COMP39/9900 Computer Science/IT Capstone Project
School of Computer Science and Engineering, UNSW

Project Number: P24

Project Title: Enhancing Code Security with AI-Driven Vulnerability Detection and Explanation

Project Clients: Dipankar Chaki and Helen Paik

Project Specializations: Software development; Artificial Intelligence (Machine/Deep Learning, NLP); Security/Cyber Security.

Number of groups: 2

Background:

In the digital age, software security is a critical concern, with vulnerabilities posing significant risks. This capstone project aims to harness the power of Artificial Intelligence (AI) and Large Language Models (LLMs) to create an innovative tool that revolutionizes how developers understand and fix vulnerabilities in their code. Utilizing CodeBERT, a state-of-the-art LLM specifically designed for understanding programming languages, this project will develop a system that not only identifies potential security vulnerabilities in source code but also provides clear, understandable explanations of these issues. The aim of this project is to empower developers with an AI assistant that enhances code safety while educating them on best security practices, thereby fostering a culture of proactive security mindfulness.

Requirements and Scope:

Programming Languages: The tool will support essential languages such as C/C++, Python, and Java, covering a broad spectrum of software development scenarios.

Tools and Technologies: The project will leverage CodeBERT for its powerful code comprehension capabilities. We will utilize Python to handle backend development and integrate with advanced Machine Learning (ML) and Deep Learning (DL) frameworks like PyTorch or TensorFlow to optimize and manage the AI models.

Development Stages: The project involves multiple phases, including the initial setup of the AI environment, the fine-tuning of CodeBERT to adapt to specific types of code vulnerabilities, the development of a user-friendly VS Code extension, and extensive testing to ensure the tool's reliability and ease of use in a real-world development setting.

Model Training: Fine-tuning the pre-trained CodeBERT model requires adjusting its parameters to specialize in detecting and explaining specific code vulnerabilities. This task will involve manipulating training hyperparameters such as learning rate, batch size, and epoch count to refine the model's accuracy and performance. This stage will leverage DL techniques to train the model effectively on a curated dataset of annotated code samples.

Interface Development: Developing a seamless and intuitive plugin for Visual Studio Code, which will allow users to analyse code directly within their development environment. This plugin will interact with the AI model to provide real-time feedback and explanations, enhancing the developer's ability to quickly understand and rectify software vulnerabilities.

Testing and Validation: Ensuring the tool's effectiveness through comprehensive testing, including automated unit tests to confirm each component's functionality and user acceptance

testing to gauge the tool’s practical utility. Feedback from these tests will be crucial in iterating and improving the tool's design and functionality.

Required Knowledge and skills:

Required Skills:

Programming Proficiency: Advanced proficiency in Python is essential for handling the backend development and AI integration tasks. This skill is crucial as Python serves as the backbone for most machine learning operations and scripting within this project.

Machine Learning Knowledge: A strong foundation in ML and DL, especially in natural language processing and neural networks, is crucial. Hands-on experience with ML frameworks like PyTorch or TensorFlow is required to effectively manage the AI models' training and operational phases.

Data Handling: Effective skills in data manipulation and analysis are necessary to manage and preprocess datasets during the AI model training phase. This includes cleaning, structuring, and optimizing data inputs to ensure that the model training is efficient and productive.

Desired Skills/Good to Have:

Security Basics: While not strictly necessary, having a fundamental understanding of common software vulnerabilities and security practices would greatly benefit students in implementing and testing the AI model's vulnerability detection capabilities.

VS Code Extension Development: Familiarity with developing extensions for Visual Studio Code is highly desirable. Knowledge of the VS Code API and the ability to integrate complex functionalities within the VS Code environment would enhance the project’s implementation and user experience but is not required for initial participation in the project.

Expected outcomes/deliverables:

Source Code: Fully functional source code for both the AI model's training algorithms and the VS Code extension.

Documentation: Comprehensive documentation that details every aspect of the project, from system architecture and AI model configurations to step-by-step setup and usage instructions.

User Guide: A detailed guide providing clear instructions on how to install and efficiently use the VS Code extension. This guide will include troubleshooting tips and user FAQs to assist developers in leveraging the tool to its fullest potential.

Final Report: An in-depth report documenting the entire project process, from conception through development to final testing. The report will evaluate the tool's impact on improving code security and developer practices.

Presentation: A polished presentation designed to showcase the project's innovative features, demonstrate its functionality within the VS Code environment, and highlight the practical applications and benefits of integrating AI into software development workflows.

Supervision:

Dipankar Chaki

Additional resources: