**PhishNet**

****

Updated on 11/25 Version 2.0

Aarshia Hukmani

Gianmarco Bonventre

AJ Whitman

Julia Pallazola

**Table of Contents**

1. **Introduction**...................................................................................................3

1.1 Purpose

1.2 Document Conventions

1.3 Scope

1.4 Intended Audience

1.5 Definitions of Words, Acronyms & Abbreviations

1.6 Software License

1. **System Overview & Requirements**.................................................................................................5

2.1 Product Perspective

2.2 Product Functions

2.3 Operating Environment

2.4 Design & Implementation Constraints

2.5 Assumptions & Dependencies

1. **External Interface Requirements**…………………………………………7

3.1 User Interfaces

3.2 Hardware Interfaces

3.3 Software Interfaces

3.4 Communication Protocols

1. **Functional Requirements**……………..…………………………………15

4.1 Scanning Methods

4.2 Real-Time Protection

4.3 Scheduled Scanning

4.4 Quarantine & Removal

4.5 Threat History & Reports

4.6 Updating Virus Bytecode Database

4.7 Data Management & Storage

1. **Non-Functional Requirements**………..…………………………………17

5.1 Performance Requirements

5.2 Security Requirements

5.3 Software Quality Attributes

5.4 Reliability & Availability

5.5 Accuracy of Malware Cleaner

1. **References**……………………..………..…………………………………20
2. **Introduction**

**1.1 Purpose**

The purpose of this document is to provide a comprehensive overview of the requirements for the Cybersecurity Anti-Malware Scanning Application (PhishNet). It details the core functionalities, constraints, and system interfaces of PhishNet, addressing key aspects crucial for current and future developers, users, and maintainers. This document serves as a reference for understanding the project's scope, design considerations, and operational guidelines, and it is subject to updates as the project evolves.

**1.2 Document Conventions**

For any future updates to this document, it must adhere to the following:

* Times New Roman font
* 14pt bold font for headings
* 12pt bold font for subheadings
* 12pt non-bolded font for any other text
* Double-spaced font, unless in the case of lists, which use 1.15 spaced font
* Requirements will be tagged in the format [LETTER-NUMBER] and will use the italicized font (i.e. *[A-1]*)

**1.3 Scope**

PhishNet is a supplemental malware scanning tool that allows users to scan locations within their Ubuntu-based computer for malicious software using the definitions within the ClamAV open-source project. Users can scan the system as a whole, frequently infected directories (hereafter referred to as a “Quick Scan”), or a set of user-specified directories. When the user selects to scan a directory, all of the files within that directory will be passed through the scanner function. Users can schedule a quick scan or system scan to be run every day, week, or month at a chosen time to ensure their system is protected. PhishNet provides malware detection and removal for the Ubuntu operating system. Additional features include definition updates, quarantine options, and detailed threat reports.

**1.4 Intended Audience**

* Everyday Users: Individuals who use the Ubuntu operating system and seek additional security for their personal computers. These users are looking to enhance their confidence in the overall security of their system by incorporating an extra malware scanner.
* Advanced Users: Individuals who are more experienced with Ubuntu operating systems and require an advanced security solution to address malware on potentially infected systems. This group includes users who might need additional protection or specialized tools to manage and remove malware.
* Technical Enthusiasts and Developers: Advanced computer users interested in exploring the open-source ClamAV project from a different perspective. This audience includes those who wish to understand and potentially contribute to implementing and developing the ClamAV library through practical application in PhishNet.

**1.5 Definitions of Words, Acronyms & Abbreviations**

| **Words, Acronyms & Abbreviations** | **Definitions** |
| --- | --- |
| Git | A distributed version-control system for tracking changes in source code during software development. It is designed to coordinate work among programmers, but it can also be used to track changes in any set of files. |
| GitHub | A web-based platform for version control and collaboration using Git. |
| GUI | Graphical User Interface |
| Open-Source | Refers to software with source code that is freely available for modification and distribution. |
| PhishNet | A lightweight antivirus/anti-malware tool available on Ubuntu 20.04. With a multi-functional scanning tool and an easily accessible history of scan results. |
| Ubuntu | An open-source Linux-based operating system. |

**1.6 Software License**

The license used for PhishNet is the GNU General Public License, Version 2, in compliance with the licensure requirements of the ClamAV library (LibClamAV). Any third-party frameworks used in PhishNet are free-licensed and open-sourced, as the LibClamAV license requires.

1. **System Overview & Requirements**

**2.1 Product Perspective**

PhishNet is an application designed for users seeking to enhance the security of their

systems. In addition, PhishNet aims to provide a system designed to handle

various use cases or scenarios, providing a user experience that reaches as many people

as possible. PhishNet is an open-source project that relies on ClamAV’s database for virus

signatures, which helps those looking to gain insight into implementing ClamAV.

PhishNet is designed for use strictly on Ubuntu 20.04.

**2.2 Product Functions**

PhishNet is designed to let users select and scan specific disks, directories, or files for scanning and analyzing each file to determine if it contains malware. Moreover, the program quarantines the affected files once malicious software is detected. Additionally, PhishNet updates virus signature databases automatically, allowing it to identify newer malware threats. The software records all files placed in quarantine, allowing users to review and analyze past malware detection incidents. PhishNet’s user interface offers a simplified, one-click scanning along with easy-to-use custom scheduling and selection for scans.

**2.3 Operating Environment**

PhishNet only rus non and is available for users on the Ubuntu 20.04 Operating System *[O-1]*.

**2.4 Design & Implementation Constraints**

PhishNet is implemented using C++ *[D-1]* and integrates ClamAV as its score backend

for malware detection and scanning *[D-2]*. The user interface is also developed in

C++ *[D-3]* and provides a native look and feel that seamlessly integrates with

Ubuntu. The development environment is Visual Studio *[D-4], and* testing

and debugging are done using Oracle VM Virtualbox *[D-5]*.

**2.5 Assumptions & Dependencies**

PhishNet is developed in C++. Users have Ubuntu 20.04 installed and meet the

necessary hardware requirements. The software depends on ClamAV for core virus

signatures and bytecode detection. Regular updates and support from ClamAV open-

source platform is essential for PhishNet’s detection methods. ClamAV has the

following external library dependencies: libcheck, bzip2, zlib, libxml2, libpcre2,

openssl, json-c, libmspack, pthreats-win32, libcurl *[D-6]*.

1. **External Interface Requirements**

**3.1 User Interfaces**

Below are images of the current GUI rough mockup for PhishNet. These are not final mockups and do not exactly reflect the final product's appearance. This section will be updated as the mockups are edited and finalized.

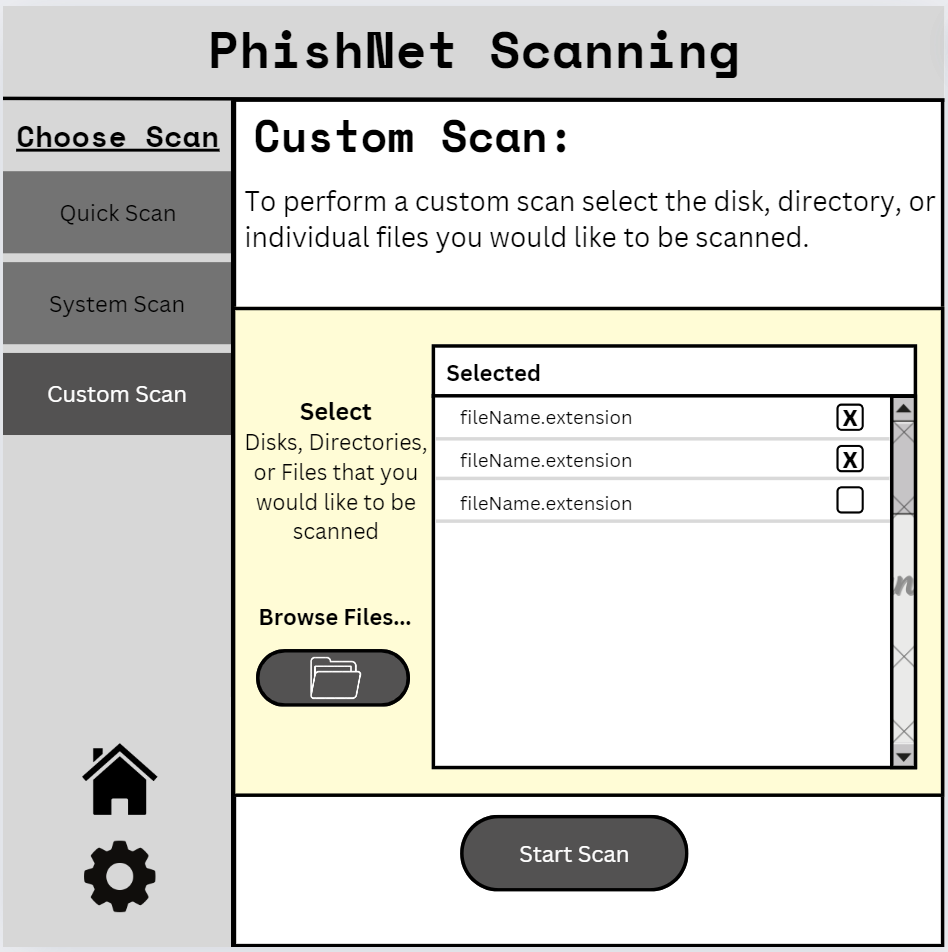
**3.1.1 Homepage Mockup**

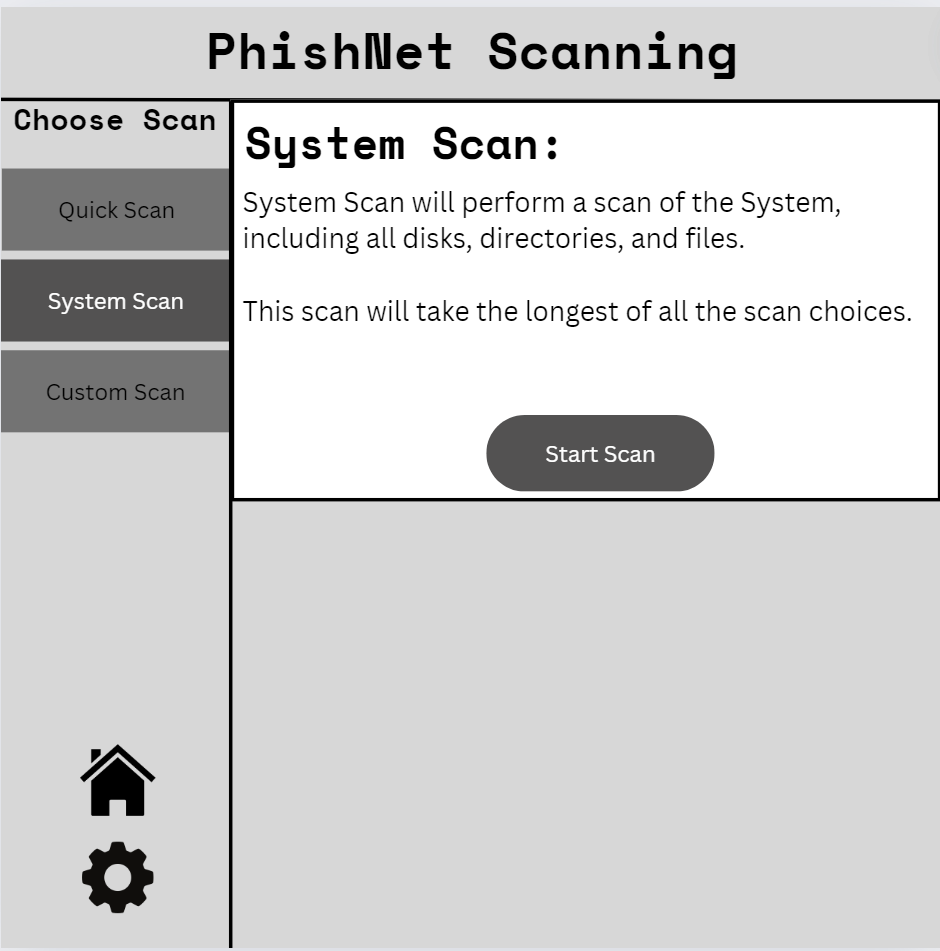
The user will automatically start on this page every time that the PhishNet application is opened after a restart. The user will have a selection of 5 menu option buttons to click as well as a settings icon in the top right corner to navigate to the ‘Settings Page’. The menu selections will be formatted as three centered square buttons on top with an icon in the middle of each, as well as the text below to indicate where the user will be brought. There will also be two larger rectangular buttons centered below the three squares, with icons in the middle of each and text below to indicate where the user will be brought.

****

**3.1.2 Scanning Options Mockup**

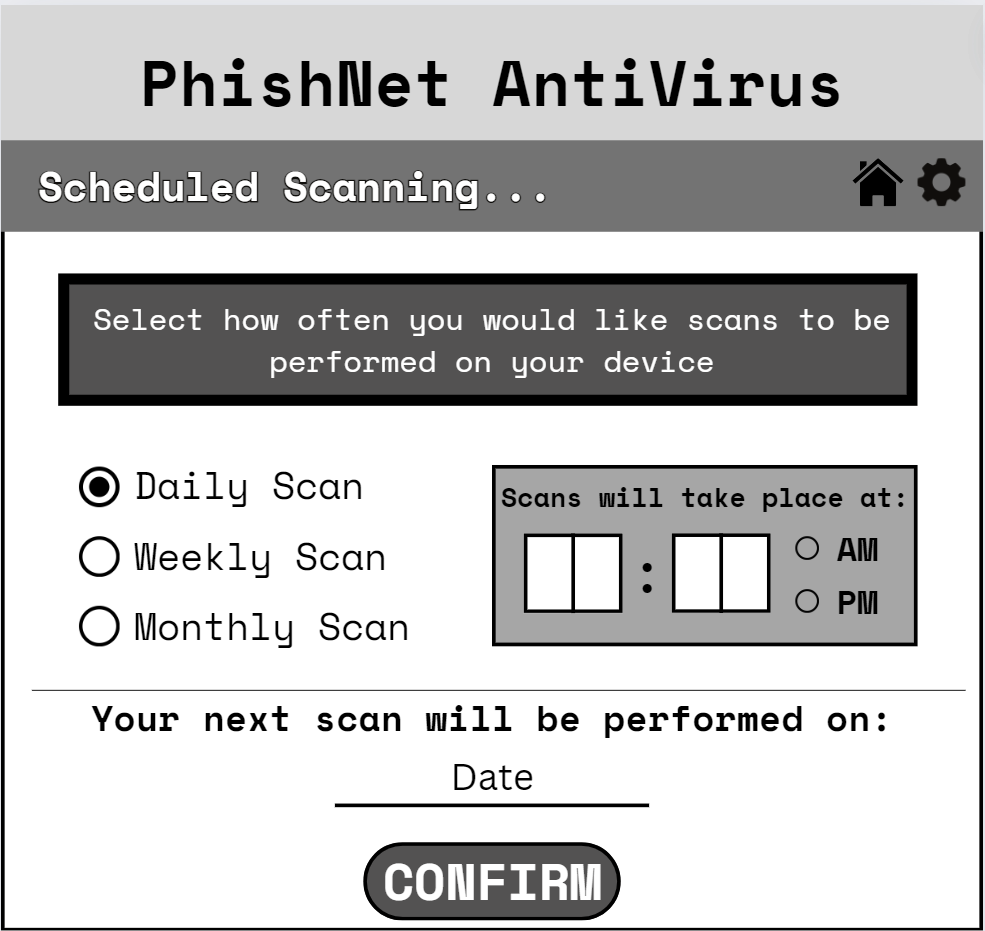
* General Layout:
  + The page is divided into two primary sections: a left sidebar and a main content area. The left sidebar occupies about one-quarter of the page’s width and serves as the navigation area. It contains three rectangular buttons for selecting scan types: “Quick Scan,” “System Scan,” and “Custom Scan.” At the bottom of the sidebar is a home icon followed by a settings icon. When a scan type is selected, its button’s background changes to a darker color and the text turns white, visually indicating the current selection. The remaining three-quarters of the window comprises the main content area, which dynamically updates to display details and options related to the selected scan type.
* Scan Options Pages
  + The “Quick Scan” and “System Scan” pages share a similar design. Each page has a large, bold header at the top, clearly displaying the selected scan type. Below the header is a white description box that outlines the purpose of the scan and the areas or items it will target. At the bottom of the page, a rounded rectangular “Start Scan” button is centered below the description box, allowing the user to initiate the selected scan.
  + The “Custom Scan” page offers more user interaction options. At the top of this page, a large bold header reads “Custom Scan:” followed by a white instruction box that briefly explains the custom scan’s purpose and user instructions for selecting files or directories to scan. Below this is a pale yellow box prompting the user to “Browse Files,” which includes a rounded rectangular button with a folder icon for opening a file browser. Below the file selection prompt is a white “Selected Files” box, displaying a scrollable list of chosen files or directories. Each item in the list appears in its own row and includes a checkbox and a delete button for easy management. At the bottom of the page, a rounded rectangular “Start Scan” button is centered beneath the file selection selection.

