Edversity.

AI AGENTS & AUTOMATION ENGINEERING

PROGRAM SYLLABUS

A 12-week, cohort-based program that produces an engineer who can build AI systems that do real work: three deployed products, a public portfolio, and an individual technical defence.

Agents & Tools · Retrieval & Memory · MCP · Evaluation · Security · Production

DURATION	LIVE HOURS	SCHEDULE	BUILD TIME
12 WEEKS	72 HOURS	3 DAYS / WEEK	10–12 HRS
live cohort	instructor-led	2 hours per day	per week, building

PROGRAM POSITIONING

This program produces an engineer who can take a business process and build a reliable AI system that runs it — not a generalist who has watched tutorials. It is **architecture-first**, because frameworks change every few months and engineering judgment does not. Agents, tool design, retrieval, state, evaluation and security form the spine, because that is the actual daily work. **Claude** is the primary implementation environment, with portability to other providers verified throughout.

TARGET FIRST ROLES

AI Automation Engineer, AI Integration Engineer, Junior Agent Developer, Applied AI Engineer, Solutions Engineer.

SECONDARY PATH

Contract and project work: automation, agent builds and system integration for international and local clients.

Honest expectation set with students up front: dedicated entry-level AI engineering roles remain comparatively limited, so the first opportunity is often an integration, automation or solutions role, or project work — not "AI architect" on day one. Students who skip the weekly build will not be ready; three deployed systems, not attendance, are what get them hired. Edversity does not offer an employment guarantee.

LEARNING OUTCOMES

By completion, students will be able to:

01	Choose the right degree of autonomy for a business process, and justify why an agent is or is not required
02	Build AI-enhanced workflows with schema-validated outputs, defined failure behaviour and approval gates
03	Engineer event-driven systems with retries, backoff, idempotency and timeout handling
04	Build agents from first principles — loops, tools, termination conditions, iteration limits and cost budgets
05	Design tools an agent can use safely, with clear contracts and non-overlapping responsibilities
06	Ground answers in organizational knowledge with retrieval, citation and correct refusal
07	Build and publish a custom MCP server that other agents and hosts can consume
08	Persist state, resume after interruption, and put humans in the loop for consequential actions
09	Evaluate agent behaviour with golden datasets, regression tests in CI, traces and cost metrics
10	Defend a system against prompt injection, privilege misuse and unsafe autonomy — and ship it to production

ENTRY & FOUNDATIONS

There is no theoretical entrance examination. Entry is by demonstrated readiness, through either route. **No professional software engineering experience is required — foundations are included.**

ROUTE	WHO	WHAT HAPPENS
A · Direct entry	Already works comfortably with Python, REST APIs, JSON and Git	Straight into Week 1
B · Foundations Sprint	Everyone else, including career changers and students	2-week included sprint, then a practical readiness exercise

THE FOUNDATIONS SPRINT — 2 WEEKS, ~12 HOURS, INCLUDED AT NO EXTRA COST

F1	Python essentials: functions, data structures, control flow, reading an error
F2	Working with Claude Code: specifying, reviewing, running and correcting generated code
F3	APIs, JSON, HTTP requests and responses, webhooks
F4	Git and GitHub: clone, branch, commit, pull request
F5	Environment setup: cloud development environment, API keys, repository templates
F6	Professional operations: client contracts, invoicing, receiving international payments through formal banking channels

THE READINESS EXERCISE — PRACTICAL, NOT A QUIZ

Retrieve data from a public API, transform it, save the result, push your work to GitHub, explain in your own words what each part of your code does, and repair one deliberately broken function. Those who complete it start Week 1. Those who do not are offered a further preparation sprint and a place in the next cohort at no additional cost — nobody is turned away, some people start later than they hoped.

ENGINEERING STANDARDS

Applied from Week 1 — not introduced in the security week.

Deployment	Externally accessible to mentors, with authentication appropriate to the interface. Endpoints that execute tools or write data are never anonymous public endpoints.
Data	Synthetic or sanitized by default. Real data requires documented permission and a stated retention approach.
Cost	Every system reports its cost per run. Cost is an engineering constraint, not an afterthought.
Evidence	Every claim about system quality is backed by a measurement on a defined dataset.

THREE SHIPPED PRODUCTS

One per month. Each deployed, measured, documented and individually defended.

PRODUCT 01 INTAKE-TO-ACTION SERVICE

Weeks 1–4. Receives unstructured business input, extracts validated structured data, classifies and routes it, requests human approval where a rule requires it, updates a system of record, and logs every run with cost and latency.

