Student Name and Number : Sakshi Umesh Teli -20079303

Hardik Shailesh Rathod - 20083974

Course Title : M.Sc. Cyber Security

Lecturer Name: Swati Dongre

Module/Subject Title: Advanced Programming Techniques

Assignment Title: CA_TWO_(70%)

No of Words: 1576

By submitting this assignment, we are confirming that:

- **This assignment is all our own work;**
- **Any sources used have been referenced;**
- **we have followed the Generative AI instructions/ scale set out in the Assignment Brief;**
- **we have read the College rules regarding academic integrity in the QAH Part B Section 3, and the Generative AI Guidelines, and understand that penalties will be applied accordingly if work is found not to be our own.**
- **we understand that all work uploaded is submitted via Ouriginal, whereby a text-matching report will show any similarities with other texts.**

TABLE OF CONTENTS

1. INTRODUCTION

The number of cyber threats is growing every day and most users find it difficult for identifying a file or a site is trustworthy. To assist with this, we designed **ViroScan**, a basic web application that is a URL and file scanner of malware using **VirusTotal API.**

It verifies the findings of many antivirus engines and logs the scan history to a database so users can view it later. The system has also got the provision of login to access the system. We applied the **SDLC approach of iteration**/agile to develop and enhance the project in small phases

2. PROBLEM DOMAIN & OBJECTIVES

2.1 Problem Domain

Individuals are now exposed to numerous threats online such as **viruses, fake websites and malicious downloads**. When an individual opens a dangerous **file or a link**, then his or her computer or personal information may be destroyed. Many users don't know how to check safety before using something. So, we need a simple tool that can quickly tell whether a file or website is safe or dangerous.

2.2 Objectives

- Make a user-friendly website to scan URLs and files.
- Use the VirusTotal to determine if they are secure.
- Store all of the scan results in a database for later viewing.
- For users understanding the scan results, show them helpful reports.
- Help to secure people against threats on the web.

3. SYSTEM REQUIREMENTS

3.1 Functional Requirements- (What the system should have the capacity to do)

- Uploaded files checked against virus.
- Present the user with the scan results.
- Store every scan result in a database.
- Give users access to the complete scan history.
- Allow users to report on any suspicious files or URLs.
- Provide the facility to create accounts and log-in to allow users to access their accounts.

3.2 Non-Functional Requirements- (How the system should work)

- The system should be user friendly.
- User data should be protected by the system.
- It should function across various web browsers.

- The system should to be dependable and not crash.
- It will be easy to update it in the future.

4. SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)

4.1 SDLC Model Used (Agile/Iterative)

In our project, the model we used **Agile/Iterative**. We built the system in small bits and perfected each bit upon trial.

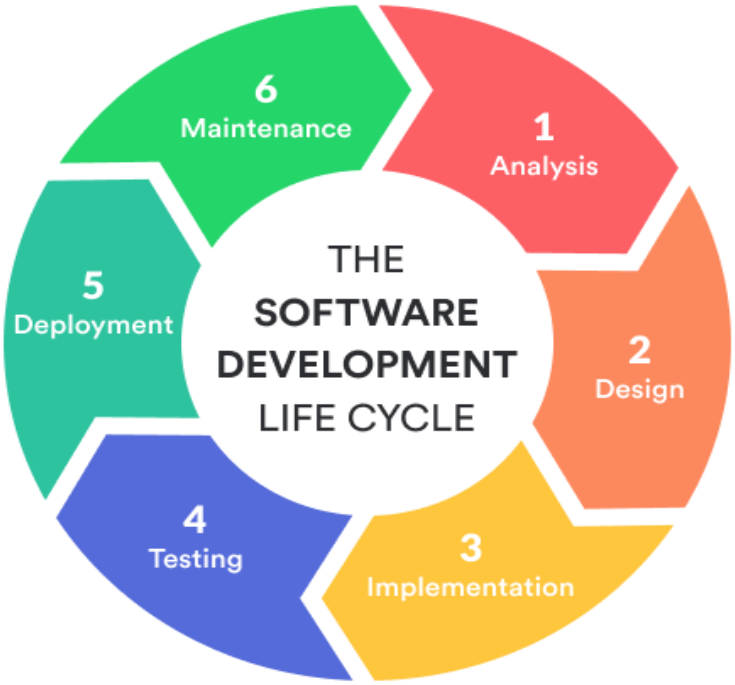


FIGURE 1: SDLC DIAGRAM THAT INDICATES THE STAGES OF SYSTEM DEVELOPMENT.

4.2 Justification for Selection

We chose Agile because:

- We was able to add functions one at a time **(URL scan → File scan → History → Login)**
- Problems were easier to acquire and correct early.
- We was able to update the system any time if something changed.
- It allowed me to complete the project better and faster .

