RED-TEAMING AI SYSTEMS

CYB-4203/6203: SECURE AND TRUSTWORTHY AI

Unit 9.3 — Monday, April 20, 2026

Dallas Elleman — Spring 2026

Interactive version at
https://dallaselleman.github.io/cyb-4203-6203-spring-2026/course_materials/presentations/revealjs/pres-19.html

RED-TEAMING

The interactive and iterative process of simulating real-world attacks to identify vulnerabilities in systems.

What kinds of systems?

Physical | Technical | Human | Hybrid

Validates defenses

Realistic assessment of how security controls stand up to true threats

Improves detection

Identifies gaps in security monitoring and response times

Proactive risk reduction

Uncovers vulnerabilities before malicious actors do

TERMINOLOGY

Red Team

Attackers — simulate adversaries, probe for weaknesses.

Blue Team

Defenders — detect, respond, harden the environment.

Purple Team

Collaboration — attackers & defenders working together in the open.

WHITE-BOX TESTING

Full system information provided to the attacker — source code, architecture diagrams, credentials, internal documentation.

BLACK-BOX TESTING

Only basic information provided — e.g. a company name or a public URL. The attacker must discover everything else.

White Team / Cell — referee and oversight role; controls rules of engagement, adjudicates disputes

PENTESTING VS. RED-TEAMING

PENETRATION TESTING

Scope	Narrow — a specific web app, API, or network segment.
Duration	Short — days to weeks.
Goals	Identify as many system vulnerabilities as possible; demonstrate compliance; patch technical flaws.
Awareness	Security team is usually aware of the test.

RED-TEAMING

Scope	Broad — physical security, social engineering, full stack, people and process.
Duration	Long — weeks, months, or continuous.
Goals	Identify technical *and other* vulnerabilities; evaluate security processes and incident response; simulate advanced persistent threats (APTs).
Awareness	Security team may or may not be aware of the test.

Different tools for different questions.
Pentests answer "is this thing broken?"
Red teams answer "could we actually detect and stop a real adversary?"

THE RED-TEAMING LIFECYCLE

Common across frameworks (PTES, OWASP WSTG, MITRE, NIST)

We'll walk through each step, then apply the same lens to AI systems.

STEP 0

PRE-PLANNING & SCOPING

Define objectives, goals, and rules of engagement.

- What is the **target system**? (specific apps, networks, physical locations)
- What system components are **within scope**?
- What **safety considerations** exist? (production impact, customer data, legal constraints)
- What red-team methods and behaviors are **allowed**, and which are **out of bounds**?

Establish communication, escalation, and oversight — who gets paged if something breaks? Who has the authority to stop the exercise?

STEP 1

RECONNAISSANCE

Information gathering — learning the target without (yet) touching it.

OSINT

Open-source intelligence — what can you learn about the system without touching it? Public docs, DNS records, employee LinkedIn profiles, leaked creds, code on GitHub.

External Surveillance

How does the system behave externally? What response patterns, timings, and error messages leak information about internal architecture?

Map the External Attack Surface

Enumerate every public-facing component — domains, subdomains, APIs, login portals, ports, cloud assets, mobile apps, third-party integrations.

STEP 2

THREAT MODELING

Identify, enumerate, and analyze potential vulnerabilities.

? What **data or other systems** are touched?

? What **actions** can the system take on its own or on behalf of users?

? What is the **blast radius** — potential scope of damage, disruption, or unauthorized access if the system is compromised?

STEP 3

ATTACK PLANNING

Select attack targets — which vulnerabilities are you going to try, and in what order?

Build a test matrix of payloads and expected results.

Determine the attack sequence and phases — initial access, persistence, privilege escalation, lateral movement, exfiltration.

STEP 4

EXECUTION, MOVEMENT & ITERATION

Launch attacks

Execute against the plan. Live environments surprise you.

Carefully document results

Every payload, every response, every timestamp. This becomes your report and your evidence.

Scan for newly emergent vulnerabilities

A foothold opens up a new interior attack surface — credentials, tokens, internal services.

