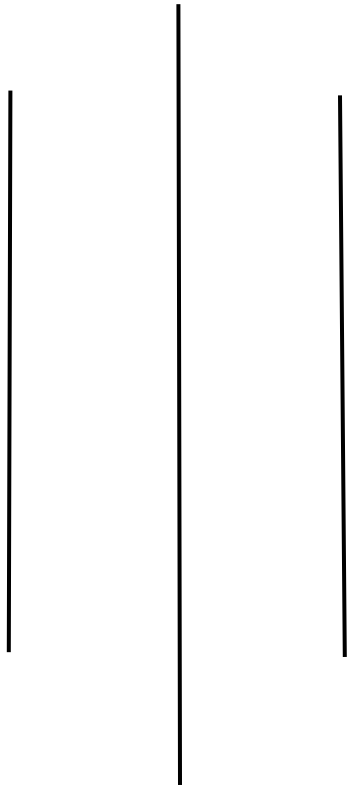
VULNERABILITY SCANNER

VULNERABILITY VANGUARDS (TEAM B2)



Team members:

Akshay (500218533) (akshay3@loyalistcollege.com)

Gagandeep Sheemar (500220112) (gagandeepsheemar@loyalistcollege.com)

Monarch Giri (500217684) (monarchgiri@loyalistcollege.com)

Prateek Chaudhary (500221065) (prateekchaudhary@loyalistcollege.com)

Ramandeep Kaur (500218644) (ramandeepkaur99@loyalistcollege.com)

Shubham Arora (500221607) (shubhamarora@loyalistcollege.com)

Abstract

Technology and innovations are growing, and new advancements are being made. Along with technological advancements, we are also prone to attacks, and our safety should be our priority. People and organizations face many issues related to vulnerable systems that malicious actors can exploit. Vulnerabilities are lurking in the systems and it should be removed promptly to safeguard the system and ourselves from being a victim of cyberattack. To address these issues, the Vulnerability Vanguard project has created a sophisticated vulnerability scanner program that is intended to quickly find and report system vulnerabilities. For security experts, the tool's contemporary, user-friendly GUI, created with PyQt6, makes it accessible and simple to use. It makes use of potent scanning technologies, such as Nmap, to carry out exhaustive network system scans and spot possible security threats. The application also has an email notification system to swiftly notify users of scan findings and a scheduling function that uses crontab to enable automated scans at predetermined intervals.

Making an interface that is easy to use and improves navigation was our top priority. The PyQt6 framework was selected because it can be used to create slick, contemporary graphical user interfaces. Nmap's robustness in network scanning led to its integration. As a result, the tool can carry out a variety of scans, including port scanning, service discovery, and vulnerability detection, giving users a thorough picture of the security posture of the network. The tool creates comprehensive reports outlining the results of the scans in a variety of formats (such as PDF and XML). These reports contain information on the seriousness of vulnerabilities as well as doable remedy suggestions.

The Vulnerability Vanguard project offers a comprehensive solution that improves the efficacy and efficiency of vulnerability management by merging various technologies. By enabling businesses to proactively detect and address security threats, it eventually fortifies their cybersecurity defenses.

Table of Contents

Table of Figures

1. Introduction

In a world where technology is an integral part of our daily lives, everyone owns a gadget. We can see everything and get information on every topic in seconds. We frequently hear stories like "This person was defrauded and fell victim to identity theft," "Hackers are on the rise," "Our data is being exposed online," "We are being observed," and so on. But what does this mean? Should we be worried about this? If yes, why? And what do we do about it?

The term "cybersecurity" is very broad and can mean different things to different people and organizations. Consider it as locking your house when you leave for work, which entails safeguarding your belongings inside when you are not home. So, cybersecurity simply refers to safeguarding your identity, private information, and critical data while you are not around. It is the process of preventing malicious actors (Hackers) from having any unauthorized access or damaging our computers, smartphones, and other devices, as well as the data stored on them. Cybersecurity seeks to defend against computer viruses, complex and expensive ransomware attacks, and other threats to people's and organizations' systems, apps, computing devices, sensitive data, and financial assets (synopsys, 2024).

Figure 1 CIA Triad (Walkowski, What Is the CIA Triad, 2019)

What is CIA, you might be wondering. Let's take a close look at that as well.

Confidentiality – Confidentiality ensures that your information is only available to you and only to those who are authorized to view it.

Integrity – Integrity ensures that your information is accurate and unchanged unless altered by you or by someone authorized to do so.

Availability – Availability ensures that you have access to your information and resources when you need it or some other authorized personnel (Walkowski, What Is the CIA Triad?, 2019).

Consider this scenario: you have a password-protected phone (which I strongly believe you have), and you want your personal information to be available only to you and not to others. This ensures the confidentiality of the information. There will be some very important documentation on your phone that you can and should only alter, change, or remove; this is the CIA's integrity factor. Finally, availability means that you should be able to access your personal information without interruption from any external factors. This principle of cybersecurity is the one that any user or organization would rely upon.

Businesses can be disrupted, damaged, or destroyed by cyberattacks, and the costs to the victims are always going increasing. For instance, the Cost of a Data Breach 2023 report from IBM states that

- 2023 saw an average cost of USD 4.45 million for data breaches, a 15% increase over the previous three years (IBM, 2023).
- In 2023, the average cost of a ransomware-related data breach was $5.13 million USD. The ransom payment cost, which averaged an additional USD 1,542,333 and increased 89% from the previous year, is not included in this figure (IBM, 2023).

It should come as no surprise that a recent survey discovered that there was a 3.4 million worker deficit in cybersecurity globally—that is, the difference between the number of cybersecurity professionals in the field and the number of open positions. In order to combat cyber threats more successfully and lessen the impact of cyberattacks, security teams with limited resources are

concentrating on creating comprehensive cybersecurity plans that make use of automation, artificial intelligence, and sophisticated analytics (IBM, 2023).

1.1 Our Team

The success of the Vulnerability Vanguards project demonstrates our dedicated team's collective effort and skill. Each person provided their own set of talents and viewpoints, which helped to shape and polish the tool. Here's a summary of our team members' roles and the processes they used to attain their goals.

Team Members and Contributions

1. Monarch Giri - Project Leader

- Role: Monarch led the project, ensuring timely delivery of milestones and coordinating tasks across the team.
- Process:
    - Developed a detailed project plan and timeline.
    - Conducted regular team meetings to track progress and address any challenges.
    - Facilitated communication between team members and stakeholders.

2. Akshay - Lead Developer

- Role: Akshay was responsible for the core development of the vulnerability scanning tool, including the integration with Nmap and the creation of the main scanning logic.
- Process:
    - Analyzed user requirements and designed the software architecture.
    - Implemented the core functionalities using Python and integrated Nmap for network scanning.
    - Conducted rigorous testing to ensure the accuracy and reliability of scan results.

3. Gagandeep Sheemar - Documentation Specialist

- Role: Gagandeep created user documentation and provided support to users.
- Process:
    - Authored detailed user guides and manuals, ensuring they were clear and easy to understand.
    - Developed FAQ sections and troubleshooting guides to assist users in resolving common issues.
    - Provided direct support to beta testers, gathering feedback and suggesting improvements.

4. Prateek Chaudhary - Research Coordinator

- Role: Prateek provided expertise in cybersecurity research, ensuring the tool's scanning capabilities were comprehensive and effective in identifying vulnerabilities.
- Process:
    - Conducted a thorough analysis of common network vulnerabilities and threats.
    - Collaborated with the development team to incorporate advanced scanning techniques.
    - Validated the effectiveness of the tool by running multiple test scans and refining detection algorithms.

5. Ramandeep Kaur - Testing and QA

- Role: Ramandeep was in charge of testing the tool to identify and resolve any bugs or issues, ensuring a high-quality end product.
- Process:
    - Developed comprehensive test plans and cases covering all aspects of the tool’s functionality.
    - Executed manual and automated tests to identify defects and performance bottlenecks.
    - Worked closely with the development team to address issues and verify fixes.

6. Shubham Arora - User Interface Designer

- Role: Shubham designed the user interface, focusing on creating an intuitive and visually appealing experience for the users.
- Process:
    - Researched and identified best practices in UI/UX design for cybersecurity tools.
    - Created wireframes and mockups based on user feedback and project requirements.
    - Collaborated with the development team to implement the design using PyQt6.

Figure 2 Our team members

1.2 Problem Statement

Before diving straight into the problem statements, let's look at what a vulnerability is and why it should be our concern. After that, we can discuss the problem caused by the vulnerability and the reason for developing this project in the first place.

What is a vulnerability?

A vulnerability is a fault or weakness in a system, application, or network that can be used by a threat actor to carry out unauthorized operations. Vulnerabilities can come from a variety of causes, including software defects, misconfigurations, and even human mistakes. They may be found in operating systems, applications, network protocols, or hardware components. When left untreated, vulnerabilities can be exploited to obtain unauthorized access, interrupt services, steal sensitive data, or run malicious code (Tunggal, 2024).

Why should vulnerabilities be a concern?

Vulnerabilities are a major concern in cybersecurity for various reasons:

- Security Breach: Exploiting vulnerabilities can result in security breaches, in which attackers gain unauthorized access to systems and data.
- Data theft can result in substantial financial and reputational damage.
- Service Disruption: Exploits can trigger denial-of-service attacks, making systems and apps inaccessible, which can be very detrimental for enterprises that rely on consistent uptime.
- Malware and Ransomware: Vulnerabilities can be exploited to spread malware or ransomware, jeopardizing data integrity and availability and demanding ransom payments for restoration.
- Compliance Violations: Organizations are frequently expected to follow industry standards and laws. Unaddressed vulnerabilities can lead to noncompliance, legal penalties, and a loss of confidence.

Vulnerability is in fact a real problem that everyone can face and based on that, our entire project is based – on finding and helping in the removal of that vulnerability.

In today's cybersecurity landscape, the number and sophistication of threats are constantly increasing. Attackers routinely exploit vulnerabilities in systems, applications, and networks to gain unauthorized access, steal sensitive data, disrupt services, or distribute harmful malware. Despite awareness, many businesses struggle to discover and address these vulnerabilities quickly, leaving their systems vulnerable to prospective assaults. Regular vulnerability assessments are required to identify and mitigate threats in a timely way. Manual processes, on the other hand, are labor-intensive and subject to errors, resulting in unpredictable scanning schedules and gaps in security coverage. Automated and scheduled vulnerability scanning is critical for ensuring ongoing security and proactive threat management. Identifying vulnerabilities is merely the first step in ensuring system security. Security teams require precise and actionable reports that provide insights into the type and severity of vulnerabilities, as well as clear suggestions for remediation. Without detailed reports, companies may struggle to prioritize response activities and efficiently address the most pressing security threats.

Our vulnerability scanner, Vulnerability Vanguard, is designed to address these critical issues by providing a robust, automated, and user-friendly solution for continuous vulnerability assessment and management.

1.3 Technology and software used

- Nmap
- Python
- PyQT framework
- SQL lite 3 Database
- Visual studio

2. Literature Review

In the article titled " How To Use Nmap for Vulnerability Scanning: Complete Tutorial" written by Chad Kime. It covers Nmap installation, basic and advanced commands, and how to expand capabilities with built-in and custom scripts. The article discusses how to run several sorts of scans, such as port scanning and particular vulnerability identification, and emphasizes the significance of safeguarding scan data. It also analyzes the advantages and disadvantages of Nmap and compares it to other vulnerability scanning tools (Kime, 2023).

In the article titled "Vulnerability Scanning: What It Is & Why It's Important for Security and Compliance" written by Celine Pham. The article from SecureFrame explains vulnerability scanning, a critical security practice that employs automated technologies to detect flaws in computer systems, networks, and applications. This procedure is critical for proactive defense against cyber attacks and regulatory compliance. It distinguishes between vulnerability scanning, penetration testing, and vulnerability management, demonstrating that, while similar, each has unique goals and approaches. Furthermore, vulnerability scanning provides early detection of vulnerabilities, efficient risk management, compliance adherence, cost savings, and an improved security posture (Pham, 2023).

In the article titled " Nessus" written by Rahul Awati. Tenable's Nessus platform is well-known for its strong vulnerability scanning capabilities. It scans networks, systems, and apps for security flaws and provides detailed reports to assist enterprises to manage and minimize risks. Nessus provides a variety of compliance checks and may simulate assaults to help evaluate defenses. Its high configurability and extensive support for several settings make it appropriate for a wide range of IT infrastructures. The platform is notably notable for its constantly updated plugin library, which plays an important role in detecting the most recent vulnerabilities. Nessus is intended to scan for security vulnerabilities in a variety of network resources and includes features such as predicted prioritizing and live findings for effective security management. The platform is well-known for its huge plugin library, which helps find and resolve issues quickly (Awati, 2023).

In the article titled "What is TOTP and why do you need it?" written by Joel Coutinho. The essay digs deeper into how TOTP improves user security over static passwords by adopting a time-sensitive algorithm. Each TOTP is only valid for a limited time, typically 30 seconds, decreasing the window of opportunity for abuse if intercepted. This approach requires that both the server and the user's device know the secret key, which synchronizes time to ensure correct code generation and verification. Despite its reliance on safe secret key storage, TOTP is still a popular solution for securing user logins, particularly for services that require additional security precautions.

In the article titled "What Makes Good UI Design?" written by Fuzzy Math. The Fuzzy Math article explains the essential concepts of excellent UI design, emphasizing the significance of understanding user demands and context. Maintaining consistent navigation, visual components, and interactions across the interface is critical for improving usability. It emphasizes the need of a simple, intuitive design that reduces user confusion while increasing efficacy. The essay also emphasizes the importance of responsive communication via system status updates and thoughtful error prevention to provide a positive user experience (Math, 2024).

In the article titled "Writing a Vulnerability Scanner using Python" written by Mohamed Ezzat. Mohamed Ezzat's blog post describes how to develop a vulnerability scanner in Python. It entails transforming a previously created port scanner script into a class that detects open ports and potential vulnerabilities. The vulnerability scanner checks the found services to a list of known vulnerabilities. The procedure is outlined step by step, beginning with the import of the port scanner class, prompting user input for target specifications, and reviewing banners to identify any security issues. The scanner's usefulness is illustrated through a realistic example in a controlled virtual environment (Ezzat, 2020).

3. Methods

Here in this section, we are going to discuss the technologies used in detail, the methods used to gather any information for the project, and the methodology used for this project.

3.1 Technologies used

Technologies are one of the key elements required for this project; without them, building a good scanner is impossible. The following are the technologies that we are going to use to complete this project:

- Python – Python is a high-level, interpreted programming language that is well-liked by both novice and seasoned programmers due to its readable syntax and easy-to-understand structure. Guido van Rossum was the creator, and it was originally published in 1991. Python's design philosophy, which makes extensive use of whitespace, prioritizes code readability. There is less need for additional code because Python has a sizable standard library that contains modules and packages for a variety of tasks. Python is used in many different programming languages and technological fields, including artificial intelligence, scientific computing, data analysis, web and software development, and more. In our project, the majority of coding is done in Python due to its ease of development, extensive libraries, and rapid prototyping capabilities (Pawlan, 2024).
- Nmap – An effective and flexible open-source tool for network exploration and security auditing is Nmap (Network Mapper). Although it was initially intended to scan large networks, it is effective when used against single hosts. Nmap makes creative use of raw IP packets to identify hosts on a network, the services (name and version of the application) they provide, the operating systems (and OS versions) they run, the kinds of firewalls and packet filters they have in place, and a plethora of other features. At the backend of our project, Nmap plays a crucial role when it comes to host and network scanning. It helps to perform a detailed analysis of the host to identify open services and ports for potential vulnerabilities. The use of the Nmap Scanning Engine (NSE) is highly efficient when it comes to running scripts and helps in identifying potential vulnerabilities. The use of Nmap helps in developing robust, efficient, and comprehensive vulnerability scanners (occupytheweb, 2014).

- Database – A computer system's organized collection of data that can be electronically accessed and stored is called a database. Because they make data management, retrieval, storage, and manipulation efficient, databases are essential in a wide range of applications. They facilitate structured data storage and are made to handle massive amounts of information. In our project, the database plays an important role as we have downloaded 11 and stored databases of known vulnerabilities that we use for the scanner, which will compare the results of the scanned system to the database of the known vulnerabilities to identify the vulnerabilities present in our scanned system. Some of them are cve.csv, exploitdb.csv, openvas.csv (Oracle, 2020).

- Visual Studio – Microsoft offers an integrated development environment (IDE) called Visual Studio. Computer programs, websites, web apps, web services, and mobile apps are all developed with it. Numerous programming languages are supported by Visual Studio, such as C#, Visual Basic.NET, C++, F#, JavaScript, Python, and more. With the help of the IDE, developers can write code, test it, identify errors, and launch their apps all from within a single application. For our project, all the coding and editing are done in Visual Studio (Awan, 2023).

3.2 Methodology

Within the framework of a project, a methodology denotes an organized strategy that directs the project's conception, implementation, and conclusion. It is made up of predetermined procedures, tasks, deliverables, and practices that aid in the effective and efficient management of the project. Project management methodologies are vital because they offer a structured approach to project management, guaranteeing repeatability, consistency, and predictability.

3.2.1 Selected Methodology

To do our project we have selected spiral methodology. Combining aspects of traditional Waterfall and iterative development processes, the Spiral Model is a flexible and risk-driven software development methodology. It has a spiral structure with several loops, each of which represents a different stage of the development process. The Spiral Model's salient characteristics encompass its focus on risk mitigation, its iterative structure that facilitates the integration of user feedback and flexibility in response to modifications, and its appropriateness for extensive and intricate undertakings with unpredictable requirements and elevated risks. Although the model is resource-5 intensive and needs a highly qualified team, it works incredibly well for handling difficult software development projects. As our project requires constant changes and user feedback, the spiral methodology is a good way to go (Singh, 2024).

3.3 Methods for information gathering

1. Surveys and questionnaires:

- Methodology: We created structured surveys and questionnaires to get quantitative data from a large number of consumers.
- Reason for Choosing: This strategy was chosen due to its efficiency in acquiring large amounts of data quickly. It allowed me to quantify user preferences, actions, and opinions, laying a strong statistical foundation for the project. Surveys provided for anonymity, which encouraged honest responses and improved data reliability.

2. Interviews:

- Method: Semi-structured interviews were conducted with selected individuals to collect qualitative data.
- Reason for Choosing: Interviews were picked to acquire a better understanding of user experiences and motives. Unlike surveys, interviews allowed for follow-up inquiries, resulting in a more nuanced grasp of user viewpoints. This approach was critical for investigating difficult subjects that necessitated thorough explanations and personal stories.

3. Observations:

- Method: We observed people in their natural setting as they interacted with the system or product.
- Reason for Choosing: Observations were essential for capturing real-time behaviors and interactions without the biases that can arise from self-reported data. This strategy supplied contextual information about how users interact with the system in everyday situations, emphasizing practical usability issues and possibilities for development.

4. Document Analysis:

- Method: I reviewed existing records, reports, and documents related to the project.
- Reason for choosing: Document analysis was used to collect historical and contextual information. It offered background information and assisted in identifying trends throughout time, allowing for a more comprehensive grasp of the project's context. This solution was both cost-effective and time-efficient, as it made use of existing data.

5. Web Scraping:

- Method: Automated data acquisition from relevant websites using web scraping tools.
- Reason for choosing: Web scraping was used to collect massive datasets from online sources quickly. This strategy proved very beneficial for market analysis, competitive benchmarking, and sentiment analysis. It delivered up-to-date information and insights from a variety of sources that would be difficult to obtain manually.

6. Sensor Data Collection:

- Method: Sensors were used to capture real-time data on certain variables such as environmental conditions or user behaviors.
- Reason for choosing: This method was chosen because of its accuracy and dependability in collecting continuous data. It was vital for initiatives that required long-term monitoring, such as environmental studies or health research. Sensors generated objective data that could be studied to detect patterns and correlations.

Each of these strategies was chosen to enable extensive data collection while meeting the project's different requirements. By integrating quantitative and qualitative methodologies, I hoped to gain a balanced and comprehensive knowledge of the study objectives. This multi-method strategy increased the robustness and validity of the findings, allowing for more informed conclusions and suggestions.

4. Findings

Here we discussed all the findings and our progress throughout the project, findings from our research, and technology advancements.

4.1 Introduction to our product

A vulnerability is a flaw in a system that can be exploited by cyber attackers to gain unauthorized access or damage the system. Vulnerabilities may exist in software, hardware, network configurations, or procedures used within an organization. Understanding vulnerabilities is critical because they represent potential entry points for threats.

Vulnerability scanner: A vulnerability scanner is a tool for detecting and diagnosing vulnerabilities in a network or software system. It automates vulnerability detection and management, which entails scanning systems, networks, or applications for security flaws and weaknesses. Here's how vulnerability scanners typically work.

- Scanning involves a systematic examination of the target system, network, or application. It scans a database of known vulnerabilities and signatures to look for flaws such as outdated software versions, misconfigurations, and missing patches.
- Analysis: After the scan is completed, the scanner examines the data to determine actual vulnerabilities. This analysis can also consider the context in which potential vulnerabilities could be exploited.
- Reporting: The scan results are compiled into a report that details the discovered vulnerabilities. The report typically ranks vulnerabilities by severity, providing critical information to help prioritize remediation efforts.
- Remediation Guidance: Most scanners provide instructions on how to address each identified vulnerability. This could include patches, configuration changes, or other mitigation strategies.

Architectural Design

In project management, especially in software development, architectural design refers to the system's high-level structure. It includes how the hardware and software components are arranged, how they work together, and the guiding ideas that have shaped its creation. The project's and the system's blueprint are both provided by architectural design. It covers the primary elements, their functions, and how they work together. It entails making important choices regarding the coding standards, platforms, tools, frameworks, and technical standards. The entire development process is shaped by these choices. Architectural design tackles significant issues that could have an impact on the project's success early in the lifecycle. It offers solutions for problems with performance, security, scalability, and dependability (Shilpa, 2024).

Architectural design is one of the most important elements that should be done before starting to develop any software or any products. Features like security, dependability, and performance are improved by good architectural design. It guarantees that the system can manage the specified requirements and grow to meet additional demands. Because the architecture is well-designed, maintenance is made easier because the system's structure is logical and clear. It also facilitates the addition of new features and system updates without requiring significant redesigns.

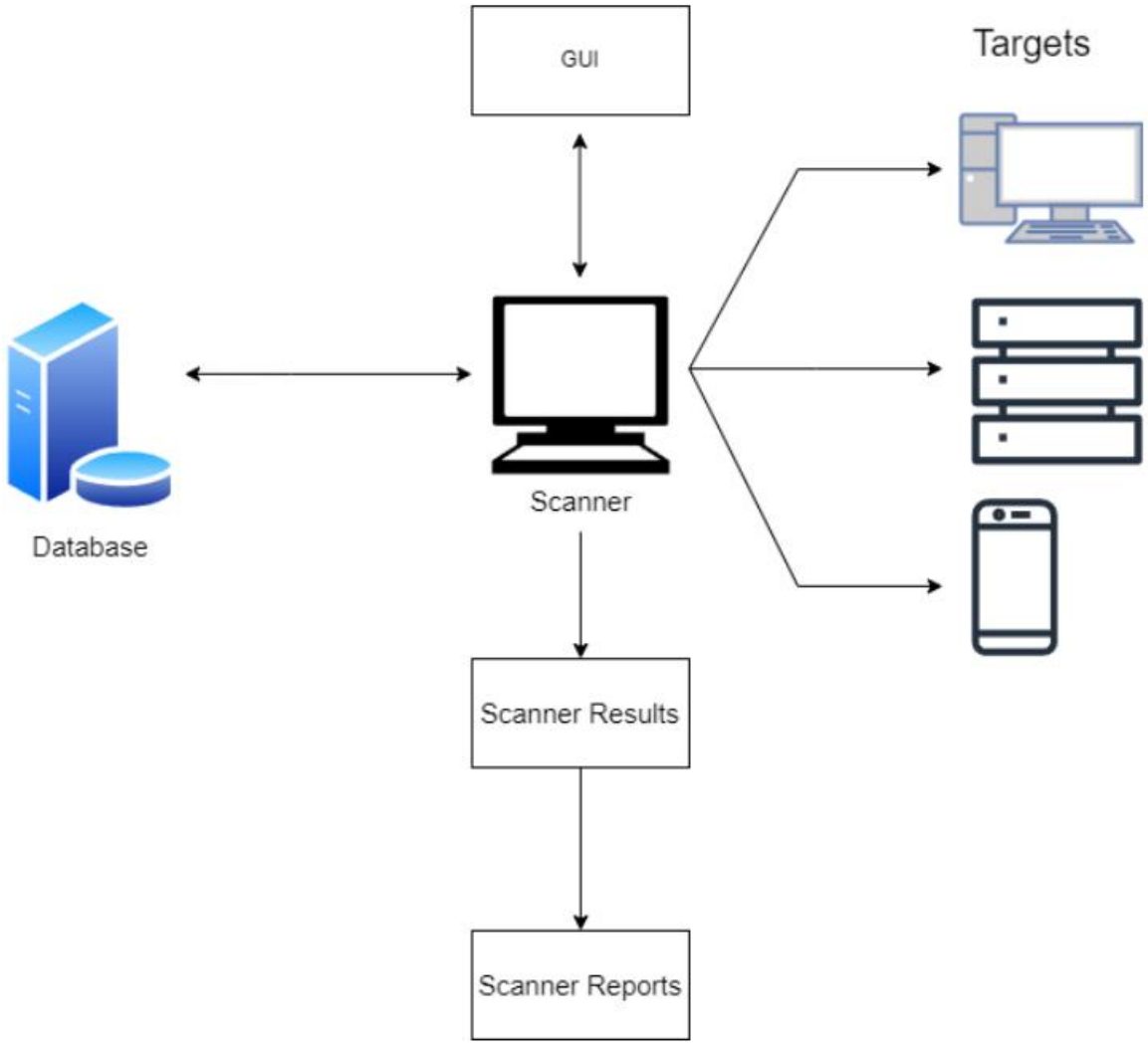


Figure 3 Architectural design

Above is the visual representation of an architectural layout of the vulnerability scanning system. Below is a summary of every part and how it fits into the system:

- GUI (Graphical User Interface) – Users communicate with the scanning system via this front-end interface. Users can enter settings, start scans, and view the results through this interface.
- Scanner – The scanner, the system's central component, handles user requests from the GUI. By comparing the targets to known vulnerabilities kept in the database, it manages the actual scanning of the targets. Here in the backend, Nmap is extensively used to scan the target host, to identify the open ports and the services running, and information related to the system and the ports.

- Database – This storage component has a database of vulnerabilities that are known to exist. To obtain the data required to find weaknesses in the targets, the scanner makes use of this database. Some of them are cve.csv, exploitdb.csv, openvas.csv. These databases are subject to change when we move forward according to the implementation of the scanning system.
- Targets – These are the objects that the system is currently scanning. Targets in this context can be any kind of system or device, including computers, servers, and mobile devices. The target types shown in the diagram demonstrate the scanner's versatility in handling a range of systems.
- Scanner Results – Following the targets' scanning, the scanner produces results that list any vulnerabilities discovered. These outcomes are directly related to the scanning procedure.
- Scanner report – This part generates comprehensive reports by processing the scanner data. These reports probably contain details about vulnerabilities that have been found, their severity, possible effects, and suggested mitigations. Users can understand the vulnerabilities and take appropriate action with the help of these reports.

This is the initial design and initial planning of the scanner, these are subject to change when we move along the weeks according to user feedback, user testing, and system modification.

Flowchart

A flowchart is a diagrammatic depiction that shows the steps or workflow involved in resolving an issue or carrying out a procedure. It makes understanding and analysis easier by using standardized symbols and shapes to represent different kinds of actions or steps in a process. Flowcharts are widely used because they are a useful tool for streamlining complex processes and promoting a shared understanding among various stakeholders in a variety of fields, including business, education, engineering, and programming. Using flowcharts, one can more easily comprehend how decisions impact the final result and how tasks are connected. They simplify and streamline work procedures by assisting in the identification of redundant and needless steps in a process. Because they make the process easy for newcomers to understand, flowcharts are a great choice for training materials. All things considered, flowcharts are useful tools that can help with both the design and analysis stages of process management. This makes them indispensable in many situations where following protocol and maintaining order are crucial (IN-COM, 2024).

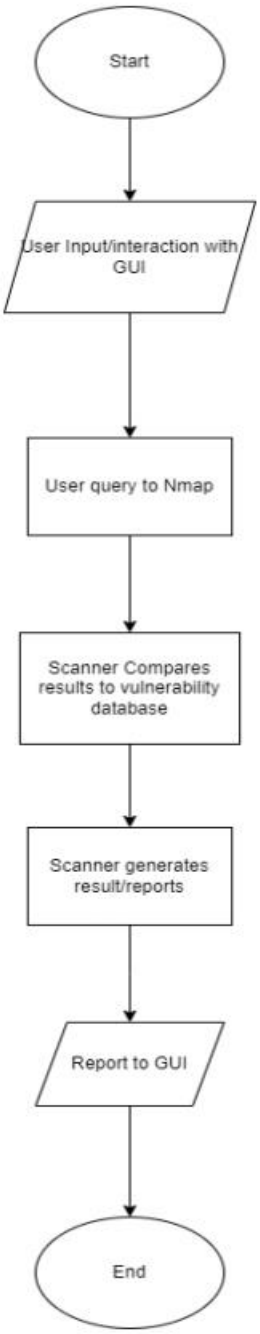


Figure 4 Flowchart of the system

GUI (Graphical User Interface)

Unlike text-based interfaces, typed command labels, or text navigation, a GUI, or Graphical User Interface, enables users to interact with electronic devices using graphical icons and visual indicators. GUIs are intended to improve the intuitiveness and usability of software without requiring users to type or remember commands. By offering clear interfaces and visual cues, graphical user interfaces (GUIs) increase the accessibility and ease of use of computer systems, particularly for novice users. They facilitate faster software navigation, which lowers the learning curve and boosts user output. Especially for those with specific disabilities, graphical user interfaces (GUIs) facilitate the use of pointing devices such as mice, which can be more user-friendly than command-line interfaces. The software can feel more engaging and pleasurable to use when it has a graphical user interface (GUI), which can improve the overall user experience (Rouse, 2021).

A screenshot of our tools GUI is shown below.

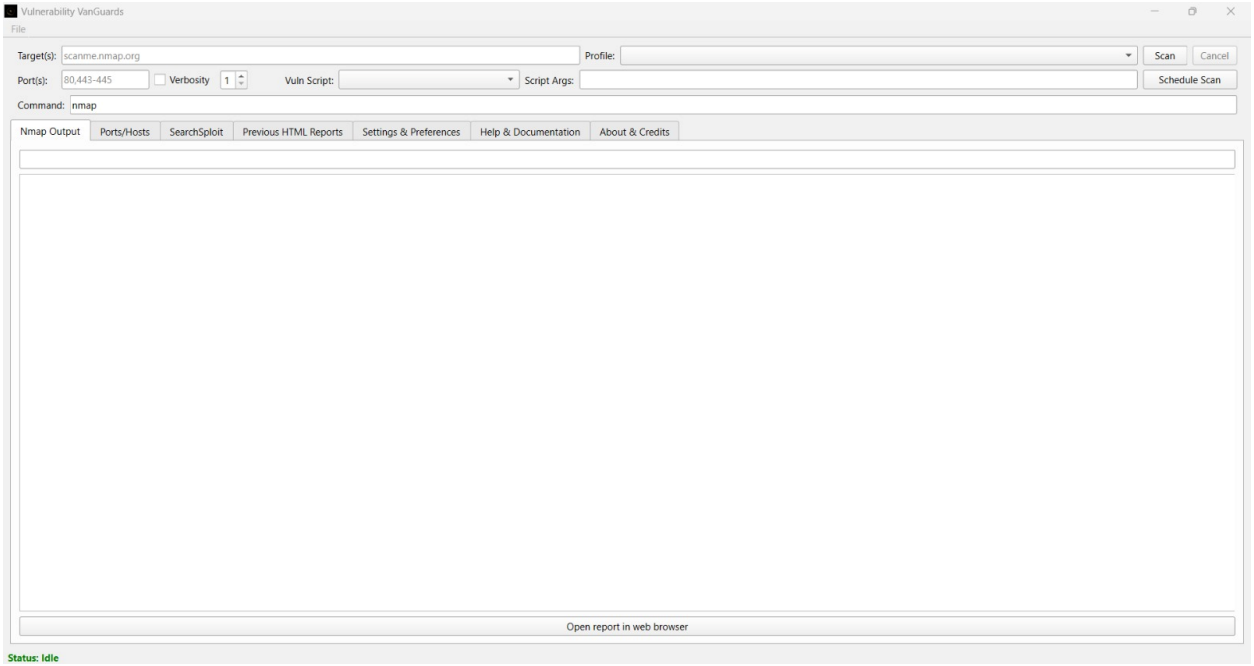


Figure 5 System GUI

4.2 Software development process

The real software development process occurs during the development phase of a software development project, which is frequently referred to as the project lifecycle's center. It's a crucial phase that usually comes after thorough planning and design and precedes deployment and testing. Here's an introduction to what the development phase involves:

- Coding and Implementation – In the development phase, this is the main task. With the programming languages and tools of their choice, developers write code according to the requirements and design specifications that were specified in previous stages. The aim is to convert the conceptual design and documentation into a usable software product.
- Unit Testing – Unit testing is a common practice among developers as they write code to make sure that the software's components—or units—function as intended. Before being incorporated into the bigger system, every unit is independently tested. This aids in the early detection and correction of bugs throughout the development cycle.
- Integration – The different software modules or components are integrated to work as a cohesive unit after unit testing. The linking of independent modules may need more coding in this step, and managing dependencies and possible conflicts between interacting modules frequently takes a significant amount of work.
- Integration Testing – Integration testing is done after integration to ensure that various application components function as intended. This kind of testing makes sure the software functions as a cohesive system and finds problems with the interfaces between components.
- Review and Iteration – Iterations are common during the development phase, particularly in agile development models. Regular reviews and revisions are made in response to test results and stakeholder feedback. The software can be continuously improved, features can be refined, and it can be adjusted to meet changing requirements thanks to this iterative process.
- Preparing for Deployment – As the development phase comes to an end, the emphasis switches to getting the software ready for deployment. This entails completing optimizations, evaluating security, and guaranteeing performance benchmark compliance. 14 It is necessary to get the software to a point where users or clients can dependably release it.

The dynamic nature of the development phase necessitates striking a balance between project management, technical expertise, and a thorough comprehension of end-user needs. During this stage, making sure that testing, documentation, and stakeholder engagement are all done thoroughly can greatly improve the final software product's quality and success (Nikolaieva, 2024).

Software Development Life Cycle (SDLC)

Software engineers use the Software Development Life Cycle (SDLC) as a methodical approach to design, develop, and test high-quality software. Coding standards and best practices must be integrated into the SDLC to guarantee consistency, maintainability, and software product quality. These guidelines and procedures reduce the likelihood of mistakes and security flaws while also resulting in a codebase that is simple to comprehend, maintain, and expand. There are some of the key steps and practices involved which are as follows:

- Requirement analysis
- System design
- Implementation
- Testing
- Deployment
- Maintenance

Best practices during the requirement analysis phase center on using requirement management tools, communicating with stakeholders on a regular basis, and documenting requirements in a clear and comprehensive manner. While coding standards aren't directly relevant at this stage, it's still important to make sure the requirements make sense and follow the accepted standards. This alignment aids in laying a strong basis for the stages that follow.

Best practices in system design include generating thorough design documents, applying design patterns when appropriate, and taking performance and scalability into account. Here, coding standards come into play by guaranteeing that classes, methods, and variables have consistent naming conventions, and by visualizing the design with UML diagrams. These procedures aid in the creation of a blueprint that directs the development team and guarantees a solid and orderly system architecture.

It's important to follow coding standards and guidelines during the implementation (coding) phase. Regular code reviews, pair programming, and writing unit tests alongside the code are among the 32 best practices during this phase. Coding standards are crucial and include things like adhering to a consistent code style guide, using appropriate indentation, meaningful naming conventions, and thorough commenting and documentation. The long-term success of a project depends on the code being clear, readable, and maintainable, all of which are ensured by these standards.

Using automated testing tools, creating thorough test cases covering every scenario, and utilizing continuous integration (CI) to run tests on a regular basis are all examples of best practices during the testing phase. Test code should adhere to the same coding standards as the implementation phase to ensure clarity and consistency. Creating dependable and effective tests is further aided by the application of stubbing and mocking techniques for unit tests.

Best practices like utilising automated deployment tools, putting rollback protocols in place, and keeping an eye out for problems during deployments are beneficial to the deployment phase. Using configuration files for environment-specific settings and avoiding hardcoding environment configurations are two examples of coding standards in this phase. These procedures guarantee dependable and efficient deployment procedures.

Lastly, best practices during the maintenance phase center on patching and updating the program on a regular basis, keeping an eye out for errors and performance problems and reworking the code to make it more maintainable. The most important things are to make sure that backward compatibility is maintained, apply coding standards consistently for new code, and keep thorough documentation. These procedures aid in maintaining the software's functionality over time and keeping it current.

To sum up, coding standards and best practices must be followed throughout the software development life cycle (SDLC) to guarantee high-quality, maintainable, and secure software. It guarantees that the program can be effectively maintained and expanded over time, fosters teamwork, and lowers the possibility of errors. These procedures are essential to accomplishing any software development project since they support creating dependable and durable software solutions (Clark, 2024).

4.3 Findings for surveying audience

Here we have made a Google form and, recorded the response of the audience and analyzed the results of the survey to implement in our project.

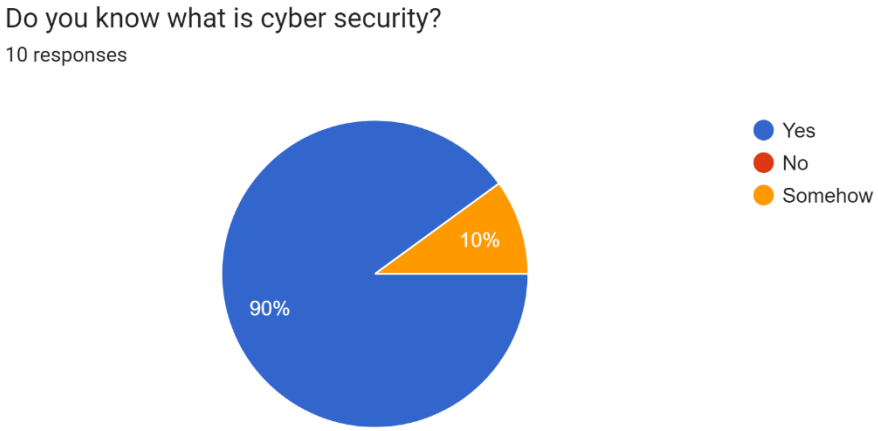


Figure 6 Survey results 1

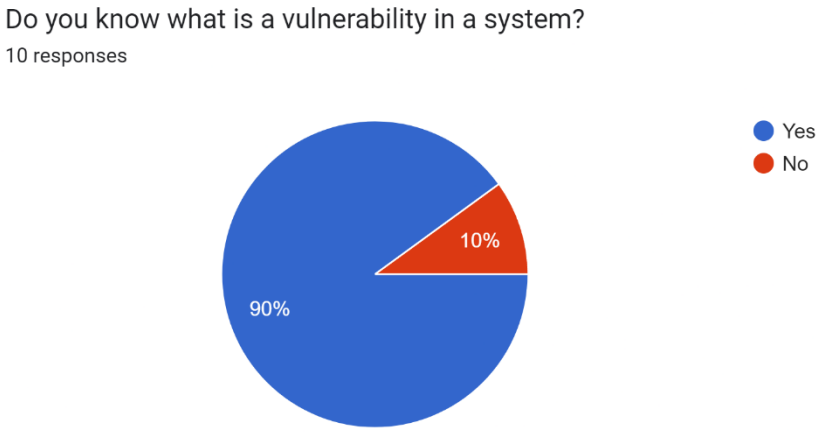


Figure 7 Survey results 2

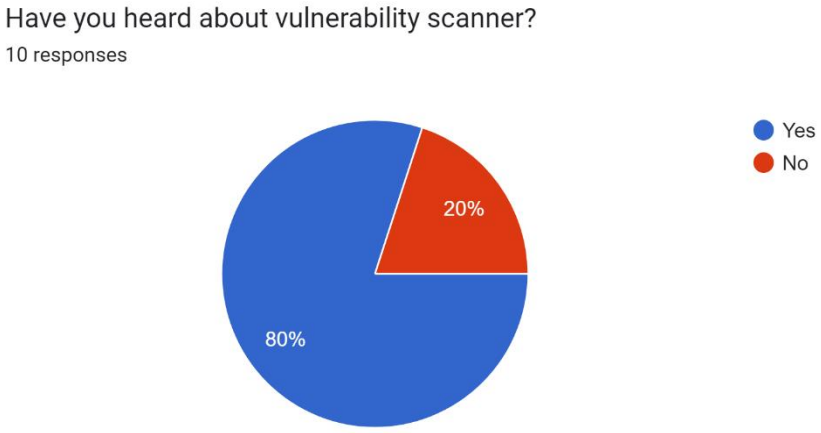


Figure 8 Survey results 3

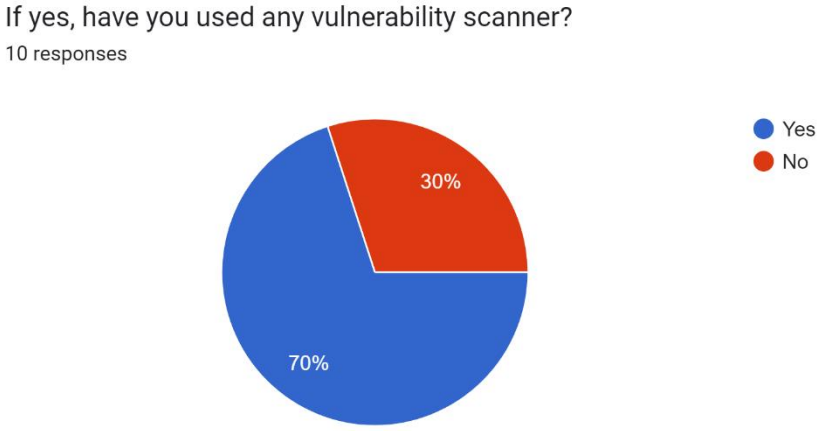


Figure 9 Survey results 4

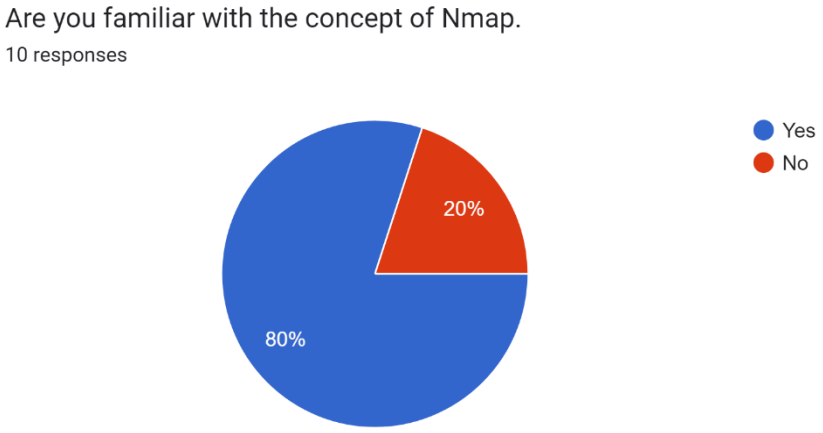


Figure 10 Survey results 5

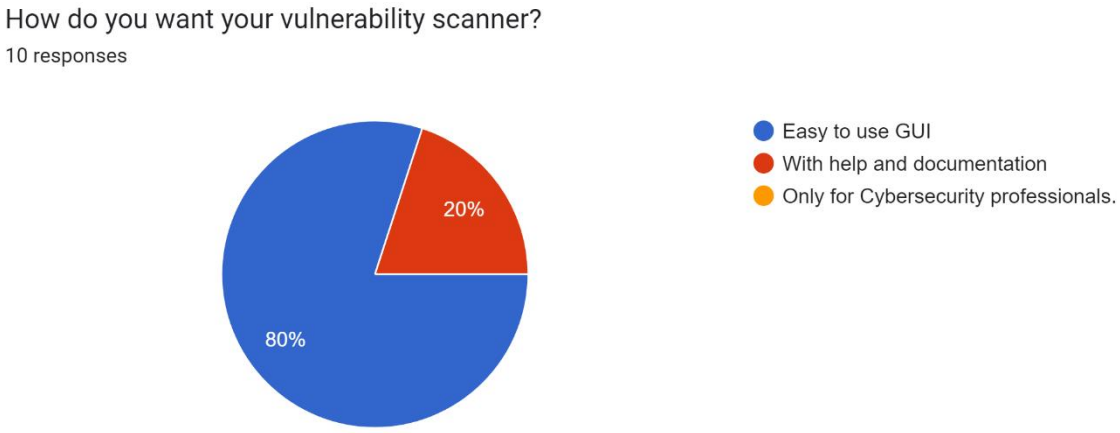


Figure 11 Survey results 6

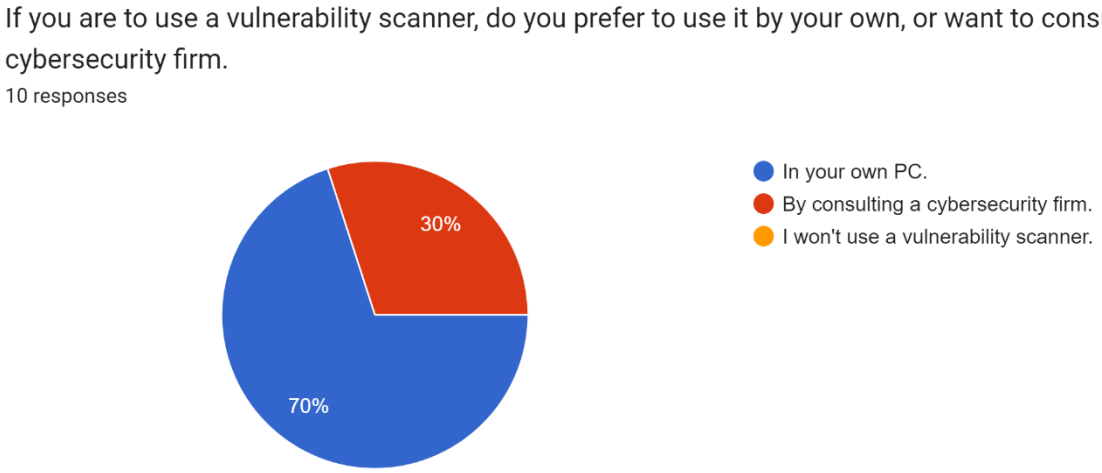


Figure 12 Survey results 7

Based on the analysis of the Google Form responses, the following insights were gathered regarding users' understanding and preferences related to cybersecurity and vulnerability scanning tools:

1. Understanding of Cybersecurity:

- 90% of respondents know what cybersecurity is.
- 10% are somewhat familiar with the concept.

2. Understanding of System Vulnerabilities:

- 90% of respondents are aware of what a vulnerability in a system is.
- 10% do not know about system vulnerabilities.

3. Awareness of Vulnerability Scanners:

- 80% have heard about vulnerability scanners.
- 20% have not heard about them.

4. Usage of Vulnerability Scanners:

- Among those aware of vulnerability scanners, 70% have used them.
- 30% have not used any vulnerability scanners.

5. Familiarity with Nmap:

- 80% of respondents are familiar with the concept of Nmap.
- 20% are not familiar with Nmap.

6. Preferences for Vulnerability Scanner Interface:

- 80% prefer an easy-to-use GUI.
- 20% prefer help and documentation.
- 0% indicated that the tool should be only for cybersecurity professionals.

7. Usage Preference of Vulnerability Scanner:

- 70% prefer to use a vulnerability scanner on their own PC.
- 30% prefer consulting a cybersecurity firm.
- 0% indicated they would not use a vulnerability scanner.

The replies to the Google Form poll provide useful information about consumers' understanding and preferences for cybersecurity and vulnerability screening solutions. The vast majority (90%) of respondents are aware of fundamental cybersecurity principles and understand what a system vulnerability is. This demonstrates a high level of awareness among the target population.

Vulnerability scanners are likewise well-known and widely utilized, with 80% of respondents having heard of them and 70% using them. This demonstrates a positive reaction and acceptance of such tools throughout the cybersecurity community.

A large percentage of respondents (80%) are familiar with Nmap, a popular network scanning tool, indicating that the target audience has some technical knowledge and expertise with cybersecurity solutions.

When it comes to vulnerability scanner interfaces, the majority (80%) choose a simple graphical user interface (GUI), whereas 20% prefer extensive support and documentation. This emphasizes the need for user-friendly design and the availability of support services to accommodate varying user preferences.

When it comes to vulnerability scanners, 70% of respondents prefer to use them on their own computers, showing a preference for hands-on control and rapid access to the tool. Meanwhile,

30% would prefer to contact a cybersecurity firm, indicating a need for expert guidance or a lack of confidence in utilizing the technology on their own.

Overall, our findings highlight the importance of creating a vulnerability scanning tool that is user-friendly, well-documented, and accessible, as well as offering professional consultation options to meet users' various degrees of competence and confidence.

4.4 Findings that set us apart from others

4.4.1 Intuitive User Interface

Our user research and feedback sessions demonstrated the need of a user-friendly graphical user interface (GUI). Unlike many other vulnerability detection tools, which have steep learning curves and crowded interfaces, our application has a modern, soft-colored, and straightforward GUI. This design approach ensures that both rookie and experienced users may easily explore and operate the product, making cybersecurity more accessible to a wider audience.

4.4.2 Comprehensive Help and Documentation

Many programs on the market lack comprehensive documentation and user support, which might be an impediment for new users. Our tool stands out for its rich support resources, which include a detailed user manual, FAQs, and video lessons. These materials enable users to fully exploit the tool's functionality without requiring expert support, hence improving overall user experience and satisfaction.

4.4.3 Automated Scanning and Real-Time Alerts

Our product includes advanced features including automated scan scheduling and real-time warnings. These features enable customers to schedule regular scans without manual involvement and receive fast notifications of discovered vulnerabilities. This proactive approach distinguishes our product from others, which may involve more manual operation and provide delayed or infrequent updates.

4.4.4 Enhanced Security Measures

Our solution includes strong security features including Time-based One-Time Passwords (TOTP) for user authentication and encrypted email alerts for scan results. These procedures give an extra degree of protection, protecting both user data and scan results from unauthorized access. This kind of security integration is not common in vulnerability scanners, thus our tool stands out in terms of data protection and user trust.

4.4.5 Standalone and Self-Operable

Understanding that a large number of users prefer to run the vulnerability scanner on their own PCs influenced the creation of a standalone application. In contrast, other technologies rely largely on cloud-based operations or require professional setup and upkeep. Our tool's ability to be readily installed, adjusted, and operated independently gives users more autonomy and freedom.

We have also talked about the research, findings, and survey impacts in the discussion section in the heading following 5.3, in more detail.

5. Discussion

This report's discussion portion focuses on the development of the vulnerability scanning tool, looking at how our research, conclusions, and user survey responses influenced the final product. This analysis explains how each phase of the project influenced the tool's design, functionality, and user experience. By investigating these factors, we hope to highlight the strengths of our development process, solve the obstacles faced, and identify opportunities for future improvement. This debate will bridge the gap between the project's initial objectives and the actual results, providing a detailed appraisal of the tool's development journey.

5.1 Development

5.1.1 GUI of the scanner

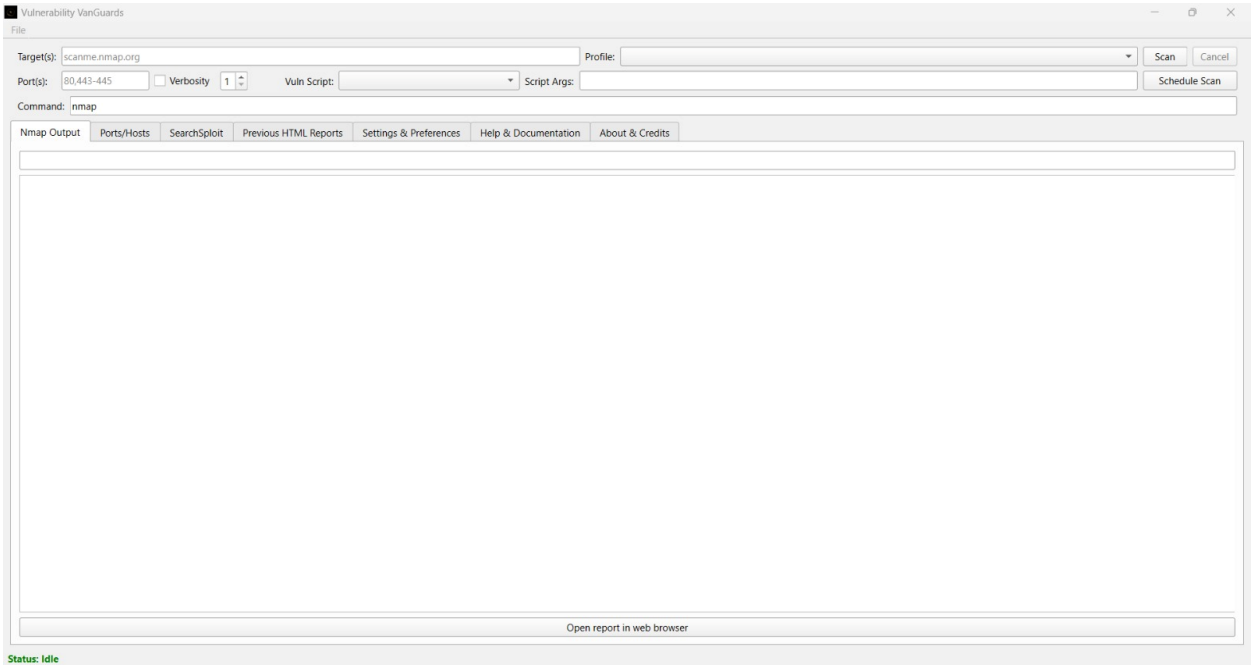


Figure 13 GUI interface

1) File Menu (Top-Left Corner)

- Options for file operations, like opening, saving, or exporting scan results, are probably included in this menu.

2) Target(s) Field

- A text field where users can enter the domain name or target IP address to be scanned. It is configured to "scanme.nmap.org" in the screenshot.

3) Port(s) Field

- An input field where the port range to be scanned can be entered. The screenshot displays "80,443-445," which denotes ports 443, 80, and 445.

4) Verbosity Dropdown

- A drop-down menu to choose the scan's verbosity level, which regulates how much detail is output in the scan result. The screenshot shows that it is set to "1".

5) Vuln Script Dropdown

- A drop-down menu that allows users to choose which vulnerability script to run when the scan is running.

6) Profile Field

- A text field or dropdown menu where users can choose a scan profile, some of which may have preset parameters for various scan kinds.

7) Script Arguments Field

- A text field where you can type in extra script arguments that you might need for more intricate scans.

8) Command Field

- A text field with the command to be performed displayed. In this instance, it displays "nmap," signifying that the scanning tool being used is Nmap.

9) Scan Button

- A button to start the scan with the chosen parameters.

10) Cancel Button

- A cancel button for a running scan.

11) Schedule Scan Button

- A button to schedule scans for a later time, allowing users to automate scans.

12) Tab Panel

A number of tabs to switch between the application's various sections:

- Nmap Output: Shows the Nmap scan's findings.
- Ports/Hosts: Offers comprehensive details regarding the ports and hosts that have been scanned.
- SearchSploit: Combines SearchSploit results to locate verified exploits for vulnerabilities found.
- Settings & Preferences: Allows users to configure application settings and preferences.
- Help & Documentation: Offers access to user manuals and help files for the scanner.
- About & Credits: Provides details about the application and gives the developers credit.

13) Output Display Area

- A sizable space beneath the tab panel that will be used to show other data and scan results.

14) Status Bar (Bottom-Left Corner)

- Shows the application's current state, as "Idle" in the screenshot.

When combined, these elements offer an extensive interface for administering and performing vulnerability scans inside your application with Nmap and SearchSploit.

5.1.2 Profile tab

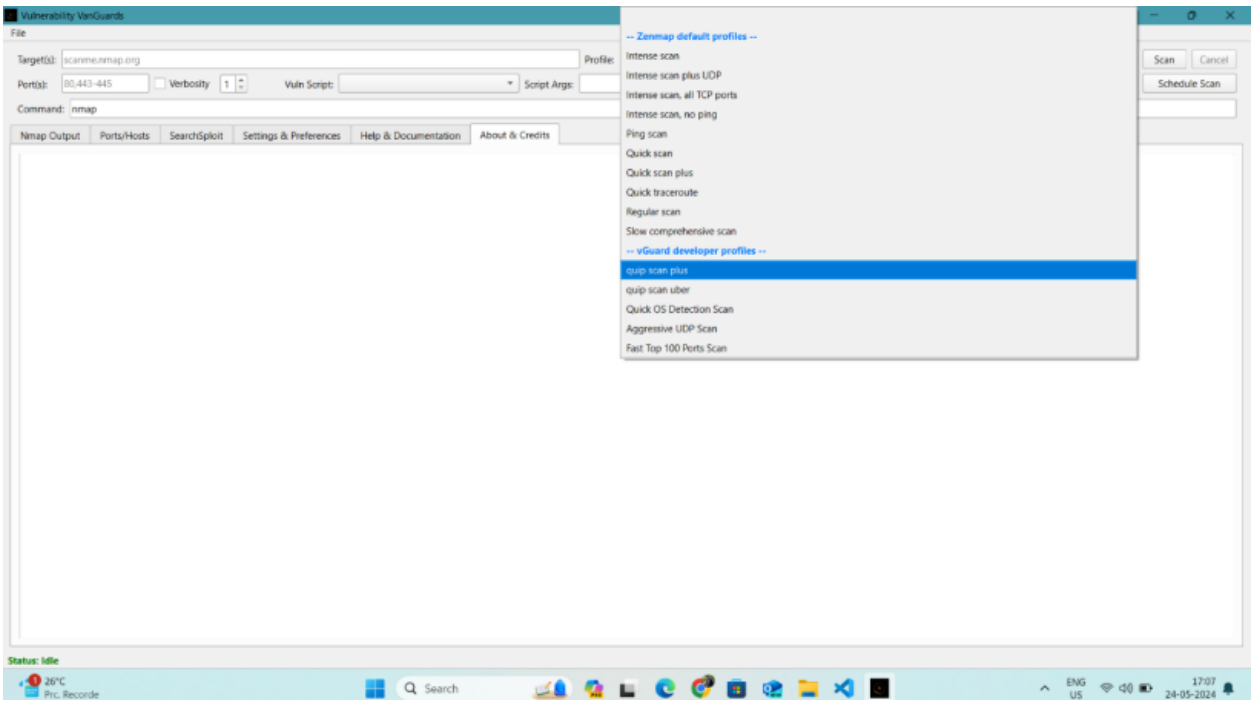


Figure 14 Profile tab

A number of predefined profiles are displayed when the dropdown menu for choosing a scan profile is expanded:

- Zenmap default profile: Features profiles such as Ping scan, Quick scan, Quick scan plus, Quick traceroute, Regular scan, Slow comprehensive scan, Intense scan, Intense scan plus UDP, Intense scan (all TCP ports), Intense scan (no ping), and so on.
- vGuard developer profile: Features include profiles such as Fast Top 100 Ports Scan, Aggressive UDP Scan, Quick OS Detection Scan, Quip Scan Plus, and Uber.

Users can choose various scanning configurations quickly and easily with these profiles without manually adjusting every setting.

Objective	To test the working of the profile tab
Expected result	As there are many types of scans to perform in the tab, all the types will run properly.
Actual result	The scans were running properly.
Conclusion	The test was successful.

5.1.3 Vulnerability script tab

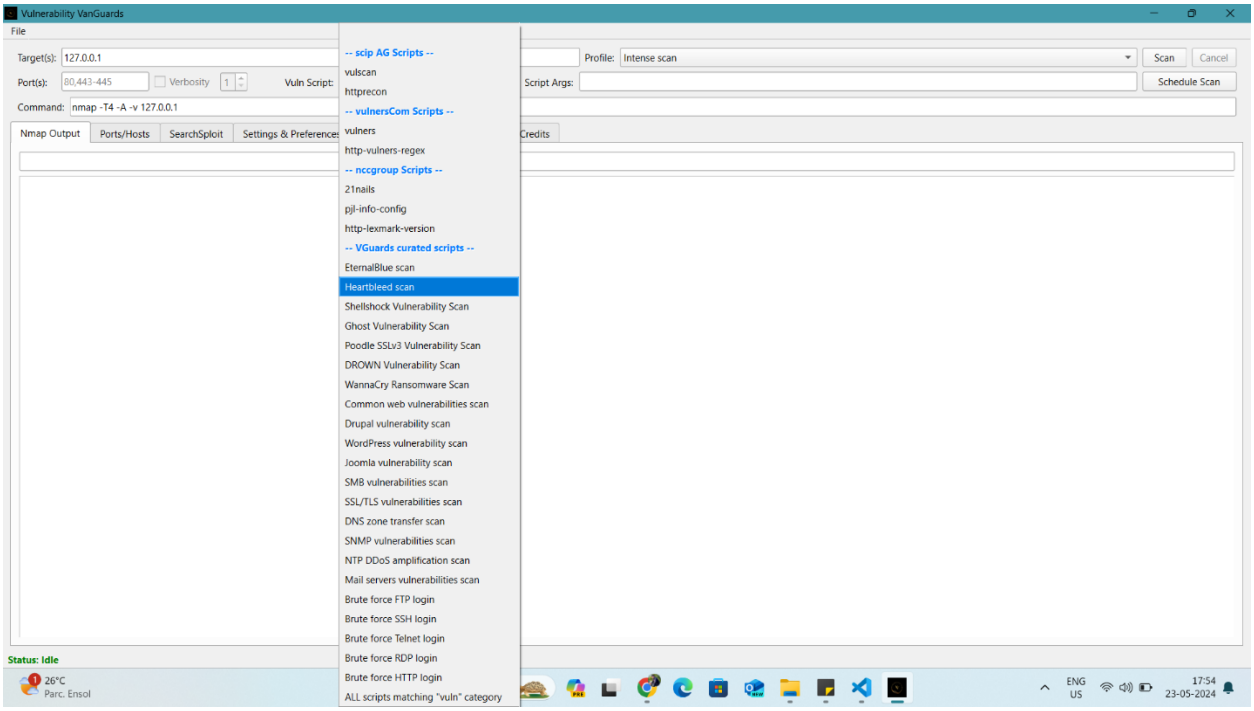


Figure 15 Script tab

This dropdown provides an extensive list of scripts tailored for various vulnerability checks, allowing users to perform targeted scans based on specific needs.

The dropdown menu for selecting a vulnerability script is expanded, showing several script categories:

1) scip AG Scripts: Includes scripts like vulscan and httprecon.

2) vulnersCom Scripts: Includes scripts like vulners and http-vulners-regex.

3) nccgroup Scripts: Includes scripts like 2nails, pjl-info-config, and http-lexmark-version.

4) VGuards Curated Scripts: Includes scripts for specific vulnerabilities and scenarios such as:

- EternalBlue scan
- Heartbleed scan
- Shellshock Vulnerability Scan
- Ghost Vulnerability Scan
- Poodle SSLv3 Vulnerability Scan

- DROWN Vulnerability Scan
- WannaCry Ransomware Scan
- Common web vulnerabilities scan
- Drupal vulnerability scan
- WordPress vulnerability scan
- Joomla vulnerability scan
- SMB vulnerabilities scan
- SSL/TLS vulnerabilities scan
- DNS zone transfer scan
- SNMP vulnerabilities scan
- NTP DDoS amplification scan
- Mail servers vulnerabilities scan
- Brute force FTP login
- Brute force SSH login
- Brute force Telnet login
- Brute force RDP login
- Brute force HTTP login
- All scripts matching "vuln" category

The interface is designed to provide users with a highly customizable and powerful tool for conducting vulnerability scans using a variety of predefined scripts and profiles.

Objective	To test the working and display of the vulnerability script tab and the databases displaying in the GUI.
Expected result	The selected scan will be displayed and the selected database will be triggered.
Actual result	The scan is working properly when the script is selected and the comparison between the scan and the databases is being done successfully.
Conclusion	The test was successful.

5.1.4 Nmap output tab

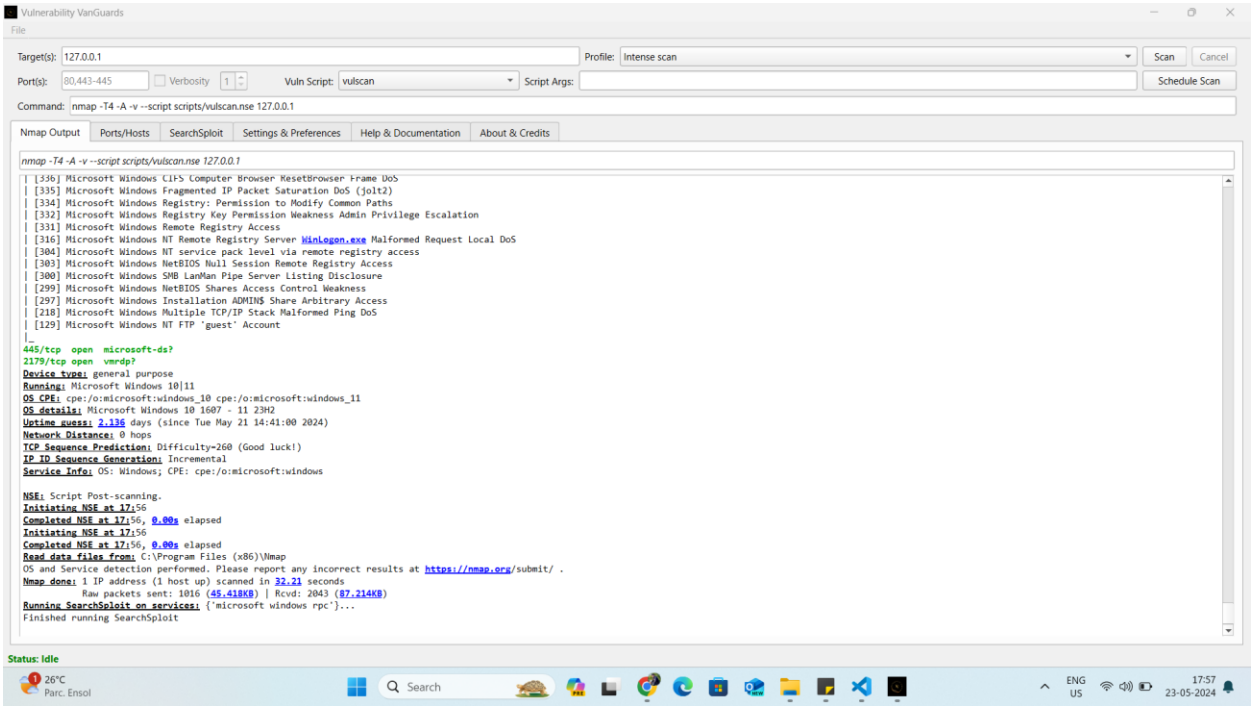


Figure 16 Nmap output tab

As shown in the screenshot above, the result of the Nmap scan is displayed for example open ports: Lists open ports and services running on the target. For example:

445/tcp open microsoft-ds?

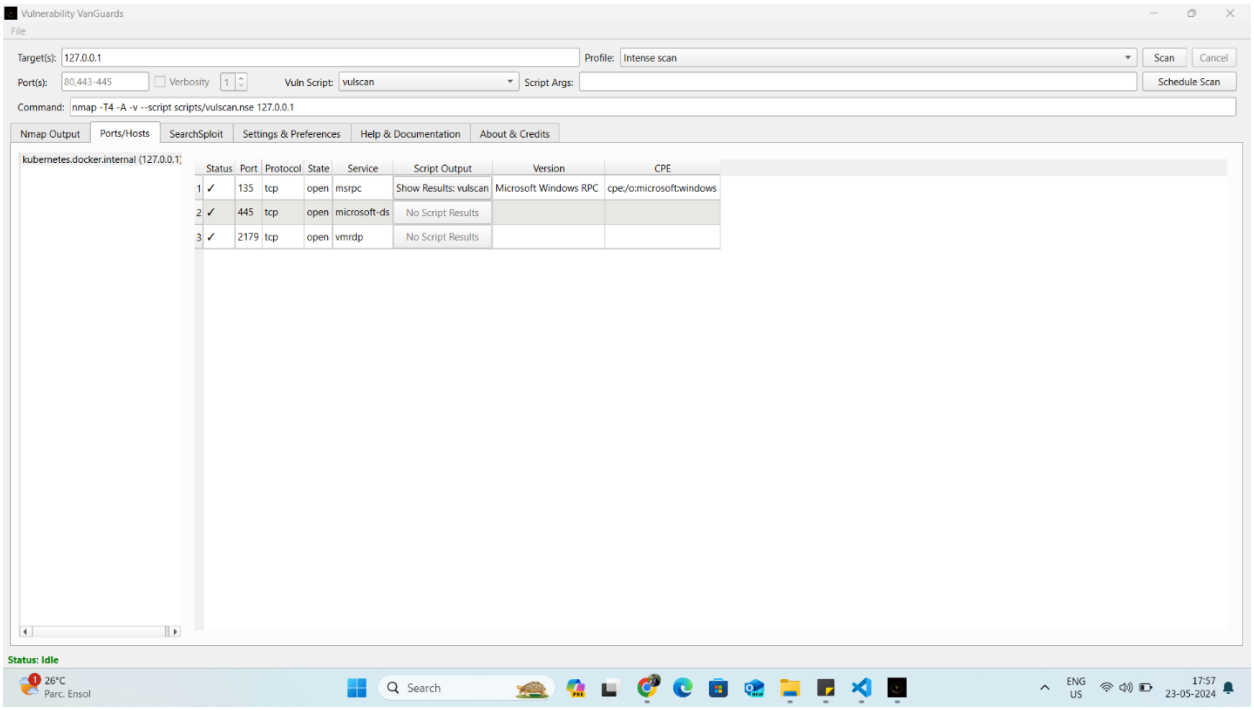
2179/tcp open vmrdp?

These are then compared with the vulnerability script which can be selected as described in the earlier screenshots.

These specifics offer a thorough overview of the scan results, emphasizing open ports, vulnerabilities found, and system data—all of which are combined with extra exploit data from SearchSploit.

Objective	To test the working and display of the Nmap results about ports.
Expected result	It will display the open ports and information about the selected target.
Actual result	The tab displayed the open ports and information about the selected target.
Conclusion	The test was successful.

5.1.5 Port/Host tab



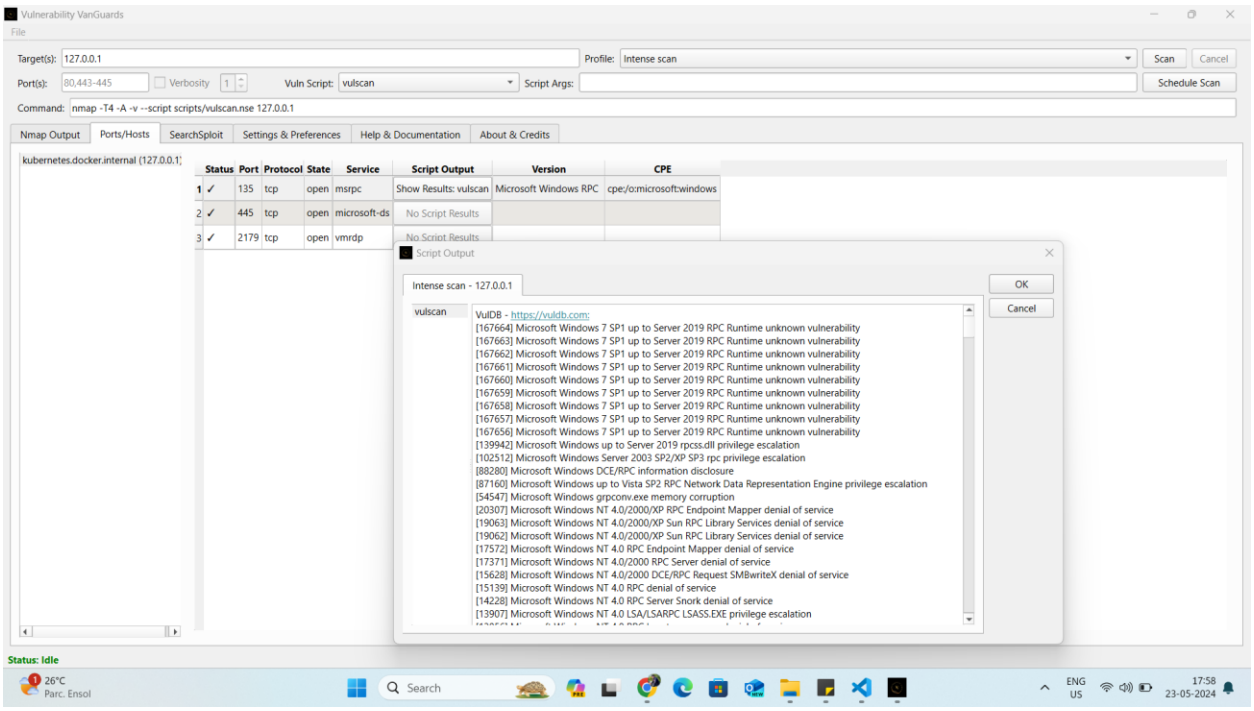


Figure 17 Port/Host tab

This screenshot describes the ports that the Nmap has identified.

Section for Displaying Scan Results:

The Nmap scan's results are displayed in this section.

- Kubernetes.docker.internal (127.0.0.1) is the host.

Details of the open ports and services are listed in the table:

- Status: Denotes the current state (each is numbered consecutively).
- Port: Displays 135, 445, and 2179 as port numbers.
- protocol: TCP
- Status: All are open
- Service: Identifies the Microsoft-ds, vmrdp, and SRPc services that are operating on each port.
- Script Output: Displays the vulnerability script's findings:
- The message "Show Results: vulscan" appears for port 135.
- Under ports 445 and 2179, "No Script Results" is displayed.
- Version: Shows the service's version information.

CPE: Displays the services' Common Platform Enumeration (CPE).

Objective	To test the working and display of the further details of the Nmap results about ports.
Expected result	When clicked on the script output button as shown in the screenshot above, it will display further details about the scan.
Actual result	The tab displayed further details about the scan.
Conclusion	The test was successful.

5.1.6 Help and Documentation tab

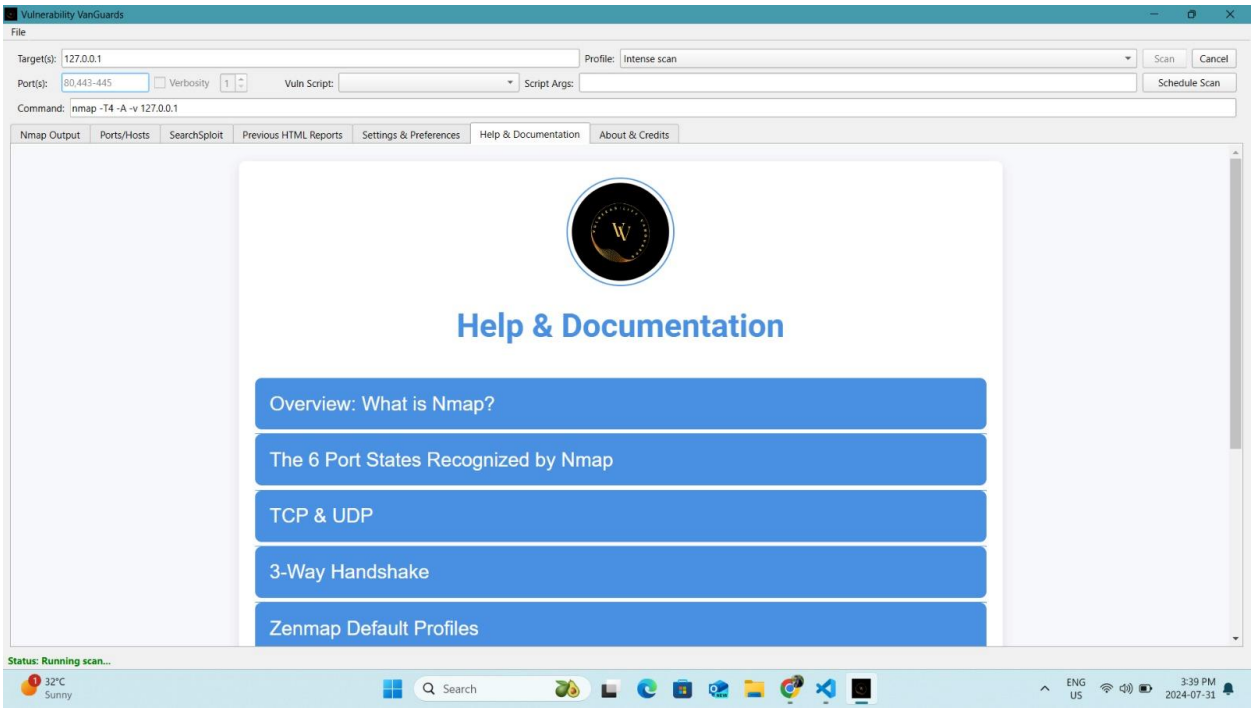


Figure 18 Help and Documentation tab

This tab is all about helping the customer how to use the scanner, what the scanner does, and all the important components of the scanner.

5.1.7 About and Credit tab

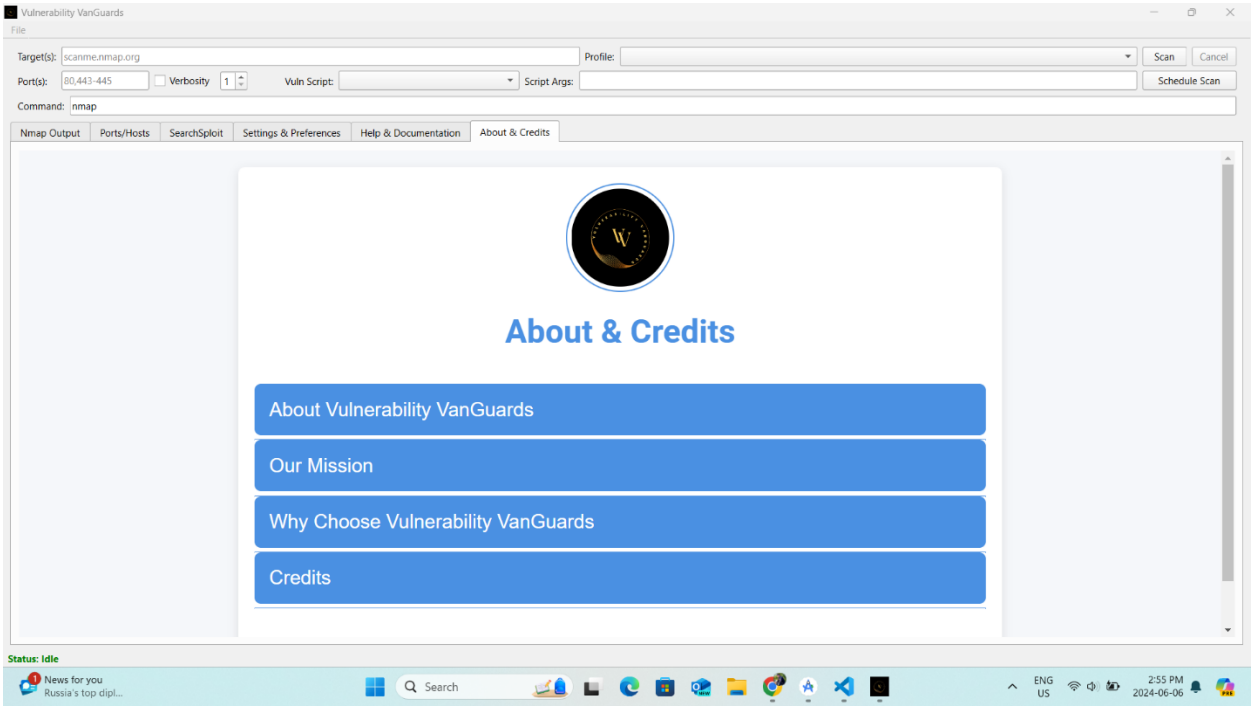


Figure 19 About and Credit tab

Our scanner's "About & Credits" section contains important details about the project's goals, team, and mission. This section aims to clarify for users the purpose of the application, and the motivation behind its development, and to thank all those who contributed to its creation.

Subsections:

- About Vulnerability VanGuards

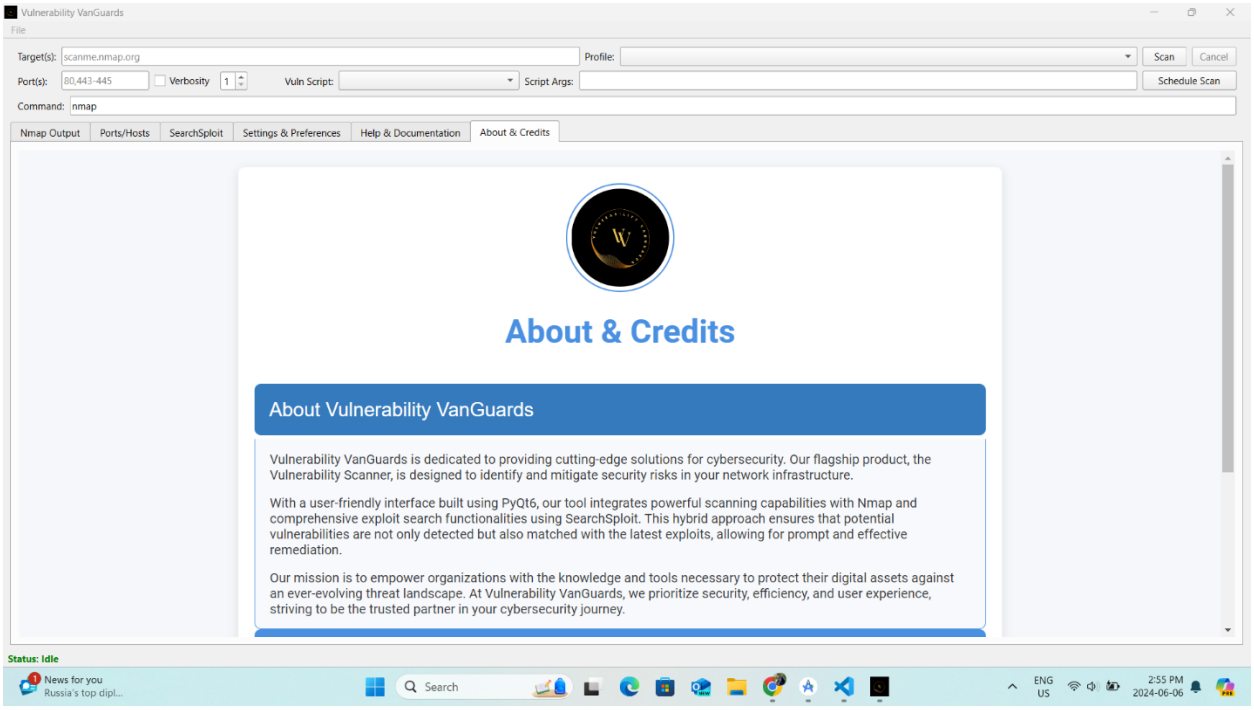


Figure 20 About Vulnerability VanGuards

An overview of the Vulnerability VanGuards project is provided in this subsection. It describes the primary goals of the application, its significance in the field of cybersecurity, and how it assists users in locating and resolving security vulnerabilities. For new users to grasp the extent and power of the tool, this section is crucial.

- Our Mission

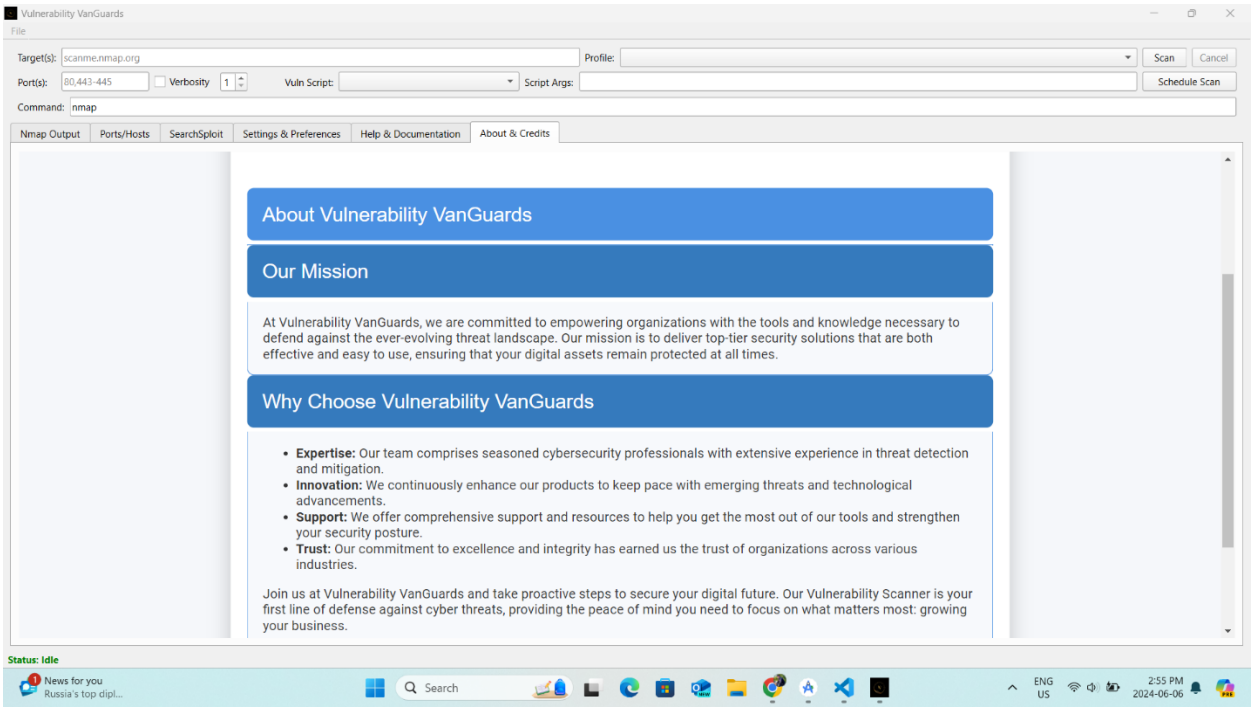


Figure 21 Our mission

This is a summary of the Vulnerability VanGuards project's mission. It draws attention to the main objectives and principles guiding the application's creation and ongoing enhancement. This could involve the dedication to offering dependable security scanning solutions, raising awareness of cybersecurity issues, and cultivating a safe online environment.

- Why Choose Vulnerability VanGuards:

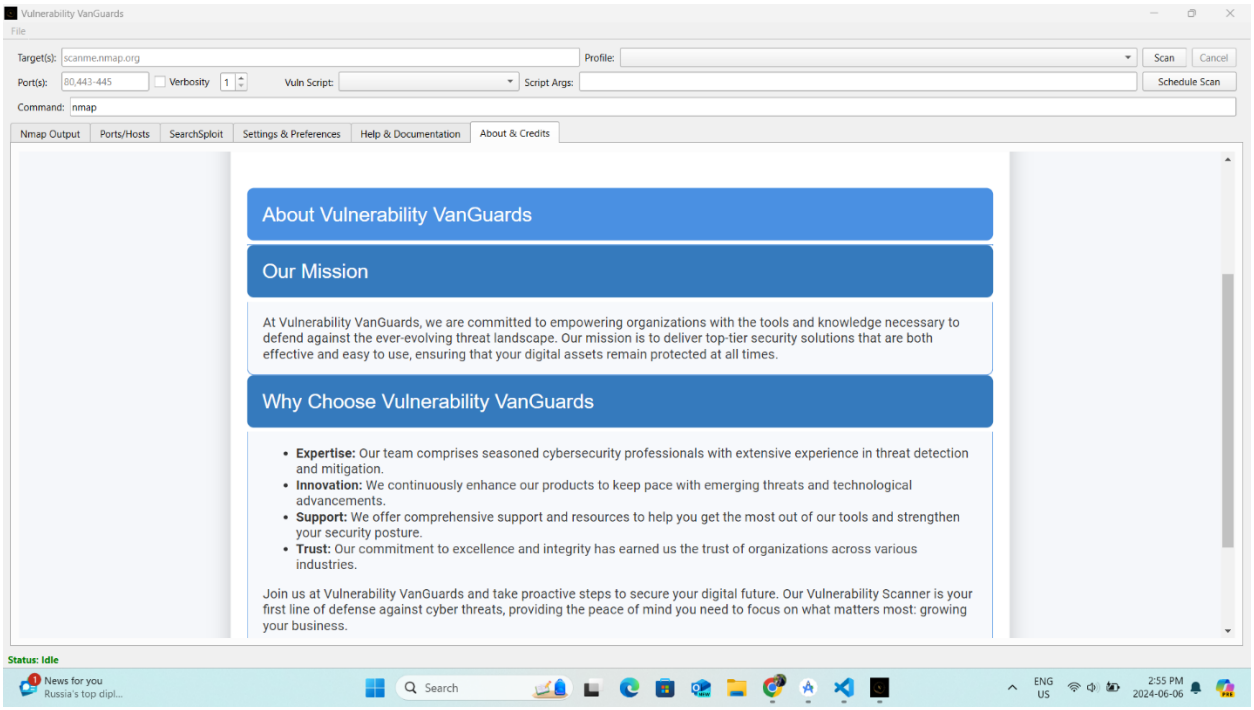


Figure 22 Why vulnerability vanguards

The purpose of this section is to describe the special attributes and advantages of the Vulnerability VanGuards program. It might cover the tool's accuracy, ease of use, extensive scanning capabilities, and compatibility with different protocols and standards. Users can better grasp why this tool is unique in comparison to other vulnerability scanning solutions with the help of this information.

- Credits:

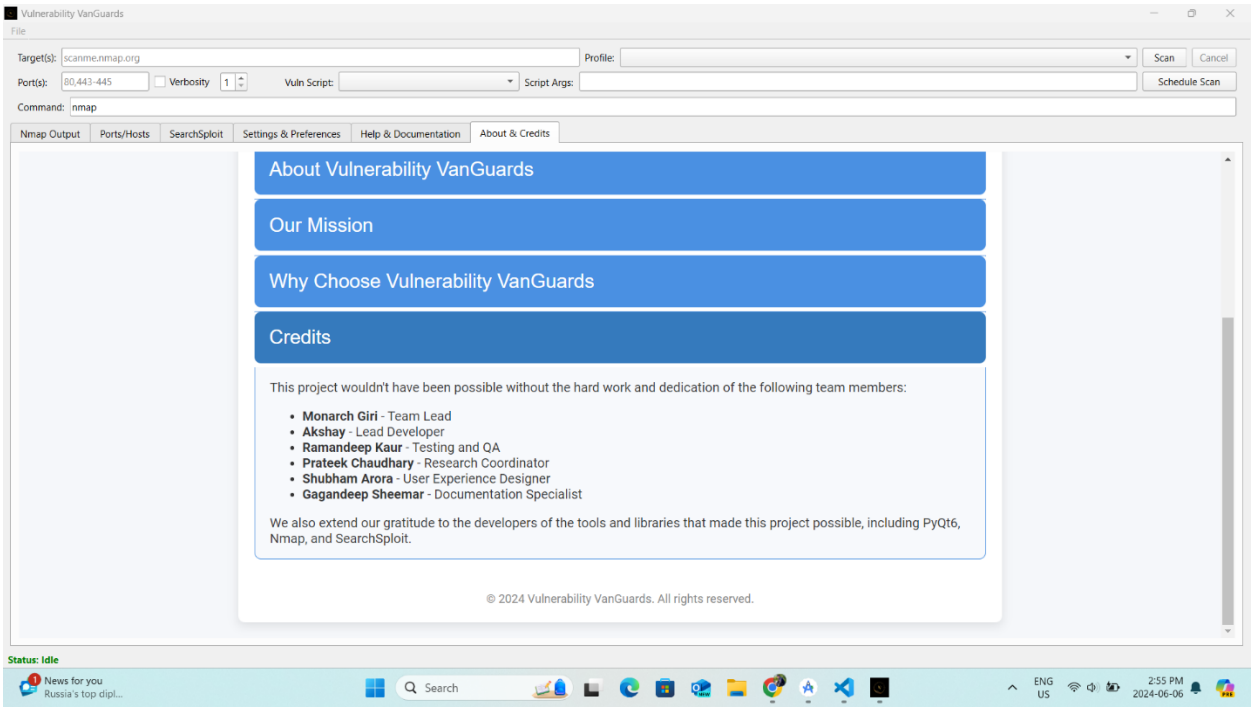


Figure 23 Credits

The people and groups who helped to build the Vulnerability VanGuards project are recognized in the credits subsection. Developers, designers, testers, and any other outside parties who contributed work or organizations that supplied resources or support can all be included in this. Acknowledging these contributors promotes transparency and ensures that everyone gets the credit they deserve.

In general, the "About & Credits" section informs users about the application, the team that created it, and its goal. It establishes credibility and gives users a basis to recognize the work and knowledge that went into developing the scanner.

5.1.8 Login GUI

As shown below, little adjustments and colors are changed for the registration and login pages. The goal was to make the GUI more interactive and user-friendly compared to the previous ones. The buttons are updated and the overall GUI is a bit bigger.

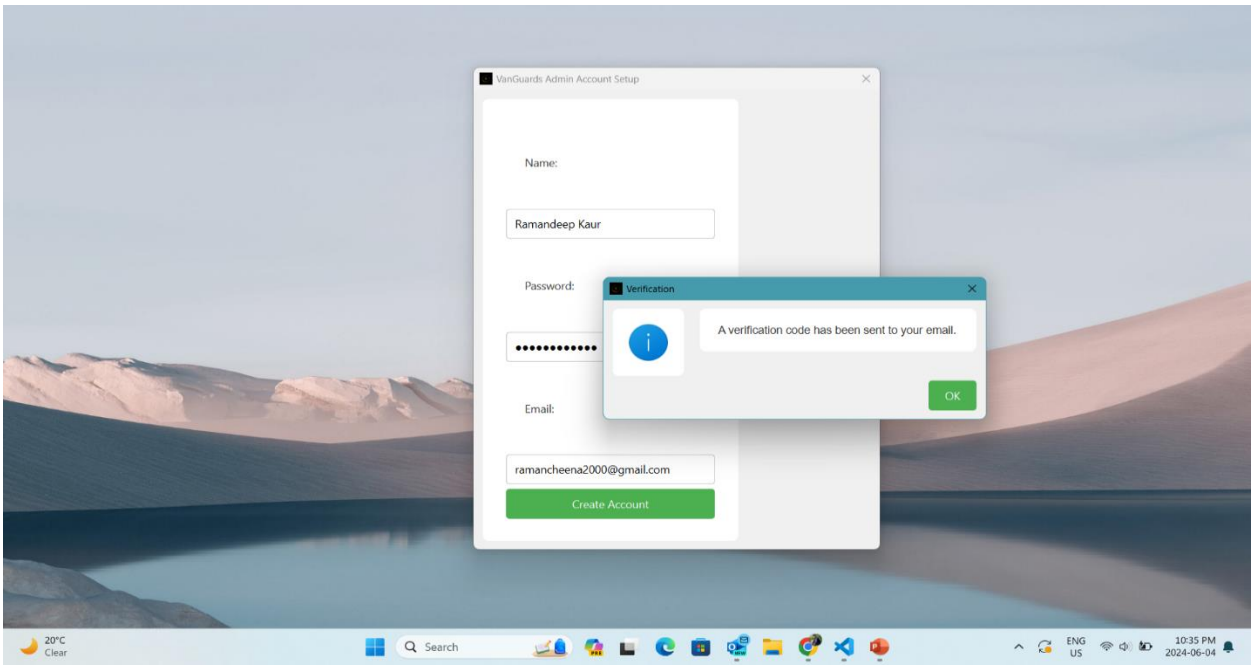


Figure 24 User setup

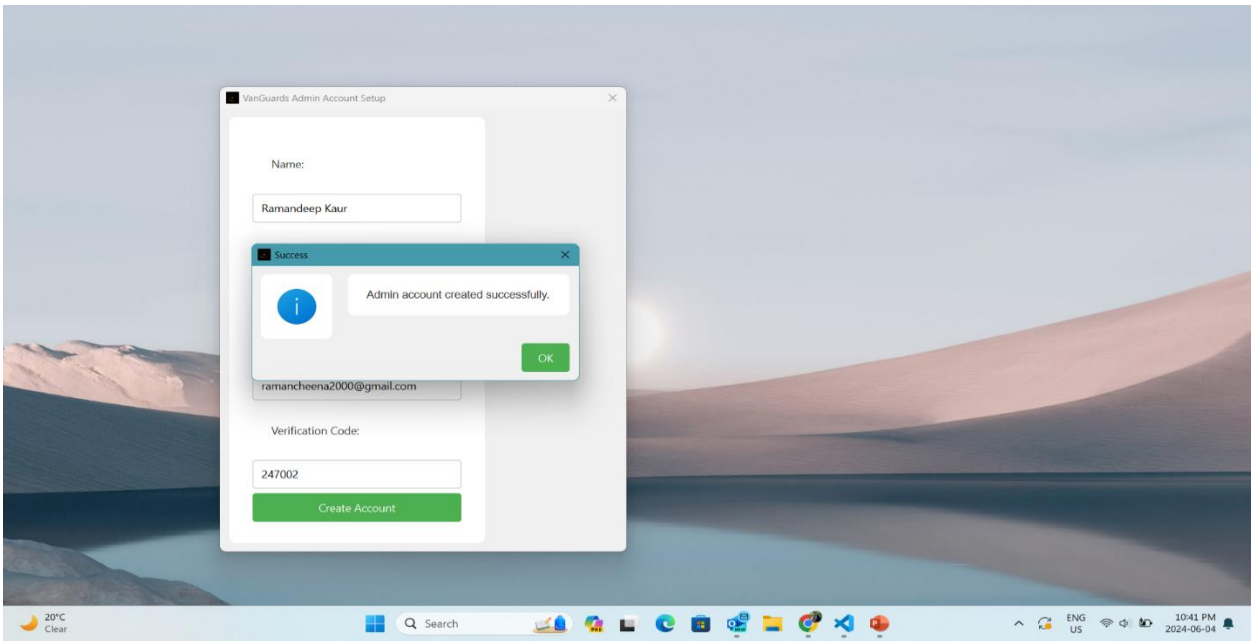


Figure 25 Successfully account created

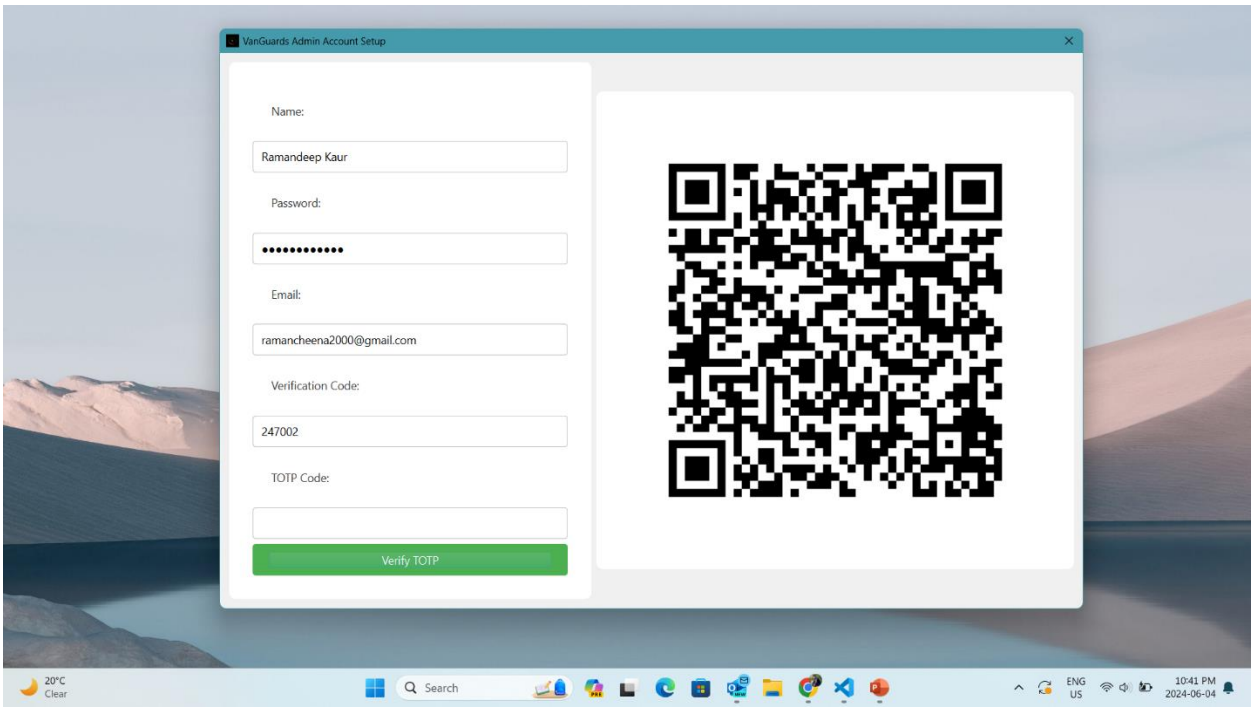


Figure 26 QR for registration

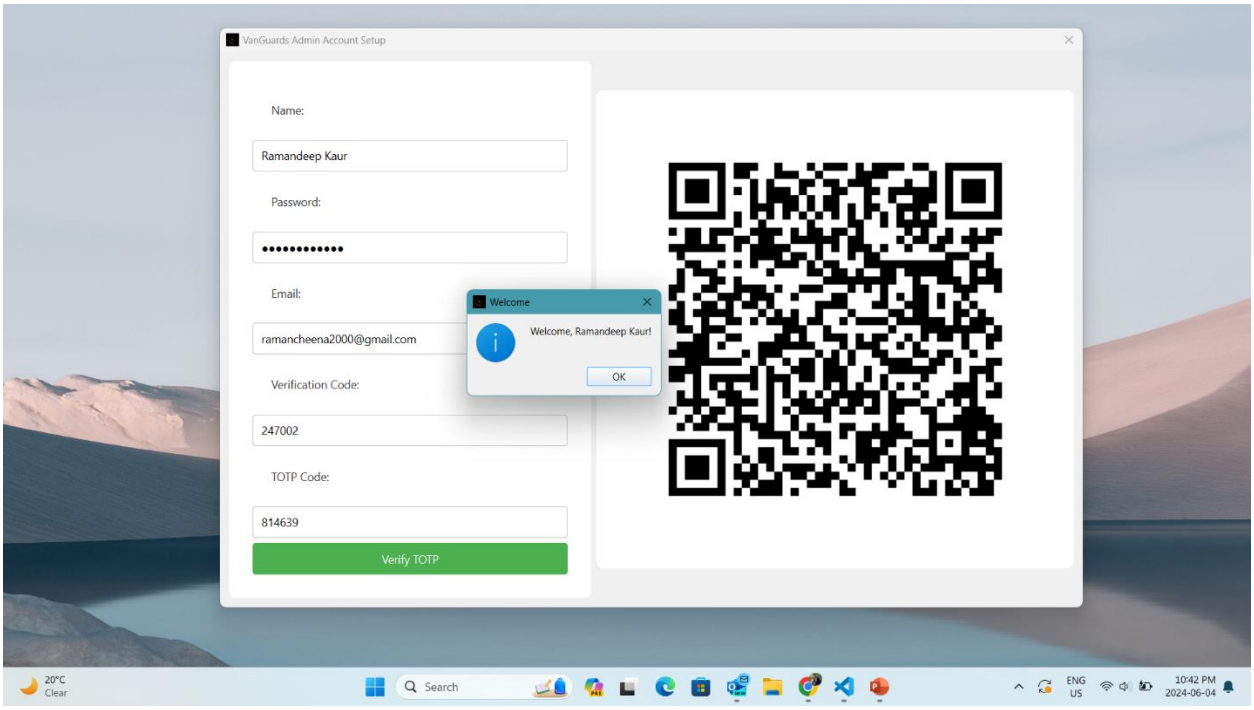


Figure 27 User logged in

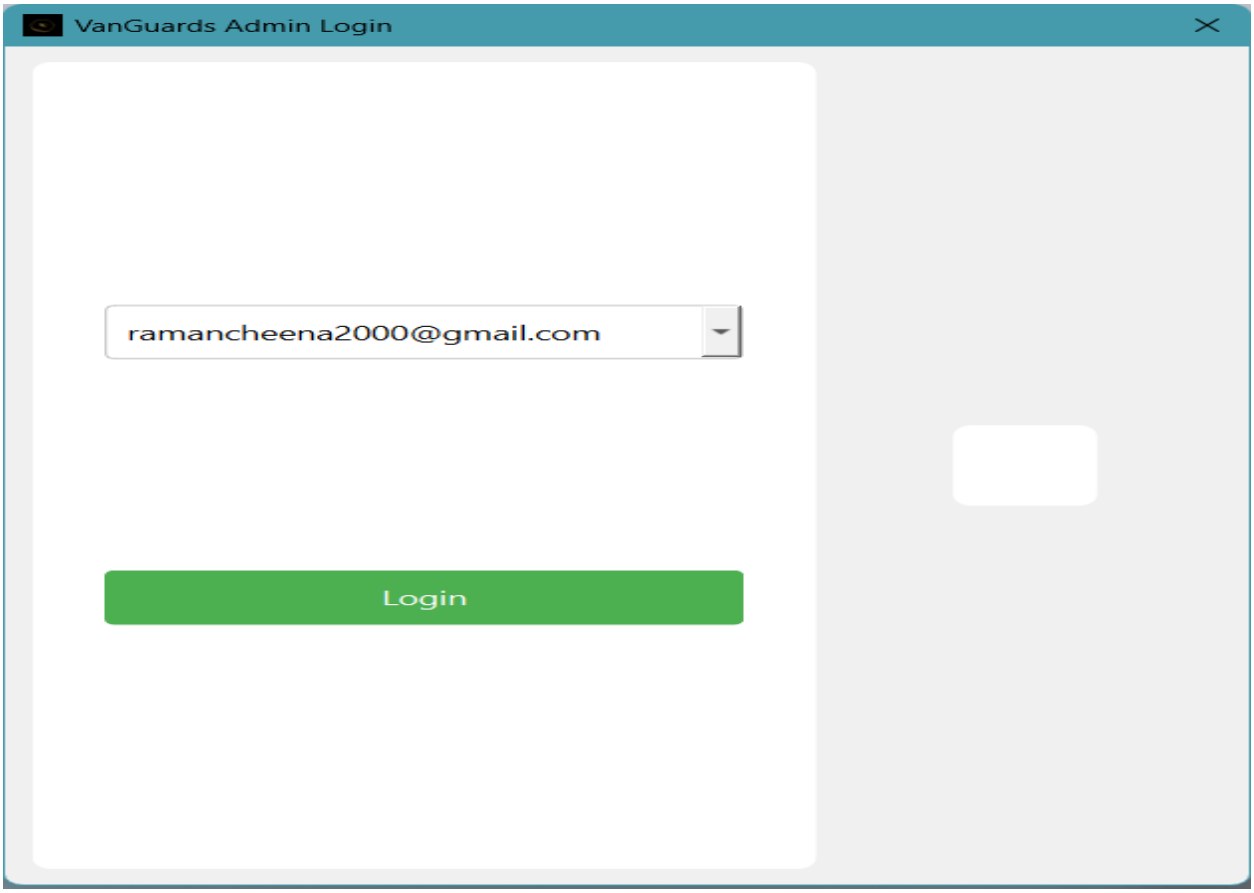


Figure 28 The user registered permanently in the tool

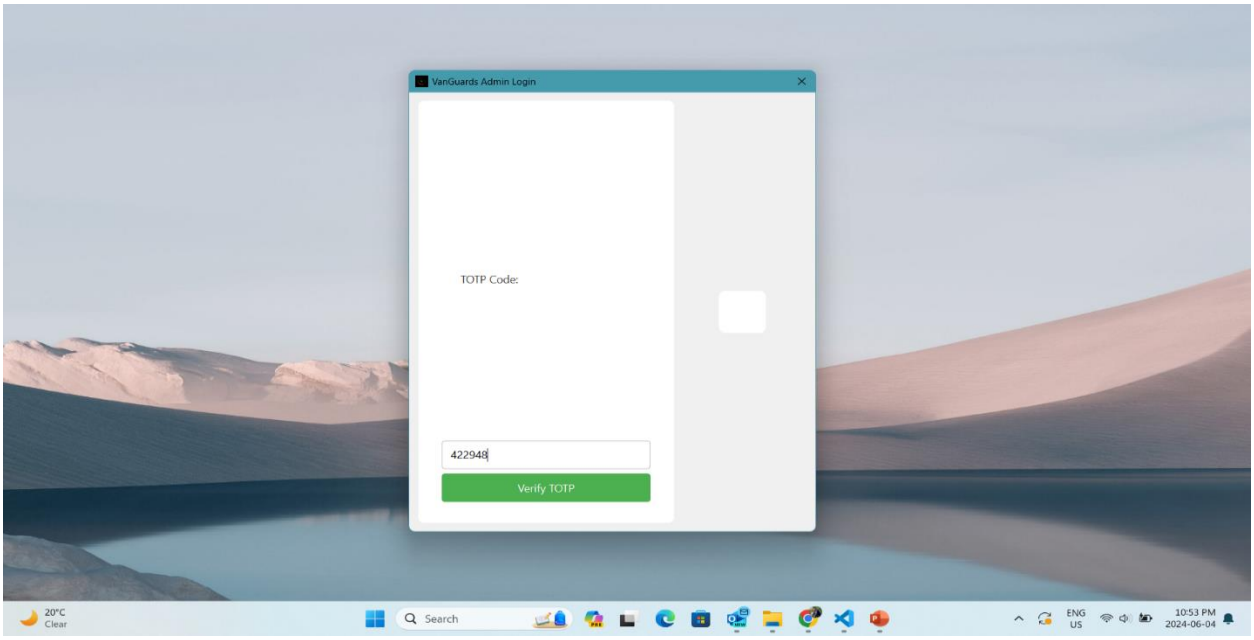


Figure 29 TOTP code for the tool

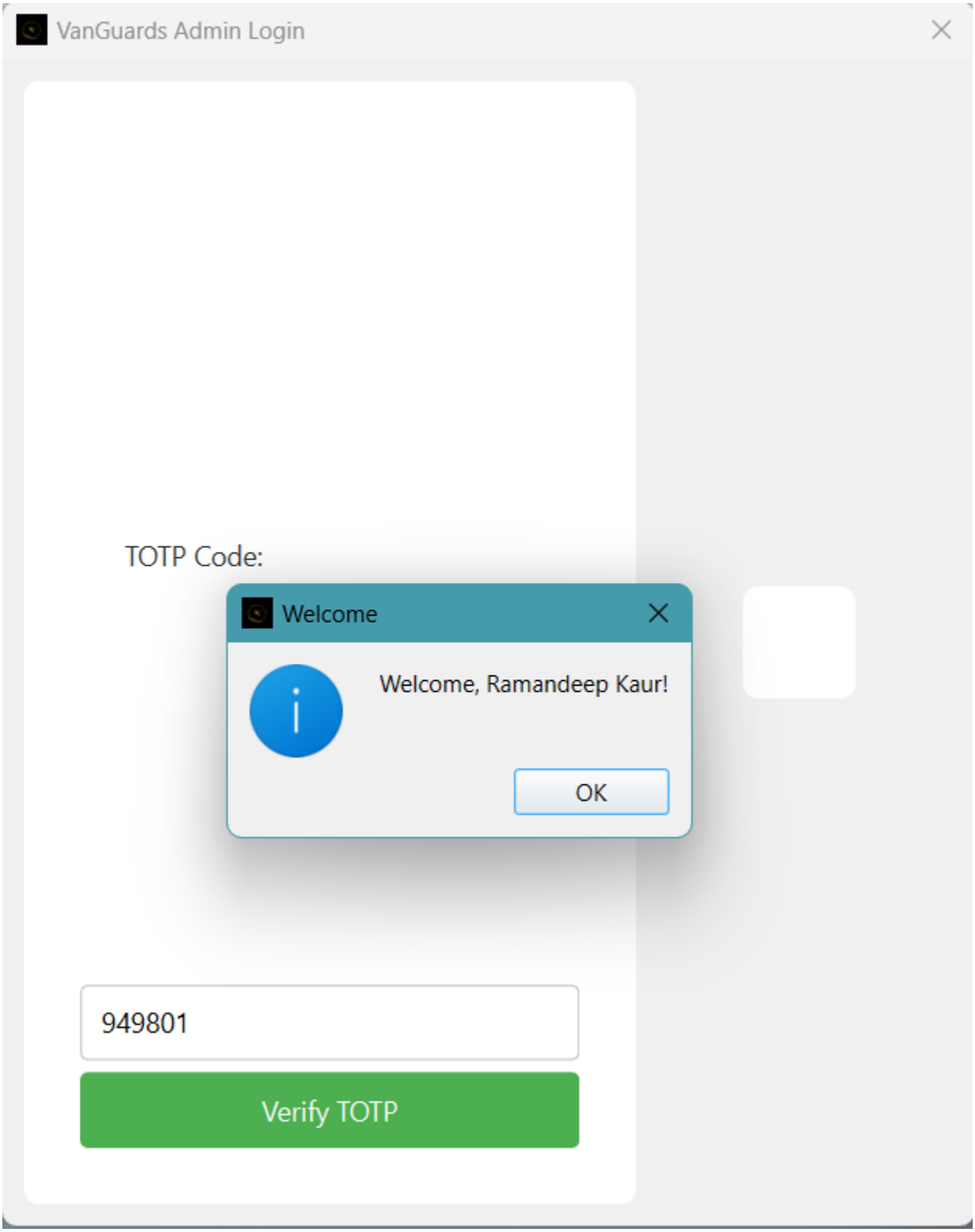


Figure 30 User welcomed

This is the entire process of the user account setup and account login as properly shown in the screenshots above. The user registers their account first through the username, email, and password. Upon creation user should scan the QR to obtain the TOTP and after verifying the code the user is logged in and the tool is their to explore.

Objective	To test the working of the account creation and validation code.
Expected result	The account will be created and the code will be sent to the email.
Actual result	As the steps were followed properly and the respective fields were filled, the account was successfully created and the verification code was sent to the email.
Conclusion	The test was successful.

5.1.9 Scan report

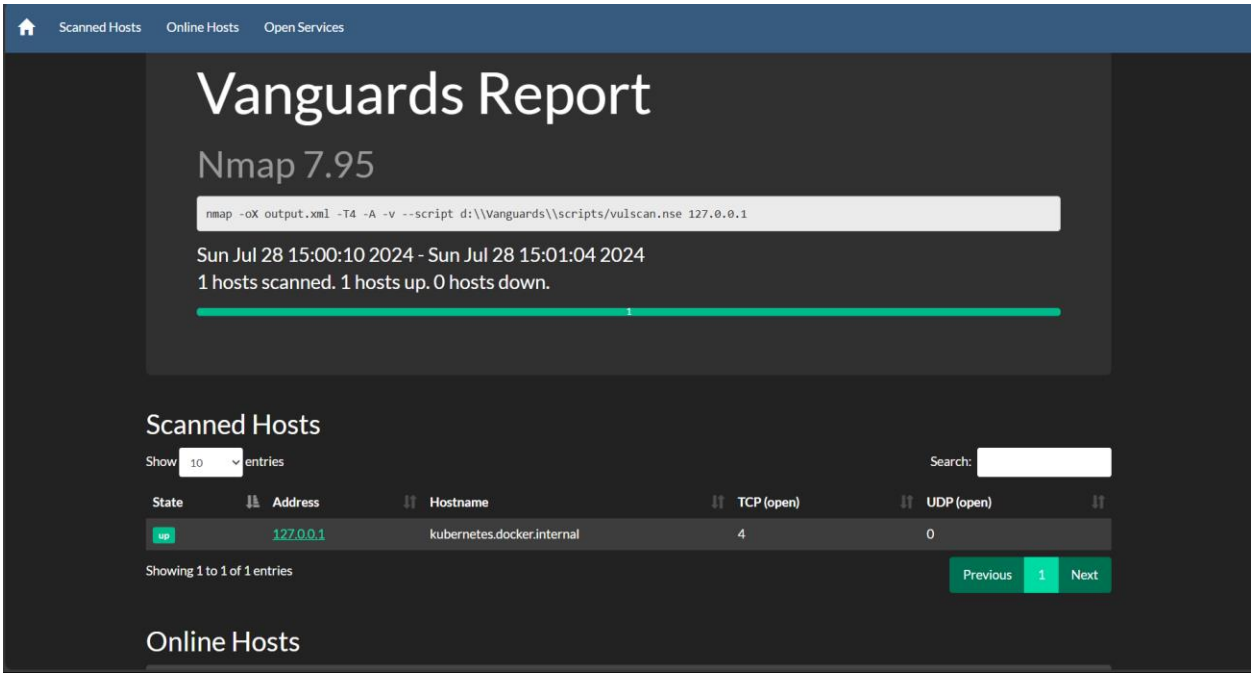


Figure 31 Scan report 1

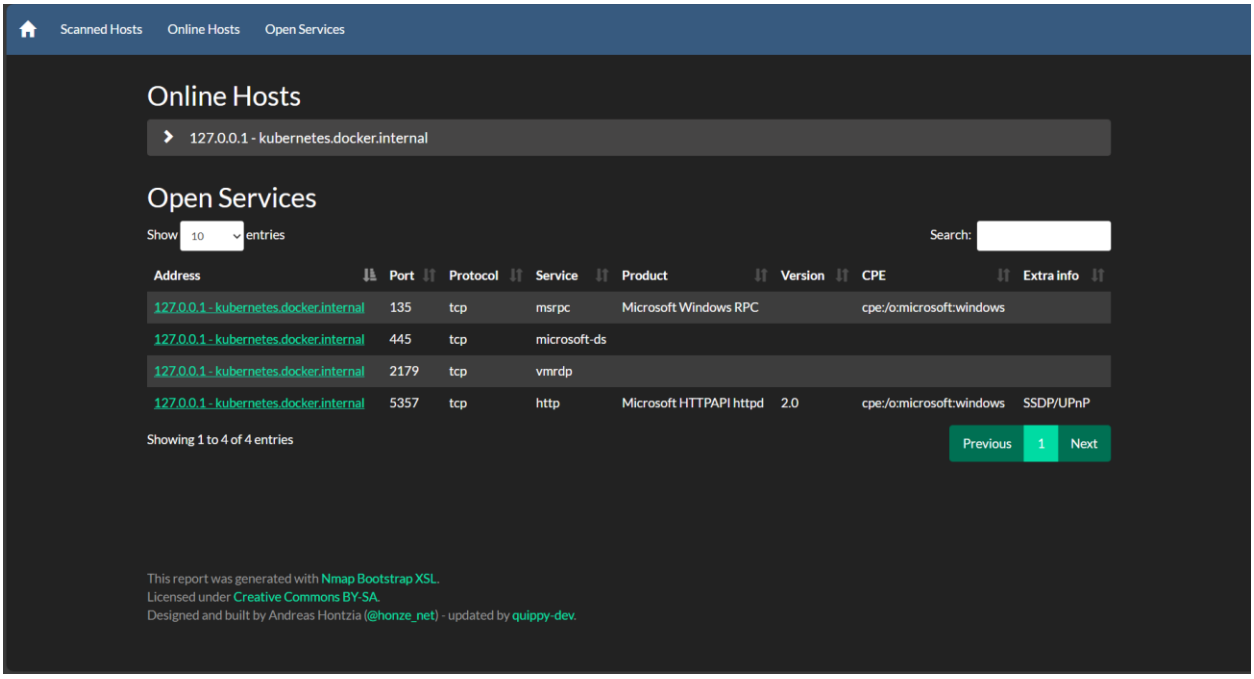


Figure 32 Scan report 2

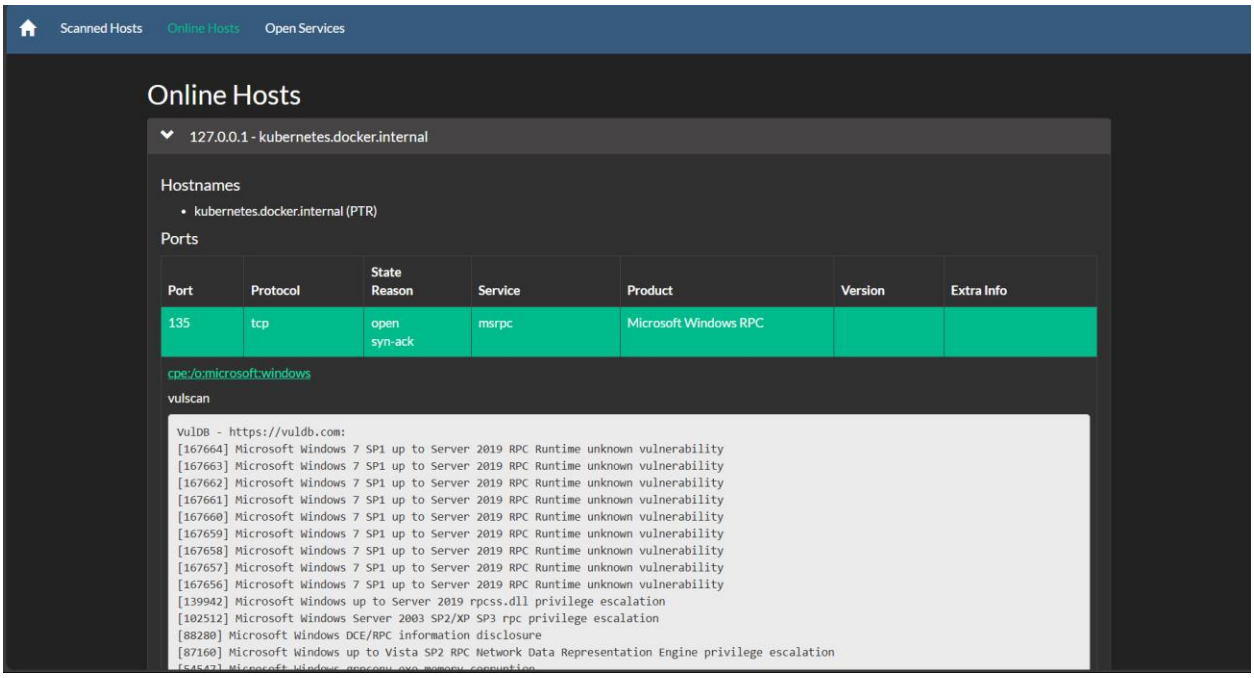


Figure 33 Scan report 3

This is a comprehensive scan report generated by our scanner which is clear and precise in the data and very user-friendly to navigate through the report. The Vulnerability Vanguards utility successfully ran a network scan with Nmap version 7.95 against the IP address 127.0.0.1. The scan was run with enhanced settings, such as aggressive timing (-T4), verbose output (-A -v), and a vulnerability scanning script (vulscan.nse).

The scan was successful, illustrating the usefulness of the Vulnerability Vanguards program in finding active hosts and their open ports. The full output contains the precise command executed, the scan's period, and a visual representation of its progress.

This thorough analysis allows users to understand the current network state, identify potential vulnerabilities, and take the required steps to improve network security. Future scans can be planned to guarantee that network security is continuously monitored and upgraded.

5.2 Coding standards

Achieving excellent code quality and dependability is a key goal in software development. This aim is crucial for software functionality, performance, and maintaining the development process's integrity. As a result, regular unit testing and code reviews become necessary operations.

During code reviews, developers other than the original author conduct a systematic examination of the code. This technique aims to identify and correct problems, improve code readability, and ensure compliance with coding standards. Having many reviewers detect potential issues early on can prevent faults from spreading later in the development process. Code reviews promote knowledge sharing and collaboration among team members, fostering best practices and collective code ownership.

Unit testing isolates individual software modules or components from the overall program. These automated tests ensure that all units function properly. Unit testing is the first line of defense against bugs, ensuring that changes to the code do not introduce new issues. Unit tests provide a safety net for continuous integration and deployment, proving the operation of individual components and providing developers confidence to make changes.

Incorporating unit testing and code reviews into the development process offers numerous benefits. This ensures long-term codebase maintenance, enhances software stability and performance, and reduces the risk of problems in production. These procedures support the development team's culture of accountability and continuous improvement.

This paper discusses the benefits of unit testing and routine code reviews, including their techniques, advantages, and best practices. This paper aims to emphasize the significance of quality assurance methods in software development and provide practical insights for teams seeking high-quality solutions.

5.2.1 Reviews and Feedback

To ensure optimal quality and usefulness in the Vanguards project, we have created a rigorous approach for gathering and integrating feedback. Analyzing these methods is crucial for maintaining project features, usability, and performance.

Internal Code Reviews

Our development team conducts internal code reviews on a regular basis to maintain high quality standards. Colleagues review each code sample and provide feedback on organization, coding guidelines, errors, and optimization potential. These evaluations promote team best practices and help identify problems early on. Collaborative code reviews promote knowledge exchange and provide opportunities for less experienced engineers to learn from more experienced colleagues.

User Feedback

Because our end customers are critical to the project's success, we regularly solicit their feedback. We obtain valuable insights about user experiences by providing beta versions of the Vanguard app to a select number of users. Users provide feedback on the app's usefulness, user interface, speed, and any issues they encounter. We collect feedback using surveys, in-person interviews, and an integrated feedback mechanism in the application. We collected user input via Google Forms to make necessary adjustments as shown in the findings section above.

Continuous Improvement

Feedback from external users and internal reviews informs future development cycles. Integrating feedback into our agile development methodology allows us to modify the Vanguard application in small steps. Regular updates and patches are published based on user feedback to address bugs and enhance functionality. This iterative process improves user involvement, contentment, and application quality.

Review Meetings

Periodic review sessions are held to plan the next steps and discuss any comments received. These sessions are attended by important stakeholders such as project managers, developers, and user reps. The team prioritizes projects for upcoming sprints, evaluates proposed modifications, and analyzes feedback from meetings.

Transparency and Documentation

We ensure that every mechanism for reviews and feedback is thoroughly documented. We preserve detailed records for bug reports, feature requests, user comments, and code reviews. This documentation promotes transparency and consistency throughout the project's lifetime, making it a valuable resource for ongoing and future development. The Vanguard project's success relies heavily on detailed reviews and comments. Actively connecting with external users and internal team members allows us to continuously improve and ensure the application meets the highest quality and dependability criteria.

5.3 Research impact on the technology

5.3.1 Foundational Research

Our first research provided the foundations for the entire project by analyzing the existing landscape of cybersecurity threats and the most effective ways to address these weaknesses. This study emphasized the necessity for a comprehensive vulnerability scanner that could detect a wide range of threats while also adapting to new and emerging security concerns. By incorporating information from industry reports, academic papers, and expert interviews, we ensured that our tool was founded on a strong grasp of the most recent cybersecurity trends and best practices.

5.3.2 Technology Integration

The insights gathered from our research were critical in influencing our technology decisions during development. Understanding the value of automated and real-time scanning, for example, prompted us to create features that continuously check for vulnerabilities without the need for manual involvement. Furthermore, a study into the benefits of secure communication and data encryption had a direct impact on how we integrated these technologies into our application, guaranteeing that critical information is always protected.

5.3.3 Enhanced Security Measures

One of the important findings from our research was the significance of multi-layered security measures. This knowledge prompted us to include features like Time-based One-Time Passwords (TOTP) for user authentication and encrypted email notifications for scan results. These modifications were intended to give an extra degree of security, protecting both users and the data they handle.

5.4 Findings Impact on Technology

5.4.1 User-Centric Design

The data from our user surveys and feedback sessions helped shape the vulnerability detection tool's design. Users clearly preferred a graphical user interface (GUI) that was intuitive and easy to traverse. In response, we created a modern, soft-colored GUI that promotes usability and accessibility, guaranteeing that both novice and experienced users can operate the tool effectively.

5.4.2 Functional Enhancements

We discovered numerous critical functionalities that users considered essential through repeated testing and user input. This included automatic scan scheduling, detailed reporting, and real-time notifications. By implementing these functionalities, we improved the tool's capacity to give timely and actionable insights, allowing users to proactively address system risks.

5.4.3 Security Integration

Our findings also highlighted the necessity for strong security measures within the tool. Users emphasized the significance of secure data transfer and storage, which prompted us to integrate encrypted communication channels and secure storage solutions. These safeguards not only secure the data's integrity, but also increase user confidence in the tool's dependability and security.

5.5 Survey Impact on Technology

5.5.1 Understanding User Awareness

The poll findings revealed that our users have a high level of cybersecurity awareness and understanding. This finding justified the inclusion of precise technical information in our scan results, which cater to consumers with a solid basic understanding of cybersecurity principles. By providing in-depth analysis and explanations, we guaranteed that our product met the needs of a knowledgeable audience.

5.5.2 Preferences for Interface Design

The study revealed a high preference for an easy-to-use GUI. This input immediately affected our design decisions, resulting in a user-friendly interface with intuitive navigation and clear visual elements. In addition, we incorporated comprehensive help and documentation to help users understand and successfully use all of the tool's functions.

5.5.3 Usage Preferences

Understanding that a large number of users prefer to run the vulnerability scanner on their own PCs influenced the creation of a standalone application. This desire for self-use underlined the need for a tool that is simple to install, configure, and use independently. As a result, we concentrated on providing a seamless user experience that allows people to manage their own cybersecurity without the need for expert assistance.

5.5.4 Consultation and Support

According to the report, some consumers prefer to engage with cybersecurity businesses for professional counsel. To satisfy this choice, we created comprehensive user manuals, troubleshooting guides, and professional support services. These resources ensure that users, regardless of technical competence, receive the essential support and instruction to effectively use the product.

The detailed study, findings, and survey responses had a significant impact on the development of our vulnerability scanning tool. We designed a product that satisfies our users' different needs by combining strong security features, user-friendly design, and comprehensive support resources. This discussion focuses on the iterative and user-centric approach we took to ensure that the end product not only meets technical criteria but also provides a seamless and secure user experience. We hope to increase the tool's capabilities and keep it relevant in the ever-changing world of cybersecurity by providing constant feedback and upgrades.

6. Conclusion

The Vulnerability Vanguards project sought to create a powerful, user-friendly vulnerability scanning solution that satisfies modern cybersecurity requirements. Throughout the project, significant progress was made to improve the tool's functionality, user interface, and overall performance.

Project Achievements

- Comprehensive Scanning Capabilities: The program works seamlessly with Nmap to run detailed network scans that uncover potential vulnerabilities across multiple devices and systems. This allows users to fix security issues before they are exploited.
- User-Friendly Interface: Using PyQt6, the team created a modern, intuitive GUI that facilitates user interaction with the tool. The design decisions emphasized soothing colors and stunning toolbars and buttons, resulting in an aesthetically attractive experience without sacrificing utility.
- Automated Scheduling and Notifications: The crontab scheduling function enables users to automate regular scans, assuring ongoing security monitoring. Additionally, the email notification system, which is connected with SMTP, warns users when the scan is complete, delivering the scanned report and necessary facts in a professional manner.
- Scalability and Flexibility: The tool is meant to be scalable, allowing for future expansions and integrations. Its adaptable architecture enables for simple updates and alterations, ensuring that it meets increasing cybersecurity standards.

Lessons Learned

During the development phase, some significant lessons were learned.

- Importance of User-Centric Design: Emphasizing the end-user experience was critical to making the tool accessible and effective.
- Challenges with Secure Email Integration: Ensuring safe email operation necessitated close attention to configuration details and security standards.
- Continuous testing and iterative development were critical for discovering and correcting bugs, optimizing performance, and increasing overall reliability.

Future Directions

While the current version of Vulnerability Vanguards is extremely functional, there are various areas for further enhancement.

- Advanced Reporting Features: Improving reporting capabilities by including more extensive analyses and scan result visualization.
- Integration with Additional Security products: Enhancing the tool's capabilities by integrating with other security products and platforms to provide a more comprehensive security solution.
- User input Mechanisms: Adding features that allow users to offer input directly within the product, hence guiding future developments based on real-world use.

Final Thoughts

The Vulnerability Vanguards project has successfully created a strong, user-friendly vulnerability scanning tool that meets the critical requirements of current cybersecurity. By combining thorough scanning capabilities, a sleek and straightforward UI, and robust automation tools, the tool is a tremendous asset for users looking to keep their networks secure and resilient. The project's success reflects the meticulous planning, thorough execution, and continual improvement efforts that went into its creation.

7. Recommendation

1. Advanced Reporting and Visualization

- Graphical Reports: Include charts, graphs, and other visual features in scan reports to provide a better understanding of the vulnerabilities discovered.
- Detailed Vulnerability Analysis: Provide deeper information into each vulnerability, such as potential impact, severity levels, and remedy processes.

2. Integration with Other Security Tools

- SIEM Integration: Allow for integration with Security Information and Event Management (SIEM) systems to provide a more comprehensive security perspective and correlate vulnerability data with other security events.
- Threat Intelligence Feeds: Integrate with threat intelligence systems to improve vulnerability detection accuracy and provide context for new attacks.

3. User Experience and Accessibility

- Multi-Language Support: Enable support for multiple languages to make the tool more accessible to a wider audience.
- Mobile Application: Create a companion mobile app that allows users to monitor scans, receive alerts, and see reports while on the go.

4. Performance Optimization

- Scalability Enhancements: Increase the tool's performance to handle larger networks and more extensive scans.
- Resource Management: Optimize the tool's resource utilization to reduce the impact on the host system during scanning.

8. References

Awan, A. A. (2023). Setting Up VSCode For Python: A Complete Guide. *datacamp*.

Awati, R. (2023). Nessus. *techtarget*.

Clark, H. (2024). The Software Development Life Cycle (SDLC): 7 Phases and 5 Models. *theproductmanager*.

Ezzat, M. (2020). Writing a vulnerability scanner using python. *github*.

IBM. (2023). What is cybersecurity? *IBM*.

IN-COM. (2024). What is a Flowchart in Software Development? Understanding Process Flow Diagrams & Maps. *IN-COM*.

Kime, C. (2023). How To Use Nmap for Vulnerability Scanning: Complete Tutorial. *esecurityplanet*.

Math, F. (2024). What Makes Good UI Design? *Fuzzy Math*.

Nikolaieva, A. (2024). Software Development Process: Definition, Methodologies and Key Steps. *up tech*.

occupytheweb. (2014). Advanced Nmap for Reconnaissance. *wonder how to*.

Oracle. (2020). What Is a Database? *Oracle*.

Pawlan, D. (2024). What Is Python Development? The Complete Guide For 2024. *aloa*.

Pham, C. (2023). Vulnerability Scanning: What It Is & Why It's Important for Security and Compliance. *secureframe*.

Rouse, M. (2021). Graphical User Interface. *techopedia*.

Shilpa. (2024). Architectural Design in Software Engineering. *artoftesting*.

Singh, A. (2024). Spiral Model in Software Engineering. *shiksha*.

synopsys. (2024). Cyber Security. *synopsys*.

Tunggal, A. T. (2024). What is a Vulnerability? *UpGuard*.

Walkowski, D. (2019). What Is the CIA Triad. *F5 labs*.

Walkowski, D. (2019). What Is the CIA Triad? *F5 labs*.