FINAL YEAR PROJECT SPECIFICATION AND PLAN

Project Title: Vulnerability Management Personal Assistant

Created By

Khairul Amirin Bin Syahrean

Student Number: C00265680

4th Year (Hons) Cybercrime and IT Security

South East Technological University Carlow Campus

Supervised by

Richard Butler

October 27, 2023

Table of Contents

Project Specification

Title: Vulnerability Management Personal Assistant

Brief Description

An AI-Powered Personal Assistant to oversee threats and vulnerabilities of an organisation and execute scanning tasks through Python.

Technologies

1. Python (Base Programming Language)
2. GPT 3.5 (Base Large Language Model)
3. ChromaDB (local database application)
4. Streamlit (Python-based web framework)
5. PyCharm (Code Editor)
6. GitHub (Code repository)
7. Nmap (Network discovery tool)
8. Nessus Essentials (Basic Vulnerability Scanner)

Full Description

There has been a meteoric rise in the development of Artificial Intelligence. Technological advancements have led to machines capable of learning from data and make intelligent decisions. Nowadays major industries and organizations are clamouring for AI-based applications to streamline operations and automate tedious tasks (Uzialko, 2023). IT users are encouraged to learn, adapt, and implement AI into their workflow.

For my final year project, I am creating an AI-powered personal assistant that oversees vulnerability management of a company's IT security infrastructure. It would mainly be a user-friendly personal chatbot allowing users to obtain insightful information on threats or vulnerabilities of the company's systems at a moment's notice. Example information would be listing known vulnerabilities from exported reports of Nessus scanners, historical data of company machines and possibly remediation recommendations. To make it truly stand out from competitors, I have added user personalization to the assistant. In other words, the AI model would be able to identify needs through constant interaction and tailor the conversation and

workflow to the user. This configuration would be carried over the next time the user logs in to the application.

The purpose of this project is to showcase how AI can be integrated into vulnerability management of IT systems for businesses and organizations. It will mainly assist cybersecurity personnel with their daily routines of discovering and examining known vulnerabilities. For later iterations of the application the bot would feature the capability to run impromptu scans outside established scanning policies using the available tools such as Nmap for network scanning and Zap for a more comprehensive vulnerability scan. The application can also triage vulnerabilities to appropriate response teams and notifying other members through email.

Project Deliverable List

Mandatory

GUI - Simple GUI with user input box and AI output box.

Chatbot capability – Can interact with user using Large Language Model.

Database – Database to store scan reports

Vulnerability Reporting – List known vulnerabilities from scan reports.

Discretionary

GUI – Overall interface fits the description of a capable cybersecurity tool.

Remediation – Provide recommendations based on existing knowledge base.

Vulnerability Insight – Analyse database and historical data and provide better insight to vulnerabilities.

User Account – Cybersecurity members can log into the application using their own credentials.

Exceptional

Release - Release quality product including attractive GUI and features optimized for speed and ease of use.

Advanced Chatbot – Human-like interaction and responses.

Personalized Workflow – Model learns the user's needs and preferences thus able to tailor the workflow to them for current and future application sessions.

Comprehensive Asset Rundown – Provide insight to assets using historical data (common vulnerabilities)

Scanning capability – Can execute python scripts for running Nmap and Nessus functionalities by installing the appropriate libraries. Results will be replied to the user on the web-interface.

Targeted Remediation – Provide recommendations based on existing knowledge base.

Incident Triaging – Assign vulnerabilities to appropriate response team through email.

Project Plan

Project Timeline

There are various directions we can take to create our Personal Assistant application. Before delving into project requirements and recommendations we will explore our estimated project timeline.

