AlgoProfessor

AI R&D Solutions

Innovating the Future with Artificial Intelligence

3-Month Immersive Internship Programme

From AI Foundations ⟶ Agentic AI Systems

90	12	16+	3	44
Days	Milestones	Projects	Phases	AI Tools

Phase 1 (Days 1–30)

Foundations: Git, Python, All Libraries, ML, Deep Learning, Databases, RAG

Phase 2 (Days 31–60)

LLM Engineering: Frontier Models, Fine-tuning, Prompt Engineering, APIs

Phase 3 (Days 61–90)

Agentic AI: Multi-Agent Systems, MCP Servers, LangGraph, Deployment

B.Tech | M.Tech | Ph.D | Post-Doctoral

ceo@algoprofessor.com
www.algoprofessor.com

A Message from the CEO

AlgoProfessor AI R&D Solutions — Internship Batch 2026

SS

Dr. S.Satyanarayana Ph.D., PDF(AI)
Founder & CEO, AlgoProfessor AI R&D Solutions
ceo@algoprofessor.com www.algoprofessor.com

Dear Future AI Professional,

Welcome to the AlgoProfessor AI R&D Solutions 3-Month Internship Programme — one of the most intensive, hands-on AI training experiences available today. I am genuinely excited that you are considering this journey, and I want to speak to you directly about what it means and what it will demand of you.

When I founded AlgoProfessor, I had a singular vision: **to close the gap between academic AI education and the real-world demands of the industry.** Universities teach theory beautifully. But the industry needs professionals who can build, ship, and iterate on intelligent systems at speed. That is precisely what this programme delivers.

Over 90 days, you will not watch lectures or complete toy exercises. You will write production-grade code every single day. You will build AI agents that browse the web, negotiate deals, write and test software, conduct deep research, and manage entire autonomous workflows. You will work with the same tools — OpenAI, Anthropic Claude, Meta Llama, CrewAI, LangGraph, MCP — that power the world's most advanced AI applications right now.

Each of the **16 milestone projects** is drawn from real industry challenges our team has faced working with enterprises across healthcare, finance, e-commerce, and technology. These are not hypothetical exercises; they are the kinds of systems that companies actively deploy and pay premium salaries to build.

I want to be honest about what this programme demands. **The daily GitHub submission requirement is not bureaucratic formality — it is your daily commitment to yourself.** The discipline of committing working, documented code every single day, even when the task is hard, is what separates a capable engineer from an outstanding one. We hold this standard without exceptions because the industry holds you to it without exceptions.

To students from **B.Tech, M.Tech, Ph.D, and Post-Doctoral backgrounds** — I especially encourage you to bring your domain expertise. AI is most powerful when

combined with deep knowledge of medicine, law, finance, materials science, or any other field. The diversity of our interns' backgrounds enriches the programme, and we have structured the projects so your domain knowledge becomes a genuine competitive advantage.

The AI revolution is not happening to us. **We are building it — one commit at a time.** I look forward to witnessing what you create, and I am personally invested in your success.

With the highest expectations and full confidence in your potential,

Dr. S.Satyanarayana Ph.D., PDF(AI)
Founder & Chief Executive Officer
AlgoProfessor AI R&D Solutions
ceo@algoprofessor.com www.algoprofessor.com

1

Contents

ceo@algoprofessor.com

ceo@algoprofessor.com

1

Programme Overview & Learning Roadmap

Mission: This 90-day programme takes interns from Python and Git fundamentals to production-grade Agentic AI. Every 10 days a milestone project synthesises the learning; each 30-day phase ends with a major capstone. Interns commit code to GitHub daily and receive structured mentor reviews every 10 days.

Phase	Days	Theme	Milestones	Reviews
Phase 1	1–30	Foundations: Git, Python, ML, Deep Learning, RAG	5	3
Phase 2	31–60	LLM Engineering, Fine-tuning, Evaluation	4	3
Phase 3	61–90	Agentic AI, Multi-Agent, MCP, Deployment	7	3
Total			**16**	**9**

GitHub Daily Submission Requirements

- Repository: `algoprofessor-rd-internship-2026` (public or access granted)
- Push all code and documentation by **5:00 PM daily** — no exceptions, no extensions
- Email GitHub link to `ceo@algoprofessor.com` after each commit
- Every folder: `README.md`, `requirements.txt`, `outputs/` subdirectory
- Commit format: `[Day-N] Short descriptive title of work completed`

