HW05 – Performance Testing

HW05 – Performance Testing

1. General Information

Exercise ID	**HW05-AI**
Duration	10 hours
Deadline	Please refer to the submission link on Moodle
Form	Individual Assignment
Submission	Moodle (report)
Lecturers & TAs	Dr. Lam Quang Vu / Dr. Tran Duy Hoang / MSc. Tran Thi Bich Hanh / MSc. Truong Phuoc Loc / MSc. Ho Tuan Thanh
Contact	lqvu@fit.hcmus.edu.vn / tdhoang@fit.hcmus.edu.vn / ttbhanh@fit.hcmus.edu.vn / tploc@fit.hcmus.edu.vn / hthanh@fit.hcmus.edu.vn
AI Policy	Open — a declaration and an attached AI Audit Report are **mandatory**
Required Bloom-AI Level	G9.1 → G9.6, depending on the homework (see the *CLO Mapping*)

2. Guiding Principles

These principles define how you are expected to work throughout the series of assignments in this course. Read them carefully before you begin, as your submission will be evaluated against them.

- **AI-First strategy.** You are required to apply AI to the testing techniques covered in class. However, this does not mean issuing a single, generic prompt such as *"run a load test and tell me whether the performance is good."* Instead, you must guide the AI through every step of the technique as it was taught, using the AI as a disciplined assistant rather than a black box.
- **Human review.** Every result produced by the AI must be carefully reviewed by you, the student. You are fully responsible for the correctness of these results. You are expected to make any necessary corrections and

refinements — submitting the raw AI output without review is not acceptable.

- **AI Audit Report.** The entire process of using AI must be recorded in a complete log. You are encouraged to build Agent Skills that can automatically perform these activities on similar exercises. If you do **not** use AI, you must still declare this explicitly.
- **Documentation.** The whole working process must be documented in a text-based format such as Markdown.
- **Quality over completion.** Your work will be graded not merely on whether it is complete, but on the quantity and quality of the deliverables: the test plans, data files, raw logs and report views, resource/hardware evidence, the demo video, the AI analysis critique, and referenced links.

3. Learning Outcomes

By completing this assignment, you will be able to:

- Design and run Load, Stress, and Spike performance tests against the SUT's backend API using JMeter (or k6).
- Collect and present performance metrics with resource monitoring and multiple report views, and determine the endurance threshold on your own hardware.
- Use AI to analyse the results, then critically review its analysis — identifying where it misinterprets metrics and which of its proposed optimizations are feasible.
- Propose a continuous performance-testing pipeline.
- Demonstrate Bloom-AI competencies at levels **G9.2 (Apply)**, **G9.3 (Analyse)**, **G9.4 (Collaborate)**, and **G9.6 (Disrupt)**.

4. System Under Test (SUT)

SUT: EShop — a Vietnamese e-commerce demo application designed for testing practice.

Repository: https://github.com/ttbhanh/eshop-sut

The application's features are organised into the following pools:

- **Pool A — Authentication, Categories, and Products**

    - FR-01: Account registration
    - FR-02: Login and account lockout
    - FR-03: Forgot password and password reset (two steps)
    - FR-04: Personal profile management
    - FR-05: Product listing and search
    - FR-06: Product detail view
- **Pool B — Shopping Cart and Checkout**
    - FR-07: Shopping cart
    - FR-08: Checkout
    - FR-09: Discount coupons
    - FR-10: Order state machine
    - FR-11: Order history view (user)
- **Pool C — Web Admin**
    - FR-12: Access control
    - FR-13: Dashboard
    - FR-14: Category management (CRUD)
    - FR-15: Product management (CRUD)
    - FR-16: Product import from CSV
    - FR-17: Coupon management (CRUD)
    - FR-18: Order management (admin)
    - FR-19: User management (admin)
- **Pool D — Mobile App**

The SUT exposes a REST backend API that the web frontend consumes; consult the repository for the exact endpoints and ports.

5. Scope — Endpoint Selection

Target three backend API **endpoint groups**, mapping each to the SUT's API:

- **Read-heavy** — for example, product listing / search and product detail.

- **Auth-heavy** — for example, login, taking the account-lockout behaviour into account.
- **Transactional** — for example, add-to-cart and checkout / order creation.

As in the previous assignments, ensure that your selection is **not duplicated** among the members of your group: no two members may test the same workflow.

6. Requirements

For each of the following tasks, document your process in the main report and attach the required evidence. Review the relevant course lectures on performance testing before you begin.

Task 1 — AI-assisted test design and execution

Following the AI-first strategy, use an AI tool to design and generate the test plans, then review, fix, and take full responsibility for them.

- **Design and generate with AI.** Drive an AI tool — step by step, not with a single generic prompt — to design and generate three test plans: **Load**, **Stress**, and **Spike**. All three test plans must exercise the same end-to-end workflow, covering all three endpoint groups: **auth-heavy**, **read-heavy**, and **transactional**. For example, a virtual user may log in, browse or search products, then add an item to the cart and complete checkout. Have the AI help choose realistic parameters (think-time, ramp-up, thread / virtual-user counts) for each scenario, and briefly justify how the workflow covers each endpoint group.
- **Make the workflow data-driven.** Use CSV input data in the end-to-end workflow to parameterize requests (e.g., credentials, product IDs, or order payloads). You may use one or more CSV files, as appropriate for your workflow.
- **Use three different report views.** Across the three test plans, use three distinct listener / report types (e.g., View Results Tree, Summary Report, Aggregate Report); do not repeat a type. *(JMeter terminology; k6 users provide the equivalent distinct outputs.)*
- **Name each test plan** `{StudentID}_{ScenarioType}_{YYYYMMDD}`.
- **Review and fix (human review).** Critically review the AI-generated test plans and correct them. Report what the AI got wrong or missed — for

example, unrealistic ramp-up or think-time, wrong thread counts, weak assertions, or missing account-lockout handling — and explain *why* it missed them (prompt quality, model limitations, or characteristics of the endpoint). You are fully responsible for the final test plans.

- **Run as completely as possible, with evidence.** Execute all three scenarios and capture, for each run, a screenshot of the tool together with the backend process's resource usage (htop / Task Manager / Activity Monitor), plus a hardware report (a dxdiag / screenfetch screenshot and a spec table). When Stress/Spike runs trigger the 3-fail login lockout, reset it between runs and document the steps. Produce the raw `.jtl` logs and the HTML report folders.
- **Determine the endurance threshold.** Run a short endurance / soak test (around 10–15 minutes at sustained load) to empirically find your hardware's threshold, reported with concrete numbers (e.g., maximum stable RPS, memory ceiling).
- **Record a demo video.** An unlisted YouTube video of **at least 6 minutes total** (you may split it into one clip per scenario), showing the tool and the resource monitor **in the same frame**, with your own Vietnamese narration.
- **Report issues.** Log any genuine bugs or performance issues (error responses, crashes, functional regressions) on your GitHub Issues page with screenshots. Logging performance issues such as high latency or elevated error rate is encouraged but not penalised if absent.

Task 2 — AI analysis and misinterpretation hunt

Following the AI-first strategy, use AI to analyse your results, then critically review what it produces — the analysis is the AI's output, and the review is yours.

- **Analyse with AI.** After collecting the raw results, prompt an AI tool to analyse the `.jtl` logs and suggest performance thresholds.
- **Review and correct (human review).** Critically review the AI's analysis and identify where it misinterprets or misreads the metrics. For each misinterpretation, cite the **correct value from your raw `.jtl` log** and explain the error.
- **Judge the AI's recommendations.** Have the AI propose optimizations (e.g., adding a database index, a connection pool, or enabling SQLite WAL) and

classify each as **feasible or hallucinated**, with reasoning.

Task 3 — Continuous Performance Testing proposal (Disrupt)

- In your conclusion, propose a **continuous performance-testing model** that watches the SUT's commits, decides whether to run performance tests, and flags p95 regressions. Include a **flow chart** and a discussion of the **trade-offs** (cost, false alarms).

7. Agent Skill

- You are encouraged to build an Agent Skill that applies this performance-testing and log-analysis workflow, so that it can be reused on additional endpoints in future testing tasks.
- Submit the skill together with a demonstration video (YouTube link) that shows, end to end, how you used the skill on a complete endpoint group.

8. Allowed Tools and Bloom-AI Level

You may use the following tools, and you must declare them in your AI Audit Report:

- JMeter (default) or k6 (bonus).
- Any AI tool of your choice (e.g., ChatGPT, Claude, Gemini) — for log analysis.
- A resource monitor (htop / Task Manager / Activity Monitor).

The required Bloom-AI level for this homework is **G9.2 (Apply)**, **G9.3 (Analyse)**, **G9.4 (Collaborate)**, and **G9.6 (Disrupt)**.

9. AI Audit Report (Mandatory Appendix)

Attach the AI Audit Report as an appendix. Use the content of the given AI Templates if needed.

- If you did not use AI, declare: *"I do not use any AI help in this exercise."*
- If you did use AI, declare: *"I use AI tools for the following tasks,"* and include the following information for each interaction:
  - Name of the AI tool
  - Date and time

- Your prompt
- The AI output

To simplify this process, you are encouraged to create a skill or rule that extracts the information above automatically after an AI session.

10. AI Critique (200–300 words, Mandatory)

Write a paragraph of 200–300 words critiquing the AI. Address the following questions: Where did the AI get something wrong, biased, or incomplete? Why did it fail to catch the issue? What principle have you learned about collaborating with AI during this assignment?

Use the content of the given AI Templates if needed.

11. Anti-AI-Cheat Constraints

This homework relies on real, attributable execution evidence. The following must not be AI-generated or fabricated, and the TAs verify them during grading:

- The test-plan filenames, which must match `{StudentID}_{ScenarioType}_{YYYYMMDD}`.
- The raw `.jtl` log files, attached in full — not only the summary.
- The demo video, which must show the tool and the resource monitor in the same frame with your own voice narration.
- The hardware report, whose hostname matches your previous homework deployments.

12. Git Commit Log

- Create a new Git commit for each step of the procedure (for example: each scenario's test plan, the AI analysis, and the continuous-testing proposal).
- Provide the Git commit log in a text-based file format.

13. Oral Defense

A randomly selected **30% of students** may be invited to a 5–7-minute oral defense during the week following the deadline, to explain how they completed this homework.

14. Submission Regulations

- **Filename format:** `<StudentID>_HW05_AI_Performance_<SelfAssessedGrade>.zip`
  - *SelfAssessedGrade:* a 3-digit number in the range [000, 100].
  - *Example:* `25127001_HW05_AI_Performance_090.zip`
- **Required contents of the `.zip`:**
  - Main report (Markdown + PDF), including the performance-testing report and your AI-analysis critique.
  - The public GitHub repository link (test plans and data files).
  - The three test plans (Load / Stress / Spike) following the filename convention.
  - The three raw `.jtl` logs and the three HTML report folders.
  - The resource-monitor and hardware-spec screenshots.
  - The unlisted YouTube demo video link.
  - AI Critique and AI Audit Report (Markdown + PDF).
  - Git commit log (text file).
  - Bug report, with screenshots of any issues on the GitHub Issues page (if any).
  - A `README.md` containing the self-assessment table (below) and a test summary report: scenarios run; endpoint groups covered; the endurance threshold (with numbers); number of bugs / performance issues; and the demo video link.
  - Any other supporting materials.
- Submit to Moodle. For the deadline, refer to the submission link.

15. Assessment Template

No.	Criteria	Grade	Self-Assessed Grade
1	Task 1 — Load testing	30	
2	Task 1 — Stress testing	20	
3	Task 1 — Spike testing	20	

No.	Criteria	Grade	Self-Assessed Grade
4	Task 2 — AI analysis + misinterpretation hunt (with correct values from raw logs)	10	
5	Task 3 — Continuous Performance Testing proposal (G9.6)	10	
6	Agent Skills	10	
	Total	**100**	

16. References

- ISTQB Foundation Level Syllabus (latest edition).
- Hardman, P. (2025). *A Post-AI Learning Taxonomy.*
- Fuster Rabella, M. (2025). *OECD Education Working Paper No. 338.*
- Anthropic (2025). *Building Reliable AI Test Agents* — engineering blog.
- DeepEval & Promptfoo documentation — LLM testing frameworks.

17. Other Regulations

- Late submission is **not permitted**.
- Missing any required document results in **0 points**.
- Copying between students — **including prompts** — results in a **grade of 0 for both parties**.