Gantt Chart of Project Timeline

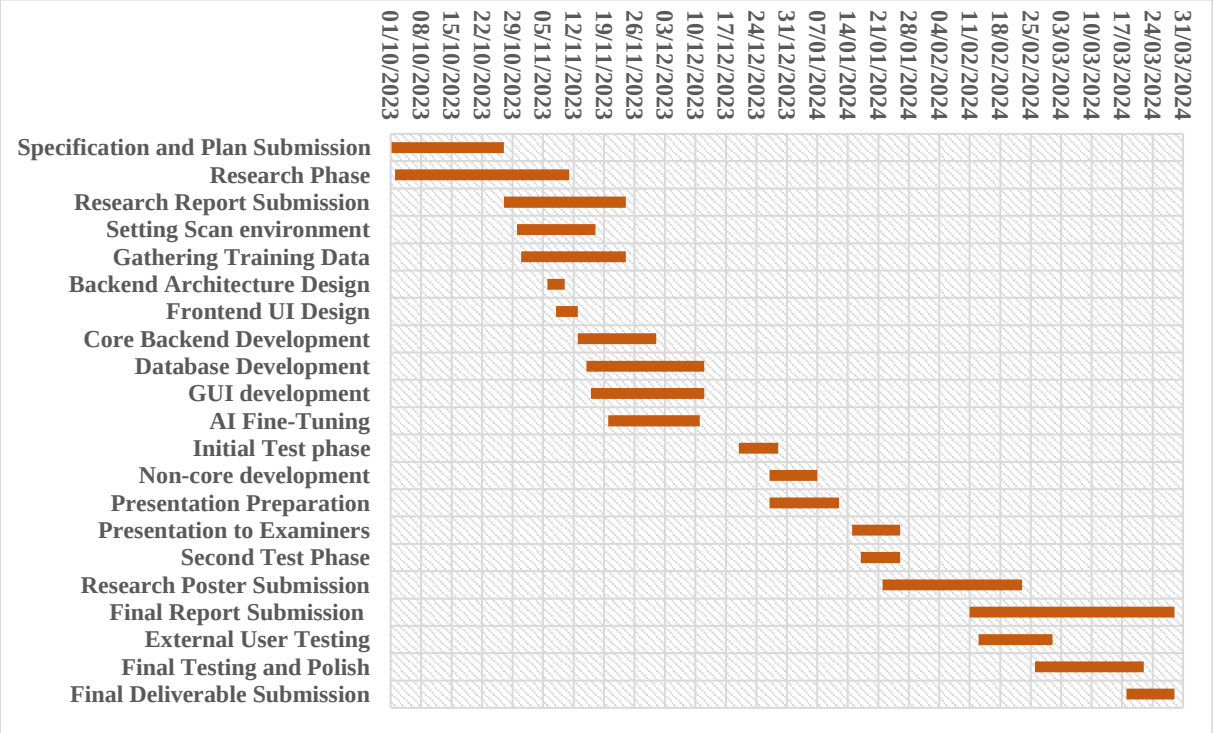

Figure 1: Project Timeline

It should be noted that the milestones are influenced by the deliverable deadlines.

Key milestones to achieve:

Tasks	Given Deadline
Specification and Plan Submission	27/10/2023
Research Report Submission	24/11/2023
Presentation to Examiners	Starting from 15/02/2024
Research Poster Submission	23/02/2024
Final Deliverables and Report Submission	29/03/2024

Table 1: Listed Deadlines

Project Timeline Summary

Figure 2: Timeline Summary

Our project is divided roughly into 5 phases.

Research Phase - is where we explore AI functionalities and gauge the feasibility of the overall project. Having to see the potential roadblocks early on can greatly reduce future downtime and speed up designing and developing our application. A research report encompassing all the work done here must be submitted by the designated deadline.

Designing Phase - This is where the coding architecture is drafted. Potential problems when developing can be identified here early on and be addressed or prepared for when the situation arises. GUI design will also be drafted using wireframes and shown to project supervisor.

Development Phase – Backend and frontend will begin being developed. Backend development is prioritized as more attention is needed to train the LLM and prepare it for chatbot functionality. Choosing a good web framework allows for creation usable and attractive Graphical User Interface. A prototype must be ready for presentation at the start of 2024.

Testing Phase – Can be conducted in tandem with development phase where the application is tested against FURPS, a model used to evaluate the application's overall attributes and see if it performs within the project's expectations. Prototypes can also be shared with project supervisor and other cybersecurity specialists for outside user testing.

Delivery Phase – Project deliverable would be ready to submit, including application scripts and instructions on how to operate the Personal Assistant. Project report must be done by the final deadline.

Use Case Scenario

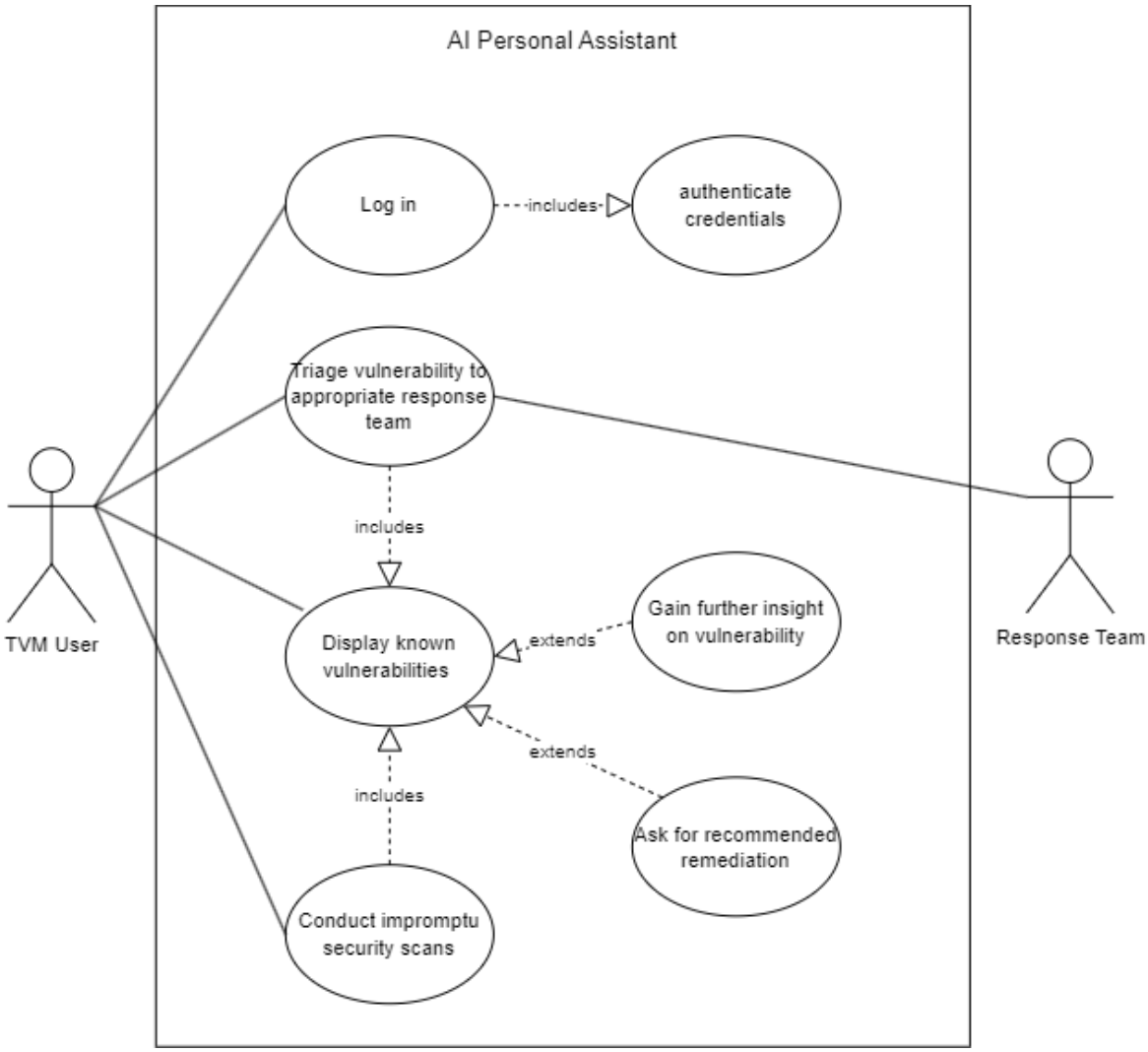

Figure 3: Use Case Diagram

Diagram Legend:

1. The rectangle represents AI Personal Assistant, the application of the project.
2. The ovals within the rectangle represent use case functions.
3. The humanoid figures represent users of the application.

Diagram 1 is a simple visualization of high-level use cases of the AI Personal Assistant. The user must authenticate before using the other functions. The application would check for records of the user's interaction and begin the session with a general idea of how the user will operate.

It is evident in the diagram that the application's primary task is to display vulnerabilities from security scanners stored in the database. Threat and Vulnerability

Management relies on security platforms to provide accurate readings, ensuring IT security of the organization.

To enhance vulnerability listing function, we implement LLMs (Large Language Models) that utilizes NLP (Natural Language Processing) to analyse vulnerability list, comparing with historical data and provide insight and targeted remediations.

Leveraging the LLM model, the user can easily issue instructions using text to the Personal Assistant. It is limited to the functions coded into the application. This is further explored in architecture breakdown section.

Such functions include conducting security scans outside of established schedules to test for further vulnerabilities on a host. The Personal Assistant should also be equipped with the functionality to triage a vulnerability to the appropriate response team through email containing relevant information such as Vulnerability ID, risk level, remediations, etc.

Application Architecture Breakdown

Fine Tuning the Language Model

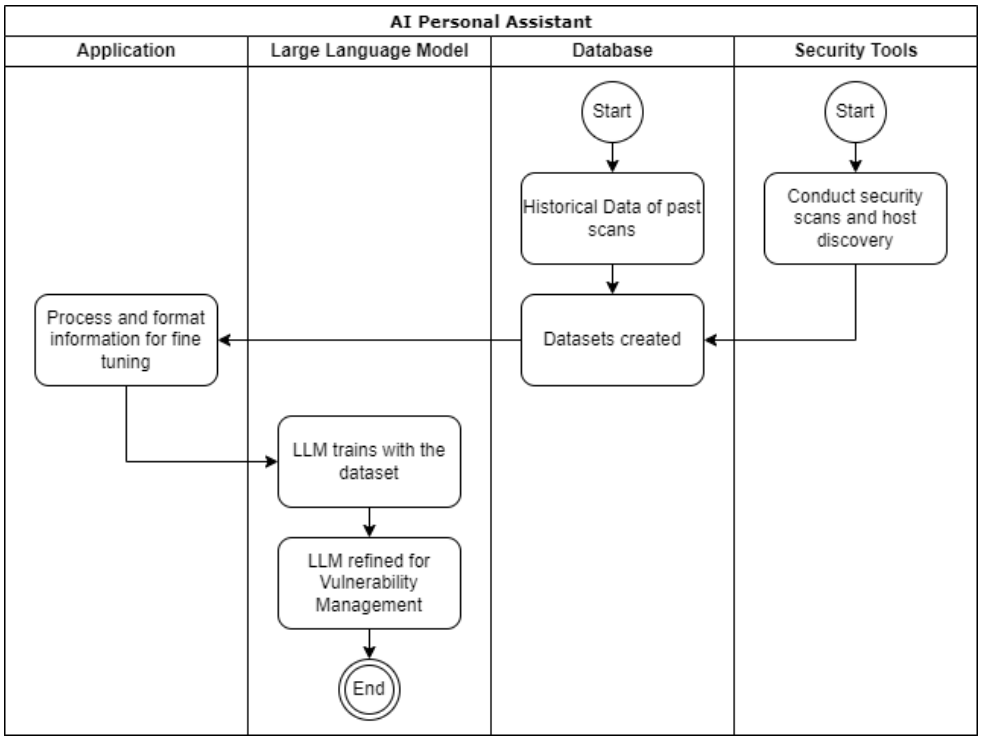

Figure 4: Fine Tuning Plan

Large language models are flexible and can be used for various tasks out of the box. These models however should be trained with datasets appropriate for its intended purpose (OpenAI, 2023).

Training with historical security data such as past scans give the LLM artificial memory and knowledge on the organization's systems. LLM output would be more consistent and avoids hallucinating false information or knowledge outside the intended database.

In the diagram above, we start by obtaining the datasets. Our application processes the dataset and uses appropriate libraries for fine-tuning. Op6enAI has functions designated for fine-tuning their LLMs.

Fine tuning can be conducted manually if the output of the LLM is dated. It is also possible to automate fine-tuning on a consistent schedule to ensure the LLM is up to date with latest information.

Scheduled vulnerability scanning

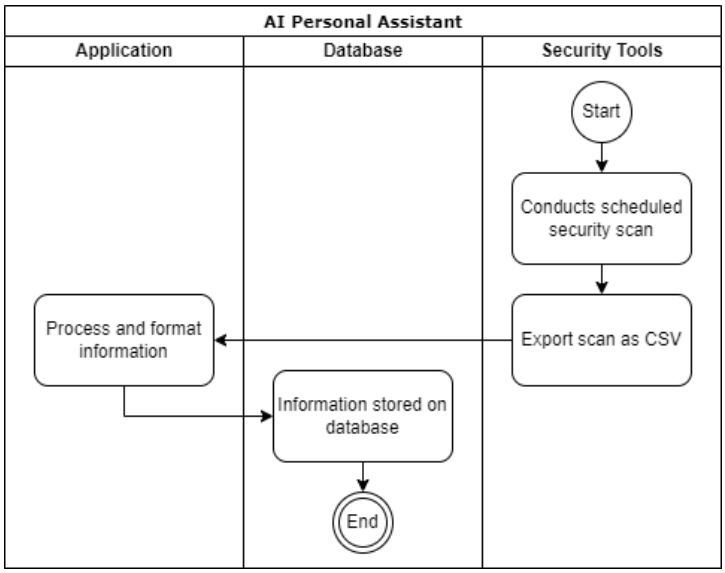

Figure 5: Automated Scan Scheduling

Diagram above is an example of scans being conducted on the organization IT environment on a designated schedule.

For the project, VMs will be used to replicate an organization computing infrastructure. A single VM acts as a host and multiple VMs form an environment in which we can conduct security scans in. VMs can be manipulated to house vulnerabilities for the security scanners to detect and report.

Example scenario would be where Nessus security scans are conducted over the weekend. Scan results are exported in various formats, including CSV, PDFs and XMLs.

Our application should be coded to receive the scan report using Nessus's API integration capability. The application can process and format the data to be stored in database. Data will also be encrypted for confidentiality.

Vulnerability Listing

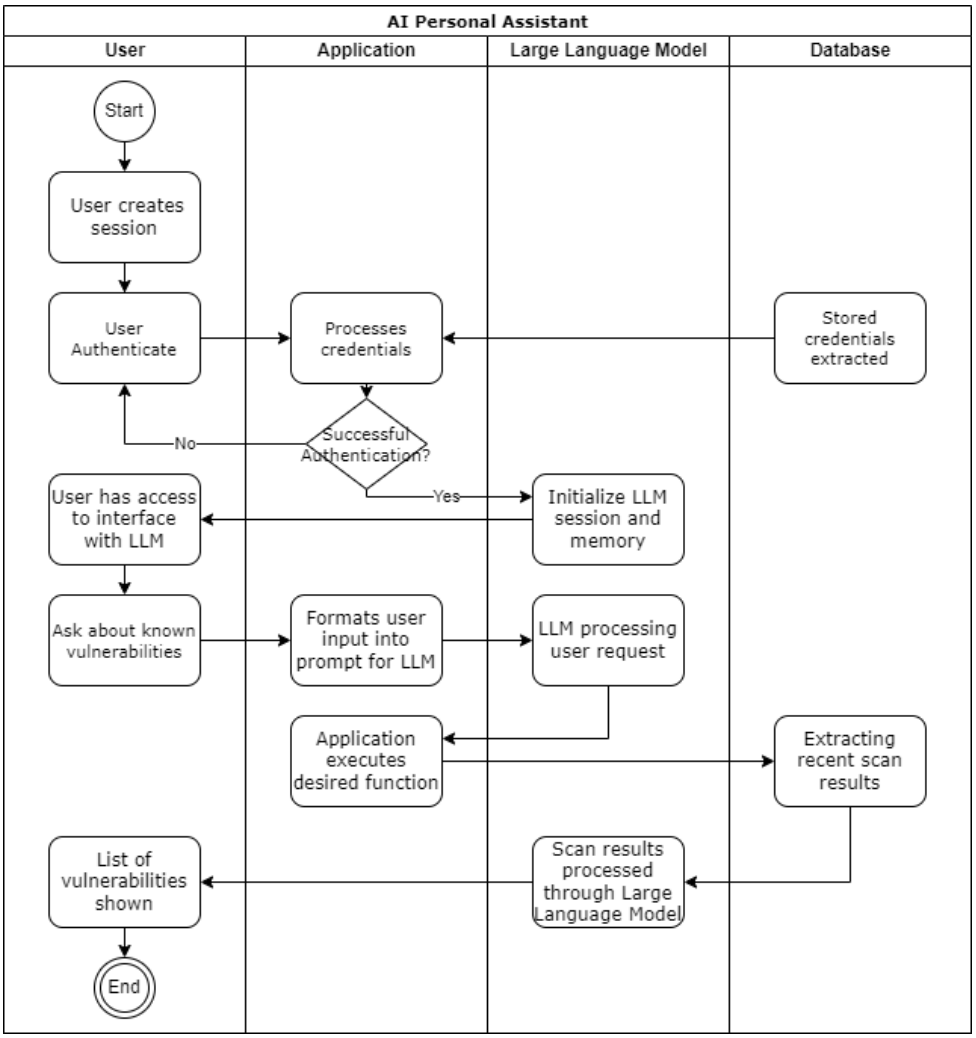

Figure 6: Processing Vulnerabilities

TVM member enters the office for the new week. They open Personal Assistant and authenticate credentials through the login and hashing functions in place within the application. Appropriate encryption protocols will be in place to ensure confidentiality of data credentials.

Once authenticated, the user will be taken to a chatbot-like GUI. They first prompt the bot about security scans conducted over the weekend by entering instructions through the text box. Application takes input, formats it into a prompt and sends it to the LLM.

We will be utilizing LangChain, an application framework will be used for this project. It will be explained in the Basic Requirements page. In brief terms it allows the LLM to select and execute our application's functions based on the user's needs (LangChain, 2023).

For the example above, the application will extract scan results for the weekend scan and sends it to the large language model to reformat into a readable format. The application will then display the LLM output it on the text interface for the user to see.

Vulnerability Insight and Response

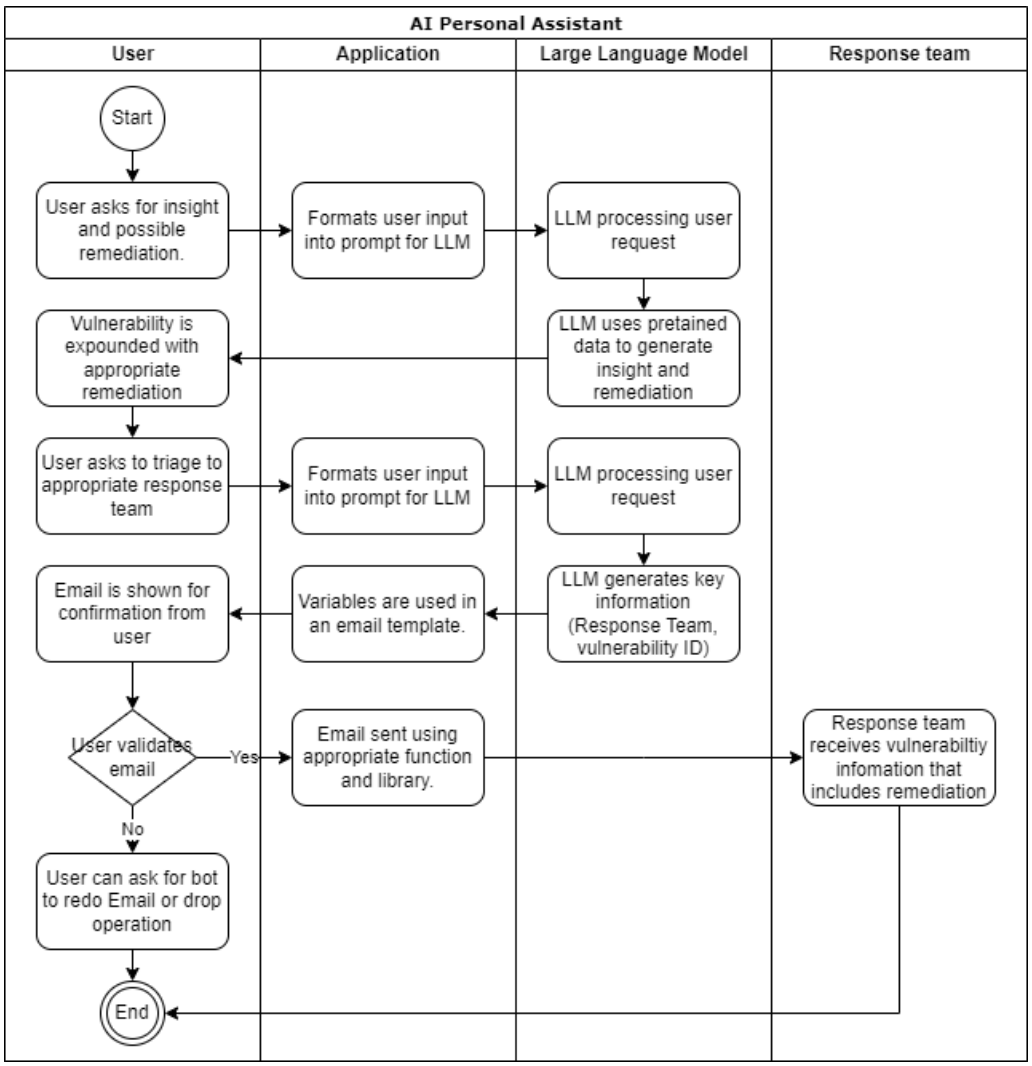

Figure 7: Vulnerability Response Process

TVM member can prompt the bot to expound on detected vulnerabilities. Personal Assistant gives insight based on previous scans on the targeted host. and generate an appropriate remediation recommendation.

The application should have the capability of triaging the incident to the appropriate response team. The application prepares a notification email based on a template containing vulnerability ID, vulnerability description and possible remediations. The LLM should be able to decide on the best response team through its reasoning capability.

Application will display the email drafted to the user. If the email is deemed valid to send, they will confirm with application and the email will be sent to the appropriate response team.

Impromptu Security Scan

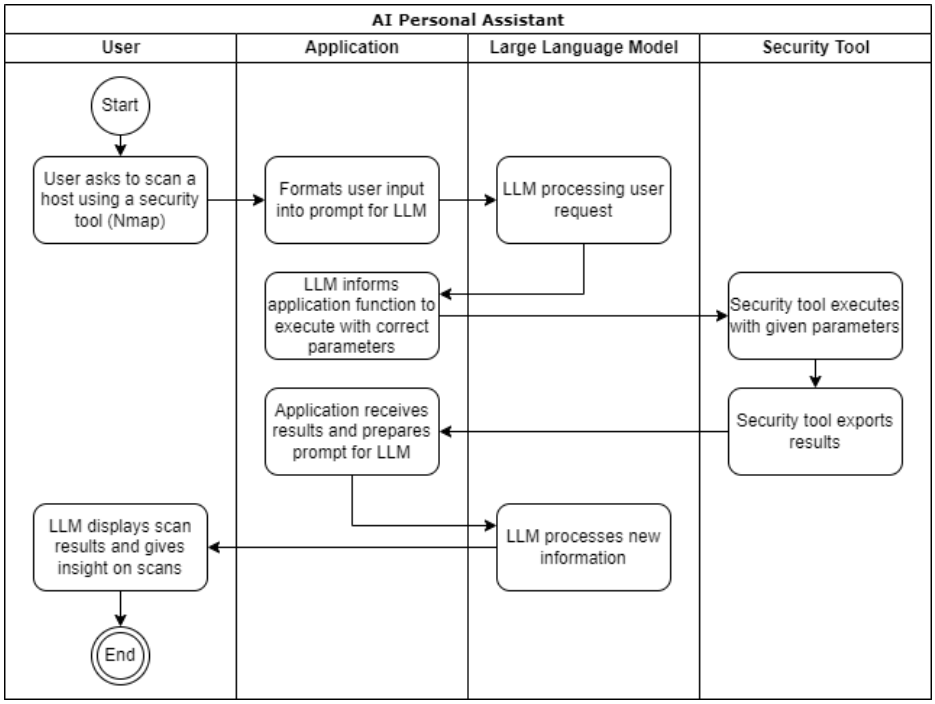

Figure 8: Self-Scan Integration

Another functionality of the application is to allow for security scans when prompted by the user. Nmap for example, through python integration the application can directly run a network scan on a host within the organization environment.

The user must provide the instruction and appropriate parameters such as host IP in the text input. LLM should be able to parse the instructions and then communicate with the application using LangChain as the bridge. Application will execute the correct function which is running an Nmap scan using given parameters.

The LLM will receive the scanning results from the application and process it. The application will take the LLM output and display it to the user.

Basic Requirements

This section contains all the requirements needed to conduct the project.

The requirements of the project shall be based on the FURPS model that is mainly used to classify software quality attributes (Gekht, 2020).

FURPS breakdown:

Attribute Name	Description
Functionality	Capability to execute tasks within project expectations
Usability	Ability for users to operate the application
Reliability	Ability to operate continuously and be resistant to complications and unforeseen circumstances (e.g., Malicious user behaviour)
Performance	Speed and quality of tasks executed
Supportability	Capable of being maintained and serviced to ensure application longevity.

Table 2: FURPS

Software Requirements

Most of the requirements are software related except for host machines which are hardware. Software requirements include:

1. Software requirements include:
2. Large Language Models
3. Programming Language
4. Code Editors
5. Web Framework Libraries
6. Virtual Machines
7. Security Scanning Software
8. Database Options
9. Supporting Applications
10. Extension Libraries
11. LangChain (Crucial Extension)

Large Language Models

The core of our application. Choosing the right Language Model ensures tasks are executed within the project's expectations.

In FURPS terms, differing language models does not affect the application's overall functionality but may affect performance such as output response of the LLM which in turn affects Usability.

Criteria	Large Language Model		
	GPT 3.5 Turbo	GPT 4	Llama 2
Developed By	OpenAI	OpenAI	Meta
Description	Capable of understanding natural language and creative, useful for mimicking human communication.	Advanced Reasoning capability. Can follow instructions better and solve complex problems. Can also process images.	Open Source LLM. Knowledge base is trained on newer data, making it up to date with more recent events than OpenAI models
Speed	Turbo version is quick at the possible cost of accuracy	Slower than GPT 3.5 but highly accurate	Dependent on host used to run the model.
Running Environment	Hosted on OpenAI platform	Hosted on OpenAI platform	Run locally
Context Size	4096 tokens	8192 – 32k	4096
Pricing Rate	$0.0015 / 1K tokens (Input) $0.002 / 1K tokens (Output)	$0.03 / 1K tokens (Input) $0.06 / 1K tokens (Output)	Free to download and use.

Table 3: LLM Comparison

Ideally GPT 4 would be the LLM to use. However, GPT 3.5 Turbo is capable enough to run our application and is more cost effective. Llama 2 is free to run but can be computationally expensive to maintain at an efficient rate (Luzniak, 2023).

High Level Programming Language

There are various programming languages that can be used to create AI applications (Aurora, 2023). Below is a table listing comparisons based on several key criteria when choosing main language to focus on.

Criteria	Programming Language			
	C++	Java	JavaScript	Python
Ease of development	Complex syntax may induce risk of human errors. Static typing may slow down coding but allows to detect errors early.3	Syntax is straightforward compared to C++ but can still be lengthy. Static typing akin to C++.	Easy to use syntax. Can be used for backend development through node.js. Dynamic Typing for quicker coding.	Easy to use and read syntax even compared to JavaScript and has dynamic typing, reducing risk of errors and promotes flexibility.
Integration with AI and LLM	Slightly limited library ecosystem making it harder to build AI applications.	Has a relatively robust library for AI development.	Contains a wide range of libraries and frameworks for AI.	Contains vast range of well documented libraries and frameworks for AI development.
Community Support and Learning Resources	Small but dedicated community, capable of providing highly specialized and excellent support for AI development.	Substantial community that provides adequate support for AI development.	Growing community but smaller than Python's and documentation may be less comprehensive.	Incredibly large community with lots of resources, tutorials for AI development.

Table 4: Programming Languages

Python as of now is the forefront programming language for AI development and machine learning and shall be used for the project. Based on the FURPS model python development would lead to great supportability due to its popularity when developing AI applications and conduct machine learning tasks and the community resources available to

view. OpenAI encourages development using Python with tutorials and resources provided by the organization.

Code Editors

Type of coding environment can greatly affect development speed and reliability.

Criteria	Code Editor		
	PyCharm	Visual Studio Code	JupyterLab
Language Support	Python-focused	Multi-language focused	Python-focused
Extensions	Substantial collection of plugins and extensions.	Very large collection of plugins and extensions	Limited extensions
Interface	User friendly and visible interface and is customizable.	User friendly and visible interface and is customizable.	Web-based interface may be limiting.
Web Development	Excellent support for web frameworks such as Django and Flask.	Multi-language support allows for web development languages. (HTML, CSS, JavaScript)	Contains built-in support for web development such as HTML, CSS and JavaScript).
Debugging	Wide range debugging tools suited for Python.	Basic debugging tools.	Limited debugging capability.

Table 5: IDE comparison

PyCharm shall be the main IDE and coding environment for the project as Django and Flask support is excellent for building the Personal Assistant.

Web Framework Libraries

Web frameworks are a set of modules used for assisting in writing web application code. Technically it is not necessary but highly recommended as it can increase user usability and performance of the application. There are various options. For this project, I have narrowed it down to three.

Criteria	Web Framework		
	Django	Flask	Streamlit
Purpose	Full-stack Python framework for complex, large-scale web applications	Lightweight framework for making smaller apps and prototyping larger ones.	Easy to use web framework for prototyping and deploying web applications based on generative AI.
Database Support	Django supports the most popular relational database management systems like MySQL, Oracle etc.	No default model means it can support multiple database types and have more control. Uses SQLAlchemy for database requirements.	Like Flask, no built-in database support requires external library such as SQLAlchemy.
Flexibility	Less flexible due to built-in features and tools. Developers cannot make changes to the modules.	Extensive libraries supported allowing for flexibility when developing.	Somewhat limited flexibility but has features suited for AI application building.
AI integration	Not much support for AI-specific features.	Not much support for AI-specific features.	Extensive integration with LLMs such as GPT and Llama.
Community Support	Large community capable of providing resources for web app building.	Medium sized community tailored for beginner developers and startups.	Community is rapidly growing with resources available for AI app development.

Table 6: Python Web Frameworks

Streamlit seems to be the best fit based on FURPS criteria. It offers reliability and supportability due to it being developed solely for data applications including generative AI, a new form of content that has revolutionized workflow for various industries (Kelly & Treuille , 2023). LangChain libraries also help alleviate database implementations. Streamlit also excels at testing LLM output due to its simplicity and light overhead.

Virtual Machines

Virtual machines will be used to set up a test scanning environment. Each VM will represent a target host to be scanned by security tools such as Nessus and Nmap.

Criteria	VM Tool		
	VirtualBox	VMware Workstation Player	Hyper-V
Operating Systems	Multiple OS options, including older versions	Multiple OS options	Windows only
Hypervisor Type	Type 2 Hypervisor - runs on the operating system installed on a host.	Type 2 Hypervisor – runs on the operating system installed on a host.	Type 1 Hypervisor - that runs directly on a computer's hardware.
Performance	Slow to medium speed	Mid to high speed.	High performing.
Ease of use	Very easy to set up and utilize VMs and functionalities.	Easy to set up VMs.	Not too difficult to set up.

Table 7: Virtual Machine Hosts

According to FURPS criteria, all VM tools are capable of simulating organization IT environment. VMWare Workstation would be the preferred VM application as it has good performance levels and decent supportability. Hyper-V however will be further explored if the need for faster performance from a Type 1 Hypervisor VM arises (Manikandan, 2023).

Security Scanning Software

The main way to create a dataset is to create a test environment to gather information. Dataset will be used as knowledge base to fine tune the bot.

Criteria	Security Tool			
	Nessus Scanner	Network Mapper (Nmap)	Burp Suite	Zap
Descriptions	Security tool allowing to scan the network environment and manage vulnerabilities.	Open-source network discovery and security auditing tool.	Flexible security tool for testing web applications	Open-source security tool to scan and fix vulnerabilities of web applications
Key functions	Automated scans. Can generate detailed reports with remediations.	Focused on network scanning with various techniques.	Extensive range of tools for professional web application testing.	Flexible tool with scripting capabilities for automating scans.
API-integration	Yes	Yes	Yes	Yes
Pricing	Has free tier (Nessus Essentials) but limited features	Free to use	Has free tier (Burp Suite Community) but limited features	Free to use

Table 8: Scanning Software

Nessus Essentials is the preferred security scanner as in terms of FURPS it provides excellent functionality features, capable of conducting holistic scans of targeted hosts, identifying a wider range of vulnerabilities and has result export feature.

Database Options

We need to store our security reports and scans in a database. This will also act as the location for the knowledge base. We can choose several ways to host our database.

Criteria	Database Hosts			
	XAMPP-MariaDB	ChromaDB	MongoDB	Pinecone
Functionality	Easy to use SQL database application for local hosts	Experimental vector database that can be hosted locally or on cloud.	Widely used cloud database service that allows SQL and vector versions. with AI integration feature.	Cloud based vector database tailored for building applications for AI and fine-tuning Large Language Models.
Pricing	Free to use	Free to use	$70/month (paid tier)	$57/month (paid tier)
Ease of use	Easy to set up with tutorials available online.	Available resources online and syntax is easy to utilize	Vast resources available due to popularity and community support.	Resources available on website to assist with building.
Application Integration	Libraries available to integrate database with application	Vector database makes it suited for AI applications	Can integrate with application and has AI support.	Can integrate with application and encouraged for AI apps.

Table 9: Database Types

Due to the experimental nature of the project, we will use ChromaDB as main database to store our scan results, historical data, user credentials and other vital data for vulnerability management. Data must be encrypted through proper hashing functions to maintain confidentiality and increase reliability for users. Local databases also ensure performance as it is locally hosted and no need for internet connections to cloud services.

Supporting Applications

I will be actively using GitHub that should be integrated with my preferred IDE or coding environment to keep track of my scripts, application drafts and prototypes (Coursera, 2023).

Software Name	Description
Git	Version control system for developers to record changes made on their script and files.
GitHub	Web-based Git repository for storing all scripts and drafts of our applications.

Table 10: Git Descriptions

GitHub repository also allows for viewing millions of other repositories of other users that are publicly made, allowing developers to take inspiration and expedite their own projects, akin to this one.

Extension Libraries

As Python is the preferred programming language for this project, there are several key libraries to install. Pip is used to install most of the libraries mentioned.

Python Libraries	Description
Transformer	Library created by Hugging Face to install and train external LLMs for that website.
OpenAI	OpenAI's library to use its various plugins for AI application building.
LangChain	Application framework primarily used for AI projects. Offers various plugins that enhance the application's functionalities.
Tiktoken	Tokenizer used to manage and calculate tokens for AI models to process. Tokens are lengths of characters which the LLM use as memory currency.
MySQL	Needed to connect application to external database server and use SQL queries.

Table 11: Python Libraries

LangChain

This extension library calls for its own section. LangChain is an application framework that acts as the primary link between the user and LLM with the application functions (LangChain, 2023). LangChain has vital modules which enhances the application's functionality such as database integration, document crawling, autonomous function execution, etc.

Module Name	Description
Model Input/Output	Ability for user to interface with LLM and parse information to be used for application functions
Retrieval	Allows for fetching data from data sources such as database or public websites.
Chains	Way to utilize LLM outputs into single functions and connects them together to carry out complex functions
Agents	Reasoning module that helps the LLM to decide which function on the application to use.
Memory	Ability to store information on previous tasks and maintain a session.
Callbacks	Module that allows user to monitor and log tasks being carried out in the application through 'CallbackHandlers' objects.

Table 12: LangChain Breakdown

Using the FURPS model to evaluate LangChain's importance, it gives the application the functionality of a human-like security agent. Users can tell the AI Personal Assistant through text input to conduct a specific task. With the help of agents and model I/O, the LLM can understand the user and execute the desired operation such as listing vulnerabilities of a host machine.

These ready-made modules also improve application performance by instilling code structure and avoid writing complex and suboptimal configurations that may slow down function execution.

LangChain being a popular framework contributes to the supportability of the project. There are consistent updates to the framework and various resources and tutorials available online to assist with operating the application.

Hardware Requirements

A private-owned physical machine is recommended as it can centralize application development and store vital data.

Machine name	Specifications	Description
Student computer	System RAM: 15.2 GB Disk Size: 930 GB CPU: AMD Ryzen 7 6800HS	Physical machine capable of hosting coding environments, VMs for scanning simulations and even Large Language Models if needed.
Google Collab	System RAM: 12.7GB Disk Size: 107.7 GB CPU: Tesla K80 GPU	Cloud-based computing resource available for experimenting scripts and hosting open-source LLMs.

Table 13: Hardware Requirements

Student computer is the default option to use for creating the application. If the need arises Google Collab service can be used if working remotely from student computer or needing to host open source LLMs such as Llama 2 through cloud without leveraging local computing power.

References

1. Aurora, S. (2023) Best Programming Language for AI Development in 2023 [online] Available at: https://hackr.io/blog/best-language-for-ai [Accessed 15 Oct. 2023].
2. Cousera (2023) What Is GitHub and Why Should You Use It? [online] Available at: www.coursera.org/articles/what-is-git [Accessed 20 Oct. 2023].
3. Gekht, N. (2020) Create Better Backlog and Engage the Development Team with FURPS. [online] Available at: https://gehtsoftusa.com/blog/create-better-backlog-and-engage-the-development-team-with-furps/ [Accessed 20 Oct. 2023].
4. Kelly, A., Treuille A. (2023) Generative AI and Streamlit: A perfect match. [online] Available at: blog.streamlit.io/generative-ai-and-streamlit-a-perfect-match/ [Accessed 16 Oct. 2023].
5. LangChain (2023) Introduction [online] Available at: python.langchain.com/docs/get_started/introduction [Accessed 14 Oct. 2023].
6. Luzniak, K. (2023) Is Llama 2 Better Than GPT Models? 6 Main Differences Between Llama 2 vs. GPT-4 vs. GPT-3.5 [online] Available at: https://neoteric.eu/blog/6-main-differences-between-llama2-gpt35-and-gpt4/ [Accessed 15 Oct. 2023].
7. Manikandan, S. (2023) Client Hyper-V vs VirtualBox. [online] Available at: www.bdrsuite.com/blog/client-hyper-v-vs-virtualbox/ [Accessed 22 Oct. 2023].
8. OpenAI (2023) Fine Tuning [online] Available at: platform.openai.com/docs/guides/fine-tuning [Accessed 18 Oct. 2023].
9. Raf (2023) What are tokens and how to count them? [online] Available at: help.openai.com/en/articles/4936856-what-are-tokens-and-how-to-count-them [Accessed 18 Oct. 2023].
10. Uzialko, A. (2023) How Artificial Intelligence Will Transform Businesses [online] Available at: www.businessnewsdaily.com/9402-artificial-intelligence-business-trends.html [Accessed 10 Oct. 2023].

Appendix

TABLES

FIGURES