

# 4. Opportunities Beyond the Classroom

This is one of the most important aspects of the Trafy AI Engineer Cohort.

The objective is to connect technical learning with real-world execution.

## A. Real Product Development Experience

Trafy is building a social networking platform designed to connect:

- Students and graduates.
- Working professionals.
- Founders.
- Trainers and educators.
- Recruiters and hiring teams.

Participants will have opportunities to contribute to the Trafy product and experience how real technology products are developed.

This gives students exposure beyond isolated tutorial projects.

## B. Resume Sharing with FAANG Companies

Resumes of cohort performers will be shared with FAANG companies for consideration.

This is intended to give participants additional visibility and potential career opportunities.

Resume sharing does not guarantee interviews, shortlisting or employment.

## C. MNC Industry Opportunities

Top-performing participants may get opportunities to work with leading MNC companies, subject to selection, eligibility and available opportunities.

## D. Investor Interactions

Students who develop promising product ideas may get opportunities to interact with investors and discuss their ideas.

These interactions are intended to provide exposure to the startup ecosystem. Funding is not guaranteed.

### E. Paid Internship Opportunities at Trafy

The top three performers will have the opportunity to join Trafy as paid interns.

Selected candidates may get the opportunity to work alongside the Trafy team, contribute to product development and gain practical experience.

Internship selection will be based on performance, project quality, consistency and evaluation.

# 5. Learning and Evaluation Framework

Students are expected to participate actively throughout the programme.

The recommended evaluation structure includes:

| Evaluation Area | Purpose |
|---|---|
| Attendance | Consistent participation |
| Weekly assignments | Understanding and implementation |
| Project 1 | Generative AI implementation |
| Project 2 | Full-stack application development |
| Project 3 | RAG implementation |
| Capstone | Independent project execution |
| GitHub contributions | Evidence of technical work |
| Final presentation | Technical understanding and communication |

### Recommended Capstone Evaluation

| Criteria | Weight |
|---|---|
| Problem identification | 10% |
| Application architecture | 15% |
| AI implementation | 25% |

| | |
|---|---|
| Backend and functionality | 20% |
| Deployment and testing | 15% |
| Documentation and presentation | 15% |
| **Total** | **100%** |

Performance across the programme should be considered when identifying top performers and internship candidates.

# 6. Expected Learning Outcomes

By successfully completing the programme and its practical assignments, students should be able to:

1. Write Python programmes independently.
2. Process and analyse datasets.
3. Understand fundamental machine learning concepts.
4. Train and evaluate basic ML models.
5. Understand LLMs and Generative AI.
6. Integrate AI models through APIs.
7. Build backend applications using FastAPI.
8. Work with SQL and PostgreSQL.
9. Build interactive AI interfaces.
10. Implement embeddings and semantic search.
11. Develop RAG-based applications.
12. Understand AI agents and tool calling.
13. Build multi-step AI workflows.
14. Deploy AI-powered applications.
15. Maintain projects using Git and GitHub.
16. Present and explain their own AI applications.

These are intended learning outcomes, not a guarantee of professional mastery or employment after 32 hours of live teaching.