



AI BUILDER • LLM APPS & RAG FOUNDATIONS

Build AI Apps. Think Like an AI Engineer.

6 Months • Weekly 60-Minute LIVE Class • Small Skill-Based Cohorts

Designed for students with Python foundations who are ready to connect LLMs to code, data and tools — and build useful RAG applications.

100% LIVE

AI ENGINEERING LAB

PROJECT-BASED

AI ENGINEERING MINDSET

Strong Python foundations let students move from using AI to engineering real AI applications.

LLM APIs → structured outputs → RAG → tools → AI workflows.



PROGRAM OVERVIEW

Six months. From AI user to confident AI Builder.

24 LIVE CLASSES

Approximately 24 weekly sessions
60 minutes each
Hands-on coding every week
Real-time instructor guidance

AI APPLICATION ENGINEERING

LLM APIs → structured outputs →
embeddings → retrieval → RAG → tools → AI
workflows

Students move beyond standalone prompts
into coded AI systems.

LEARN BY BUILDING

AI Engineering Lab • evaluation • LLM apps •
RAG • tool use • automation • capstone

Every concept becomes part of a larger
system.

OUTCOME

Students finish able to design, build, evaluate and debug grounded AI applications in Python with growing independence.



MONTHS 1-2

AI APPLICATION ENGINEERING + STRUCTURED THINKING

01 • LLM APIs + APP DESIGN

model APIs • messages • parameters • environment variables • error handling • reusable AI functions

AI Lab: turn one-off prompts into reusable Python components

02 • STRUCTURED OUTPUTS + EVALUATION

JSON/schema concepts • extraction • classification • rubrics • test cases • output validation

Build: classifier, extractor and structured study assistant

GOAL Students stop treating AI as a chat window and start designing it as reliable, testable parts of a program.

MONTHS 3–4

EMBEDDINGS + RETRIEVAL + AI DEBUGGING

03 • EMBEDDINGS + SEMANTIC SEARCH

embeddings intuition • similarity • chunking
concepts • vector search • metadata • retrieval
quality

Build: search and Q&A over a small curated
knowledge set

04 • RAG FOUNDATIONS

retrieval → context → generation • citations •
grounding • prompt templates • failure modes

Build: a grounded assistant that answers from
provided content



AI DEBUGGING LAB

Trace bad answers • inspect retrieved context • log
prompts and outputs • read tracebacks • live
debugging • repair broken AI pipelines

AI challenge: verify and fix generated code instead
of trusting it.

GOAL Build AI apps that answer from real content, cite their sources and can be debugged systematically instead of by guessing.



MONTHS 5–6

TOOLS + WORKFLOWS + REAL AI APPLICATIONS

05 • TOOL USE + EXTERNAL DATA

tool/function calling concepts • external APIs • JSON • safe parameters • failures • permission boundaries

Build AI apps that can retrieve information or perform controlled actions.

06 • AI WORKFLOWS + AI BUILDER CAPSTONE

multi-step AI workflows • retrieval • tools • evaluation • GitHub • documentation • final AI Builder capstone

Plan → design → build → evaluate → document → present.

GOAL Combine LLM APIs, retrieval, tools and evaluation into a useful AI application the student can defend and extend.



LEARNING BY BUILDING

Not just prompts — AI Builders create grounded, testable AI apps.

Structured AI Apps

schemas • validation • evaluation

LLM API Apps

models • JSON • error handling

RAG Assistants

retrieval • context • citations

Tool-Using Apps

APIs • tools • permissions

AI Engineering + Debugging Lab

evaluation + tracing + diagnosis

AI Builder Capstone

design • build • evaluate • document

GitHub becomes part of the workflow: organize versions, document projects and present work like a developing software engineer.



AI BUILDER CAPSTONE

Real problems. Grounded AI solutions.

RAG STUDY ASSISTANT

Course notes • chunking • retrieval •
grounded answers • citations • evaluation

Designed to answer from a controlled
knowledge base.

AI RESEARCH ORGANIZER

Gather approved sources • extract
structured facts • compare results • cite
context • handle uncertainty

Great for learning retrieval, structure and
verification.

MY OWN IDEA

Student proposes an AI application, scopes
it with Ankit, builds a realistic Builder version
and documents it on GitHub.

The capstone proves the student can combine multiple concepts, make design decisions, debug independently and explain how the system works.



WHAT COMES NEXT

AI Builder is where prompting becomes AI application engineering.

1 • AI EXPLORER

AI literacy checkpoint
prompt + context foundations
AI Engineering + Debugging Lab
AI Explorer projects

2 • AI BUILDER

LLM APIs + RAG
structured outputs + retrieval
tools + workflows
AI Builder capstone

3 • AI CREATOR

agents + orchestration
advanced RAG
evaluation + guardrails
portfolio AI systems

Recommended placement: students should demonstrate AI Explorer readiness plus working Python fundamentals before AI Builder. Experienced students can take a readiness assessment for direct placement.

Know where your child stands in the age of AI → thinkpythonai.com/will-ai-take-my-kids-job



THINKPYTHONAI YOUNG CODERS

From “I can prompt AI” to “I can build AI applications.”

AI BUILDER • 6 MONTHS • LIVE • SMALL SKILL-BASED COHORTS

LLM APIs • RAG • AI Debugging Lab • tools • evaluation • GitHub • final capstone

Young Coders Coordinator: Crystal

+1 (650) 444-3020

[AI Readiness Quiz: thinkpythonai.com/will-ai-take-my-kids-job](https://thinkpythonai.com/will-ai-take-my-kids-job)

THE AI BUILDER PROMISE

We move beyond prompting.

Students learn to engineer grounded AI applications.

They evaluate systematically, connect models to data and tools, and build with increasing independence.

Then they are ready for AI Creator: agents, advanced RAG and AI systems.

Strong Python foundations let students move from using AI to engineering real AI applications.