- Deployed and mentor-accessible, appropriately authenticated
- Schema validation with defined failure behaviour
- Two logic branches plus one human approval gate
- Retries, backoff, timeout handling; no duplicate writes
- 20-case golden set with measured accuracy
- One documented improvement, before and after

EXAMPLE DOMAINS · Lead intake and qualification · invoice processing · admissions intake · ticket triage · candidate screening.

PRODUCT 02 GROUNDED OPERATIONS AGENT

Weeks 5–8. An agent that interprets an objective, selects and uses tools, answers from organizational documents with cited evidence, exposes its capabilities through a custom MCP server, and talks to real people on a real channel with escalation to a human.

- Agent loop with termination conditions, iteration limit, cost budget
- Four or more tools with defined contracts
- Retrieval with citations; correct refusal on weak evidence
- MCP server verified against a second host
- Live on WhatsApp, email or Slack with human handoff
- Tool-selection accuracy measured; 8+ injection tests

REFERENCE INFRASTRUCTURE PROVIDED · Service template, database schema, auth, tracing, CI and channel adapters — so build time goes into design, not plumbing.

PRODUCT 03 DURABLE OPERATIONS SYSTEM

Weeks 9–12 · Capstone. A multi-stage system running a named business process end to end. It persists state and resumes after interruption, requires approval before consequential actions, and has been tested adversarially by another team.

- Named process with a measurable objective
- Three or more external integrations
- Runtime forcibly terminated mid-task — resumes without repeating irreversible actions
- Approval gate with audit trail and timeout behaviour
- 30+ case evaluation suite, baseline and improvement
- Adversarial report from another team, fixed and retested

CAPSTONE DOMAINS · Sales operations · customer operations · talent operations · research and intelligence · education operations.

12-WEEK CURRICULUM

Three live sessions per week, each pairing an engineering concept with a build that goes into your product.

THE ARCHITECTURE DECISION LADDER — THE RULE BEHIND EVERY BUILD	USE
Predictable fixed steps	**Deterministic workflow**
Fixed steps with bounded AI judgment	**AI-enhanced automation**
Dynamic tool selection and decision making	**Single agent**
Long-running tasks with checkpoints and approvals	**Stateful agentic workflow**
Independent reasoning that measurably improves outcomes	**Multi-agent system**

Students must always be able to answer: why is an agent required here? Where a plain workflow is more reliable, the workflow is the correct engineering choice and the agent is a defect.

WEEK 01 SYSTEMS, NOT CHATBOTS

Goal: Learn to decide before you build. Most failed AI projects are architecture mistakes, not model mistakes.

DAY	TOPIC	THEORY	PRACTICAL / LAB
1	**How LLM Systems Work**	Tokens, context windows, inference, latency; cost per run; deterministic vs probabilistic components	Cost-model a real workflow before writing any code
2	**Process & Architecture**	Triggers, events, state, side effects, irreversibility; the Architecture Decision Ladder	Map a real business process; classify every step
3	**Standards & Scoping**	Deployment, authentication and data standards; working effectively with Claude Code	Architecture diagram + approval map; scope sign-off

INDEPENDENT BUILD 10–12 HRS · Finalise the Product 1 scope. No student proceeds without an approved architecture.

WEEK 02 STRUCTURED INTELLIGENCE & EVALUATION

Goal: Make a model produce reliable structured data — and measure it from the very first week.

DAY	TOPIC	THEORY	PRACTICAL / LAB
1	**Model APIs & Structured Output**	Request lifecycle, system instructions, streaming, retries; JSON Schema and Pydantic validation	Schema-validated extraction with defined failure behaviour
2	**The Five Operations**	Classification, extraction, transformation, routing, summarization; model selection across capability, cost, latency	Build the decision core of Product 1
3	**Evaluation From Day One**	Golden datasets, success criteria, failure tables, error analysis	Build a 20-case set; measure, improve, measure again

INDEPENDENT BUILD 10–12 HRS · Extend the golden set. Clinic: each student presents their three worst failures.

WEEK 03 APIs, EVENTS & RELIABILITY

Goal: Make it survive the real world. This is the week the system goes live.

DAY	TOPIC	THEORY	PRACTICAL / LAB
1	**APIs & Authentication**	REST, methods, headers, OAuth, pagination, rate limits; reading unfamiliar API documentation	Integrate an external service end to end
2	**Events & Workflow State**	Webhooks, polling, schedules, queues; branching, state machines, approval gates	Workflow runtime lab in n8n; wire the approval gate
3	**Reliability & Deployment**	Retries, exponential backoff, duplicate events, idempotency, timeouts, partial failure, secrets	**Deploy Product 1.** Survive live failure injection

INDEPENDENT BUILD 10–12 HRS · Harden against malformed payloads, duplicates and slow dependencies.

WEEK 04 OBSERVABILITY & DELIVERY

Goal: Know what your system is doing, what it costs, and how to hand it over. Product 1 ships.

DAY	TOPIC	THEORY	PRACTICAL / LAB
1	**Logging, Tracing & Cost**	Structured logs, run IDs, traces; what to log and what must never be logged; cost and latency measurement	Instrument the service; produce a cost-per-run figure
2	**Runbooks & Handover**	Operational support expectations, runbooks, handover documentation, commercial scoping	Complete the Ship Kit
3	**Review & Defence**	Presenting architecture without the code; defending design decisions	Verification: mentor sends malformed and duplicate input

GATE · PRODUCT 1 DEFENCE WINDOW · Individual, 20 minutes, editor closed. Scheduled by clinic group.

WEEK 05 AGENTS FROM FIRST PRINCIPLES

Goal: Build the loop yourself before any framework hides it — and meet your first attacker.

DAY	TOPIC	THEORY	PRACTICAL / LAB
1	**The Agent Loop**	Goals, observations, actions, termination conditions; iteration limits, cost budgets, kill switches	Build a working agent loop from nothing
2	**Tool Engineering**	Schemas, semantic naming, input/output contracts, safe failure, the accuracy cost of overlapping tools	Build your first tools with enforced contracts
3	**Injection & Frameworks**	Direct and indirect prompt injection; least privilege and permission models; the same agent hand-rolled, in a provider SDK, and in a graph framework	Attack your own agent through a tool result

INDEPENDENT BUILD 10–12 HRS · The hardest week of the program. Reference implementation and extra mentor cover available.

WEEK 06 CONTEXT, RETRIEVAL & GROUNDING

Goal: Answer from real organizational knowledge — with evidence, and with the good sense to refuse.

DAY	TOPIC	THEORY	PRACTICAL / LAB
1	**Context Engineering**	Context types; budgeting, compaction, relevance filtering, and what stale context does to an agent	Build a context budget for your agent
2	**Retrieval**	Embeddings, chunking strategies, metadata, hybrid search, reranking; PostgreSQL and pgvector	Knowledge layer over a real document corpus
3	**Grounding & Retrieval Evaluation**	Citations, refusal and escalation; hit rate, citation correctness, hallucination rate	Measure retrieval quality; demonstrate a correct refusal

INDEPENDENT BUILD 10–12 HRS · Clinic: retrieval failure autopsy — bring one query the system got wrong.

WEEK 07 MODEL CONTEXT PROTOCOL (MCP)

Goal: Publish your capabilities so any agent can use them. The most transferable week in the program.

DAY	TOPIC	THEORY	PRACTICAL / LAB
1	**MCP Architecture**	Hosts, clients, servers, tools, resources; capability discovery, schemas, transports	Consume a public MCP server from your own agent
2	**Building a Server**	Designing interfaces agents can use reliably; authentication, authorization, error handling	Build and document your own MCP server
3	**Portability & MCP Risk**	Tool poisoning, server substitution, confused deputy, over-scoped tokens	Verify against a second host; cross-test a peer's server

INDEPENDENT BUILD 10–12 HRS · MCP is the leading open protocol for connecting agents to tools and context, with cross-vendor adoption and neutral governance under the Linux Foundation.

WEEK 08 CHANNELS & HUMAN HANDOFF

Goal: Put the agent where users actually are, and know when to get out of the way. Product 2 ships.

DAY	TOPIC	THEORY	PRACTICAL / LAB
1	**Messaging Channels**	WhatsApp, email, Slack, web; threading, deduplication, out-of-order messages, media, delivery	Connect the agent to a live channel
2	**Multilingual & Handoff**	Urdu and Roman Urdu code-switched input; evaluating quality in a language your test set was not written in; escalation and return	Build the handoff path; evaluate non-English handling
3	**Review & Defence**	Tool boundary justification; injection surface walkthrough	Verification: mentor messages the agent from their own phone

GATE · PRODUCT 2 DEFENCE WINDOW · Individual, 20 minutes, editor closed.

WEEK 09 STATE, DURABILITY & HUMAN OVERSIGHT

Goal: Build something that survives a crash, and that asks permission before it does anything expensive.

