Forward Deployed Engineer (FDE) Master Syllabus

WEEK 1 - Python for Production

Goal: Write clean, typed, tested Python you can ship inside a customer's repo in week one.

Day 01: Code quality & structure: PEP 8 style, meaningful naming, small single-purpose functions, clean project layout, keeping configuration out of code.

Day 02: Packaging & environments: virtual environments, pip and pyproject.toml, dependency pinning, uv / Poetry awareness, building an installable internal package.

Day 03: Type safety & data validation: end-to-end type hints, mypy checks, Pydantic v2 models, field validators, settings loaded from environment variables.

Day 04: Async, testing & debugging: asyncio event loop, async/await, gather, semaphores, threads vs processes vs async; pytest fixtures and parametrization, mocking external APIs, coverage, profiling with cProfile.

Day 05: Reading unfamiliar codebases: finding entry points, reading tests first, tracing calls, safe small refactors, searching a large repo effectively.

Day 06: Hands-on: package a small data CLI with typed Pydantic config, an async API fetcher, and pytest coverage of the core logic.

Day 07: Review & practice: refactor your CLI for style, typing and tests; peer code review.

WEEK 2 - SQL, Backend APIs & Cloud Tooling

Goal: Explore any customer database, expose it through an API, and deploy it without waiting on a DevOps team.

Day 08: Core SQL fluency: inner, outer and cross joins, aggregations, GROUP BY pitfalls, CTEs for readable layered logic; window functions - ROW_NUMBER, RANK, LAG / LEAD, running totals.

Day 09: Schema design & query performance: normalization (1NF-3NF), keys and constraints, when to denormalize, ER diagrams; EXPLAIN plans, index types, avoiding N+1, partitioning awareness.

Day 10: Warehouses & unknown databases: PostgreSQL in depth; how Snowflake and BigQuery differ; information_schema queries, row counts, sampling, null and duplicate profiling. Hands-on: profile a mystery sales database and answer ten business questions with documented SQL.

Day 11: REST design & FastAPI: resources, verbs, status codes, versioning, pagination; routes, request/response models, dependency injection, background tasks, automatic OpenAPI docs.

Day 12: Event-driven integration & authentication: webhook receivers with signature verification, queues vs pub-sub, SQS / RabbitMQ / Kafka awareness; API keys, OAuth 2.0 flows, JWT structure and validation, SSO / SAML concepts.

Day 13: Resilience & connector building: rate limits, retries with backoff and jitter, idempotency keys, cursor vs offset pagination; wrapping a third-party API in a clean, testable Python client. Hands-on: FastAPI service with JWT auth plus a public-API connector with retries and pagination.

Day 14: Git, Linux & containers: branching strategies, pull requests, code review, conventional commits; permissions, processes, pipes, grep/sed/awk, bash scripting, cron; Dockerfiles, images, layers, docker-compose, Kubernetes awareness.

WEEK 3 - Data Pipelines & Enterprise Integration

Goal: Move a customer's messy data into clean, monitored tables they trust.

Day 15: ETL vs ELT & batch vs streaming: what changes when the warehouse does the transforming; schedules, micro-batches, change-data-capture concepts, when streaming is overkill.

Day 16: Data quality: cleaning rules, deduplication keys, null strategy, validation checks, quarantining bad rows instead of dropping them silently.

Day 17: Unstructured data: parsing PDFs, emails and scanned documents into usable rows (pdfplumber, OCR awareness).

Greens Technologies

Day 18: Orchestration: Airflow DAGs and Prefect flows, scheduling and backfills; dbt models awareness.

Day 19: Hands-on: a pipeline from CSV + REST API into PostgreSQL with quality checks and a quarantine table.

Day 20: Enterprise systems landscape: Salesforce (accounts, opportunities), SAP (ERP), ServiceNow (tickets), Workday (people), SharePoint, Google Workspace, Microsoft 365.

Day 21: Reading API docs under constraints: sandbox limits, throttling, auth quirks, version differences.

WEEK 4 - Enterprise Integration & Security Compliance

Goal: Get data flowing inside a restricted enterprise stack, and clear the customer's security review.

