35-Day Agentic AI Engineer Roadmap

6 hrs/day (1h15 LeetCode & mini-projects, 4h45 main study). Theory + hands-on + production projects — print this and check off daily.

Daily split: **1h15 LeetCode** + **4h45 Main Study**
Outcome: Production-grade Agentic AI Platform (RAG + Multi-Agent + Deployment)
Commitment: Type code yourself; weekly & final capstone projects

How to use this sheet

- Each day: follow the *theory → coding → mini-project* flow. Write all code yourself (no copy/paste).
- Use Notion / notebook to log daily reflection (0.5 hr). Mark completed tasks with the checkbox.
- Weekend mini-projects integrate the week's learnings — commit them to GitHub with tests & README.

Daily time allocation (6 hrs)

1h15 — LeetCode & Mini Projects

Focus on algorithmic problems (strings, arrays, hashing, sliding window, recursion, basic DP) and short utility mini-projects.

4h45 — Main Study (Theory + Hands-on)

~1.5h theory & notes, ~3h coding & experiments, ~1h integration of day's work, ~0.5h reflection.

Week 1 — Core Foundations (Days 1–7)

Day 1 — Python Foundations

- Theory: OOP, error handling, logging, type hints, packaging (venv/poetry), pytest basics.
- Hands-on: Build a CLI tool that fetches APIs and logs structured output.
- Project: CLI API Fetcher — include unit tests and README.

Done

Day 2 — Advanced Python

- Theory: generators, decorators, context managers, async/await fundamentals.
- Hands-on: Async API caller with concurrency, retries, and structured logging.
- Project: Async Fetcher with retry/backoff + simple local cache.

Done

Day 3 — Data Structures & Algorithms for AI

- Theory: arrays, hashmaps, heaps, stacks/queues, recursion, BFS/DFS, time & space complexity.
- Hands-on: Implement BFS/DFS and a simple top-N recommender using heaps.
- Project: small recommender (top-K similar items) → include tests.

Done

Day 4 — Statistics

- Theory: distributions, mean/variance, Bayes basics, hypothesis testing, p-values, confidence intervals.
- Hands-on: EDA + statistical tests on Titanic or Iris dataset; write observations.
- Project: Statistical analysis notebook with visualizations.

Done

Day 5 — Linear Algebra & Calculus

- Theory: vectors, matrices, dot product, matrix multiplication, derivatives, gradients.
- Hands-on: Implement gradient descent and simple linear regression from scratch (NumPy).
- Project: Gradient descent notebook with visualization of loss over iterations.

Done

Day 6 — Machine Learning Fundamentals

- Theory: linear/logistic regression, decision trees, ensembles, overfitting, bias-variance tradeoff.
- Hands-on: Build sklearn pipeline (preprocess → model → evaluate) and compare metrics (F1, ROC-AUC).
- Project: Churn or spam classifier with cross-validation and model comparison.

Done

Day 7 — Weekend Mini-Project: End-to-End ML

- Integration: EDA → preprocessing → model training → evaluation → simple API to serve prediction.
- Project: "Customer Churn Prediction" — include README, tests, saved model, and simple FastAPI endpoint.

Done

Week 2 — Deep Learning & LLMs (Days 8–14)

Day 8 — Neural Networks from Scratch

- Theory: perceptron, activation functions, forward/backprop, loss functions.
- Hands-on: Implement a small neural network in NumPy (forward + backward) for a toy task.
- Project: NumPy neural net with training loop & loss visualization.

Done

Day 9 — PyTorch Basics

- Theory: tensors, autograd, optimizers, training loop, saving/loading models.
- Hands-on: Build and train an MNIST classifier in PyTorch.
- Project: PyTorch MNIST training + inference API.

Done

Day 10 — CNNs & RNNs

- Theory: convolution, pooling, RNN/LSTM basics and when to use them.
- Hands-on: Simple CNN for images and simple LSTM for sequence classification.
- Project: Image classifier + short text classifier demos.

Done

Day 11 — Transformers & Attention

- Theory: attention mechanism, encoder vs decoder, context windows.
- Hands-on: Implement a small attention block / toy transformer encoder in PyTorch.
- Project: Attention visualization & tiny transformer experiment.

Done

Day 12 — Hugging Face & Tokenizers

- Theory: tokenization strategies (BPE, WordPiece), tokenizers API, model pipelines.
- Hands-on: Load pretrained models, run summarization and Q&A pipelines locally.
- Project: Small app to compare tokenizer outputs & model inference.

Done

Day 13 — Fine-Tuning LLMs (LoRA / PEFT)

- Theory: fine-tuning approaches, parameter-efficient tuning (LoRA/PEFT).
- Hands-on: Fine-tune a small model (or run a LoRA demo) on a toy dataset.
- Project: Fine-tuning notebook & evaluation.

Done

Day 14 — Weekend Mini-Project: Q&A Bot

- Integration: Fine-tuned model or pretrained HF model serving as Q&A engine with a simple UI/endpoint.
- Project: Q&A app (Streamlit or FastAPI) with example docs.

Done

Week 3 — Agentic AI Core (Days 15–21)

Day 15 — RAG Fundamentals

- Theory: embeddings, vector spaces, dense vs sparse retrieval, chunking strategies.
- Hands-on: Build a FAISS index for a set of documents and run retrieval experiments.
- Project: Tiny RAG pipeline (embedding → retrieve → prompt → LLM call).

Done

Day 16 — LangChain Basics

- Theory: chains, tools, memory, agent patterns in LangChain.
- Hands-on: Create a LangChain agent that uses a calculator tool + retriever.
- Project: LangChain small agent demonstrating tool use.

Done

Day 17 — LlamaIndex & Data Connectors

- Theory: index types, connectors for PDFs / web / Google Drive.
- Hands-on: Ingest documents, create searchable index, compare responses.
- Project: PDF/web data ingestion + Q&A demo.

Done

Day 18 — Multi-Agent Systems

- Theory: planner-worker, debate/committee, supervisor-specialist patterns.
- Hands-on: Implement a planner that decomposes tasks and worker agents that execute subtasks.
- Project: Planner + Worker demo with logs & traces.

Done

Day 19 — Observability & Evaluation

- Theory: metrics for LLMs (accuracy, grounding, hallucination rate), observability tooling (LangSmith, W&B basics).
- Hands-on: Instrument agent code to collect traces, tokens, latencies & simple evaluation scripts.
- Project: Agent trace collector + evaluation harness.

Done

Day 20 — Project: Multi-Agent Research Assistant

- Build: Planner + Searcher + Summarizer agents working together to answer research queries.
- Deliverable: Repo with README, tests, sample inputs and outputs.

Done

Day 21 — Project: Marketing Content Generator (Epsilon Focus)

- Build: RAG + template prompts to generate compliant, grounded marketing content.
- Deliverable: Content generator with examples and evaluation metrics (relevance / safety).

Done

Week 4 — Backend, Cloud & Deployment (Days 22–28)

Day 22 — FastAPI & REST

- Theory: REST conventions, async endpoints, request validation with Pydantic.
- Hands-on: Serve a model as REST API, add health & status endpoints.
- Project: FastAPI model service with unit tests and docs.

Done

Day 23 — WebSockets & Streaming

- Theory: WebSocket protocol, streaming token-by-token responses, client connection management.
- Hands-on: Implement a streaming endpoint with FastAPI that sends LLM tokens to the client.
- Project: Streaming chat demo (frontend or a simple WebSocket client).

Done

Day 24 — Docker & Containerization

- Theory: Dockerfile best practices, multi-stage builds, networking, volumes.
- Hands-on: Containerize FastAPI + agent services and test with docker-compose.
- Project: Dockerized agent stack with a compose file.

Done

Day 25 — CI/CD (GitHub Actions)

- Theory: build/test/deploy pipelines, artifact management.
- Hands-on: Create GH Actions workflow to run tests, lint, build docker image, and push to registry.
- Project: CI pipeline (tests & build) and manual deploy step.

Done

Day 26 — AWS Core Services

- Theory: S3, EC2, IAM, Lambda, API Gateway basics and security best practices.
- Hands-on: Upload docs to S3, retrieve them in your app; launch a small EC2 to host container.
- Project: Simple EC2 deployment and S3 document store integration.

Done

Day 27 — AWS Bedrock & SageMaker (LLM infra)

- Theory: Managed LLMs (Bedrock) and SageMaker deployment pipelines (optional depending on access).
- Hands-on: If Bedrock not available, integrate with OpenAI/Anthropic APIs securely; optionally try SageMaker endpoints.
- Project: Agent configured to call LLM provider with credentials stored securely.

Done

Day 28 — Project: Deploy RAG Agent (FastAPI + WebSocket + AWS)

- Integration: Dockerize service, deploy to EC2 or container service, connect S3 documents, expose REST + WebSocket endpoints.
- Project: Staging deployment with simple smoke tests and README for deploy steps.

Done

Week 5 — Scaling, Monitoring & Capstone (Days 29–35)

Day 29 — System Design for AI

- Theory: caching strategies (Redis), message queues (RabbitMQ/Kafka), load balancing, scaling patterns.
- Hands-on: Design and prototype cache + queue pattern for RAG requests.
- Project: Design doc + small prototype using Redis for cache and a lightweight queue.

Done

Day 30 — Monitoring & Observability

- Theory: OpenTelemetry, Prometheus metrics, Grafana dashboards, alerting basics.
- Hands-on: Instrument FastAPI service to emit traces & metrics; create a sample dashboard.
- Project: Observability pipeline with basic dashboards and alert rules.

Done

Day 31 — Security & Reliability

- Theory: secrets management (AWS Secrets Manager / Vault), rate limiting, auth (JWT), input validation.
- Hands-on: Add JWT auth, secrets fetching, and input sanitization to your API.
- Project: Secure agent API with role-based access and sanitized inputs.

Done

Day 32 — Project: Multi-Agent Platform with Monitoring

- Integration: Planner + workers + RAG + memory + telemetry.
- Project: End-to-end multi-agent system with logs, metrics, and simple UI for demonstration.

Done

Day 33 — Capstone Part 1: Design & Implement

- Design: Architecture diagram, component responsibilities, data flow, scaling plan.
- Implement: Scaffold the multi-agent platform (LangChain / custom orchestration), RAG index, memory store.
- Project: Core platform scaffolding + basic end-to-end happy path.

Done

Day 34 — Capstone Part 2: Deploy & Scale

- Deploy: Containerized platform to AWS, enable autoscaling, configure caching and queues.
- Verify: Load tests, fallback policies, cost estimation and optimizations.
- Project: Deployed platform with basic autoscale and fallback strategy.

Done

Day 35 — Capstone Part 3: Demo & Documentation

- Deliverables: Architecture doc, runbook, deployment steps, README, demo video (3–5 mins) and Day-1/30/90 plan for Epsilon.
- Publish: Push final code to GitHub, create release, write a short tech blog or LinkedIn post summarizing learnings.
- Project: Complete capstone package ready for presentation and interview discussion.

Done

Good luck — be consistent. Build daily, test, and reflect. You're ready to be production-grade for agentic AI.

Tip: Keep daily logs and short screencast (2–5 min) each weekend to teach back what you learned — this cements mastery.