Phase 1 — Foundations (Days 1–30)

Git & GitHub | Python | NumPy, Pandas, Matplotlib | ML | Deep Learning | NLP | Databases | Vector Stores | RAG

1

Week 1 — Version Control & Python Foundations (Days 1–5)

1

Day 1 — Git & GitHub Mastery

Day 1: Git & GitHub Mastery Deadline: 5 PM

1Learning Objectives

- Distributed version control concepts: commits, trees, blobs, HEAD, refs, objects
- Configure Git globally: username, email, SSH keys, GPG signing, `.gitignore` templates
- Core workflow: `init`, `clone`, `add`, `commit`, `push`, `pull`, `fetch`, `stash`
- Branching strategies: git flow, feature branches, hotfixes; rebase vs. merge tradeoffs
- Pull requests, code reviews, resolving merge conflicts, GitHub Actions basics, GitHub CLI

1Deliverables

- `day01_git_mastery/` — repo with commit history demonstrating all commands
- `branching_demo/` — feature branch created, reviewed, and merged via PR
- `git_cheatsheet.md` — annotated reference with real output examples
- `conflict_resolution_walkthrough.md` — step-by-step documented conflict resolution

Tools: Git 2.x | GitHub | GitHub CLI | VSCode GitLens | pre-commit

1

Day 2 — Python Core Syntax & Data Structures

Day 2: Python Core Syntax & Data Structures

- Python 3.10+: type hints, walrus operator (:=), match/case, f-strings advanced
- Built-ins: lists, dicts, sets, tuples, deque, Counter, defaultdict, namedtuple
- List/dict/set comprehensions, generator expressions, `itertools`, `functools`
- Decorators, context managers, dataclasses, Pydantic models, ABCs, protocols

- PEP 8; `black` formatter, `flake8`, `mypy` type checker, `pytest` basics

Deliverables: `core_python.py` • `data_structures_bench.py` • `pythonic_patterns.py` • `test_suite.py`

Tools: Python 3.10+ | pytest | black | mypy | Pydantic

1

Day 3 — NumPy, Pandas & Exploratory Data Analysis

Day 3: NumPy, Pandas & EDA

- NumPy: ndarray ops, broadcasting, `linalg`, FFT, random number generation, memory layout
- Pandas: I/O (CSV, JSON, SQL, Parquet, Excel), groupby, merge, pivot, melt, explode
- Missing values, outliers, dtype coercion, memory optimisation with categoricals
- Full EDA: distributions, correlations, skewness, kurtosis, chi-square, mutual information
- Generate profiling report; export clean datasets as Parquet for downstream ML

Deliverables: `numpy_ops.py` • `eda_pipeline.py` • `data_cleaner.py` • `outputs/eda_report.html`

Tools: NumPy | Pandas | ydata-profiling | SciPy | Parquet

1

Day 4 — Data Visualisation: Matplotlib, Seaborn & Plotly

Day 4: Data Visualisation

- Matplotlib: figure/axes API, subplots, gridspec, custom styles, animations, tight_layout
- Seaborn: categorical, distribution, pair, heat, regression plots; FacetGrid
- Plotly: interactive charts, choropleth maps, 3D scatter, sunburst; Dash dashboards
- Colour theory for data: sequential, diverging, qualitative palettes; WCAG accessibility
- Export: publication-quality PDF, SVG, PNG at 300 dpi; embed in HTML reports

Deliverables: `static_charts.py` • `interactive_dashboard.py` • `outputs/` (all figures)

Tools: Matplotlib | Seaborn | Plotly | Dash | Altair

1

Day 5 — OOP, File I/O & Environment Management

Day 5: OOP, File I/O & Project Environments

- OOP: classes, multiple inheritance, mixins, dunder methods, ABCs, protocols, slots
- File I/O: CSV, JSON, YAML, XML, HDF5, binary; `pathlib`, `shutil`; `watchdog`
- Environments: `venv`, `conda`, `pyenv`, `poetry`; Docker basics for Python apps
- Packaging: `pyproject.toml`; structured logging with `loguru`; `rich` for CLI output
- Design patterns: Factory, Singleton, Observer, Strategy, Builder applied in Python

Deliverables: `oop_design_patterns.py` • `file_manager.py` • `project_template/`

Tools: poetry | loguru | pydantic | pathlib | Docker

1

Week 2 — Machine Learning & Deep Learning (Days 6–10)