Day 22: Middleware & patterns: point-to-point vs hub-and-spoke; iPaaS awareness - MuleSoft, Boomi, Zapier, n8n.

Day 23: Restricted environments: VPNs, jump hosts, on-prem agents, IP allowlisting, working through customer IT.

Day 24: Ontology / schema mapping: mapping customer fields to the product's data model, resolving conflicts and gaps. Hands-on: integrate a CRM-style API and map its objects to a canonical schema, with a written mapping document.

Day 25: Access control & PII handling: least-privilege access, role-based access control, scoping service accounts tightly; identifying PII, masking, minimization, data-residency expectations.

Day 26: Compliance awareness & security reviews: what SOC 2, GDPR, HIPAA and India's DPDP Act ask of a deployment; questionnaires, evidence, working with a customer's security team.

Day 27: Audit logging: what to log, retention, tamper-evidence.

Day 28: Hands-on: add PII masking and audit logging to your Week-1 service; complete a sample customer security questionnaire.

WEEK 5 - AI & LLM Fundamentals

Goal: Explain to a customer, in plain language, what an LLM can and cannot do on their problem.

Day 29: The landscape: AI vs ML vs deep learning; supervised, unsupervised and reinforcement learning in one picture.

Day 30: How models work: neural networks and transformer architecture basics - attention, layers, parameters.

Day 31: Tokens to vectors: tokenization, embeddings, semantic similarity, vector representations.

Day 32: How LLMs are trained: pre-training, instruction tuning, RLHF - and why each stage changes behavior.

Day 33: Model landscape: closed vs open-weight models, reasoning models, context windows, temperature, top-p.

Day 34: Limits: hallucinations, knowledge cutoffs, failure modes; model selection across quality, cost and latency.

Day 35: Hands-on: run the same ten customer-style prompts through three models; log quality, latency and cost per call.

WEEK 6 - Prompt Engineering & LLM APIs

Goal: Get reliable, structured, affordable output from any major model API.

Day 36: Prompting craft: zero-shot and few-shot prompting; system, user and assistant message roles; instruction placement.

Day 37: Structure: task decomposition, chain-of-thought concepts, delimiters and output formatting.

Day 38: Structured output: JSON schemas, constrained decoding, validation and repair of bad outputs.

Day 39: Prompt ops: templates, versioning, A/B comparison, regression sets for prompt changes.

Day 40: APIs in practice: OpenAI, Anthropic and Gemini APIs; running open-weight models with Ollama and vLLM.

Greens Technologies

Day 41: Production behaviors & cost control: streaming, function / tool calling, error handling, retries, rate limits; prompt caching, batch processing, token budgeting.

Day 42: Hands-on: one script calling two providers with identical prompts - structured JSON output, caching enabled, cost per call logged.

WEEK 7 - Retrieval-Augmented Generation (RAG)

Goal: Put a trustworthy 'ask our documents' assistant in front of a customer.

Day 43: Why RAG & the full architecture: grounding answers in customer data instead of model memory; load, chunk, embed, retrieve, generate, cite.

Day 44: Document loading: PDFs, HTML, tables, scanned files; preserving structure while parsing.

Day 45: Chunking: sizes, overlap, metadata, per-document-type strategies.

Day 46: Retrieval: embedding models; FAISS, Chroma and pgvector; semantic vs keyword vs hybrid search; reranking.

Day 47: Trust: grounded answers with source citations; permission-aware retrieval over enterprise documents.

Day 48: Beyond documents & agentic RAG: text-to-SQL over customer databases; query rewriting, corrective retrieval, answer verification.

Day 49: Hands-on: a RAG assistant over policy PDFs with citations and a 50-question evaluation set.

WEEK 8 - Model Customization

Goal: Advise a customer on when fine-tuning is worth it - and when it isn't.

Day 50: Choosing the lever: prompting vs RAG vs fine-tuning - cost, effort, and when each wins.

Day 51: Datasets: sourcing, cleaning and formatting instruction data.