Iterate and adapt

Your plan was a hypothesis. What actually works may be something you didn't anticipate.

STEP 5

REPORTING & DEBRIEFING

Communicate findings with a well-written, professional report.

What goes in the report

Executive summary • methodology • findings ranked by severity • artifacts and diagrams • exploit evidence (screenshots, logs, PoCs) • timeline of the engagement

Recommend mitigations

Every finding gets a remediation. Be specific: technical controls, process changes, training, architectural redesign. Rank by impact vs. effort.

A great engagement with a bad report is a wasted engagement. The report is the deliverable.

RED-TEAMING AI/ML SYSTEMS

RED-TEAMING AI/ML SYSTEMS

Simulating real-world attacks to identify vulnerabilities in artificial intelligence systems and components.

HOW IS RED-TEAMING AI/ML SYSTEMS DIFFERENT?

Probabilistic behavior

Outputs are non-deterministic. A payload that fails 9 times may succeed on the 10th. You test with distributions, not single shots.

Novel vulnerability classes

Biases, hallucination / confabulation, data leakage, harmful content generation, agentic misbehavior — categories traditional security doesn't cover.

Heterogeneous systems

Computer vision, recommenders, classifiers, autonomous control, generative models, agents. Different attack surfaces per system type.

Lifecycle-wide attack surface

Testing must span the full AI/ML lifecycle: training data, model training, deployment, and inference-time behavior.

RED-TEAMING AI/ML SYSTEMS (1 OF 2)

COMPUTER VISION

Recognition & detection

ADVERSARIAL PERTURBATION

Modify image pixels slightly to cause misclassification — often imperceptible to humans.

DATA / MODEL POISONING

Alter training datasets or inject backdoors; triggered inputs produce attacker-chosen predictions.

RECOMMENDERS & CLASSIFIERS

Filters, anomaly detection, ranking

MODEL EVASION

Craft inputs that bypass security measures — spam filters, fraud detection, malware classifiers.

BIAS & FAIRNESS

Test whether outputs are discriminatory or demographic-dependent.

DATA / MODEL POISONING

Alter the model's output for selected inputs without degrading overall accuracy.

RED-TEAMING AI/ML SYSTEMS (2 OF 2)

AUTONOMOUS SYSTEMS

Self-driving cars, anthrobots, drones

ADVERSARIAL PERTURBATION / EDGE CASES

Simulate unexpected, rare, or adversarial scenarios to test decision-making robustness — especially at the long tail where safety really matters.

