CS486C – Senior Capstone Design in Computer Science

Project Description

Project Title: Developing AI-Supported Learning Environments for Static Analysis and Cybersecurity Education	
Sponsor Information:	Lan Zhang, Assistant Professor Lan.zhang@nau.edu

Project Overview:

Understanding software security and program analysis is critical in a world increasingly dependent on complex, interconnected systems. Static analysis plays a central role in identifying security vulnerabilities before code is deployed. However, mastering this domain is difficult—even for students with programming experience. Capture The Flag (CTF) challenges based on static analysis are effective learning tools, but learners often struggle without expert guidance. Human tutors are expensive and not always available. There's a clear need for scalable, accurate, and consistent educational support in this space.

We are a team building intelligent tutoring systems that support students learning advanced software topics. Our current focus is on creating an **AI tutoring agent for static analysis CTF challenges**. By combining modern AI techniques with formal knowledge representation, we aim to support learners in developing a deep understanding of program analysis, vulnerability patterns, and reasoning about code statically—without relying on dynamic execution.

Although large language models can answer general programming questions, they struggle in the domain of static analysis and CTFs due to several key limitations:

- **Lack of grounding in authoritative content**: Current models often generate plausible-sounding but incorrect or imprecise explanations.
- **Inconsistent answers**: Learners receive different answers to similar questions, leading to confusion and mistrust.
- **Weak domain knowledge**: Static analysis concepts like dataflow, abstract interpretation, or taint propagation are rarely covered well in general-purpose models.
- **No structured explanation path**: Learners have no access to concept hierarchies or prerequisite knowledge tailored to the analysis domain.
- **Limited support for inquiry-based learning**: AI agents are often passive responders rather than active tutors that guide through question-answering loops.

As a result, students engaging with static analysis CTFs face steep learning curves and limited support, slowing down their learning process and reducing long-term understanding.

The Envisioned Product:

We propose building an **AI tutoring agent** designed specifically to answer student questions about **static analysis CTF challenges**, leveraging **Retrieval-Augmented Generation (RAG)** and a **curated knowledge graph** of program analysis concepts. This will be delivered as a **web-based interactive tutoring system** that allows students to ask natural language questions and receive grounded, consistent, and educationally structured answers.

Minimum Viable Product (MVP):

- Retrieval-Augmented Generation system combining an LLM with curated static analysis CTF documentation and explanations
- A static analysis knowledge graph encoding key concepts, relationships, and example challenges

- Web interface where students can pose questions about CTF problems and receive AI-generated tutoring responses
- Context-aware follow-up interaction: students can ask clarifying questions or go deeper on specific concepts
- Consistent, canonical explanations anchored in the knowledge graph to ensure accuracy across sessions

A Useful System:

- Interactive exploration of the knowledge graph to help students understand the "big picture"
- Concept tagging of CTF challenges to help students locate related problems
- Personalized response scaffolding based on user skill level or prior questions
- Summary responses that group related questions and concepts into learning paths

Stretch Goals:

- Question templates or guided prompting to help students formulate better queries
- A feedback loop where students rate explanations to improve future responses
- Support for instructor-created content or annotations in the knowledge graph

This system will empower students to explore static analysis CTFs more effectively, providing them with expert-level guidance at scale. The tutoring agent will help learners deepen their conceptual understanding of program analysis, vulnerability detection, and formal reasoning, without requiring a human expert to be constantly available. Instructors will benefit from reduced overhead, and learners will benefit from personalized, high-quality tutoring. The long-term impact is a more confident, better-prepared generation of developers and security researchers.

Knowledge, skills, and expertise required for this project:

- **Full-Stack Web Development**
- Experience building **Retrieval-Augmented Generation (RAG)** pipelines
- Ability to design or work with a **domain-specific knowledge graph**

Equipment Requirements:

- We have a GPU server in our lab.
- No specialized software or equipment should be required for this project, beyond a standard software development stations and free IDEs, frameworks, and other tools.

Software and other Deliverables:

- **Fully Functional Web Application**
  The completed AI tutoring system will be delivered as a deployed, browser-accessible web application built on an MCP server architecture. It will be tested using real static analysis CTF challenge data to ensure that it performs reliably, responds accurately to student questions, and integrates seamlessly with the retrieval and knowledge graph components.
- **User Manual**
  A comprehensive and clearly written **User Manual** will be provided. It will cover system setup, configuration, user onboarding, and guidance for typical usage scenarios (e.g., asking questions, exploring concept relationships). The manual will be designed for both students and instructors.
- **As-Built Report**
  A detailed **as-built technical report** will document the full system architecture, development process, major design decisions, challenges encountered, and solutions implemented. It will serve as a professional reference for future development, maintenance, or research use of the system.
- **Professionally Documented Codebase**
  The full source code will be thoroughly documented using consistent, professional commenting standards and inline documentation. It will be delivered in two forms:
  - As a **hosted repository** on a version control platform (e.g., GitHub or Bitbucket)
  - As a **physical archive** (e.g., on a USB drive) to satisfy institutional backup or transfer requirements

- **Open-Source Licensing**
  All created software, including the web-based GUI and backend components, will be released under a permissive open-source license (e.g., MIT or Apache 2.0). This ensures that the scientific and educational community can freely access, use, and build upon the system.