Day 52: Fine-tuning concepts: supervised fine-tuning, LoRA / QLoRA, preference tuning (DPO) - what each actually changes.

Day 53: Multimodal AI: vision-language models, speech-to-text, text-to-speech, document OCR.

Day 54: Hands-on: build a small instruction dataset, then compare base vs prompted vs RAG answers on the same task.

Day 55: Project work: apply the right customization lever to a customer-style task and document why you chose it.

Day 56: Review & assessment: customization trade-offs clinic - given five customer scenarios, pick and defend the lever.

WEEK 9 - Building AI Agents

Goal: Build agents that are reliable, observable, and safe to run inside a customer's environment.

Day 57: Gen AI vs agentic AI: workflows vs autonomous agents, and where each fits.

Day 58: Anatomy of an agent: reasoning model, instructions, tools, memory, stopping conditions.

Day 59: The loop: the ReAct pattern; observation-action-feedback cycles.

Day 60: Reliability: planning, replanning, reflection and verification.

Day 61: Memory & context engineering: short-term, long-term, summarization, checkpoints.

Day 62: Failure handling & safety: failed tool calls, timeouts, iteration limits, graceful stops; human-in-the-loop approval for sensitive actions.

Day 63: Hands-on: an agent that answers questions over a database and asks for approval before any write operation.

WEEK 10 - Agent Frameworks, Tools & Protocols

Goal: Use the frameworks and protocols real customer agent projects are built on.

Day 64: LangGraph in depth: stateful graphs, persistence, checkpoints, human-approval nodes.

Greens Technologies

Day 65: LangChain & LlamaIndex: integrations and retrieval building blocks.

Day 66: CrewAI & Microsoft Agent Framework: role-based multi-agent teams; successor to AutoGen and Semantic Kernel (awareness).

Day 67: Vendor SDKs: OpenAI Agents SDK, Claude Agent SDK, Google ADK (awareness).

Day 68: Model Context Protocol (MCP): building servers and clients that connect agents to customer tools.

Day 69: Beyond: agent-to-agent (A2A) protocol concepts; computer-use, browser, coding and voice agents (awareness).

Day 70: Hands-on: the same agent rebuilt in LangGraph, then exposed as an MCP server.

WEEK 11 - Multi-Agent Systems

Goal: Design, build and debug an agent system a customer can run every day.

Day 71: Patterns: supervisor-worker and hierarchical designs; sequential vs parallel execution.

Day 72: Coordination: shared state, delegation, result aggregation.

Day 73: Debugging: tracing multi-agent runs and evaluating workflow quality.

Day 74: Long-running work: event-driven and scheduled agent workflows; idempotency across retries.

Day 75: Hands-on: a supervisor with two worker agents that completes a multi-step research task.

Day 76: Project work: extend your multi-agent system with a third worker and an approval step.

Day 77: Review & assessment: design review - defend your agent architecture for a customer scenario.

WEEK 12 - AI Security, Scoping & Evaluation

Goal: Prove - with numbers a customer trusts - that a system works and is safe to deploy.

Day 78: AI security: prompt injection - direct attacks and indirect injection through retrieved content; data leakage paths, PII in prompts and logs.

Day 79: Permissions, guardrails & traceability: least privilege for tools, sandboxing agent execution; output validation, content filters, refusal behavior; audit logs, run traces; OWASP Top 10 for LLM Applications. Hands-on: red-team your RAG app with ten injection attempts; patch whatever breaks.

Day 80: Scoping AI use cases: where AI adds value vs where rules or plain automation suffice; feasibility testing on real customer data before promising anything.

Day 81: Estimation & decisions: cost, latency and accuracy envelopes worked out before building; build vs buy vs configure. Hands-on: scope two real use cases end-to-end on paper - problem, data, feasibility, cost.

Day 82: Evaluation customers trust: building eval sets with customer subject-matter experts; metrics tied to business outcomes, not model vanity scores.

Day 83: Methods & observability: LLM-as-judge plus human review; RAGAS for RAG; agent task-completion rates; tracing with LangSmith, Langfuse, OpenTelemetry.

