ABSTRACT SOFT COPY

INTRODUCTION:

As we humans progress and get more and more technologically advanced, there is a threat at rise that has always been there since the birth of technology. Cybercrime. To combat this threat, over the years, antivirus software was brought into existence. It scans for vulnerabilities. Malicious files. Changes in file integrity, web protection and may more protection from cybercrimes and ensuring cyber safety. It is a rapidly growing piece of technology that is very crucial as we progress more and more towards advanced technology.

SCOPE OF THE PROJECT:

The criteria of the project is given below:

* The software will be developed using VB.NET for Windows operating system.
* It will focus on detecting malware through MD5 hash comparison.
* Target users are individuals or businesses with low end computers.
* It can scan all types of files including executables, documents and archives.
* There will be a singular file scan option and a directory scan option.
* Efficient algorithms and libraries will be used to carry out the hash calculations and hash comparisons.
* Users can see the scan history and scan log while the scan is running.
* Database will be implemented, maintained and secured for user data storage and hash storage.
* Testing will be carried out to ensure the application stability and efficiency.
* Detailed documentation will be available for support and guidance.

The project, however, does not accomplish the following functions:

* The software will not use additional hash algorithms beyond MD5.
* The software will not include additional security like firewall or password vault.
* The project will not cover support for operating systems other than Windows.

PROBLEM DEFINITION:

Antivirus softwares by default, runs in the background consuming significant amount of system resources, especially in a widely use operating system like windows. It limits the resources that you have and hinders you from having the fast and responsive computer experience for people having lower end systems. To combat this, I have found a way to make a virus scanning software that is extremely light weight which doesn’t use even a fraction of regular antivirus softwares and can run offline. This will give you a faster way to checking a folder or a singular file you need and tells you if its malicious or not.

OBJECTIVE OF THE PROJECT:

The objective of this project is to create a virus scanning application that allows you to scan a particular directory or a single file that may appear malicious to you or just to make sure you have a safe directory without having to install antiviruses that can spam you with bloat and use up your system resources to keep you covered, that too, to a certain degree, not perfect. This piece of software enables you to quickly run a scan, check for results and close it so that it doesn’t run in the background and use up all the precious system resources. This can be achieved using various algorithms but due to the limitations of vb.net, the best at hand is MD5 hash comparison which compares the MD5 hash of the file with known malware hashes.

HARDWARE SPECIFICATION:

Minimum requirements:

* Operating system: Windows 10 Home
* Processor: Intel Celeron – N3060 (up to 2.40 GHz)
* Memory: 4 GB RAM
* Storage: 512 GB HDD
* Graphics: Intel HD
* Connectivity: WiFi / ETHERNET

Testing System Specifications:

* Operating system: Windows 11 23H2
* Processor: Ryzen 7 5800H (up to 4.4 GHz)
* Memory: 16 GB RAM
* Storage: 512 GB SSD
* Graphics: Nvidia Geforce RTX 3050
* Connectivity: WiFi 6

SOFTWARE SPECIFICATION:

Programming Language support:

* VB.NET – Software code
* SQL – Database management
* HTML – Authentication redirect (if required)

Software Required:

* Visual Studio 2022
* Visual Studio Code

EXISTING SYSTEM:

There are may Antivirus detection systems in the market. Some of the commonly used detection algorithms are:

* Heuristic-based detection: most common detection algorithm that uses an algorithm to differentiate the signature of known viruses against a potential threat.
* Signature-based detection: it searched for specific digital code of a virus and once it finds it, it deletes or quarantines the risk.
* Behavioral detection: it uses an algorithm to analyze what a software does and its general behavior is compared to know virus behaviors to check if its malicious or not.
* Machine learning: it uses a plethora of algorithms, finds the pattern upon which a malware acts and can detect potential threats, new and old since it scours the internet for new vulnerabilities and keeps itself updated.

Every algorithm listed here has the drawback of relying on an active internet connection and the extensive system resource usage, paired with the elevated execution being susceptible to crashing the system as a whole if the core file of the software is corrupted.

FEASIBILITY STUDY:

A feasibility study is a critical step in assessing the viability of a project. It typically includes an analysis of technical, operational, economic, legal, and scheduling aspects to determine if the project is worth pursuing. After condensing my implementation plan, I have come up with the following:

* Technical Feasibility: To develop a vb application that calculates the md5 hash of files and compares them with known hashes, there should be an extensive knowledge about the required vb modules and os navigation methods.
* Operational Feasibility: The users must have a clean and understandable user interface with the right amount of abstraction, hiding the unnecessary working parts of the program. It should be resource efficient, fast and should be able to handle multiple number of files at the same time.
* Economic Feasibility: The development must not include any monetary costs and should avoid using proprietary software support and integration.
* Legal Feasibility: The application must adhere to data protection laws, software distribution laws and must have an opensource or closed source license. There shouldn’t be any ethical mishaps or misuse of the application.
* Scheduling Feasibility: The development of the application and the documentation must be complete before a week of the last submission to allow any changes and fixes.

PROPOSED SYSTEM:

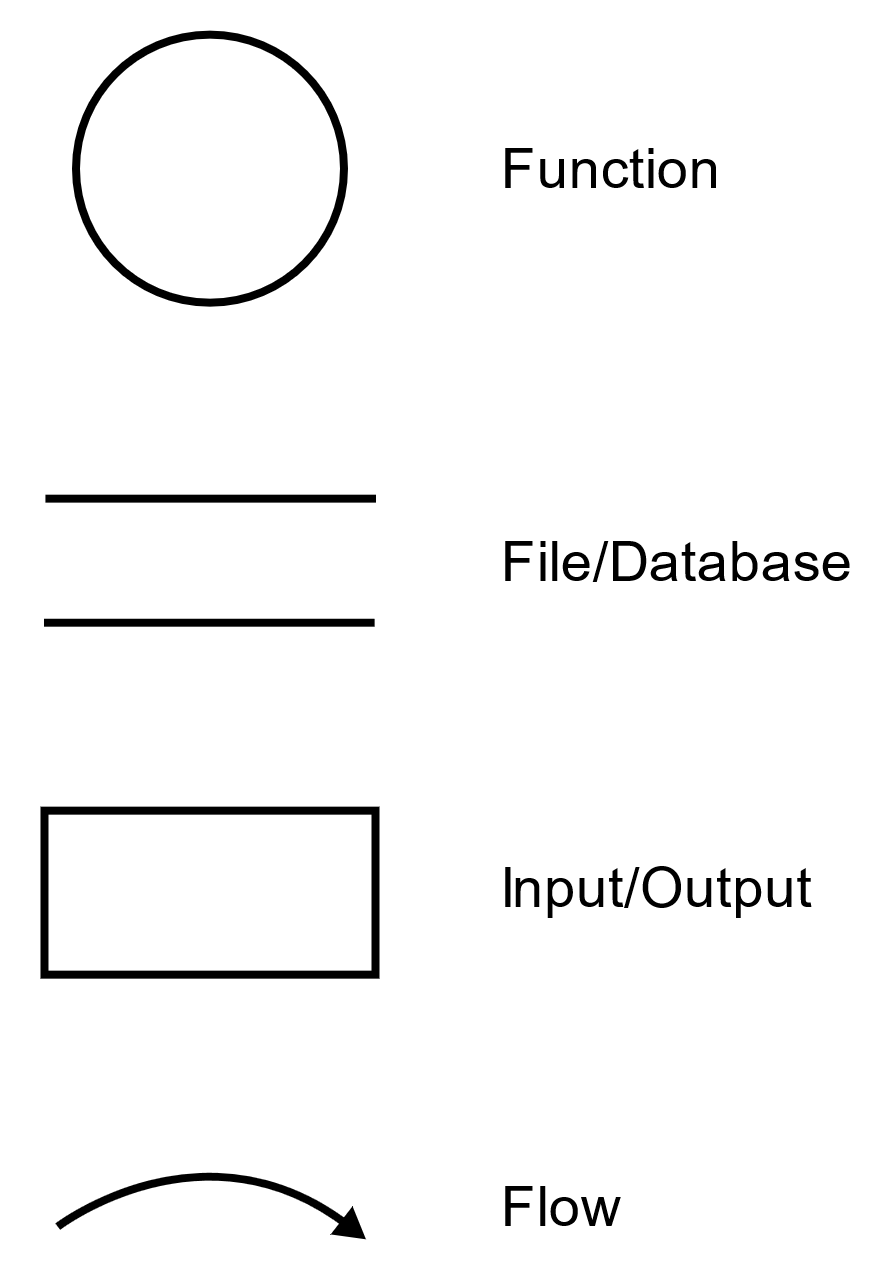
The project demonstrates a fast and lightweight standalone antivirus scanner application using vb.net. The algorithm used here is MD5 hash comparison by maintaining a virus hash list database using ADO.NET. The idea is to iterate through the files in the specified directory or scan the selected files by comparing the calculated MD5 hash of the file with the MD5 hashes of known viruses. The entire algorithm can be simplified into the following:

* System Overview: Developing a user-friendly MD5 hash comparison-based virus detection software using vb.net which integrates MD5 hashing libraries for efficient and accurate hash calculations.
* User Interface: Designing an intuitive and understandable user interface for ease of use and include features like file scanning, scan history and scan options.
* Database Integration: Maintaining a database to store known virus MD5 hash and the file hashes for integrity checks. The database must be updated regularly to detect new threats.
* Scanning Engine: Developing a scanning engine that calculates MD5 hashes for files while scanning and compare those hashes with the known hashes of the viruses from the database.
* Testing: The virus detection capabilities must be tested and made sure that the algorithm works and the application runs resource efficient, quick and runs offline.
* Documentation: The application must have a clear and detailed documentation so that open-source peers can improve and give suggestions about the code, growing the community.
* Deployment and Future Support: If the application passes all the tests and checks all the requirement, it can be deployed for public usage.

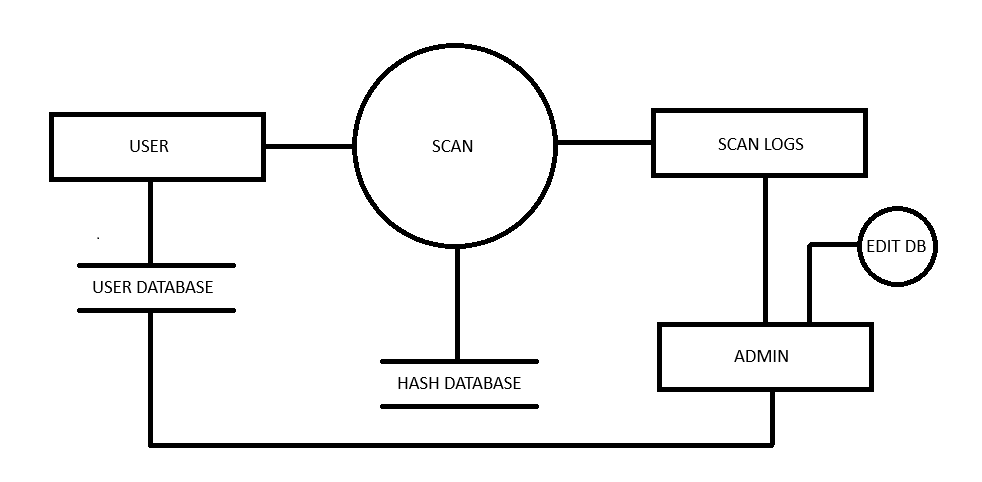
DATA FLOW DIAGRAM (DFD):

Data flow diagram (DFD) aka data flow graph is a visual representation of the information flow through a process or system. It uses defined symbols like rectangles, circles and arrows to show data inputs, outputs and storage points. They are commonly used during problem analysis. They are quite general and are not limited to problem analysis for software requirements specification. They are very useful in understanding a system and can be effectively used during analysis.

The symbols used to create a DFD are:



LEVEL 0 DFD OF VIRUS SCANNING APPLICATION:



LEVEL 1 DFD OF VIRUS SCANNING APPLICATION:

