EZ-LEARN GLOBAL × heu.ai

FRESHER TRAIN & DEPLOY PROGRAM

AI Developer Program

Agentic AI Engineering with LangChain & LangGraph

Train. Build. Deploy. Place.

220+	3	6	10
INSTRUCTIONAL HOURS	MONTHS TRAINING	MONTH INTERNSHIP	SEATS PER COHORT

SALARY POST SELECTION

₹3.5 – ₹4.5 LPA

+ 6-month paid internship up to ₹15,000/month • Subject to assessment criteria • 1-year service agreement

LOCATIONS **Chennai • Kochi • Hyderabad**

DELIVERY MODE **Online, Instructor-Led**

Program at a Glance

A three-month, intensive, online instructor-led program that trains fresh graduates to ship production-grade agentic AI systems with LangChain and LangGraph. Top performers progress to a six-month paid internship and a placement offer of ₹3.5–₹4.5 LPA as an AI Developer, subject to meeting assessment criteria.

Audience	Fresh graduates from 2023–2025 batches. No prior AI/ML experience required.
Eligibility	B.Tech / BCA / MCA / M.Tech with 60% throughout academics and good communication skills.
Total Duration	3 months online training (220+ instructional hours) + 6-month paid internship.
Cohort Size	10 selected engineers per batch — small, focused, instructor-driven.
Format	Online instructor-led + hands-on labs + 4 milestone assessments + team capstone.
Tech Stack	Python, LangChain, LangGraph, Vector DBs (Pinecone/ChromaDB/pgvector), FastAPI, Docker, LangSmith, Claude / OpenAI / Gemini APIs.
Assessments	Four milestone assessments + capstone evaluation + final certification.
Outcome	Job-ready AI Developer who can design, build, deploy, and operate agentic AI systems in production.
Placement	₹3.5–₹4.5 LPA post-selection, subject to capstone & assessment performance. 1-year service agreement applies.
Locations	Chennai, Kochi, Hyderabad (post-internship deployment).

Program Outcomes

By the end of the program, every participant will be able to:

- ✓ Build, debug, and deploy multi-agent systems using LangGraph with state management, persistence, and human-in-the-loop checkpoints.
- ✓ Compose production-grade LLM workflows with LangChain — chains, retrievers, memory, tool calling, and structured outputs — using LCEL fluently.
- ✓ Design and ship retrieval-augmented systems (chunking, embeddings, hybrid search, re-ranking) with measurable answer quality.
- ✓ Integrate with frontier LLM APIs (Claude, OpenAI, Gemini) — including streaming, function calling, and cost-aware model routing.
- ✓ Operate AI features safely in production — observability with LangSmith, prompt-injection defence, PII handling, audit logging, and cost controls.

✓ Wrap agents in FastAPI, containerise them, and ship them through a CI/CD pipeline with monitoring and rollback.

✓ Apply Python engineering discipline — typing with Pydantic, testing with pytest, async programming, packaging — to AI codebases.

✓ Deliver a team capstone end-to-end with documented architecture, tests, security review, and a recorded demo.

Why This Program

Built for placement, not for certificates.

The program is structured backwards from the job. Every module produces an artefact that maps directly to what a hiring manager looks for in an AI Developer interview.

Agentic AI is the next hiring wave.

Companies are moving from chatbots to agents. Engineers who can ship LangGraph-based multi-agent systems with proper state management and observability are scarce — and disproportionately valuable.

Frameworks change; engineering doesn't.

We teach Python fluency, typing, testing, async, and packaging as first-class skills. When LangChain evolves (and it will), our engineers adapt; framework-only learners stall.

Production discipline from day one.

Hallucinations, prompt injection, runaway costs, and silent regressions are real risks. We treat them as core curriculum — not as an afterthought after the model call works.

Small cohorts, real instructors.

10 engineers per batch. Live instruction. Live code review. Live failure recovery. No pre-recorded videos. No silent learners.

Train. Build. Deploy. Place.

3 months training → 6-month paid internship (up to ₹15,000/month) → ₹3.5–₹4.5 LPA AI Developer role on selection. The path is the program.

Deliverables & Toolkit

Every participant walks away with the following actionable materials, working code, and shipped artefacts:

#	Deliverable	Format / Description
1	Python Foundations Codebase	Personal Python utility library with typing, testing, and packaging — reused throughout the program.
2	Personal Prompt Library	100+ tested, version-controlled prompt templates for code, retrieval, and agent tasks.

#	Deliverable	Format / Description
3	RAG Reference Implementation	Working end-to-end RAG with chunking, embeddings, hybrid search, re-ranking, and citation.
4	LangChain Agent Portfolio	Multiple LCEL chains and tool-using agents with structured output and LangSmith tracing.
5	LangGraph Multi-Agent System	Supervisor-worker agent system with persistent state and human-in-the-loop checkpoints.
6	Production-Ready Agent API	Agent wrapped in FastAPI, containerised with Docker, instrumented with LangSmith.
7	CI/CD Pipeline	GitHub Actions pipeline with testing, prompt versioning, and deployment automation.
8	AI Security & Governance Checklist	Prompt-injection defences, PII handling, audit logging, cost controls — applied to your stack.
9	Capstone Project	Team-built, end-to-end agentic AI system with architecture docs, tests, security review, and recorded demo.
10	Job-Ready Certification	Issued upon successful capstone evaluation — basis for internship and placement progression.

Curriculum Overview

The program is delivered in 6 progressive modules followed by a team capstone. Each module ends with a milestone assessment that gates progression to the next.

#	Module	Hours
1	Python Foundations for AI	40
2	AI/ML Fundamentals & Retrieval-Augmented Generation	28
3	LangChain Mastery	44
4	LangGraph & Agentic Systems	60
5	Production & Deployment	25
6	Capstone Project — Team-Based Build	40
	Milestone Assessments (4) + Final Evaluation	Included
	TOTAL	237+

INSIGHT

The 220+ headline figure is instructional time only. With milestone assessments, remediation slots, peer-review sessions, and the team capstone, total engaged learner time is approximately 240+ hours over the three-month training window.

Module 1 — Python Foundations for AI

40 hours

Build a rock-solid Python base before touching any LLM or framework. Engineers who skip this step end up debugging Python — not their agents — during capstone. This module covers everything from data types to async programming, type-safety with Pydantic, and testing discipline.

Topic	Theory Content	Hrs
Data Types & Control Flow	Variables, primitives, collections, operators, branching, loops, list comprehensions	3
Functions & Modules	Definitions, args/kwargs, lambda, decorators, closures, built-in & custom modules	3
OOP in Python	Classes, inheritance, polymorphism, abstract classes, magic methods, dataclasses	3
File I/O & Exception Handling	Reading/writing text, CSV, JSON, YAML; context managers; logging module	3
Iterators, Generators, Context Managers	Iterator protocol, yield, contextlib, async generators introduction	3
Data Structures & Algorithms	collections module, heapq, graph basics (BFS, DFS), complexity analysis	4
Virtual Environments & Packaging	venv, pip, poetry, requirements.txt, pyproject.toml, dependency management	3
Working with APIs	HTTP methods, requests, authentication (API keys, OAuth2), rate limiting	4
Async Programming	asyncio, async/await, event loop, aiohttp, threading vs multiprocessing vs async	3
Type Hints & Code Quality	Type annotations, mypy, pydantic, ruff/flake8, black, pre-commit hooks	3
Testing in Python	unittest, pytest, fixtures, mocking, parametrize, coverage, TDD basics	4
Python for Data Handling	pandas (DataFrame, Series), data cleaning, numpy fundamentals, basic visualization	4

MILESTONE ASSESSMENT 1 — PYTHON PROFICIENCY

A timed coding exam covering all Python Foundations topics. Pass mark is required to advance to Module 2 — engineers who do not clear it receive targeted remediation and re-attempt before progressing.

INSIGHT

The single biggest predictor of capstone success is Python fluency, not framework knowledge. This module is non-negotiable; every later module leans on async, typing, and class design that are taught here.

Module 2 — AI/ML Fundamentals & Retrieval-Augmented Generation

28 hours

Set the mental model right before touching LangChain. This module covers the working developer's intuition for how LLMs behave, how embeddings work, and how RAG turns a stateless model into a knowledgeable assistant — without the math overhead of a research course.

Topic	Theory Content	Hrs
Introduction to AI & ML	AI landscape, ML types, neural networks overview, transformers high-level	4
NLP Fundamentals	Tokenization, stemming, lemmatization, TF-IDF, word & sentence embeddings	4
Introduction to LLMs	GPT architecture, attention, pre-training vs fine-tuning, scaling laws, tokens	3
LLM APIs & Providers	OpenAI, Anthropic Claude, Google Gemini, open-source (Llama, Mistral); parameters	4
Prompt Engineering for Developers	Zero/few-shot, chain-of-thought, system roles, templates, structured outputs	3
Vector Databases & Embeddings	Embedding models, similarity metrics, vector DBs (Pinecone, ChromaDB, pgvector)	3
RAG Basics	Architecture, document loading, chunking strategies, context window management	4
Advanced RAG Techniques	Hybrid search, re-ranking, query transformation, multi-query, self-querying	3

MILESTONE ASSESSMENT 2 — RAG IMPLEMENTATION

Build an end-to-end RAG system with chunking, retrieval, and generation. Evaluation is on retrieval quality (recall@k), answer faithfulness, and latency budget — not just whether it runs.

INSIGHT

Most engineers think the LLM is the hard part. The hard part is the retrieval layer — chunking, embeddings, and re-ranking decide whether your agent feels smart or hallucinates.

Module 3 — LangChain Mastery

44 hours

LangChain is the connective tissue between LLMs, tools, retrievers, and memory. By the end of this module, every engineer composes LCEL chains fluently, builds tool-using agents, and instruments everything with LangSmith — the same way production teams ship today.

Topic	Theory Content	Hrs
LangChain Architecture & Core Concepts	Ecosystem, LCEL, Runnables, chains, input/output schemas	4
Models & Prompts	ChatModels, PromptTemplates, MessagesPlaceholder, FewShot, output parsers	4
Chains & LCEL Deep Dive	RunnableSequence/Parallel/Lambda/Branch, fallbacks, batching, streaming	4
Document Loaders & Text Splitters	PDF/web/CSV/DB loaders, RecursiveCharacterTextSplitter, SemanticChunker	4
Retrievers & Vector Stores	MultiQuery, ContextualCompression, Ensemble, SelfQuery; ChromaDB/Pinecone/FAISS	3
Memory & Conversation	BufferMemory, SummaryMemory, ChatMessageHistory, RunnableWithMessageHistory	4
Tools & Tool Calling	Tool interface, @tool decorator, structured tools, binding tools to LLMs	3
LangChain Agents	AgentExecutor, create_tool_calling_agent, ReAct, agent memory, streaming	4
Output Parsing & Structured Output	PydanticOutputParser, JsonOutputParser, with_structured_output, retry parser	3
Callbacks & Observability	Callback handlers, LangSmith tracing, evaluation, token tracking, debugging	4
Building RAG with LangChain	End-to-end RAG, indexing pipeline, stuff/map_reduce/refine, citations	4
Integration Patterns	External services, API chains, SQL chains, web scraping, multi-modal, caching	3

MILESTONE ASSESSMENT 3 — LANGCHAIN IMPLEMENTATION

Build a complex RAG agent with tools, memory, and structured output parsing. The agent must handle ambiguous queries, fall back gracefully, and emit traces consumable in LangSmith.

INSIGHT

LangChain's surface area is large; the engineers who win pick a small set of LCEL primitives and master them. Cleverness with chains rarely beats discipline with retrievers and prompts.

Module 4 — LangGraph & Agentic Systems

60 hours

The largest module in the program. LangGraph is where engineers move from linear chains to graph-based, stateful, human-in-the-loop multi-agent systems. Every concept here maps directly to what production-grade agentic systems require.

Topic	Theory Content	Hrs
Introduction to LangGraph	Why LangGraph, StateGraph, nodes, edges, conditional routing	4
State Management	TypedDict state, Annotated state with reducers, channels, message state	4
Nodes & Edges	Node functions, conditional routing, START/END, branching, transformations	4
Human-in-the-Loop Patterns	Interrupt mechanisms, approval flows, breakpoints, edit-and-resume	4
Persistence & Checkpointing	MemorySaver, SqliteSaver, PostgresSaver, recovery, replay, threading	4
Multi-Agent Systems	Agent-as-node, supervisor, communication, shared state, delegation	4
Advanced Multi-Agent Patterns	Swarm, handoff, routing, collaborative & competitive agents, map-reduce	4
Subgraphs	Nested graphs, composition, parent/child state mapping, modular design	3
Tool-Using Agents	ToolNode, error handling, parallel tool execution, retry strategies	4
Streaming & Real-time	Stream modes (values/updates/messages/events), async, SSE integration	4
LangGraph Platform & Deployment	LangGraph Server, Cloud, assistants API, threads, cron, webhooks	4
Reflection & Self-Correction	Self-reflection, output validation, retry with feedback, critic agents	4
Planning Agents	Plan-and-execute, dynamic replanning, task decomposition, hierarchical	3
RAG Agents with LangGraph	Adaptive RAG, corrective RAG, self-RAG, query routing, iterative retrieval	4
Testing & Evaluating Agentic Systems	Evaluation strategies, LangSmith evaluators, trajectory testing, metrics	3

Topic	Theory Content	Hrs
Security & Production Considerations	Prompt injection defence, tool permissions, rate limiting, sandboxing, audit	3

MILESTONE ASSESSMENT 4 — AGENTIC AI SYSTEM DESIGN

Design and implement a complex multi-agent system in LangGraph: supervisor + workers, persistent state, human approval checkpoint, graceful failure on tool errors, and a recorded trace walkthrough.

INSIGHT

Agentic systems fail at the seams — bad plans, runaway loops, hallucinated tool calls, missing checkpoints. Engineers who ship them invest as much in the loop's guardrails as in the loop itself.

Module 5 — Production & Deployment

25 hours

The work that separates a notebook demo from a system someone pays for. FastAPI, containerisation, CI/CD, observability, scaling, cost controls, and responsible AI — covered as first-class engineering concerns, not as afterthoughts.

Topic	Theory Content	Hrs
FastAPI for Agent APIs	Async endpoints, WebSocket, Pydantic models, DI, middleware, CORS	4
Containerisation & Docker	Dockerfile, multi-stage builds, docker-compose for agent stack, health checks	4
CI/CD for AI Applications	GitHub Actions, testing pipelines, model & prompt versioning, rollback	3
Monitoring & Observability	LangSmith in production, custom metrics, alerting, cost & latency tracking	4
Scaling Agentic Systems	Horizontal scaling, queues (Celery/Redis), caching, connection pooling	3
Cost Optimisation & Performance	Token optimisation, response caching, model routing, batching, compression	4
Ethics, Responsible AI & Best Practices	Bias, hallucination management, transparency, PII handling, regulation	3

PRODUCTION HARDENING SPRINT

Engineers take their best agent from Modules 3–4 and harden it: wrap in FastAPI, containerise, add CI/CD, instrument with LangSmith, define cost budgets, and document a responsible-AI checklist. Reviewed end-to-end before capstone selection.

INSIGHT

Production AI fails silently — wrong answers, runaway costs, and regression on edge cases never throw exceptions. The discipline taught here is the difference between a project and a product.

Module 6 — Capstone Project — Team-Based Build

40 hours

The integration test of everything in the program. Teams of 2–3 engineers ship an end-to-end agentic AI system over four sprints — with documented architecture, tests, security review, and a recorded demo. Capstone performance is the primary input to placement decisions.

Sprint	Focus	Hrs
Sprint 0 — Kickoff & Architecture	Project scoping, system design, technology selection, success metrics	8
Sprint 1 — Core Build	Core agentic workflow, tool integration, RAG setup, state management, LangGraph design	16
Sprint 2 — Hardening	Testing, error handling, monitoring setup, UI/API layer, deployment prep	8
Demo Prep & Mock Run	Rehearsal, presentation technique, architecture explanation, Q&A;	4
Final Evaluation	Live demo, code review, architecture defence, project discussion	4

FINAL EVALUATION — JOB-READY AGENTIC AI ENGINEER CERTIFICATION

Comprehensive assessment of all course competencies plus capstone quality. Top-performing engineers progress to the 6-month paid internship (₹15,000/month) and are considered for placement at ₹3.5–₹4.5 LPA, subject to meeting assessment criteria.

INSIGHT

Capstone is where intent meets execution. Engineers who treated earlier modules as deliverables — not lectures — show up to capstone with code they can be proud of, and they place.

Delivery Methodology

Build, don't watch.

Every module ends with the engineer having built or shipped something running on their own machine. Theory is paired tightly with code.

Live instruction, not pre-recorded videos.

All teaching is live. All demos are unscripted. Engineers see real failure modes, real recoveries, and real engineering judgement — not polished slideware.

Pairing and code review.

Hands-on exercises rotate between solo, pair, and team work. Code review is a first-class part of every module, not an afterthought.

Milestone-gated progression.

Four milestone assessments gate movement between modules. Engineers who do not clear a milestone receive targeted remediation before progressing.

Eval-driven assessment.

Engineering output is measured, not just submitted. Prompts, retrievers, and agent trajectories are scored against eval suites — the same discipline taught in the program.

Small cohort, deep attention.

10 engineers per cohort with one lead instructor and one TA. No engineer goes silent; no question goes unanswered.

Internship integration.

Selected engineers move into a 6-month paid internship (up to ₹15,000/month) applying capstone skills to real client work delivered through Ez-Learn Global and heu.ai.

Tools & Technologies Covered

Tool / Technology	Role in the Program
Python 3.11+	Primary language. Typing, async, packaging, testing — production-grade discipline.
LangChain	LCEL chains, retrievers, memory, tool calling, structured output, callbacks.
LangGraph	Stateful, graph-based, multi-agent systems with persistence and human-in-the-loop.
LangSmith	Tracing, evaluation, monitoring, debugging — for both development and production.
Claude API (Anthropic)	Agentic workflows, extended thinking, tool use, structured outputs.
OpenAI API	GPT model family, function calling, structured outputs, embeddings.

Tool / Technology	Role in the Program
Google Gemini API	Long-context use cases and multi-modal experimentation.
Vector Databases	Pinecone, ChromaDB, Weaviate, pgvector — embedding storage and retrieval.
FastAPI	Async API layer for serving agent endpoints with WebSocket and streaming support.
Docker & docker-compose	Containerisation of agent stack — API, vector DB, cache, observability.
GitHub Actions	CI/CD pipelines with testing, prompt versioning, and deployment safety.
pytest, mypy, ruff, black	Testing, type-checking, linting, and formatting — applied to AI codebases.
Pydantic	Schema-driven structured outputs and validation across the stack.
Celery / Redis	Queue-based scaling patterns for production agentic workloads.

Key Takeaways

✓ Agentic AI engineering is a new discipline — engineers who treat it as one (with Python rigour, eval discipline, and production hygiene) compound their advantage every quarter.

✓ Frameworks are temporary; engineering is durable. The Python foundations module is the highest-ROI investment in this program.

✓ LangChain teaches composition; LangGraph teaches systems thinking. Together they cover the surface area of every modern agentic application.

✓ Retrieval quality, not model choice, decides whether your agent feels smart. Master chunking, embeddings, hybrid search, and re-ranking before reaching for a bigger model.

✓ Production AI fails silently. Observability, evals, cost controls, and prompt-injection defence are taught as first-class topics — not afterthoughts.

✓ Train. Build. Deploy. Place. The program path leads to a job, not a certificate.

Apply

Only 10 seats per cohort. Selection is on a first-eligible basis with a screening call and a short technical aptitude test.

Eligibility	B.Tech / BCA / MCA / M.Tech, 2023–2025 passouts, 60% throughout, good communication skills.
Application	Email your resume to contactus@ez-learn.global with the subject line 'AI Developer Program – [Your City]'.
Phone	+91 9080872896
WhatsApp	+91 9884146187
Website	www.ez-learn.global
Delivered with	heu.ai — www.heu.ai
Service Agreement	1-Year Service Agreement applies on placement.

Build the AI engineering career that compounds.

Ez-Learn Global × heu.ai • AI Developer Program • Fresher Train & Deploy