Day 84: Iteration & economics: error analysis, regression testing, improvement loops; latency and token-cost tracking; reporting accuracy and limitations plainly. Hands-on: a full evaluation report on your RAG app - scores, failure modes, cost, and a go / no-go recommendation.

WEEK 13 - Discovery & Solution Design

Goal: Find the real problem and design the answer the customer says yes to.

Day 85: Discovery calls: structure, question banks, listening for the real problem.

Day 86: Stakeholder interviews & workflow mapping: mapping who cares and why; current state, pain points, volumes and costs.

Day 87: Framing & prioritization: problem statements and measurable success criteria; impact vs effort, picking the first win.

Day 88: Architecture diagrams: the one picture everyone aligns on.

Day 89: Design docs and scope docs: what belongs in each, and how much is enough.

Greens Technologies

Day 90: Decisions & planning: trade-off analysis and architecture decision records; timeline estimates and risk registers.

Day 91: Hands-on: turn a mock discovery call into a one-page scope doc and an architecture diagram.

WEEK 14 - Prototyping, Demos & Stakeholder Management

Goal: Own the room - from the first prototype to the executive readout.

Day 92: Prototyping speed: a working demo in days, not weeks; realistic or synthetic stand-in data when customer data is not ready.

Day 93: Storytelling: lead with the business problem, not the technology.

Day 94: Live demo skills: handling questions and demo failures gracefully.

Day 95: Updates three ways: the same progress told to an engineer, a manager and an executive.

Day 96: Expectations: scope changes, saying no, negotiating trade-offs.

Day 97: Impact & the loop: presenting results and ROI to leadership; relaying customer feedback to product teams.

Day 98: Hands-on: a mock discovery call, a one-page scope doc, an architecture diagram, and a 5-minute demo video.

WEEK 15 - From POC to Production & Business Knowledge

Goal: Ship to production, prove business value, and read the customer's business.

Day 99: Lifecycle: POC to pilot to production to expansion.

Day 100: Deployment & operations: customer clouds, VPCs, on-prem; monitoring, alerting, incident response.

Day 101: Adoption & handoff: user training and change management; documentation, runbooks, enabling customer teams to own the system.

Day 102: How enterprises buy & ROI: pre-sales, procurement, renewals; time saved, cost reduced, revenue gained - measured, not claimed.

Day 103: Domains & reading a business: retail, supply chain, banking / finance, healthcare, manufacturing basics; KPIs and org structure.

Day 104: Project 1 (beginner): data integration service - pull data from two public APIs and a CSV export, clean and join it in PostgreSQL, expose it through a FastAPI service.

Day 105: Project 2 (intermediate): enterprise document assistant - RAG over policy documents with per-user access control, citations, and a 50-question evaluation set.

WEEK 16 - Projects & Interview Preparation

Goal: Finish a five-project portfolio and prepare for every interview round.

Day 106: Project 3 (intermediate): text-to-SQL analyst agent - answers business questions over a sales database, shows its SQL, and refuses unsafe queries.

Day 107: Project 4 (advanced): workflow automation agent - reads incoming emails or tickets, classifies them, updates a CRM via API or MCP, and asks for approval on sensitive actions.

Day 108: Project 5 (capstone): end-to-end customer deployment - pick an industry use case; discovery notes and scope, build, evaluate, deploy with Docker on a cloud, write a runbook, present ROI.

Day 109: Portfolio: write every project up in the same format - problem, architecture, results, and your role; existing client demos and pre-sales prototypes also count.

Day 110: Interview prep - coding round: practical Python and SQL problems, data manipulation, API integration tasks.

Day 111: Interview prep - decomposition / case & system design rounds: break down a vague customer problem aloud; design an AI system covering data flow, integrations, security, evaluation, deployment.

Greens Technologies

Day 112: Interview prep - take-home build & behavioral: build a small working solution in a time box with documented assumptions and trade-offs; STAR stories - ambiguity, a difficult stakeholder, a failed demo, shipping under a deadline.

Greens Technologies