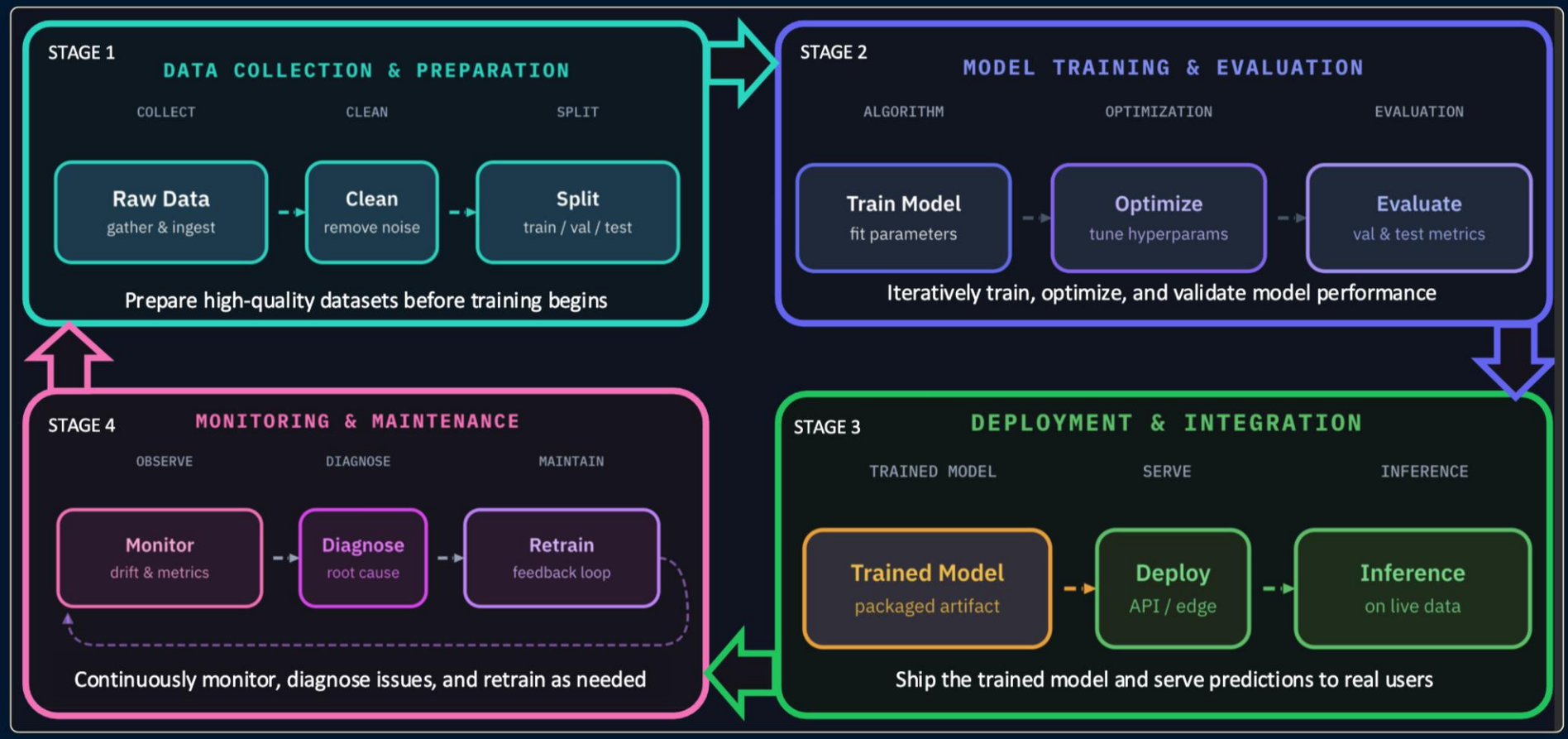
GENERATIVE AI

Model-only, non-agentic

INPUT / OUTPUT VULNERABILITIES

Probe across modalities — **text**, **audio**, **image**, **video**. Each modality introduces its own attack vectors and safety failure modes.

RED-TEAMING ACROSS THE AI/ML LIFECYCLE



FOCUS FOR THE REST OF THIS SESSION

Phase 3 — **Deployment & Integration (inference time)**

RED-TEAMING FOR LLMS

(model-only, non-agentic)

INHERENT VULNERABILITIES

Built into how LLMs work

MODEL STRUCTURE

Data ↔ control path confusion • context-window limits

MODEL BEHAVIOR

Hallucination / confabulation • sycophancy • deception

ADVERSARIAL VULNERABILITIES

Introduced by a motivated attacker

TARGETING THE MODEL / DATA

System prompt extraction • model inversion / training-data extraction • model distillation

TARGETING MODEL BEHAVIOR

Getting the model to do bad things — jailbreaks, policy violations, harmful outputs.

Full taxonomy covered in Presentation 11.

OWASP TOP 10 FOR LLM APPLICATIONS

LLM01:2025 Prompt Injection

A Prompt Injection Vulnerability occurs when user prompts alter the...

Read More

LLM02:2025 Sensitive Information Disclosure

Sensitive information can affect both the LLM and its application...

Read More

LLM03:2025 Supply Chain

LLM supply chains are susceptible to various vulnerabilities, which can...

Read More

LLM04:2025 Data and Model Poisoning

Data poisoning occurs when pre-training, fine-tuning, or embedding data is...

Read More

LLM05:2025 Improper Output Handling

Improper Output Handling refers specifically to insufficient validation, sanitization, and...

Read More

LLM06:2025 Excessive Agency

An LLM-based system is often granted a degree of agency...

Read More

LLM07:2025 System Prompt Leakage

The system prompt leakage vulnerability in LLMs refers to the...

Read More

LLM08:2025 Vector and Embedding Weaknesses

Vectors and embeddings vulnerabilities present significant security risks in systems...

Read More

LLM09:2025 Misinformation

Misinformation from LLMs poses a core vulnerability for applications relying...

Read More

LLM10:2025 Unbounded Consumption

Unbounded Consumption refers to the process where a Large Language...

Read More

genai.owasp.org/llm-top-10

LLM JAILBREAKING

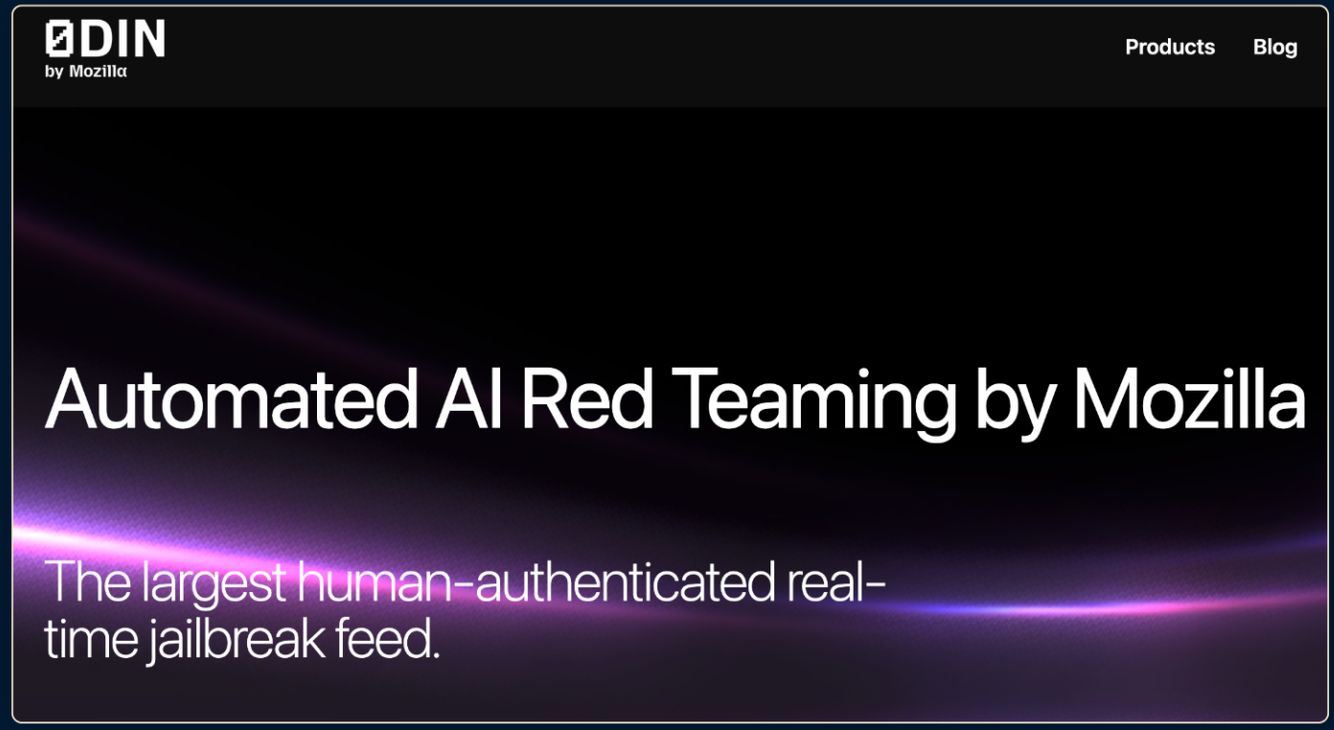
The class of attacks that attempt to subvert built-in safety filters placed by model developers.

Many goals

Restricted outputs, dangerous / unethical content, operational abuse, policy bypass, extraction of refused information.

Many strategies

Role-playing • formatting / encoding tricks • model "social engineering" • multi-turn pressure • context stuffing • adversarial suffixes.



0din.ai/research/taxonomy/techniques

LLM TRAINING DATA EXTRACTION

Probing the model to leak sensitive information (including PII) from its training data.

Extracting Training Data from LLMs

Carlini et al. — USENIX Security 2021

- Recovered hundreds of verbatim sequences from GPT-2, including names, addresses, and code
- Established memorization as a real, reproducible attack surface

arxiv.org/abs/2012.07805

Scalable Extraction from Production LLMs

Nasr et al. — 2023

- "Repeat the word 'poem' forever" caused ChatGPT to emit training data verbatim
- Gigabytes of data extracted from a deployed, aligned model

arxiv.org/abs/2311.17035

PII Leakage in Language Models

Lukas et al. — IEEE S&P 2023

- Systematic measurement of PII memorization and extraction rates
- Showed common defenses (DP, scrubbing) reduce but don't eliminate leakage

arxiv.org/abs/2302.00539

RED-TEAMING AGENTIC SYSTEMS

Models + tools + memory + orchestration

BROADER SCOPE

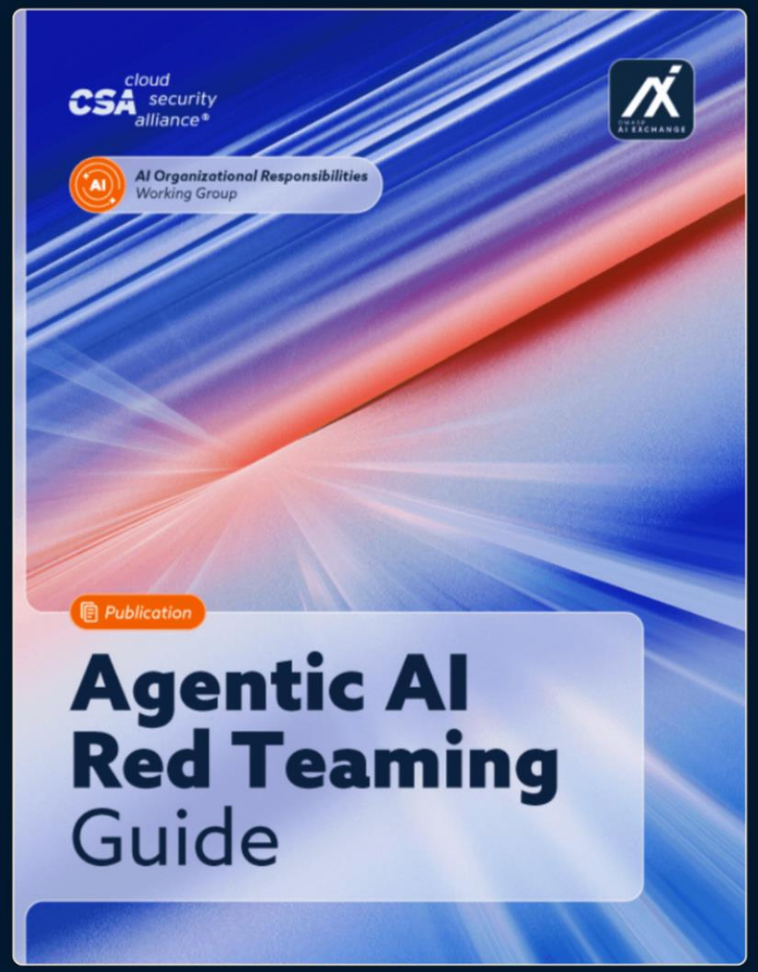
Tests the full agentic surface: **pipelines, plugins, tools, inter-agent communication, memory stores, and broader system dynamics** — not just the model in isolation.

PROMPT INJECTION — OWASP LLM01

A class of attacks against systems and applications built on top of LLMs that work by **concatenating untrusted input with trusted input**.