** **

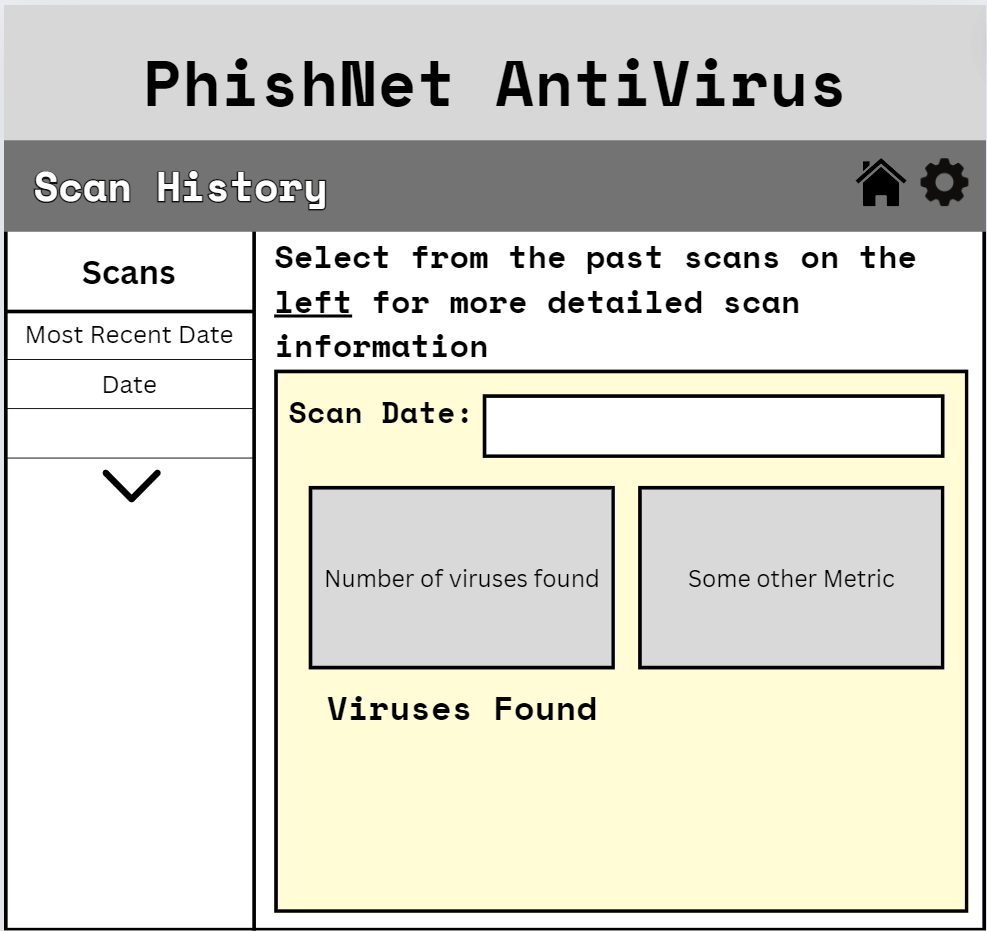
****

**3.1.3 Scheduled Scanning Mockup**

* Header & Navigation
  + The top of the page features a wide gray border acting as the header, which spans the full width of the page. On the left side of this header is bold text reading “Scheduled Scanning.” On the right side of the border, two navigation buttons are present: a home button followed by a settings button. The settings button is positioned to the right of the home button, allowing users to navigate to the Home or Settings pages quickly.
* Main Content Area
  + Beneath the header, the remaining portion of the page consists of a large white box with thin black borders on its left, bottom, and right sides. Centered at the top of this box is a dark gray rectangular section with a thick black border. This box contains centered white text that likely serves as an important label or instruction for the page’s functionality.
* Schedule Selection
  + Below the centered gray box on the left side of the page are three vertically aligned radio buttons. Each button has descriptive text to its right, enabling users to choose their preferred scanning schedule. A thin black horizontal line, spanning most of the page’s width, separates this section from the next. [Not pictured: The user can choose between a quick and system scan to be performed.]
* Next Scheduled Scan
  + Under the thin divider, a bold, centered subheading introduces the next section. Directly beneath this subheading is a small black horizontal line in the center of the screen, which displays text indicating the date of the next scheduled scan. This section ensures users can clearly see when the next scan is set to occur based on their selections.
* Confirmation Button
  + At the bottom of the page, a dark gray rectangular button with rounded edges is centered. The button contains bold white text reading “CONFIRM.” Users click this button to finalize their schedule and save their scanning preferences.



