M08 — DevOps, CI/CD & Automation Lab

Lab — Generative AI Boot Camp for Developers — Exported 2026-09-02

Time: ~20 min | **Day 3 AM** (shared with M09)

2026-09-02

Learning Objectives

Build an AI-assisted CI pipeline with automated PR review — one of the highest-ROI AI integration points for development teams.

2026-09-02

Concept Review: AI in CI/CD

Where AI Adds Value in Pipelines

Pipeline Stage	What AI Can Do	Example
Linting	Catch semantic issues linters miss	"This function always returns None in this branch"
PR Review	First-pass automated review	Style, missing error handling, security anti-patterns
Security Scan	Explain vulnerabilities + suggest fixes	"This SQL query is vulnerable to injection. Use parameterized queries instead."
Test Analysis	Identify flaky tests, coverage gaps	"These 3 tests fail intermittently — possible race condition"
Pipeline Config	Generate CI configs for new projects	"Generate a GitHub Actions workflow for a Python FastAPI project"

The AI-Enhanced PR Review Pattern

```
Developer pushes PR → CI runs AI review → AI posts review comments →
Developer addresses → Human reviewer approves → Merge
```

What AI catches: Style violations, missing error handling, hardcoded values, security anti-patterns, obvious bugs.

What AI misses: Business logic correctness, architectural fit, team conventions, whether the change actually solves the problem.

2026-09-02

Exercise: AI-Assisted CI Pipeline

Step 1 — Create an AI Review Script (10 min)

Create a script that reviews code changes using AI. Use **OpenCode**:

```
opencode run "Write a bash script called ai-review.sh that takes a filename as
an argument, reads the file, and sends it to an LLM API for code review. The AI
should check for: missing error handling, hardcoded values, security issues,
unclear variable names. Output the review as Markdown. Use OpenCode's built-in
LLM integration or curl to call the API directly."
```

Step 2 — Test with Sample Code (5 min)

Create a file with intentional issues (missing error handling, hardcoded credentials) and run your review script:

```
# Create a test file with issues
echo 'def connect_db():
    password = "admin123"
    conn = psycopg2.connect(host="localhost", password=password)
    return conn' > test_code.py

# Run your AI review
bash ai-review.sh test_code.py
```

2026-09-02

Did the AI catch: the hardcoded password? The missing error handling? The lack of type hints?

Step 3 — CI Integration Concept (5 min)

Sketch (don't build) how you'd integrate this into a real CI pipeline. Consider: - When should AI review run? (On every PR? Only on changed files?) - Where do review comments go? (PR comments? CI logs? Slack?) - What's the threshold for blocking a merge?

Deliverable

Working `ai-review.sh` script + test results + CI integration sketch.

2026-09-02

Troubleshooting

Symptom	Fix
Review script fails to run	Check file permissions: `chmod +x ai-review.sh`
AI review returns nothing	Verify API key and endpoint; check for JSON parsing errors
Review is too verbose	Add "Be concise. Limit to 5 findings max." to the AI prompt
Can't call LLM from script	Use OpenCode: `opencode run "Review this code: $(cat $1)"`

2026-09-02

Hints

- **Start simple:** A bash script that calls `opencode run` is perfectly valid.
- **The CI integration sketch (Step 3) is the most valuable part** — this is what you'd actually propose to your team.
- **Test with code you know has issues** so you can validate the AI's accuracy.

2026-09-02