Faysal Aziz

Harness Engineering Foundations

Designing controlled environments for AI coding agents

FA-0625 · 1 day

Agentic & Generative AI · Software Development

Review the instructions, context, tools, permissions and feedback that surround an AI coding agent, then outline a small harness design.

Introduction

This one-day introduction examines the environment around an AI coding agent: task instructions, repository knowledge, tools, permissions, state, progress records and verification. These components influence how an agent works, but do not make model output automatically correct or safe.

Participants review repository-readiness examples and outline a controlled workflow for one small engineering task. The term harness engineering is used descriptively for emerging practices rather than as a formal qualification or universal standard; the course is not a production harness implementation programme.

Learning outcomes

The course teaches participants to:

- Explain model, agent, harness and execution environment in an AI-assisted development workflow.
- Distinguish prompt instructions, context selection and operational controls.
- Identify a basic harness's tools, state, permissions and feedback components.
- Review repository guidance, documentation and command discoverability.
- Define verification evidence, human review and escalation conditions.
- Recognise ambiguous instructions, excessive access, stale context and unsupported completion claims.

- Outline a starter harness for a narrowly scoped development task.

Prerequisites

- General understanding of software development
- Basic familiarity with source-code repositories
- Awareness of generative AI and AI coding assistants
- Basic knowledge of testing and command-line operations
- No machine-learning development experience required

Training outline

1. 1. Harness Engineering Concepts

- Definition and purpose
- How prompt and context practices relate to the broader execution environment; these practices overlap rather than form a universal historical sequence.
- Model, agent, harness and environment
- Potential benefits and limitations; model errors and weak controls can still cause failures.

2. 2. Essential Components

- Task instructions and acceptance criteria
- Context and repository knowledge
- Tools and execution environment
- State and progress tracking
- Enforced permissions and operational boundaries; prompt instructions alone are not an access-control mechanism.
- Observability and activity records

3. 3. Guidance and Operational Boundaries

- Repository instruction files as concise guidance and navigation, not a substitute for enforced controls.
- Coding and architectural standards
- Tool-use guidance
- Human review or approval points appropriate to the task and its effects.
- Escalation conditions

4. 4. Feedback and Verification

- Build and compilation checks
- Formatting and linting
- Automated testing
- Relevant security/policy checks with documented scope; passing a check is not proof of complete safety.
- Completion evidence tied to explicit acceptance criteria, including checks not performed.
- Human review

5. 5. Repository Readiness

- Predictable project structure
- Current and accessible documentation
- Discoverable development commands
- Clear architectural boundaries
- Reliable testing workflows
- Review stale instructions and artefacts; preserve useful records and remove only what is authorised.

6. 6. Controlled Task Workflow

- Task intake
- Context discovery
- Planning
- Controlled implementation
- Automated validation
- Correction and targeted revalidation; keep the original acceptance criteria unless an authorised requirement changes.
- Completion reporting

7. 7. Common Failure Patterns

- Ambiguous instructions
- Missing or excessive context
- Unrestricted tool access
- Weak verification
- Stale repository documentation

- Unsupported completion claims

8. 8. Starter Design Exercise

- Defining the supported task
- Selecting minimum required context
- Selecting permitted tools
- Define task-specific permissions, isolation, recovery and escalation boundaries at introductory design level.
- Defining verification requirements
- Establishing human oversight

Outline the instructions, context, allowed tools, review boundaries, progress record and completion evidence for one small sample task. Explain remaining risks rather than claim the design guarantees reliability.

© 2026 Faysal Aziz · Kuala Lumpur