AI CLASSROOM COURSES · ADVANCED ENGINEERING

AI Coding Agent Development

Build coding agents, connect tools, automate workflows, and verify what your agents do.

6 months	300	12	₹58,000
DURATION	GUIDED HOURS	MODULES	FOUNDING BATCH FEE

Who it is for

Developers and technical learners who already write code and want to engineer agent systems, not just chat with them.

Before you begin

Working JavaScript/TypeScript, asynchronous APIs, Git, terminal use and backend fundamentals. A practical readiness check precedes enrollment.

What you will build

- A tool-using LLM application
- A reviewable workflow with human approval
- RepoGuide — repository-understanding milestones
- TestFix — a controlled coding agent evaluated on twenty unseen tasks

What you will be able to do

- Build LLM applications with contracts and evidence, where the model proposes and the system enforces
- Design workflows with state, human approval, retrieval and MCP within a context budget
- Let an agent read a repository and propose reviewable patches inside a sandbox
- Evaluate agents on unseen tasks, control safety, privacy and cost, and turn the agent into a product

Tools and technologies

TypeScript · LLM APIs · Tool calling · LangGraph · MCP · Next.js · NestJS · PostgreSQL

Evals & tracing

Modules

MODULE 1 · 25 H LLM Applications: Predictions, Contracts and Evidence

- Narrow task selection
- The automation spectrum
- Engineering readiness
- Model behaviour
- Token budgets
- Output variability
- Reproducibility limits
- Instruction boundaries
- Input trust
- Model-API integration
- Secret boundaries
- Latency perception
- Structured streaming
- Schema versus domain validation
- Abstention design
- Provider-failure handling
- Backoff reasoning
- Usage accounting
- Prompt versioning
- Data-conscious tracing
- Prompt experiments
- Baseline design
- Small-sample interpretation
- LLM-exercise integration
- Foundation assessment

MODULE 2 · 25 H Tools: The Model Proposes, the System Enforces

- Tool-call lifecycle
- Tool-schema design
- Path scope
- Bounded outputs
- Result contracts
- Tool granularity
- Read-first development
- Input validation
- Tool-source trust
- Read permissions and privacy
- Step budgets
- Termination criteria
- Loop detection
- Tool concurrency
- Error feedback
- Fallback safety
- Contract testing
- Run transparency
- User-facing rationale
- Approval scope
- Approval idempotency
- Cancellation semantics
- Tool-use integration
- Capability assessment
- Module review

MODULE 3 · 25 H Workflows, State and Human Approval

- Deterministic versus dynamic control
- Workflow graphs
- Structured state
- State representation
- Step granularity
- State reducers
- Persistence
- Run isolation
- Durable execution
- Replay safety
- Approval placement
- Rejection paths
- Stale approvals
- Time budgets
- Recovery classification
- Progress invariants
- Observable workflows
- Join semantics
- Fallback decisions
- Approval authorisation
- Debug artifacts
- Orchestration integration
- Recovery lab
- Assessment
- Module review

MODULE 4 · 25 H Retrieval, MCP and the Context Budget

- Context selection
- Retrieval choices
- Chunking trade-offs
- Evidence sufficiency
- Retrieval scores
- Source provenance
- Index freshness
- Authorised indexing
- Budget allocation
- Context compression
- Protocol versus trust
- MCP concepts
- Capability enablement
- Connection boundaries
- Audience-bound credentials
- Capability-change review
- Indirect prompt injection
- Layered defence
- Missing-evidence behaviour
- Evidence precedence
- Retrieval evaluation
- Integration exercise
- Optimisation lab
- Assessment
- Module review

MODULE 5 · 25 H Understand the Repository: From File Tree to Behaviour

- Repository orientation
- Untrusted project scripts
- Project detection
- Dependency reproducibility
- Execution entrypoints
- Call-site analysis
- Type dependencies
- Failure attribution
- Clarification
- Specification sources
- Search strategy
- Scope exclusions
- Sensitive-context handling
- Snapshot consistency
- Parsing trade-offs
- Scope discipline
- Test-driven investigation
- Environment dependencies
- Stack interpretation
- Task-specific summaries
- Evidence packaging
- Repository assistants
- Generalisation checks
- Investigation assessment
- Module review

MODULE 6 · 25 H Propose a Patch Without Taking Over

- Bounded action plans
- Minimal patches
- Patch provenance
- Mutation scope
- Patch validation
- Protected evaluation assets
- Dependency changes
- Behaviour-preserving constraints
- Untrusted directives
- Change classification
- Reviewable artifacts
- Content-bound approvals
- Atomic patching strategies
- Protecting user changes
- Scoped recovery
- Evidence-based summaries
- Alternative evaluation
- Search budgets
- Feedback loops
- Replanning boundaries
- Mini-project extension
- Patch critique
- Concurrency lab
- Assessment
- Module review

MODULE 7 · 25 H Sandboxing: Don't Give the Model the Keys to the Machine

- Isolation threat models
- Execution capabilities
- Structured execution
- Filesystem scope
- Egress policy
- Controlled provisioning
- Credential minimisation
- Resource limits
- Memory quotas
- Storage quotas
- Output limits
- Process lifecycle
- Least-privilege execution
- Execution identity
- Immutable inputs
- Tenant isolation
- Artifact-export validation
- Execution-outcome semantics
- Trusted test harnesses
- Safe operational logs
- Execution APIs
- Execution integration
- Concurrency lab
- Assessment
- Safety review

MODULE 8 · 25 H The Test-Fix Loop, Jobs and Review Dashboard

- Failure-feedback packaging
- Bounded repair loops
- Attempt isolation
- Independent tests
- Regression comparison
- Metric integrity
- Nondeterminism analysis
- Job architecture
- Job identity
- Job-state transitions
- Failure recovery
- Race handling
- Status delivery
- Run observability
- Outcome reporting
- Review experience
- Reviewer permissions
- Action boundaries
- Artifact authorisation
- Retention choices
- Product integration
- Fault isolation
- Small-load validation
- Assessment
- Integration review

MODULE 9 · 25 H

Evaluation: Turn the Demo into Evidence

- Success criteria
- Multidimensional scoring
- Fair baselines
- Dataset separation
- Evaluation composition
- Data leakage
- Metric arithmetic
- Attempt accounting
- Evaluation independence
- Specification alignment
- Rubric calibration
- Failure taxonomy
- Coverage versus precision
- Economic evaluation
- Distribution awareness
- Uncertainty fundamentals
- Experiment control
- Controlled experiments
- Component-value tests
- External validity
- Learning from failures
- Reproducible reporting
- Metric critique
- Assessment
- Module review

MODULE 10 · 25 H

Safety, Privacy, Quotas and Unit Economics

- Threat modelling
- Layered mitigation
- Safe adversarial fixtures
- Capability-escalation checks
- Memory isolation
- Retention policy
- Deletion propagation
- Policy composition
- Atomic quota reservation
- Budget uncertainty
- Usage reconciliation
- Cost-quality trade-offs
- Total operating cost
- Usage-based product design
- Trial controls
- Idempotent billing demonstrations
- Eventual-consistency experience
- Operational judgement
- Emergency controls
- Safe incident response
- Continuous review
- Failure-cost lab
- Isolation lab
- Assessment
- Module review

MODULE 11 · 25 H Productisation: From Agent to Usable Product

- Customer selection
- Customer discovery
- Demand evidence
- Trust and onboarding
- First-use experience
- Onboarding diagnostics
- Compatibility promises
- Interaction design
- Actionable status experience
- Decision support
- Designing for failure
- Accurate product messaging
- Pilot planning
- Consent and authority
- Confidentiality
- Commercial scoping
- Product positioning
- Support operations
- Product deployment
- Operational configuration
- Learning metrics
- Product storytelling
- Integration lab
- Assessment
- Release review

MODULE 12 · 25 H Capstone: Unseen Tasks, Honest Results and Handover

- Evaluation freeze
- Evaluation governance
- Fair evaluation
- Safety readiness
- Reproducibility
- Held-out execution 1
- Evaluation observation 1
- Held-out execution 2
- Evaluation observation 2
- Held-out execution 3
- Evaluation observation 3
- Held-out execution 4
- Evaluation observation 4
- Held-out execution 5
- Evaluation finalisation
- Final metric reporting
- Final control assessment
- Final approval assessment
- Final reliability assessment
- Post-evaluation transfer task
- Delivery artifacts
- Pilot-proposal defence
- Capstone defence A
- Capstone defence B
- Completion and next steps

Batch details

First class	25 Nov 2026
Days and time	Mon – Fri · 12:30 pm – 3:00 pm
Ends	11 May 2027 (public holidays may move this later)
Classes	120 classes · 12.5 h/week · home practice 8-12 hours/week
Format	Udaipur, Rajasthan · offline classroom · Hinglish (Hindi + English) · up to 15 learners

Scope: A narrow agent built on existing models. Training foundation models, unrestricted shell access and automatic production deployment are excluded, and sales are not guaranteed. Learners without JavaScript/TypeScript and backend basics should start with Full Stack Development with AI.