1

Day 6 — Scikit-learn & ML Pipelines

Day 6: Supervised Learning with Scikit-learn

- Regression: Linear, Ridge, Lasso, ElasticNet; bias-variance tradeoff and regularisation
- Classification: Logistic Regression, SVM, Decision Tree, Random Forest, KNN, Naive Bayes
- Evaluation: accuracy, AUC-ROC, precision-recall, F1, SHAP explainability, calibration
- Hyperparameter tuning: GridSearchCV, RandomizedSearchCV, Optuna (Bayesian optimisation)
- End-to-end `Pipeline` with `ColumnTransformer`; model saving with joblib/pickle

Deliverables: `ml_pipeline.py` • `model_comparison.ipynb` • `shap_analysis.py` • `models/`

Tools: scikit-learn | SHAP | Optuna | joblib | imbalanced-learn

1

Day 7 — Ensemble Methods & Feature Engineering

Day 7: Ensemble Methods & Feature Engineering

- Bagging, Boosting (AdaBoost, GBM); XGBoost, LightGBM, CatBoost parameters in depth
- Feature engineering: polynomial, interaction, lag, rolling stats, target encoding

- Dimensionality reduction: PCA, t-SNE, UMAP; feature selection via importance scores
- Stacking and blending ensembles; out-of-fold predictions for stacking generalisation
- Early stopping, learning rate schedules, cross-validation strategies (StratifiedKFold)

Deliverables: `ensemble_models.py` • `feature_engineering.py` • `dim_reduction.py` • report

Tools: XGBoost LightGBM CatBoost UMAP Optuna

1

Day 8 — Neural Networks from Scratch & PyTorch

Day 8: Neural Networks & Deep Learning Fundamentals

- Implement feedforward NN in NumPy: forward pass, backprop, gradient descent from scratch
- Activations (ReLU, GELU, Sigmoid, Softmax), losses (CE, MSE, Focal), optimisers (Adam, AdamW)
- Regularisation: dropout, batch normalisation, layer normalisation, weight decay
- PyTorch: `nn.Module`, `DataLoader`, custom datasets, training loop, TensorBoard
- CNN for CIFAR-10: convolutional blocks, pooling, residual connections; target >85% accuracy

Deliverables: `nn_from_scratch.py` • `pytorch_cnn.py` • `training_pipeline.py` • `training_logs/`

Tools: PyTorch TensorBoard torchvision NumPy

1

Day 9 — Transfer Learning & Model Fine-tuning

Day 9: Transfer Learning with Pre-trained Models

- Feature extraction vs. full fine-tuning vs. partial freezing: when and why
- Fine-tune ResNet50, EfficientNetV2, Vision Transformer (ViT) on custom image dataset
- Albumentations pipeline: geometric, colour, cutmix, mixup augmentations
- LR scheduling: cosine annealing with warm restarts, OneCycleLR, reduce on plateau
- Export to ONNX; runtime benchmarking with `onnxruntime`; TorchScript

Deliverables: `transfer_learning.py` • `augmentation_pipeline.py` • `onnx_export.py` • benchmarks

Tools: timm Albumentations ONNX HF Hub onnxruntime

1

Day 10 — NLP & Hugging Face Transformers

Day 10: NLP Pipelines & Transformer Models

- Text preprocessing: tokenisation, lemmatisation, TF-IDF, n-grams, stopwords, cleaning
- Word embeddings: Word2Vec (CBOW/Skip-gram), GloVe, FastText; t-SNE embedding viz
- Hugging Face `transformers`: pipeline API, tokenisers, model hub navigation and download
- Fine-tune DistilBERT for sentiment (IMDB dataset); evaluate precision/recall/F1/AUC
- Push fine-tuned model to Hugging Face Hub; `datasets` library; `evaluate` metrics library

Deliverables: `text_preprocessing.py` • `bert_sentiment.py` • `hf_finetune.py` • `inference_api.py`

Tools: transformers | datasets | PEFT | HF Hub | evaluate

Milestone Project 1 (End of Day 10) — Web Intelligence Synthesiser

Build an AI-powered brochure generator that scrapes and navigates company websites intelligently. Combines web crawling, NLP classification, and LLM-powered content synthesis into a polished PDF brochure.