DAY	TOPIC	THEORY	PRACTICAL / LAB
1	**Durable Execution**	Persistent state, checkpointing, event history, resumption; queues, workers, scheduling; memory types	Capstone skeleton that resumes after forced termination
2	**Oversight & Orchestration**	Approval gates, timeout behaviour, audit trails; sequential, parallel, routing, evaluator-optimizer and manager/worker patterns	Wire one approval gate end to end
3	**Claude Edge Lab**	The same task three ways: hand-built loop, Claude Agent SDK, Claude Managed Agents (beta) — control, state, operations, cost, portability	Written comparison; capstone proposal sign-off

INDEPENDENT BUILD 10–12 HRS · Managed services are taught as one implementation option among three. The curriculum stays portable by design.

WEEK 10 EVALUATION & OBSERVABILITY

Goal: Prove it works. Opinions about agent quality are worth nothing without a dataset.

DAY	TOPIC	THEORY	PRACTICAL / LAB
1	**Datasets & Graders**	Task-completion, behavioural, trajectory and tool-use evaluation; deterministic vs model graders; calibrating a judge against human labels	Build the 30+ case capstone evaluation suite
2	**Regression & Tracing**	Testing prompt, model, environment and tool-failure variation; OpenTelemetry GenAI conventions; runs, tokens, latency, cost, failures	Evaluations running in CI; tracing dashboard live
3	**Architecture Experiment**	Single-agent vs multi-agent on the same sub-problem: quality, latency, cost, reliability	Measure both, then decide on evidence — and reject complexity that does not earn itself

INDEPENDENT BUILD 10–12 HRS · Improve the capstone on evidence and record the delta against baseline.

WEEK 11 PRACTICAL AGENT SECURITY

Goal: An agent holds credentials and takes actions. Learn how it gets abused, then break someone else's.

DAY	TOPIC	THEORY	PRACTICAL / LAB
1	**Threats**	Prompt and indirect injection, tool misuse, excessive autonomy, memory poisoning, supply-chain and MCP risk	Threat-model your own system
2	**Controls**	Agent identity and privilege, vault patterns where the model never sees the secret, sandboxing, allow-lists, rate limits, spend caps, audit trails	Harden the capstone and retest
3	**Red Team**	Rules of engagement; OWASP Top 10 for Agentic Applications and the Agent Control Standard	Adversarial test of a paired team's system; documented exploits

INDEPENDENT BUILD 10–12 HRS · Both the attack report and your hardening report are graded.

WEEK 12 PRODUCTION, DELIVERY & DEFENCE

Goal: Ship it, price it, hand it over, and defend it.

DAY	TOPIC	THEORY	PRACTICAL / LAB
1	**Deploy & Operate**	Service architecture with FastAPI, background workers, secrets, containers, CI/CD, monitoring and alerting	Capstone deployed to its production configuration
2	**Cost & Commercial**	Caching, model routing, context optimization; scoping, pricing, maintenance agreements, handover to non-technical owners. Enterprise governance primer: NIST AI RMF, ISO/IEC 42001, EU AI Act, GDPR	Commercial scope and handover pack
3	**Defence & Demo Day**	Individual technical defence; public showcase to invited employers, agencies and partners	Verification: mentor terminates the runtime mid-task

GATE · PRODUCT 3 DEFENCE WINDOW & DEMO DAY · Every student is defended individually. Demo Day showcases a selected group of systems.

ASSESSMENT & GRADING

Assessment is practical and continuous. Attendance earns nothing. Three deployed products and three individual defences are the core of the credential.

COMPONENT	WEIGHT
Product 1 — Intake-to-Action Service (Week 4)	15%
Product 2 — Grounded Operations Agent (Week 8)	20%
Product 3 — Durable Operations System (Week 12)	30%
Evaluation quality across all three products	10%
Security and adversarial exercise (Week 11)	10%
Lab and clinic participation	10%
Peer review quality	5%

GRADUATION GATES: MANDATORY, PASS / FAIL

All three products deployed and meeting their acceptance criteria; all three individual technical defences passed, conducted with the code editor closed; a complete Ship Kit for each product; and a public portfolio linking to three live systems. The Edversity certificate is issued only when these are met, so an employer can rely on it. One two-week remediation window is available per product.

THE SHIP KIT

Submitted with every product, three times. A working demo without documentation does not count as shipped.