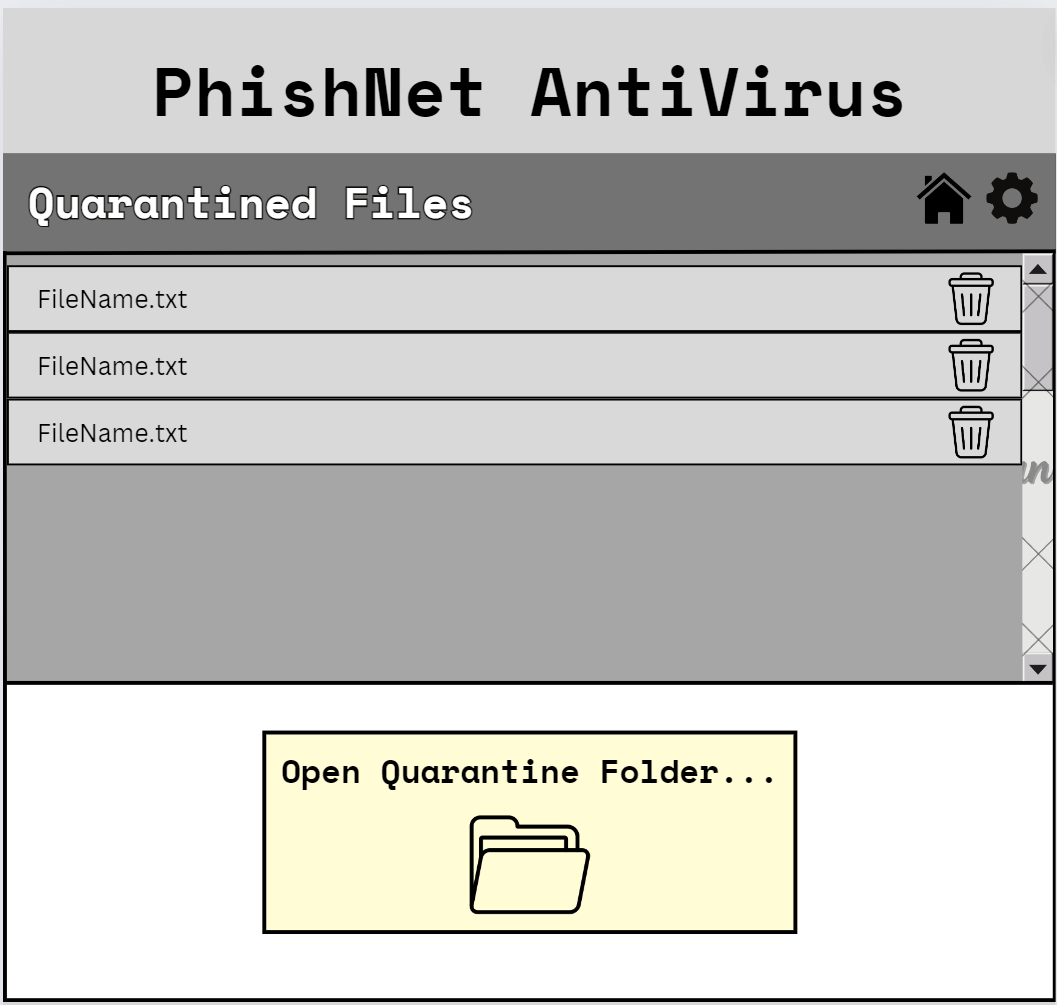
**3.1.4 Scan History Mockup**



* Header & Navigation
  + At the top of the page, a wide gray border spans the width of the screen, serving as the header. Positioned on the left side of this border is the title “Scan History,” formatted as bold text. On the right side of the gray border are two navigation icons: a home button followed by a settings button to the right. These buttons allow the user to quickly navigate to the Home or Settings page.
* Main Content Area
  + Beneath the gray header is a large white box occupying the page's remaining space. This box features thin black borders on its left, bottom, and right edges. Inside, a vertical divider splits the box into two sections. The left section, taking up about one-quarter of the page, displays a bold header labeled “Scans.” Below this header is a horizontal divider, followed by a selectable, scrollable list of scan dates. The list is sorted from the most recent scan at the top to the oldest at the bottom.
* Details Section
  + The right section of the page displays details about the selected scan. At the top left of this section, bolded instructional text provides guidance on how to select a scan. Below this text is a centered pale yellow box with a thin black border. Inside this box, a bolded label reads “Scan Date,” with a smaller white rectangular box to its right. This smaller box displays the date of the selected scan, reflecting the user’s choice from the list on the left.
* Summary
  + Below the “Scan Date: section are two light gray rectangular boxes with thin black borders, aligned horizontally. Each box contains a bold number in the center, accompanied by descriptive text underneath, These boxes are designed to display key scan measures and summaries related to the selected scan, providing users with a quick overview of the scan’s results.