Untrusted input shows up as: an attacker-controlled webpage the agent browses, a malicious document it ingests, a poisoned tool response, a user-supplied file — anything the agent treats as data but the model might treat as instructions.

RED-TEAMING FRAMEWORKS & RESOURCES



CSA Agentic AI Red-Teaming Guide

FRAMEWORKS

- AI Security Institute — Inspect Framework
- MITRE ATLAS — AI Security 101

NIST

- AI 100-2e2025 — Adversarial ML Taxonomy / Terminology of Attacks and Mitigations
- AI 600-1 — AI RMF: Generative AI Profile

GUIDES

- Promptfoo Red-Teaming Guide

PRACTICE ENVIRONMENTS

- Lakera Gandalf — LLM jailbreaking challenges
- Lakera AgentBreaker — agentic-system attack challenges

RED-TEAMING TOOLS

Promptfoo

LLM eval & red-team framework

- Pluggable attack strategies & providers
- Test matrix & CI integration

PyRIT

Microsoft — Python Risk Identification Tool

- Open-source automated red-team toolkit
- Scriptable multi-turn attacks

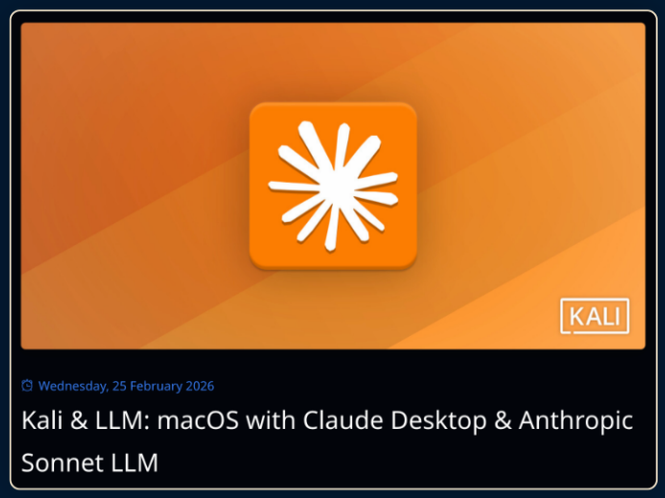
AgentDojo

ETH Zurich — agent benchmark

- Prompt-injection evaluation for tool-using agents
- NeurIPS paper · poster

Raptor — LLM pentester

Shannon — LLM pentester



Kali + Claude Desktop

MORE TOOLS & RESOURCES

PLAYGROUNDS & HANDS-ON

Microsoft AI Red-Teaming Playground Labs
Hands-on scenarios from Microsoft's AI red team

AI Red-Teaming for Complete First-Timers
Promptfoo blog — entry-level walkthrough

Top Open-Source AI Red-Teaming & Fuzzing Tools (2025)
Promptfoo blog — current tool landscape

Penligent — OpenClaw AI Security Test
Red-teaming high-privilege agents

GUIDES & METHODOLOGIES

Guide to AI Red-Teaming with MITRE ATLAS
TheCyberThrone — ATLAS-driven methodology

DeepTeam — Complete Guide to Agentic AI Red-Teaming
Comprehensive agentic coverage

Palo Alto Networks — AI Red-Teaming Guide
Vendor-neutral overview

OWASP — AI Security Solutions Landscape (Q2 2026)
Ecosystem map of red-teaming vendors and tools

ESSAYS & RESEARCH

Simon Willison — Jailbreaking vs. Prompt Injection
Canonical framing of the distinction

Anthropic — Challenges in Red-Teaming AI Systems
Lessons from a frontier lab's red team

Tree of Attacks — NeurIPS 2024
Automated jailbreaking of black-box LLMs

FINAL PROJECT

THE ENGAGEMENT

Pentest an **OpenClaw** instance running an open-source mid-sized LM deployed on cloud infrastructure.

Teams

6 teams of 2–3 — instructor-assigned

Timeline

More details later today. Full discussion on **Wednesday, 4/22**.

GRAD STUDENTS

Research paper huddle after class — let's align on topics, scope, and timeline.