

**SCHOOL OF COMPUTER & APPLIED SCIENCE**

**COMPUTER SCIENCE DEPARTMENT**

**CSC 399 - Senior Project**

**Senior Project Report**

Prepared By:

**Bahaa Khaled Yehia**

Supervisor name:

**Dr. Youssef Ajeeb**

**Summer Semester (2023)**

Contents

[Chapter One: Introduction 3](#_Toc141460413)

[**Problem Statement:** 3](#_Toc141460414)

[**Methodologies Used (Hardware, Software & Tools):** 4](#_Toc141460415)

[**Obstacles Solved:** 4](#_Toc141460416)

[**Related Works:** 5](#_Toc141460417)

[**Objectives of the system:** 5](#_Toc141460418)

[**Organization of the project:** 6](#_Toc141460419)

[Chapter Two: Theoretical Background 7](#_Toc141460420)

[Chapter Three: System Study and Analysis 8](#_Toc141460421)

[**Overview:** 8](#_Toc141460422)

[**Information gathering:** 8](#_Toc141460423)

[**Requirement Specification:** 9](#_Toc141460424)

[User Requirements: 9](#_Toc141460425)

[Technical Requirements: 9](#_Toc141460426)

[System Analysis: 10](#_Toc141460427)

[Chapter Four: System Design 10](#_Toc141460428)

[**Overview:** 10](#_Toc141460429)

[**Project Design** 11](#_Toc141460430)

[Chapter Five: Implementation 14](#_Toc141460431)

[**Overview:**  14](#Overview)

[**System Forms:** 14](#_Toc141460432)

[**Coding:** 15](#_Toc141460433)

[**Testing:**  21](#Testing)

[Chapter Six: Conclusion and future works 23](#_Toc141460435)

[**Conclusion**: 23](#Conclusion)

[**Future Works:**  24](#Future)

[References 24](#_Toc141460438)

**Table of Figures**

[Figure 1 - Usage of BeautifulSoup Library and the html.parser 12](#_Toc127283242)

[Figure 2 - System argv in order to easily execute the program 12](#_Toc127283243)

[Figure 3 - Using some symbols and focusing on identifying every step the tool is doing 13](#_Toc127283244)

[Figure 4 - Showing the exact form that has the bug in its code 13](#_Toc127283245)

[Figure 5 - Showing the attraction that the design does by detailing the process when finding 13](#_Toc127283246)

[Figure 6 - More and More about this attractive design specially in Offensive Cybersecurity Field 14](#_Toc127283247)

[Figure 7 - Presenting the different stages that the tool is passing through 15](file:///C:\Users\Amine\Downloads\Shadi_Amin_Senior_Project_Report%20-%20Updated%20Version.docx#_Toc127283248)

[Figure 8 - Detailed findings and flexibility in executing 15](file:///C:\Users\Amine\Downloads\Shadi_Amin_Senior_Project_Report%20-%20Updated%20Version.docx#_Toc127283249)

[Figure 9 - Interacting with the webpage and introducing our first function get\_all\_forms 16](file:///C:\Users\Amine\Downloads\Shadi_Amin_Senior_Project_Report%20-%20Updated%20Version.docx#_Toc127283250)

[Figure 10 - The get\_form\_details function 17](#_Toc127283251)

[Figure 11 - is\_Vulnerable function 18](#_Toc127283252)

[Figure 12 - scan\_sql\_injection function 19](file:///C:\Users\Amine\Downloads\Shadi_Amin_Senior_Project_Report%20-%20Updated%20Version.docx#_Toc127283253)

[Figure 13 - More about the amazing scan\_sql\_injection function 20](#_Toc127283254)

[Figure 14 - Our main function 21](#_Toc127283255)

[Figure 15 - Testing on a website 22](#_Toc127283256)

[Figure 16 - Another testing on the dvwa of the metasploitable machine 23](#_Toc127283257)

# Chapter One: Introduction

1. **Introduction:**

The purpose of this project is to develop a tool that can detect the presence of a critical vulnerability that can lead to catastrophic effects on businesses and which is the SQL injection vulnerability. As a matter of fact, this project will aim to detect the presence of SQL injection for website applications in order to warn business owners so that they can avoid it. This issue has been for many past years a very critical problem that is leading to information disclosure and credential leakage for many users of a certain business. Thus, it is the time to create a tool that can protect businesses from this danger trouble.

## **Problem Statement:**

Throughout the past years different businesses all over the world have been subjected to many problems and troubles due to the SQL injection vulnerability. Black hat hackers have used this methodology to attack many targets all over the world. As a matter of fact, this project will not only be for businesses of my country; however, this project is targeting business owners throughout the whole world so that they can use it as an ethical tool that can help them be ahead of black hat hackers and thus protect their businesses from any trouble that SQL injection can lead them to.

## **Methodologies Used (Hardware, Software & Tools):**

As for the methodology used, I followed the Cybermethodology. This methodology provides key insights into how modern attacks cyberattacks are being carried out. This had helped me in being able to create a project that can detect SQL injections for different types of databases that a software might be using such as MySQL and Oracle database.

In the implementation of the project, I used my own laptop as a hardware, but I did not use its windows operating system. However, this project introduced me to a very interesting operating system that I had not used before and which is the Linux operating system. As a matter of fact, I was able to surf this operating system and enjoy learning all its features. Moreover, another software I used was the virtual box machine which helped me install the Kali Linux operating system to make this project without the necessity of buying a new machine that would have this OS in it. As a result, it was my first time where I used the virtual box software that can make ethical hackers’ life much easier. In addition to that, I used a very interesting software that is called Metasploitable which I installed on my virtual box machine and that helped me in testing my project as it was very critical to test such a developed tool on any website application. Lastly, I also used python which is one of the famous programming languages worldwide; however, by this project I was able to dig deep in its libraries and get to know many libraries that can be used in cybersecurity fields. I literally can say that this project had helped me in digging deep in many software that I had known, and at the same time introduce myself to other new software that I did not used to know before.

## **Obstacles Solved:**

Creating such a project made me face many obstacles. One of them was the need to have skills of another operating system which I never worked on before and this was the Linux Operating System. Thus, with the help of this project I was able to introduce myself to this operating system where I enjoyed surfing through its amazing features and at the same time be able to create my project. Moreover, the need for a Linux operating system put me also against another obstacle and that was the importance of having a Linux Operating System machine which I was not able to buy. However, I also introduced myself into another interesting topic and which is the virtual box machine software where I was able to install a Kali Linux Operating System and use all its features on my own windows OS laptop. Moreover, due to that this project is considered critical , I was not able to test it wherever I want and thus I searched for many solutions for this until I was able to get to know more about the Metasploitable virtual machine and which was a machine that you can install on your virtual box and that can help ethical hackers to test their developed ethical hacking tools ethically. In addition to that, I became also familiar with some testing websites that are present for the sake of helping white hat hackers test their ethical hacking tools before publishing them. As a result, all of this knowledge was gained by me just because of the obstacle of not being able to test this tool ethically. Again, and as I consider this project to be a very helpful one for all business owners worldwide, it has been one of the best knowledgeable reasons that I had throughout my cybersecurity learning path.

## **Related Works:**

To effectively analyze recent similar works related to the problem mentioned in the problem statement (the catastrophic effects of SQL injections on businesses throughout the past years turning them down), it is important to conduct a comprehensive review on different businesses’ owners and organizations in order to be aware of the types of the SQL injections that they were subjected to and how did they effect them. As mentioned, this involve searching for different resources online that shows different types of SQL injection attacks that companies were subjected to; in addition to interviewing different personnel specially IT departments and CEOs in different businesses who were attacked by this critical attack.

In this part of work, it would be important to identify the type of SQL injection attack that they were subjected to, the way they handled the incident when the attack took place, the time they needed to retain their business running, the effects of the attack on their business in different fields such as moneywise and their customer’s trust.

## **Objectives of the system:**

The main objective of the project is to build a tool that is able to take any webpage in any web application and then scan it to find all forms found in it ( the ones that appear to the client-side and also the hidden ones) ;then, it scan all of these forms found, in addition to the URL of this webpage in order to alert and show the presence of any SQL vulnerability, so that the business owner in the organization can be aware of that and the developing team in this same organization can fix and patch the code having this critical vulnerability. Thus, this would help the business in avoiding and being aware of this dangerous issue. To achieve this objective, the following technical methods and tools is then used:

Virtual Box Software: This software is used I order to be able to install important operating systems that is going to be used when developing and testing the tool. (Feldman, n.d.)

Kali Linux Operating System: This operating system is an open source one that can help in being able to hit kernel and high privileges when coding the tool and testing it.

Metasploitable: This is a very interesting software that can help white hat hackers to test their tool and enjoy its success without being subjected to any criminal accusations for using this tool illegally because of its critical and powerful results.

Advanced Python Libraries: In this tool python is used in its hard way because of the need for advanced libraries in it for example to send requests, and also being able to read the http responses in order to analyze and scan them.

By employing these technical methods and tools, the project aims to deliver an interesting and powerful tool that all developers and business owners can use in order to protect their work and their organization from falling in the trap of an SQL injection that may lead to catastrophic results.

## **Organization of the project:**

Introduction: This chapter will provide a background on the problem that organizations had faced and may face due to being attacked by hackers specially in an SQL injection attack. I t will also state the effects of such an attack on them. Moreover, it will also provide an overview of the objectives and goals of the project and how such a project can help them face and be aware of this critical problem.

Problem Statement: This chapter will provide a detailed description of the problem being addressed in addition to a detailed elaboration on the effects of this problem and why it is important to use such a project in order to withstand this dangerous issue being discussed.

Design and Implementation: This chapter will describe the design and implementation of the SQL injection scanner tool, including a discussion of the hardware and software used, as well as the tools and methodologies applied. It will also provide an overview of the procedure that this tool was developed on and its various components and libraries.

Results and Analysis: This chapter will present the results and analysis of the project, including its powerful performance, user feedback, and any other relevant data. It will also discuss the strengths and weaknesses of the tool created and identify areas for future improvement.

Conclusion: This chapter will provide a summary of the results and key findings of the project, and discuss the implications and significance of the work. It will also provide recommendations for future research and development in the field.

References: This section will list all the references and sources used in the project, following a standard citation format.

Appendices: This section may include additional materials such as code snippets (Geeks, n.d.), proof of successful work, or supplementary information that support the findings and results of the project.

# Chapter Two: Theoretical Background

The main idea of the SQL injection scanner tool rose from the fact of the booming news of many software and applications being hacked by cyber criminals. Moreover, the huge amount of data being subjected to the internet in addition to businesses being connected to the internet, all of this made it dangerous to develop a software or application without taking into consideration security before everything. As a matter of fact, this resulted in the necessity of having a tool that developers and business owners can use in order to secure their coding item. In this background, we found that one of the most catastrophic and at the same time most attacked feature in a software is the database and that is because the critical information that can be found in it. As a result, creating an SQL injection vulnerability scanner and making it flexible to any database that would be invented anytime in the future would be a very important issue on the track of developing any software and at the same time in securing businesses and avoiding them from collapsing.

One of the key aspects of these studies is the fact of making this tool to not only help developers securing their code but also making their life easier by showing them the code where the flow is found and also helping them easily adapt to any database that would emerge anytime in the future.

Furthermore, this background study showed that developing a software or any application is an important work to do and specially nowadays where every business needs such a valuable feature. However, this is worth nothing if this software or application got hacked and businesses start suffering from tremendous effects due to a cyber-attack. Here comes the value and the worth of creating and a tool that targets every software and application in helping developers create a secure code that will avoid them from cyber-attacks. So how come if this tool is the one of the important scanners that scans for a major and critical vulnerability in an application and which is SQL injection.

In conclusion, the theoretical background provides a strong fact of the necessity of having such a tool to be presented for all developers and business people in order to be able to have a secure code and thus a secure business for a lifetime.

# Chapter Three: System Study and Analysis

## **Overview:**

This chapter provides an overview of the information gathering and requirement specification processes for the project. It includes a brief description of the ways used to collect the information needed for the project, as well as a detailed description of the requirements needed for the project to fulfill its objectives.

## **Information gathering:**

The first step in the project was to gather information about the problem and the needs of the target audience. This involved conducting research on the different businesses that were affected by SQL injection attacks in Lebanon, as well as the feedback of the IT department and the business owners in these organizations. As a matter of fact, information was also gathered through direct interactions with potential users and stakeholders, to gain a better understanding of their needs in order to protect themselves against such attacks and their capability of achieving this secure atmosphere in their businesses. Moreover, search engines played also important roles in the information gathering part as more evidences related to organization who were effected by SQL injection was found, and thus this all acknowledged the importance of such project and helped in pinpointing on how to make the project much more useful and flexible for the target audience.

## **Requirement Specification:**

Based on the information gathered, a comprehensive set of requirements was specified for the project. This included both user requirements, which define the needs and expectations of the target audience, and technical requirements, which specify the technical capabilities and features that the project must have in order to fulfill its objectives.

### User Requirements:

Ease of use: The tool should be very easy to work with, and at the same time the data found should be presented in a good manner for the developers and even people with no technical background can read and understand.

Secure Data Presenting: The tool should present the data in a secure environment due to that these data can be critical ones and that is why Kali Linux operating system was introduced in the project.

Wide acceptance: The tool should be widely accepted by business owners and members of developer teams.

Flexibility in Code Repairing: The tool should not only present the presence of the flaw but also should indicate the code review in which the flaw is found so that the developers’ life in fixing this flaw would be much more easier.

Quick processing time: The ability of finding the flaw and fixing it should be processed quickly and efficiently in order to maintain the continuous integration and development in the business (CI/CD).

Accessibility: The tool should be accessible only for the business owners and the IT department in the organization.

Integration with any newly developed technology: The tool should be able to adapt with any new technology that would be developed in the future specially in the database field.

### Technical Requirements:

Kali Linux Operating System: The tool will be working in Kali Linux operating system due that it is an open source and it is much faster.

Virtual Box Software: Due to the fact that the developing the project would be in a critical environment because of working with user and business credentials, so it would be much more secure to create the project and test in a virtual box software.

Python Coding Language: The tool will be written in python language because it is more flexible, and it has different libraries that are a necessity in creating the tool.

MetaSploitable Software: Again, due to the critical environment and in order to stay in a legal work when testing the tool, metasploitable virtual machine was introduced in the project to help test the tool and present the testing to the business owners and all IT members.

### System Analysis:

Python Libraries: Python libraries are different libraries that where largely used in the project and specially in the fields of networking, initiating the browser, transforming webpages data so that they can be useful in the project by the tool, and also in the different functions presented in the project that were the main engines behind the effective and the powerful results of the tool.

# Chapter Four: System Design

## **Overview:**

In this system design phase of the project, the results of the system analysis phase are transformed into a detailed design specification. The design specification defines the structure and behavior of the system and provides a blueprint for the implementation phase.

## **Project Design**

In the system design phase of the project, I focused on designing the system not in the way it looks instead in the way it is flexible, readable, and easier to be used by the target audience. As a matter of fact, I worked on trying to make the tool work just by one Enter press of the user. I used the system arguments library that is found in python in order to be able to let the user scan his/her webpage by just entering the link of it. Moreover, I worked on the user interface by using a very interesting library in python and which is called Beautiful soap that has many classes such as the html parser that can help in transforming the response into a readable html and then from this point we were able to extract the forms found in the webpage entered by the user and which are the only vulnerable objects for an SQL injection in addition to the URL of the webpage.

In addition to that, I worked on presenting the target audience with a very readable and detailed data. As a result, I made the tool show on the screen a detailed explanation of every step it is performing while executing the program, so the user was able to notice every step that the tool is undergoing on the webpage. Furthermore, I used some remarkable symbols (([-]), ([+])) in the field of security so that the user will not have a chance to disregard any detail in his webpage specially that we are dealing with its security for critical data.

Moreover, I worked hard on showing an exact copy of the form where a flaw or a vulnerability is found in a webpage. This was a very advanced part I used because I considered one of the best parts in the project that needed to be highlighted. The reason is that by presenting an exact copy of the form having the bug, this will help both the business owner and the developer, each on his side, to be familiar of what is going on and how to fix it. So, for the business owner he/she will be able to clearly read the form that has the flaw with all of its details, and thus make sure that the developer is truly correcting the exact part of the code that had the flaw and that is by just comparing the part of the code that the developer is fixing with what he saw from the form that was shown to be vulnerable by the tool. However, for the developer he/she will be able to recognize where specifically is the flaw and that is in which part of the code, which will help him both easily and quickly fix the code in order to maintain one of the most important part of a business connected to the internet and which is the CI/CD (Continuous Integration / Continuous Development).

* Some pictures of my system design are presented here:

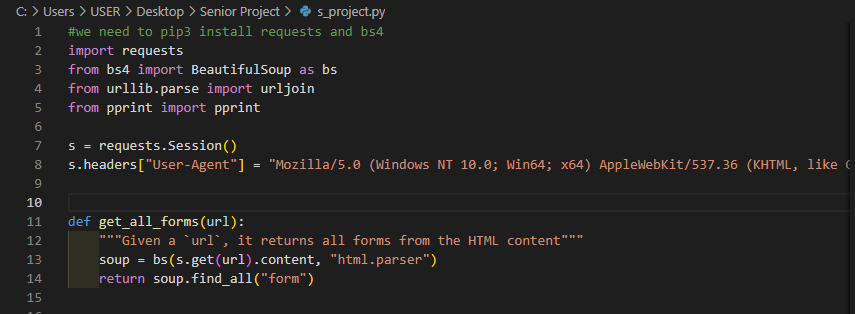


Figure 1 – Usage of BeautifulSoup Library and the html.parser

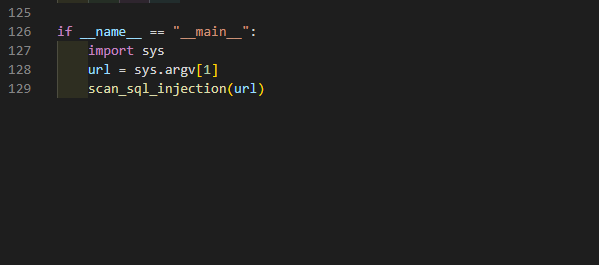


Figure 2 – System argv in order to easily execute the program

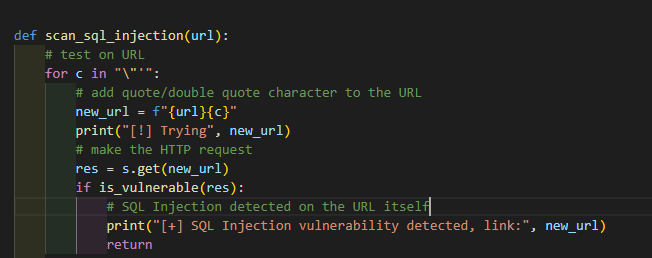


Figure 3 – Using some symbols and focusing on identifying every step the tool is doing

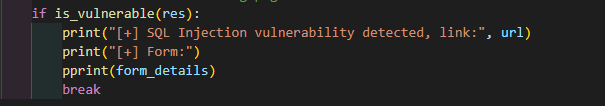


Figure 4 – Showing the exact form that has the bug in its code

* Some Pictures of the attractive design in the offensive cybersecurity field:



Figure 5 – Showing the attraction that the design does by detailing the process when finding results

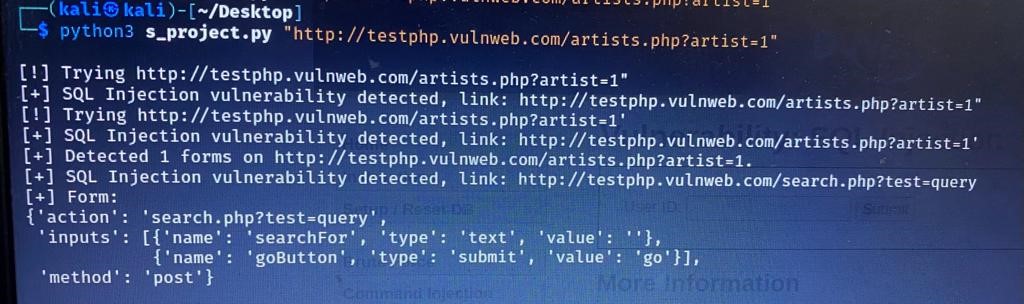


Figure 6 – More and More about this attractive design specially in Offensive Cybersecurity Field

As shown in the design, the symbols that are used attracts the user to the critical data that is being presented and make him much more focused on concerning every single detail. In addition to the detailed information presented so that the user is being able to identify every single procedure that the tool is undergoing. This all, in addition to the clearly presented form that shows all the small details of the part of the code where the flaw was found and in a very readable way even for a non-technical background person.

# Chapter Five: Implementation

**Overview**:

In the following chapter, I will be presenting more data about the implementation of this tool. As a matter of fact, I will be showing pictures of the forms that the tool will be using and presenting while undergoing different processes to reach its interesting critical findings. Moreover, this chapter will show the powerful functions that were used to create this tool with a brief description of the exiting job of each one of them. In the end, an explanation of the ethical testing of this tool on a vulnerable web application showing how interesting and helpful this tool will be in protecting businesses from shutting down.

## **System Forms:**

The system forms refer to the interfaces that make up the tool and that is what the user interact with. The system forms in my project where designed to provide both a user-flexible and usable interface, and an aggressive offensive form for users to interact with the tool and get all the needed critical data from it in an ethical attacking environment. As a matter of fact, monitoring the different stages that the tool is passing through while being executed; in addition to showing some symbolic results and also detailing the results in a very clarifying manner, all of these played an important role in making the tool practical and beneficial.

Here is the form that is presenting a detailed information about the processes that the tool is undergoing while running:



Figure 7 – Presenting the different stages that the tool is passing through

In the picture, the user can notice what is the tool doing as is stating what it is trying and then what did it get results. Moreover, its obvious the presence of some symbols such as the exclamation mark and the plus sign where the first one attracts the user attention in order to discover what is the tool doing while the other sign indicates if the result was a positive or negative one.

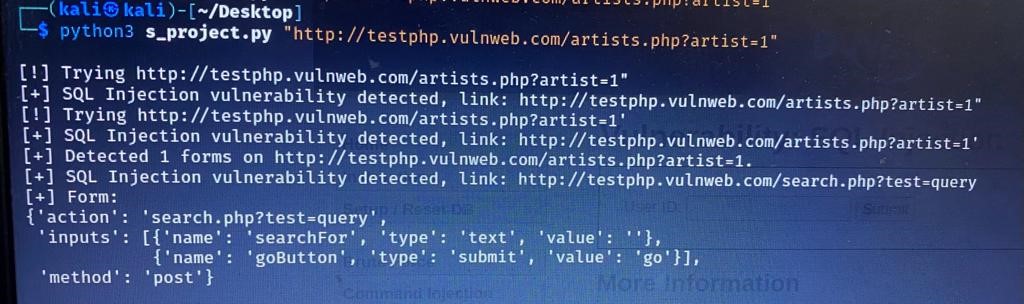


Figure 8 - Detailed findings and flexibility in executing

In the following picture the user can simply put the tool in action by just running the python3 command with the name of the tool and then the webpage that he/she is scanning. This produces usability and flexibility between the user and the tool. Furthermore, the user can see a detailed, readable, and an exact copy of the form where the flaw was found and which in its turn plays an important role in making developers life easier so that, they can directly travel to the code responsible for the form to correct it quickly and sustainably. As a result, a complete avoidance of an attack will be performed with preserving the CI/CD of the application (Continuous Integration/Continuous development).

## **Coding:**

This section is about the programming part of the project.

The code was written in Python language.

I will be presenting the code of the powerful functions that where doing all the work when executing the tool in order to secure the application in addition to their implementations

Initializing the browser for the webpage link and sending requests and getting responses:

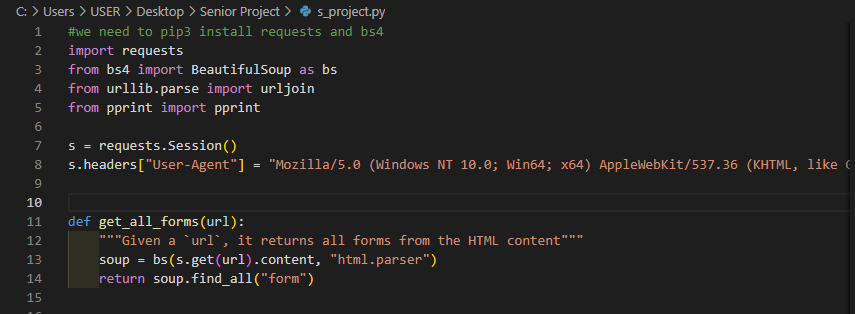


Figure 9 – Interacting with the webpage and introducing our first function get\_all\_forms

In the picture, you can see how did we first open a session in the browser in order to start interacting with the link of the webpage that the user wants to scan. Then, we introduced our first powerful function that is going to get all the forms found on the webpage and here I want to stress on the important of this function as it is a one of the game changers in this project because it does not only find forms that the user see on the webpage; however, it finds also the hidden ones and that are the most important findings in our scan as they have the highest probabaility in having the existance of a SQL injection vulnerability. As presented, you can see the usage of beautiful soap that is a class from the bs4 library in python that has the html.parser feature that can help in taking the HTML content of the response for the request that we send first when openning the session, and then degrade it using the parser in order to then start searching and finding the forms we want taking into consideration that they can be seen by the user on the webpage or they can be hidden forms.

Figure 24 - validate input function(for login and signup page)

Here comes the second powerful function in our project that is presented by get\_form details:

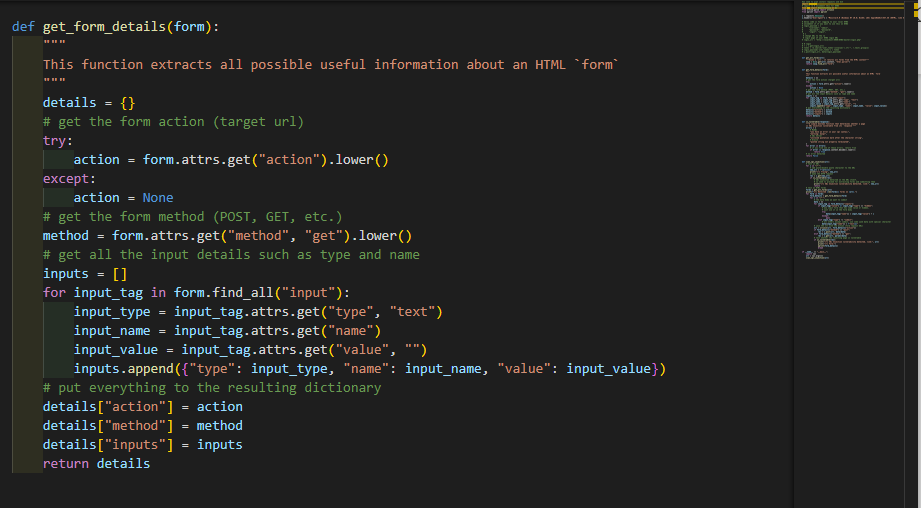


Figure 10 – The get\_form\_details function

In this function, every form found by the get all forms function will be degraded into pieces and that is in order to be able to get all the details of this form from the action it does so to know to where we will be redirected by this form, the method used to know if it is a get method so we are getting data from the server or we are posting data to the server, and at the end all of the inputs that are found in this form such as the ‘username input’ or the ‘password input’ and much more. Here, remember one of the most important part that all forms are being degraded even the most vulnerable ones and that are the hidden ones.

Going much further in our interesting functions, here comes the ‘is vulnerable’ function:

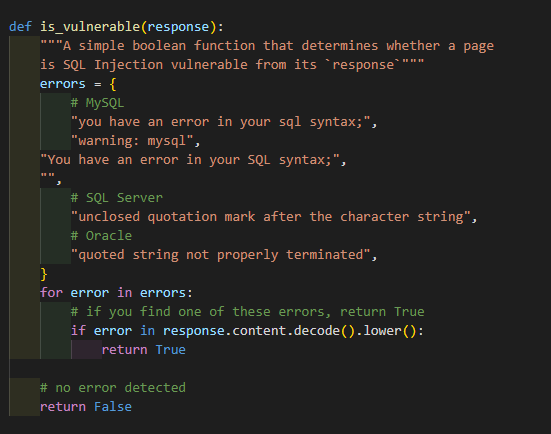


Figure 11 – is\_Vulnerable function

In this function what the tool is doing is that it is taking the response that we are getting from the handcrafted request that we created. This handcrafted request is an essential part of our project, as it is the main engine that the tool relies on. As a matter of fact, this request will be a SQL syntax that will be injected in any input of the form. As a result, after sending this request, we wait for the response and we analyze it, so if it has one of the errors presented in this is\_Vulnerble function, then this will make this function return a True value presenting that this form is vulnerable to SQL Injection vulnerability and if not again this function will return a False answer for us to notice that we are on the safe side as this form is not vulnerable to SQL Injection. Here, I would like to mention a very important idea that makes this project a unique one and at the same time very powerful and flexible. This idea is that whatever database is created or developed in the future, with just realizing the errors that this database would respond in, we can just add these errors to this interesting function and BINGO, we are again able to scan any form presented in the webpage although it is using a brand new database in order to detect if it is vulnerable to SQL Injection or not.

Diving deep in our tool we get to the function that is running and managing all the parts of the tool in order to make it powerful and reliable. This function is the scan\_sql\_injection function:

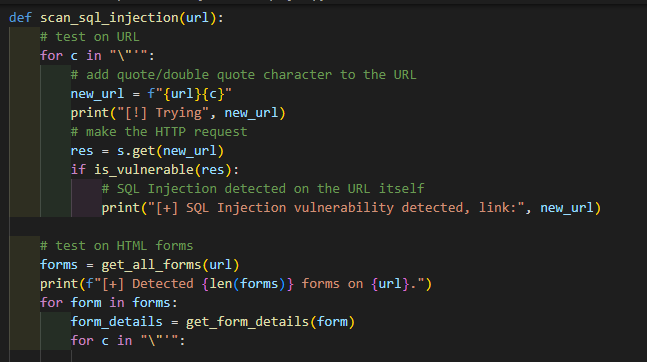


Figure 12 – scan\_sql\_injection function

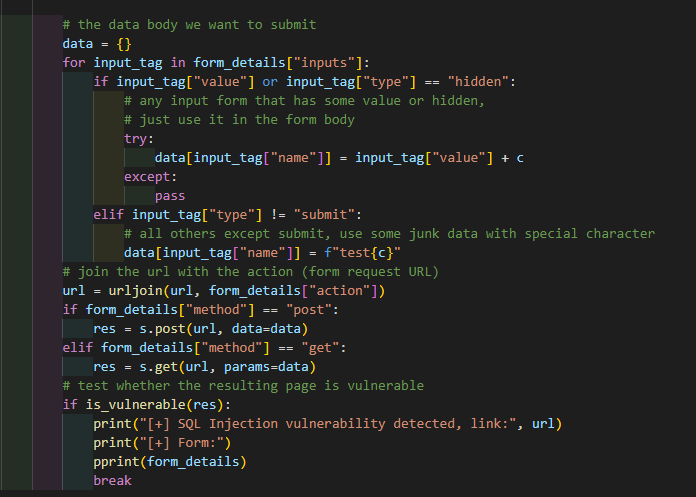


Figure 13 -More about the amazing scan\_sql\_injection function

This function and as mentioned before is the one that is managing all of the other parts of the tool making it able to be very interesting with many critical results. As a matter of fact, this function takes the URL of the webpage that we are scanning and then enters a loop that has different SQL syntax in it, and each one of these SQL syntaxes will be injected in the URL creating the handcrafted request. After that, we wait for the response which in its turn will be subjected to the is \_Vulnerable function to detect if we have an SQL injection vulnerability or no in the URL or no. MORE and MORE, this function also goes and applies both the get all forms function to get all the viewable and the hidden forms in the webpage and then on every form found on the webpage whether it is hidden or not, this function subject the form to get\_form\_detail function in order to determine all of its details. Lastly, this function makes the form also undergoes the loop of different SQL syntaxes in which every syntax of them will be injected in the form creating the handcrafted request in order to then get the response that will be again applied to the is\_Vulnerable function so that we can determine if there is a presence of an SQL Injection Vulnerability in the form or not. One of the many important parts of this function is that every step that this function is performing will be printed and showed for the user, so that he/she can know what the tool is doing. At the same time, all of the findings whether they are for example the form’s detail or the form where the flaw or the bug was found, all of this data will be printed and showed in a readable and detailed manner so that the developer can easily and quickly find the flaw and patch it maintaining the CI/CD performance of the application. All this result in a great defense against any type of any SQL Injection Vulnerability.

Finally, comes our main function that is very short in its syntax; however, it is V8 engine that is going to avoid any application form any type of an SQL injection Vulnerability

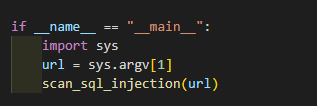


Figure 14 – Our main function

### Testing:

Testing is a crucial part of the tool building process to ensure that the procedure of creating the tool meets the specified requirements and can get the best of the results that we are working on without having any defect in it. As a matter of fact, I used the system testing in order to test my tool.

System testing is a testing methodology that focuses on the behavior of an entire system or tool as a whole. It aims to validate that the tool meets its functional and non-functional requirements and performs as expected under different operating conditions.

In the context of my SQL Injection Scanner Tool, system testing involves running the tool on a virtual machine and on a specific target due to its criticism. As a matter of fact, I tested the tool using a virtual box where I used a virtual Kali Linux operating system to target specific targets such as a vulnerable website that is known for testing and the dvwa feature that is found in the metasploitable machine. The testing procedures and result analysis can vary depending on the specific requirements of the tool, but they generally include the following:

Functionality testing: This involves testing the various features and functions of the tool to ensure that they work as expected. For example, testing the get\_all\_forms function, get\_form\_details function, and the scan\_sql\_injection function in which acknowledgement that these powerful functions are working correctly showing their process of action in details and finding critical and exact results as expected.

Performance testing: This involves testing the performance of the tool under different conditions, such as different network conditions, operating system configuration, and other variables. This can help identify performance and other issues that can affect the user experience.

Security testing: This involves testing the tool’s security features to ensure that the tool is secure and is correctly and exactly finding flaws without causing any leakage of critical data to the public unless to the tool itself. This include testing the tool on the application at the same time while the application is running to ensure that security of its data is maintained, and the flaw and the data leakage happens only on the tool’s level of action.

Usability testing: This involves testing the tool's ease of use, and other user experience factors. This can help identify areas of the tool that may be confusing or difficult for users, and thus provide good reviews in order to improve the usage of the tool and the tool’s way of presenting the critical and powerful results for the user.

The result analysis for system testing typically involves analyzing the test results and identifying any issues or defects that were found during testing. These issues can be prioritized based on their severity and impact on the user experience, and a plan can be developed to address them. The testing process can be repeated as needed to ensure that the tool is functioning as intended and meeting the requirements.



Figure 15: Testing on a website

In this testing result of the tool, the picture shows the powerful findings of the tool as it shows that it has found an SQL Injection vulnerability on the link presented by the user. In each step of the scanning, the process was explained in details and the findings were managed in a way to be clear, readable, and easy to patch in order to maintain the security of the webpage running on this link, and thus maintain a good and secure business.

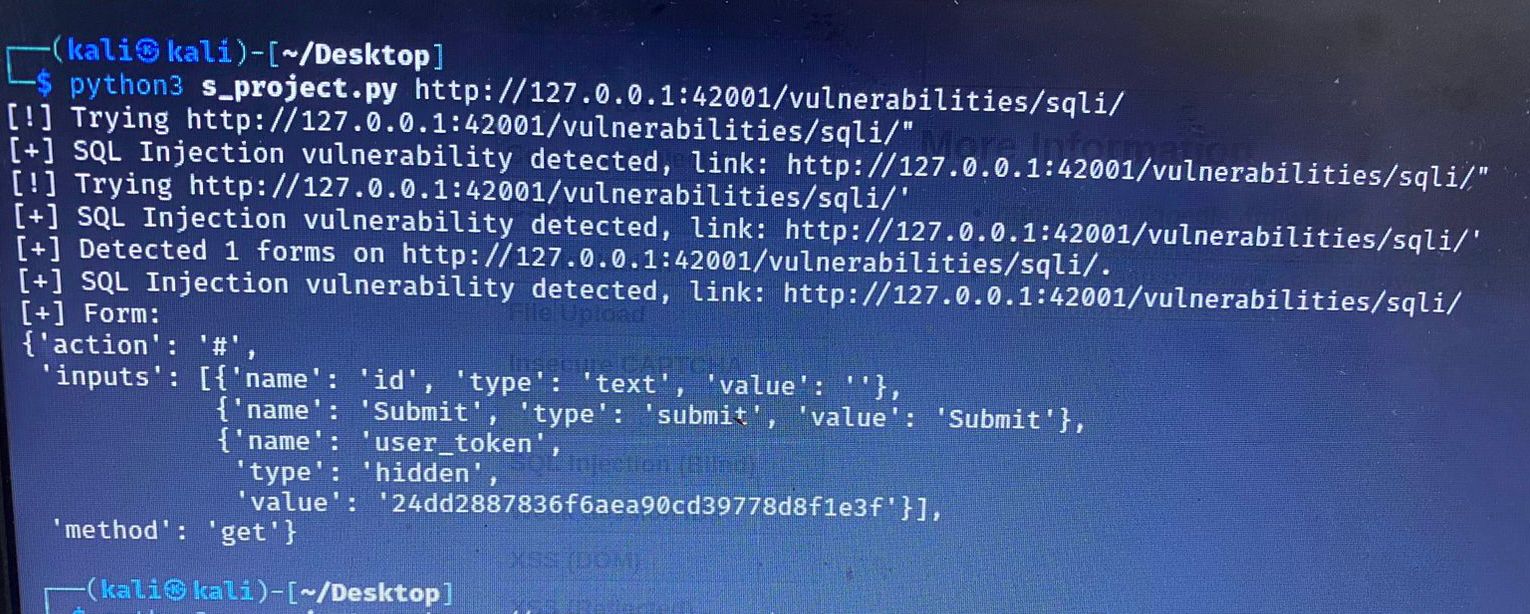


Figure 16: Another testing on the dvwa of the metasploitable machine

In this picture, another testing is done but this time on the dvwa of the metrasploitable machine and the results were astonishing. The tool was able to find a hidden form in the webpage that the user will not find when opening the webpage on their browsers. In addition to that, some data was found by the tool such as the ‘user-token’ and its value that can have a high possibility of being used to be able to access some user’s accounts in the application being tested and which can have a catastrophic issue on the business running this application. All of this was well expressed to the developer and the user using this tool in order to well patch the flaw in the application and maintain a secure business. Again, thanks for this tool and its powerful findings.

# Chapter Six: Conclusion and future works

### Conclusion:

In conclusion, the SQL Injection vulnerability scanner is a vital tool that any business needs in order to live securely and to be able to develop and maintain its presence. As a matter of fact, the ability of this tool to detect one of the most dangerous security issues worldwide is a crucial issue that any business that is running today or that will run in the future hardly needs before developing any type of application. Moreover, the flexibility of this tool with the development of technology makes it alive in the future and essential to any business that will have birth in the future. In addition to that, the tool’s usability allows not only developers to be able to use this tool but also all business owners. As a result, these stakeholders would be able to better invest their money in a secure business with tough basics.

### Future Works:

For future works, there are several improvements and additions that can be made to enhance the user experience and functionality of the tool. These include:

1. Implementing Artificial Intelligence when alerting the user to the presence of an SQL Injection vulnerability.
2. Improving the user interface, so that the user can enjoy pictures and icons while using the tool.
3. Publishing the tool with payment methods making it easily accessible to a wider audience and at the same time a good business.
4. Optimizing and enhancing the tool further using python threads, ensuring that it runs as fast as possible.

By making these improvements, the tool can continue to provide a valuable and crucial data for all users specially for business owners who can by this tool make sure that they are investing their money in a safe and long-lasting business.

# References

Feldman, J. (n.d.). *A Beginner Guide to Python for Cybersecurity*. Retrieved from https://www.coursereport.com/blog/python-for-cyber-security-with-flatiron-school

Geeks, G. f. (n.d.). *Python Network Programming*. Retrieved from https://www.geeksforgeeks.org/python-network-programming/

Kingthorin. (n.d.). *SQL Injection*. Retrieved from https://owasp.org/www-community/attacks/SQL\_Injection

NVD. (n.d.). *CVE*. Retrieved from https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=sql+injection

Portswigger. (n.d.). *SQL Injection*. Retrieved from https://portswigger.net/web-security/sql-injection