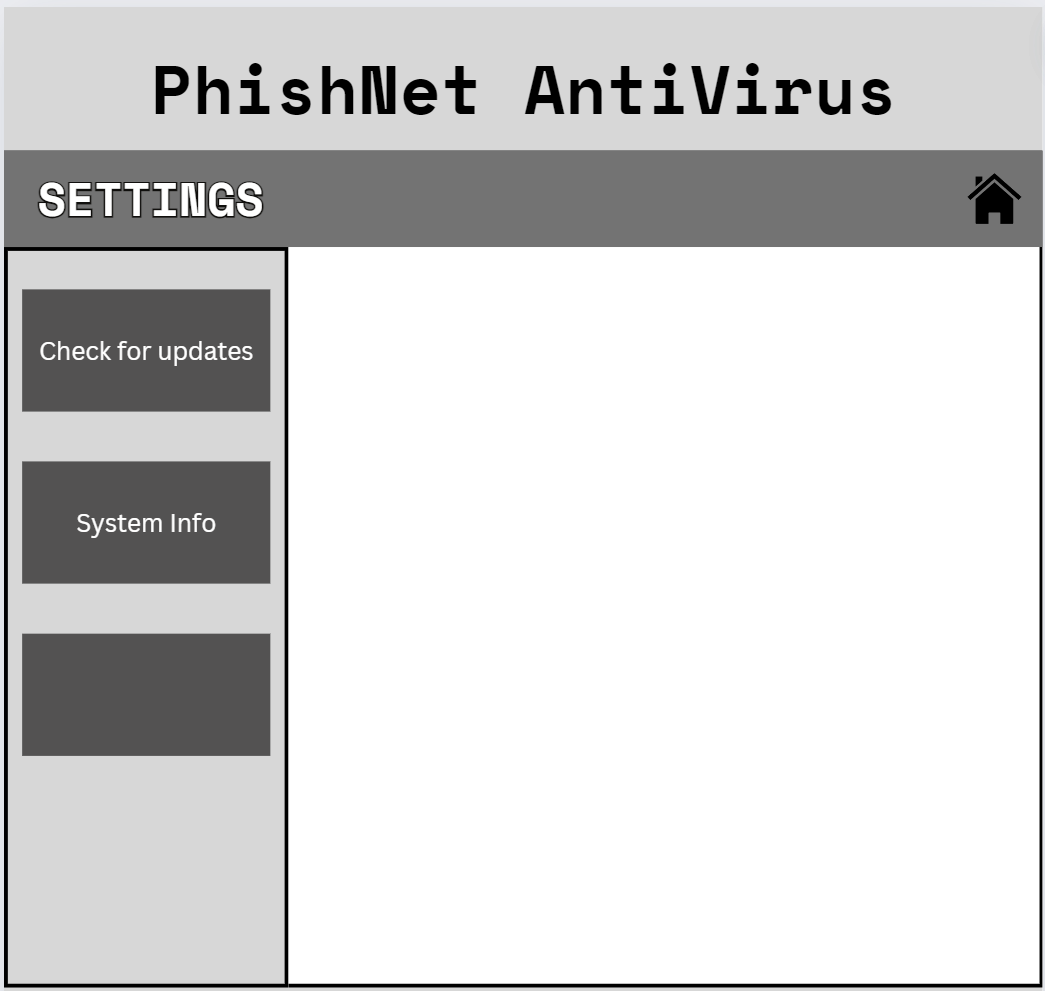
**3.1.5 Quarantined Files Mockup**

* Header & Navigation
  + At the top of the page is a wide gray header spanning the full width of the screen. On the left side of this header is bold text reading “Quarantined Files.” On the right side of the header are two navigation buttons: the home button, followed by the settings button, positioned to the right. These buttons allow users to navigate to the Home or Settings page quickly.
* Main Content Area
  + The rest of the page is occupied by a large white box with thin black borders on its left, bottom, and right edges. This box is divided into two main sections:
  + Quarantined Files List (Top Half): The top half of the page contains a light gray box with a black border that spans the entire width of the page. This section is dedicated to displaying a scrollable list of quarantined files. Each quarantined file is shown within individual, smaller, light gray boxes inside the larger section. These file boxes include a description of the quarantined file, and on their far right are trash icons. The trash icons enable users to delete files from quarantine directly from this list. A scroll bar on the far right of the larger gray box allows users to navigate through the list when it contains more files than can be displayed at once.
  + Quarantine Folder Access (Bottom Half): A pale yellow rectangular box is below the quarantined files list in the bottom-right portion of the page. This box is centered both vertically and horizontally within this section. At the top center of the yellow box, bold text reads “Open Quarantine Folder.” Directly below this text is a centered folder icon. The entire yellow box is a clickable button, which opens the PhishNet directory to view quarantined files using the regular Ubuntu file browser.



**3.1.6 Settings Mockup**

* Header & Navigation
  + At the top of the page, a wide gray border spans the full width of the screen. On the left side of this border is bold, left-aligned text reading “SETTINGS.” On the right side of the header is one navigation button: the home button. These buttons provide easy navigation back to the Home page.
* Main Content Area
  + The remainder of the page consists of a large white box with thin black borders along its left, bottom, and right edges. This area is divided into two primary sections.
  + Navigation Panel: On the left quarter of the page is a light gray panel with a thin black border enclosing it. This section serves as the navigation panel for different settings categories. Within this panel are three equally spaced dark gray rectangular buttons, each labeled to indicate the setting section it will open. These buttons do not span the full width of the left panel, leaving ample space for a clean and balanced layout. Clicking any of these buttons will display the corresponding settings content on the right side of the page.
  + Content Display Area: The right three-quarters of the page has a white background with thin black borders along its right and bottom edges. This section is reserved to display the content of the settings categories selected from the navigation panel. When clicking a button on the left, the corresponding settings section overlays this area. This section’s background and layout are currently under development, leaving room for future enhancements.



**3.2 Hardware Interfaces (Derived from ClamAV Requirements)**

Minimum Hardware Requirements *[O-2]*:

* RAM Requirement: 3 GiB+
* CPU Requirement: 1 CPU at 2.0 Ghz
* Minimum Available Hard Disk Space Requirement: 5 GiB of Free Space in addition to the recommended disk space required for the OS (Ubuntu 20.04).

**3.3 Software Interfaces**

PhishNet requires C++ to run. PhishNet uses C++ to execute the ClamAV backend and the frontend GUI.

**3.4 Communication Protocols**

PhishNet requires an internet connection for the user to update the application. Updates almost always include new virus bytecodes. Updates may include GUI changes along with a bytecode update. PhishNet uses https on port 443 to connect to the internet. The user can not check for updates or be notified of a new update release through the application unless they are connected to the internet. If the user is not connected to the internet, the virus bytecodes remain the same as that of the time of the first installation or the last update made when connected to the internet. A user who does not regularly update PhishNet is at increased risk for infection as the scan function does not have the newest bytecodes to protect against.

1. **Functional Requirements**

**4.1 Scanning Methods**

PhishNet has three different scanning options: Quick, System, and Custom. Quick Scan scans the user-designated main directory that runs the operating system and essential files to the machine. System scans the full file directory of the machine, and Custom allows the user to determine what disks, directories, or individual files to scan *[S-3]*. Each scanning method alerts the user of the location of any files found to be malicious *[S-4]*.

**4.2 Real-Time Protection**

PhishNet uses the ClamAV virus bytecode database to scan files before they are downloaded by cross-searching the file with the database *[P-1]*. When the file is deemed safe, it allows the download, and when it is found to be malicious, the download is halted, and the user is notified *[P-2]*.

**4.3 Scheduled Scanning**

PhishNet uses the Linux clock to allow the user to choose an interval for scheduled scanning from daily, weekly, and monthly selections *[S-5]*. The default selection upon software installation is daily. The scheduled scan is configured to one of two scan types, Quick Scan and System Scan, with Quick Scan being the default *[S-6]*.

**4.4 Quarantine & Removal**

When an existing file on the device is deemed to be malicious through a scan, PhishNet quarantines the file and alerts the user. PhishNet quarantines by first encrypting the file and then moving it to a protected folder, which prevents further execution of the file. The user can choose to remove the quarantined file(s) through the quarantine manager buttons on the UI.