Deployed system	Externally accessible to the mentor, authenticated appropriately
README	A competent reader can run it within ten minutes
Architecture diagram	With a written justification for every autonomous step
Interface contracts	Endpoints and tool schemas with worked examples
Evaluation report	Dataset, method, baseline, current results, weakest category
Operational evidence	Traces, cost per run, median and 95th-percentile latency
Security note	Permissions model, secrets handling, threat surface, mitigations
Runbook	Common failures and recovery procedures
Limitations	Known failure modes, stated plainly
Demo recording	Three minutes maximum
Commercial scope	One page: build effort, running cost, maintenance considerations
AI-use disclosure	What was generated, what was written, what was modified

THE TECHNICAL DEFENCE

Individual, twenty minutes, conducted by your mentor with the code editor closed. You explain your own architecture from your own diagram. AI coding assistants are required tools in this program and are used from the first week — what is assessed is your reasoning, not your typing. Work you cannot explain or debug does not meet the standard, which is exactly the standard a client or employer applies.

REPRESENTATIVE QUESTIONS

- Why an agent here rather than a deterministic workflow? Which step could be deterministic and is not?
- What does each tool do, and why is that boundary correct?
- What happens if an external service returns corrupted data, or nothing at all?
- How does the system resume after interruption without repeating an irreversible action?
- Where could injected content reach an action that writes data?
- What did you measure, on what dataset, and what improved?
- Which actions require human approval, and why those?
- What does this cost per run, and what would you charge to build and maintain it?

TOOLS COVERED

Learning outcomes are tool-independent. This is the reference implementation stack; API credits are provided with spend caps.

DOMAIN	TOOLS
Language & Assistant	Python · Claude Code
Model APIs	Claude API (primary) · a second provider for portability checks
Validation	Pydantic · JSON Schema
Workflow Runtime	n8n
Agent Implementation	First-principles loop · provider SDK · graph framework · managed service (comparison)
Interoperability	Model Context Protocol (MCP)
Data & Retrieval	PostgreSQL · pgvector
Backend & Deployment	FastAPI · Docker · GitHub Actions
Observability	OpenTelemetry GenAI conventions · managed tracing platform
Channels	WhatsApp Business API · email · Slack
Version Control & Portfolio	Git · GitHub · live-deployed systems

CAREER & READINESS OUTCOMES

Every graduate leaves with an employer-facing package, not just knowledge:

- Three live, deployed systems with shareable URLs — the core proof artifact
- A public GitHub portfolio with architecture diagrams, evaluation reports and security notes
- Three recorded technical defences demonstrating ownership of the work
- An adversarial test report and a documented hardening response
- A profile and CV built around delivered systems, not course titles
- Two mock technical interviews — architecture and evaluation, then security and failure analysis
- A scoped, priced client proposal written from your own product
- Demo Day exposure to invited employers, agencies, founders and technology partners

WHY THE PORTFOLIO, NOT THE CERTIFICATE

There is no accredited standard for agent engineering yet, so the credential carries its own evidence. Every certificate links to a portfolio page listing three live systems, three evaluation reports, an adversarial test report and three defence recordings. An employer does not have to trust us — they can inspect the work.

DESIGN RATIONALE

Internal note for the Edversity team.

Architecture-first, not framework-first: agent frameworks change every few months; the decision ladder, tool contracts, state design and evaluation methodology do not. Framework literacy gets two hours in Week 5 — enough to read a client's codebase and answer an interview question, not enough to date the curriculum.

Three products instead of twelve mini-projects: a student who has deployed three systems has a portfolio; a student with twelve notebook exercises has a transcript. Each product is independently sellable and independently defensible.

Evaluation from Week 2, security from Week 5: both are taught at the moment they first matter. Bolting evaluation on at the end produces a cohort that has already learned to ship on impressions, and introducing injection after six weeks of tool access teaches the wrong instinct.

Deployment in Week 3: a student who has put something live in the first month believes they can finish. The Week 4 ship is the single most important retention event in the program.

Defence with the editor closed: the only assessment mechanism that still discriminates now that any student can generate a working repository. It is a pass/fail gate on each product rather than a weighted component, because averaging it away would defeat its purpose.

Readiness gate rather than an entrance exam: a practical exercise after an included Foundations Sprint keeps the funnel open while protecting completion. Marketing says "no professional software engineering experience required" — not "no coding background required," which would be untrue by Week 5.

Claude as primary, portability verified: deep implementation exposure in one environment produces stronger engineers than shallow exposure to four. Portability checks at two points prevent the curriculum from becoming vendor training.

MCP gets a full week: it is the leading open protocol for agent-to-tool integration, now under neutral governance with cross-vendor adoption. It is the most transferable skill in the program.

Cohort capped at 40–45 with clinic groups of 12–15: a four-hour live engineering build degrades with scale. Growth comes from additional lab sections, not larger rooms.

Assumes 10–12 hrs/week of independent build on top of live hours, stated at enrolment. Three deployed systems cannot be produced on less, and pretending otherwise produces refund conversations in Week 6.