- Crawl any public company website (robots.txt compliant); extract text, logos, key statistics
- LLM synthesises scraped content into a 2-page branded PDF brochure with professional layout
- NLP classifier tags sections: About, Products, Leadership, Financials, News, Careers
- Streamlit UI: enter URL → view live crawl progress → download polished PDF
- SQLite caching; re-generate only on detected content changes via HTTP ETag headers

Stack: BeautifulSoup4 | Playwright | OpenAI API | ReportLab | Streamlit | SQLite

Evaluation: Functionality (40%) • UI/UX (20%) • Code Quality (20%) • Documentation (20%)

1

Week 3 — Databases, Vector Stores & RAG Foundations (Days 11–15)

1

Day 11 — Relational Databases: PostgreSQL & SQLAlchemy

Day 11: SQL, PostgreSQL & ORM

- SQL: DDL, DML, DQL; complex JOINs, CTEs, recursive queries, window functions (RANK, LEAD)
- PostgreSQL: JSONB columns, advanced indexing (B-tree, GIN, GiST, BRIN), partitioning
- SQLAlchemy ORM: models, relationships, lazy/eager loading, session management, Alembic
- Async database queries with `asyncpg` and async SQLAlchemy 2.0; connection pooling
- Query optimisation: `EXPLAIN ANALYSE`, index strategies, vacuuming, query plan reading

Deliverables: `db_schema.sql` • `orm_models.py` • `async_queries.py` • `migrations/` • benchmarks

Tools: PostgreSQL | SQLAlchemy | Alembic | asyncpg | pgAdmin

1

Day 12 — NoSQL: MongoDB, Redis & Caching Patterns

Day 12: MongoDB, Redis & Caching

- MongoDB: document model, aggregation pipeline, Atlas Search, geospatial queries
- `motor` async driver; Atlas change streams for real-time event-driven updates
- Redis: strings, hashes, sorted sets, lists, streams, pub/sub, geospatial, HyperLogLog
- Caching patterns: cache-aside, write-through, write-behind; TTL and eviction policies
- Task queuing with Celery + Redis broker; rate limiting; distributed locks

Deliverables: `mongo_crud.py` • `agg_pipeline.py` • `redis_cache.py` • `cache_benchmark.ipynb`

Tools: MongoDB | motor | Redis | Celery | redis-py

1

Day 13 — Vector Databases & Semantic Search

Day 13: Embeddings, Vector Databases & Similarity Search

- Text embeddings: dimensionality, cosine similarity, dot product, ANNS algorithms (HNSW, IVF)
- Generate embeddings: OpenAI `text-embedding-3-large`, HF SBERT, Cohere Embed v3
- Set up Chroma DB locally and Pinecone (cloud); index, upsert, and query document collections
- Compare Chroma, FAISS, Pinecone, Qdrant, Weaviate on latency, recall, scalability, cost
- Implement hybrid search: dense vector retrieval + BM25 keyword search + Cohere re-ranking

Deliverables: `embedding_gen.py` • `vector_db_setup.py` • `hybrid_search.py` • comparison report

Tools: Chroma | FAISS | Pinecone | Qdrant | sentence-transformers

1

Day 14 — RAG Foundations & Architecture

Day 14: Retrieval-Augmented Generation Architecture

- Naive RAG vs. Advanced RAG vs. Modular RAG: decision framework and tradeoffs
- Document ingestion: PDF (pdfplumber/pymupdf), DOCX, HTML, Markdown; metadata extraction
- Chunking strategies: fixed-size, semantic, recursive character splitting, parent-document
- LangChain RAG pipeline: retriever → reranker → prompt template → LLM → response

- RAGAS evaluation: faithfulness, answer relevancy, context precision, context recall

Deliverables: `doc_processor.py` • `chunking.py` • `rag_pipeline.py` • `ragas_eval.py`

Tools: LangChain | LlamaIndex | RAGAS | Cohere Rerank

1

Day 15 — Advanced RAG: HyDE, Graph RAG & Streaming

Day 15: Advanced RAG Techniques

- HyDE (Hypothetical Document Embeddings) for improved zero-shot query expansion
- Self-query retrieval: LLM generates structured metadata filters for targeted retrieval
- Multi-query retrieval: generate query variants, ensemble results, deduplicate intelligently
- Graph RAG: build knowledge graph from documents using Neo4j; entity and relation extraction
- Streaming RAG responses via Server-Sent Events; chunk-level source attribution in UI

Deliverables: `advanced_rag.py` • `graph_rag.py` • `streaming_qa_app.py` • eval notebook

Tools: Neo4j | NetworkX | LlamaIndex | Streamlit | SSE

Milestone Project 2 (End of Day 15) — Enterprise Knowledge Navigator

Build an AI knowledge-worker using advanced RAG to become an expert on all company-related matters.

- Ingest 50+ company documents (PDFs, web pages, Confluence) into a unified knowledge base
- Graph RAG captures entity relationships across documents via Neo4j knowledge graph
- Multi-query + HyDE retrieval with Cohere re-ranking for precision and recall balance
- Conversational memory: 10-turn context window with LLM-driven summary compression
- FastAPI backend + Streamlit frontend; sources shown with page numbers and highlighted excerpts
- Live RAGAS evaluation dashboard displaying faithfulness, relevancy, and context scores

Stack: LangChain | Pinecone | Neo4j | FastAPI | Streamlit | RAGAS

1

Days 16–30 — LLM APIs, Prompt Engineering & Phase 1 Capstones

Day	Topic	Key Tasks	Tools
Day 16	OpenAI & Anthropic APIs	Integration, token management, cost tracking	OpenAI SDK, Anthropic SDK
Day 17	Prompt Engineering	Zero-shot, few-shot, CoT, self-consistency, ReAct	DSPy, PromptFoo
Day 18	Function Calling & Tools	Structured outputs, JSON schema, tool orchestration	OpenAI tools, Pydantic
Day 19	Conv. AI & Memory	Multi-turn chat, summary memory, entity extraction	LangChain Memory, Redis
Day 20	**Milestone 3**	SkyAssist: Multi-modal airline support agent	Gradio, GPT-4o, Whisper
Day 21	Hugging Face Ecosystem	Model hub, Inference API, Spaces deployment	HF Hub, Gradio, Spaces
Day 22	Open-Source LLMs	Llama 3, Mistral, Phi-3 via Ollama & vLLM	Ollama, vLLM, LM Studio
Day 23	Audio AI & Whisper	Transcription, diarisation, meeting intelligence	Whisper, pyannote, assemblyai
Day 24	Code Generation AI	Python-to-C++ transpiler; profiling & benchmarking	GPT-4o, cProfile, Hyperfine
Day 25	**Milestone 4**	MeetScribe: Audio minutes (open + closed source)	Whisper, Claude, Gradio
Day 26	Evaluation & Tracing	LLM-as-judge, hallucination detection, LangSmith	LangSmith, TruLens, Arize
Day 27	Safety & Guardrails	Prompt injection, PII redaction, safety classifiers	NeMo Guardrails, Presidio
Day 28	FastAPI & Docker	REST + WebSocket streaming; containerisation	FastAPI, Docker, GHA
Day 29	Phase 1 Review Sprint	Code review, test coverage, documentation sprint	pytest, mkdocs, pre-commit
Day 30	**Milestone 5**	CodeXcelerate: Python-to-C++ AI transpiler (60,000x)	GPT-4o, Clang, OpenMP

Milestone 3 (Day 20) — SkyAssist: Multi-modal Airline Support Agent

Build a multi-modal customer support agent for an airline with full UI and function-calling. Capabilities: flight search, booking lookup, seat upgrades, baggage policy Q&A, boarding pass scanning via GPT-4o vision, voice input via Whisper. Gradio UI with conversation

history and ticket export.

Stack: GPT-4o | Gradio | Whisper | FastAPI | function-calling

Milestone 4 (Day 25) — MeetScribe: AI Meeting Intelligence Tool

Develop a tool that creates meeting minutes and action items from audio using both open- and closed-source models. Pipeline: upload audio → Whisper transcription → speaker diarisation → LLM summarisation (GPT-4o vs. Llama 3 vs. Claude 3.5 compared) → PDF export → Slack/email notification.

Stack: Whisper | pyannote | Claude 3.5 | Llama 3 | Gradio | Slack API

Milestone 5 (Day 30) — CodeXcelerate: Python-to-C++ AI Transpiler (60,000×)

Build an AI that converts Python code to optimised C++, achieving performance improvements of up to 60,000×. Multi-step LLM pipeline: GPT-4o converts Python → C++; second pass adds OpenMP parallelism; Hyperfine benchmarks quantify speedup. Streamlit UI shows code side-by-side with live performance chart.

Stack: GPT-4o | Clang | OpenMP | Hyperfine | Streamlit

Phase 2 — LLM Engineering (Days 31–60)

Frontier Models | Fine-tuning (LoRA/QLoRA) | DPO | Quantisation | Evaluation | RAG vs. Fine-tuning Analysis

1

Phase 2 Schedule — Days 31–60

Day	Topic	Key Tasks	Tools
Day 31	LLM Landscape	GPT-4o, Claude 3.5, Gemini, Llama 3, Mistral benchmarks	LMSys Arena, HELM
Day 32	Tokenisation	BPE, WordPiece, SentencePiece; context window limits	tiktoken, HF tokenizers
Day 33	Advanced Prompting	Tree-of-Thought, self-consistency, metacognitive patterns	DSPy, Outlines
Day 34	Structured Outputs	JSON mode, grammar-constrained decoding, Pydantic AI	Outlines, instructor
Day 35	**Milestone 6**	PriceOracle: Predict product prices with Frontier LLMs	GPT-4o, Claude, Gemini
Day 36	Fine-tuning Foundations	LoRA, QLoRA, adapters; dataset preparation & formatting	PEFT, bitsandbytes
Day 37	Supervised Fine-tuning	Fine-tune Llama 3.1 8B on domain-specific dataset	TRL SFTTrainer, Axolotl
Day 38	DPO & Preference Tuning	Direct Preference Optimisation; reward modelling	TRL DPOTrainer, W&B
Day 39	Quantisation & Inference	GPTQ, AWQ, GGUF; llama.cpp, vLLM serving at scale	llama.cpp, vLLM, GPTQ
Day 40	**Milestone 7**	DealHunter: Fine-tuned open-source price predictor	QLoRA, Llama 3.1, TRL
Day 41	RAG vs. Fine-tuning	Decision framework; hybrid approaches; tradeoff analysis	RAGAS, W&B Weave
Day 42	Multi-modal Models	LLaVA, GPT-4V, Gemini Vision; image+text reasoning	GPT-4V, LLaVA, CLIP
Day 43	Embeddings at Scale	Model selection; ANN at production scale; pgvector	Qdrant, Weaviate, pgvector

Day	Topic	Key Tasks	Tools
Day 44	LLM Evaluation	MT-Bench, TruthfulQA, Big-Bench; custom eval harness	HELM, lm-eval-harness
Day 45	**Milestone 8**	BargainBot: Autonomous multi-agent deal-finder + alerts	CrewAI, Twilio, Celery
Day 46	Observability	LangSmith tracing, Phoenix dashboards, Arize monitoring	LangSmith, Arize Phoenix
Day 47	Safety & Guardrails	Llama Guard, NeMo Guardrails, PII redaction, jailbreaks	NeMo, Presidio
Day 48	API Design	REST + SSE streaming; OpenAI-compatible API; LiteLLM	FastAPI, LiteLLM
Day 49	Cloud Deployment	SageMaker, GCP Vertex AI, Azure OpenAI Service	AWS, GCP, Azure
Day 50–60	Phase 2 Review & R&D	Refactor, test, document; guest lectures; research exploration	All Phase 2 tools

Milestones 6 & 7 (Days 35 & 40) — PriceOracle + DealHunter

Capstone A — PriceOracle: Predict product prices from short descriptions using GPT-4o, Claude 3.5 Sonnet, and Gemini Flash. Compare accuracy (MAE, MAPE) with a live Streamlit leaderboard.

Capstone B — DealHunter: QLoRA fine-tune Llama 3.1 8B on the same dataset to compete with Frontier models. Target sub-10% MAPE at less than $0.001 per inference. Full W&B experiment tracking.

Stack: QLoRA PEFT TRL W&B Llama 3.1 GPT-4o Axolotl

Phase 3 — Agentic AI Systems (Days 61–90)

AI Agents | LangGraph | CrewAI | AutoGen | MCP Servers | Browser Agents | Production Deployment

1

Phase 3 Schedule — Days 61–90

Day	Topic	Key Tasks	Tools
Day 61	Agent Foundations	ReAct, Reflexion, Plan-and-Execute; tool use patterns	LangChain Agents
Day 62	MCP Protocol	Build custom MCP servers; tool server architecture	Anthropic MCP SDK
Day 63	LangGraph	State machines, conditional edges, human-in-the-loop	LangGraph
Day 64	Browser Agents	Playwright agents; autonomous web navigation	Playwright, Stagehand
Day 65	**Milestone 9**	TalentTwin: Career Digital Twin (LinkedIn + GitHub)	LangGraph, FastAPI
Day 66	CrewAI Framework	Crew, Agent, Task, Process; role-playing multi-agent	CrewAI
Day 67	SDR Sales Agent	Email-writing agent with CRM integration	CrewAI, SendGrid, HubSpot
Day 68	Deep Research Agent	Planner + Searcher + Writer + Editor agent team	CrewAI, Tavily
Day 69	Stock Picker Agent	SEC filings, news sentiment, financial signal analysis	CrewAI, yfinance
Day 70	**Milestone 10**	DevSquad: 4-Agent Engineering Team in Docker	CrewAI, Coder, Docker
Day 71	AutoGen Framework	Conversable agents; group chat; nested conversations	AutoGen
Day 72	Agent Creator Agent	Meta-agent that builds and deploys new agents	AutoGen, Docker
Day 73	Browser Sidekick	OpenAI Operator-style; LangGraph + Playwright	LangGraph, Playwright
Day 74	Multi-Agent Comms	A2A protocol, message queues, event-driven patterns	RabbitMQ, Redis Streams
Day 75	**Milestone 11**	SidekickBrowser: OpenAI Operator clone for browser	LangGraph, Playwright

Day	Topic	Key Tasks	Tools
Day 76	Advanced MCP	3 custom MCP servers: DB, REST API, Filesystem	MCP SDK, FastAPI
Day 77	Agent Memory	Episodic, semantic, procedural memory for agents	Mem0, Zep, Letta
Day 78	Agent Evaluation	AgentBench, GAIA, SWE-bench; custom eval harness	AgentBench, LangSmith
Day 79	Production Hardening	Rate limiting, circuit breakers, fallback chains	Tenacity, LiteLLM
Day 80	**Milestone 12**	AgentForge: Meta-agent creates agents via AutoGen	AutoGen, Docker
Day 81–83	Grand Capstone Build	AlphaTrader: 4-agent trading floor + 6 MCP + 44 tools	All Phase 3 tools
Day 84–87	Docker & Kubernetes	Multi-container; Helm charts; CI/CD; monitoring	Docker, k8s, Helm, GHA
Day 88–89	Docs & Final Tests	Full test suite; API docs; user manual; demo video	pytest, Swagger, mkdocs
Day 90	**Final Presentations**	20-min live demo to AlgoProfessor panel; Q&A	All projects

Milestone 9 (Day 65) — TalentTwin: AI Career Digital Twin

Build and deploy your own AI agent to represent you to potential employers. It autonomously scrapes your GitHub, LinkedIn, and portfolio, synthesises a dynamic personal profile, and responds to recruiter queries with personalised, context-aware answers. Deployed publicly with Calendly integration for automated interview scheduling.

Stack: LangGraph · Playwright · FastAPI · Railway/Render · Calendly API

Milestone 10 (Day 70) — DevSquad: 4-Agent Engineering Team in Docker

Deploy your own 4-agent engineering squad: Project Manager, Developer, Code Reviewer, QA Tester. Manage, build, and test software applications with CrewAI and Coder Agents running in isolated Docker containers. Fully autonomous from specification to working tested code, with GitHub integration.

Stack: CrewAI · Coder · Docker · GitHub API · pytest

Milestone 11 (Day 75) — SidekickBrowser: AI Browser Co-pilot

Build your own version of OpenAI's Operator — a Sidekick that works alongside you inside your browser via LangGraph and Playwright. Fills forms, navigates complex UIs, extracts structured data, and executes multi-step web tasks from plain English instructions. Deployed as a Chrome extension with FastAPI backend.

Stack: LangGraph | Playwright | FastAPI | Chrome Extension | SSE

Milestone 12 (Day 80) — AgentForge: The Agent That Builds Agents

An Agent Creator that builds and launches new AI agents dynamically using AutoGen, unlocking endless AI possibilities. Takes a natural language description, generates agent code, tests it in a sandboxed Docker environment, and deploys it as a live running service — fully autonomously.

Stack: AutoGen | Docker | FastAPI | GPT-4o | GitHub Actions

Grand Capstone (Days 81–90) — AlphaTrader: Autonom

Build a complete trading floor with 4 autonomous agents making collaborative trades, powered by 6 MCP servers and 44 tools — the ultimate capstone integrating all 3 phases.

- **MarketAnalyst Agent** — real-time price data, news sentiment NLP, SEC filings
- **QuantStrategist Agent** — statistical arbitrage, momentum, portfolio optimisation
- **RiskGuardian Agent** — position limits, drawdown rules, compliance checks
- **ExecutionEngine Agent** — order placement via broker API (paper trading mode)

6 MCP Servers: Market Data • News Feed • Financial Statements • Order Management • Risk Engine • Notification Service

Stack: AutoGen | MCP | LangGraph | Alpaca API | Grafana | Docker | Kubernetes | FastAPI

Evaluation: Agent Collaboration (30%) • Risk Logic (20%) • Code Quality (20%) • Live Demo (30%)

1

Evaluation Rubric & Grading

Assessment Component	Points	Evaluated By
Daily GitHub submissions (timeliness + commit quality)	10	Automated + Mentor
Code quality, PEP 8, documentation, unit tests	15	Mentor Code Review
Phase 1 Milestones (5 projects: Days 10, 15, 20, 25, 30)	25	Technical Panel
Phase 2 Milestones (4 projects: Days 35, 40, 45, 50)	20	Senior Engineers
Phase 3 Milestones (4 projects: Days 65, 70, 75, 80)	15	AlgoProfessor Team
Grand Capstone: AlphaTrader Trading Floor	10	CEO + Board Panel
Peer collaboration, communication, professionalism	5	Team + Mentor
Total	**100**	

Score	Grade	Award & Recognition
90–100	Excellent	Certificate of Excellence + Letter of Recommendation + LinkedIn Endorsem
75–89	Good	Completion Certificate + LinkedIn Endorsement
60–74	Satisfactory	Participation Certificate
Below 60	Incomplete	Mandatory review with supervisor; programme may be terminated

1

Complete Technology Stack

1Version Control & DevOps

Git, GitHub, GitHub Actions, Docker, Kubernetes, Helm, pre-commit, mkdocs, pytest

1Python Scientific Stack

Python 3.10+, NumPy, Pandas, SciPy, Matplotlib, Seaborn, Plotly, Dash, Altair, ydata-profiling

1Machine Learning

Scikit-learn, XGBoost, LightGBM, CatBoost, SHAP, Optuna, UMAP, imbalanced-learn, joblib

1Deep Learning

PyTorch, torchvision, timm, TensorBoard, ONNX, onnxruntime, Albumentations

1Databases

PostgreSQL, SQLAlchemy, Alembic, asyncpg, MongoDB Atlas, motor, Redis, SQLite, pgAdmin

1Vector Stores

Chroma, FAISS, Pinecone, Qdrant, Weaviate, pgvector, sentence-transformers

1LLM APIs

OpenAI (GPT-4o), Anthropic (Claude 3.5), Google Gemini Pro/Flash, LiteLLM

1Open-Source LLMs

Llama 3, Mistral, Phi-3, vLLM, Ollama, llama.cpp, LM Studio, GPTQ, AWQ, GGUF

1Fine-tuning Stack

LoRA, QLoRA, PEFT, TRL (SFTTrainer, DPOTrainer), bitsandbytes, W&B, Axolotl

1Agentic Frameworks

LangChain, LangGraph, LlamaIndex, CrewAI, AutoGen, Anthropic MCP SDK, Mem0, Zep

1Web, API & UI

FastAPI, Streamlit, Gradio, SSE, WebSockets, Swagger/OpenAPI, Chrome Extensions

1Integrations & Cloud

Twilio, SendGrid, Slack API, Playwright, Stagehand, AWS, GCP, Azure, Grafana, Prometheus

Apply for the Internship

Batch 2026 — Limited Seats Available

Open to motivated students and researchers across all levels and disciplines:

B.Tech
All Engineering Branches
CS, IT, ECE, Mech, Civil

M.Tech
CS/AI/DS/ECE
Any Specialisation

Ph.D
AI / ML Research
Any Domain

Post-Doc
Research Fellows
Any Research Area

To Apply or Enquire, Contact Us Directly:

ceo@algoprofessor.com

www.algoprofessor.com

Please include your CV, a brief statement of interest, and your GitHub profile URL.

Programme Highlights: 90 Days • Daily GitHub Commits • 16+ Hands-on Projects • 3 Phase Reviews

Certificate + Letter of Recommendation for top performers • LinkedIn Endorsements • Industry Networking