**4.5 Threat History & Reports**

After a scheduled or User-initiated scan, PhishNet provides a report on any malicious files and their locations as file paths. PhishNet also provides a Threat History report, which allows the user to see which directories in the past had malicious files.

**4.6 Updating Virus Bytecode Database**

While connected to the internet, PhishNet connects with ClamAV and prompts the user to perform an update. This updates the virus bytecode database and GUI (if changes are made) *[A-1]*. PhishNet checks for an update before proceeding with any scans to provide the most accurate protection *[A-2]*. A user can also go to the settings page to manually check for updates if the application has not been restarted in a while or a scan has not been performed *[A-3]*.

**4.7 Data Management & Storage**

Through its own file directory, PhishNet stores Threat History and Reports. PhishNet also stores Quarantined files through a Quarantined file directory in which no files can be executed.

1. **Non-Functional Requirements**

**5.1 Performance Requirements**

PhishNet is designed to run efficiently on systems meeting the basic Ubuntu 20.04 requirements: 3 GB or more of RAM, a CPU with at least one core clocked at 2 GHz or higher, and 5 GB of available hard disk space *[O-2]*. As a lightweight antivirus program, PhishNet is not built using large JavaScript front-end frameworks like Electron, ensuring fast performance and a responsive user interface. Users should experience smooth operation without any performance lags or delays when interacting with the software or its GUI *[O-3]*. As the project develops, optimization, efficiency, and potentially reducing system requirements in future versions are kept in mind. Performance benchmarks include minimal latency during user interactions, real-time malware detection, and low resource usage during background scans.

**5.2 Security Requirements**

PhishNet requires administrative privileges to perform critical operations, including accessing, quarantining, and deleting system files *[O-4]*. These permissions ensure that the application can manage malicious software, even in protected directories. Security measures include:

**Administrative Acces**s: Only users with administrative privileges can review and manage quarantined files, preventing unauthorized or accidental deletions.

**Log Access Control**: Access to system logs, scan reports, and quarantine history are restricted to prevent tampering and ensure data integrity.

**File Encryption**: Quarantined files are securely encrypted to prevent unauthorized access or execution.

**5.3 Software Quality Attributes**

PhishNet is designed to focus on usability, maintainability, and scalability to accommodate novice and advanced users. Key quality attributes include:

* **Usability**: The software interface is intuitive and user-friendly, with clear prompts, easy-to-understand navigation, and automated workflows. Users can initiate scans, schedule tasks, and manage threats with minimal effort *[Q-1]*.
* **Maintainability**: The codebase and system architecture are modular, allowing for easy updates, bug fixes, and enhancements. Regular updates to the virus bytecode database ensure continued accuracy and relevancy *[Q-2]*.
* **Scalability**: The software is built to scale with system resources, ensuring consistent performance even with large file volumes or intensive scanning operations *[Q-3]*.

**5.4 Reliability & Availability**

PhishNet must maintain high availability and reliability during operation. The software is expected to function consistently under various system loads and recover gracefully from errors. Key reliability aspects include:

* **Fault Tolerance**: In the event of a system failure or unexpected shutdown, the software resumes the last known state and continues operations without data loss *[R-1]*.
* **Background Operation**: PhishNet runs scheduled and real-time scans in the background without causing performance degradation or requiring system restarts *[R-2]*.
* **System Uptime**: The application is designed to be available 99.9% of the time, ensuring users can rely on consistent protection *[R-3]*.

**5.5 Accuracy of Malware Cleaner**

PhishNet, utilizing ClamAV's signature detection engine, aims to deliver high accuracy in identifying and neutralizing malware. Regular updates to the virus signature database enhance detection accuracy over time:

**Detection Accuracy:** PhishNet achieves approximately 85% accuracy in identifying known threats, improving as ClamAV’s database evolves *[E-1]*.

**Minimization of False Positives:** Continuous refinement of detection algorithms reduces the likelihood of false positives, enhancing user trust and system stability *[E-2]*.

**Ongoing Updates:** The software periodically updates virus definitions and threat detection parameters to maintain protection against emerging threats *[E-3]*.

1. **References**
2. ClamAV Documentation. (2024). ClamAV Virus Scanning Engine. Available at: <https://www.clamav.net/documents>
3. GNU Project. (2024). GNU General Public License, Version 2. Available at: <https://www.gnu.org/licenses/old-licenses/gpl-2.0.en.html>
4. GNU Operating System (2008). GNU Free Documentation License, Version 1.3. Available at: <https://www.gnu.org/licenses/fdl-1.3.html>
5. Ubuntu. (2024). Ubuntu Server System Requirements. Available at: <https://ubuntu.com/server/docs/system-requirements>
6. C++. (2024). C++ Standard Documentation. Available at: <https://en.cppreference.com/w/>
7. Visual Studio Code Documentation. (2024). Visual Studio Code Development Environment. Available at: <https://code.visualstudio.com/docs>
8. Open Source Initiative. (2024). Open Source Definition. Available at: <https://opensource.org/osd>
9. QT Documentation (2024). QT Creator Documentation. Available at: <https://doc.qt.io/qtcreator/>