Agile made **ViroScan** more flexible, easy, and friendly to use.

5. SYSTEM DESIGN

5.1 System Architecture Diagram

This diagram presents all the principal details of the system and their interaction. **The browser is used by the user, server does processing , data is stored in the database and VirusTotal check malware.** It depicts the flow of data among these sections.

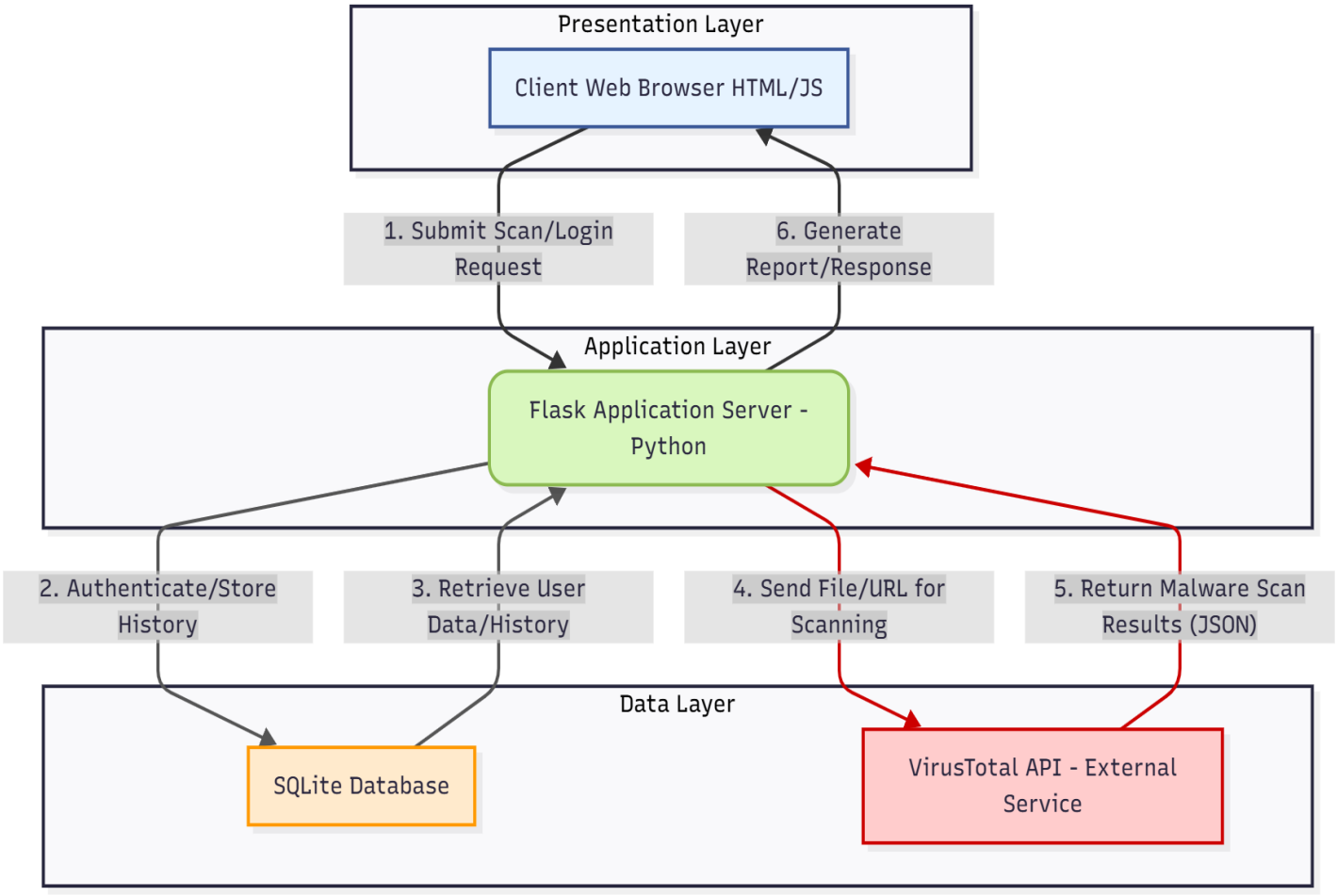


FIGURE 2 : SYSTEM ARCHITECTURE DIAGRAM SHOWING THE MAIN COMPONENTS AND THEIR INTERACTIONS.

5.2 Use Case Diagram

This diagram indicates what the users are allowed to do within the system. **Example: sign up , log in, scan files/URLs and scan history.** Admins can manage reports. It help us to know the system's features.

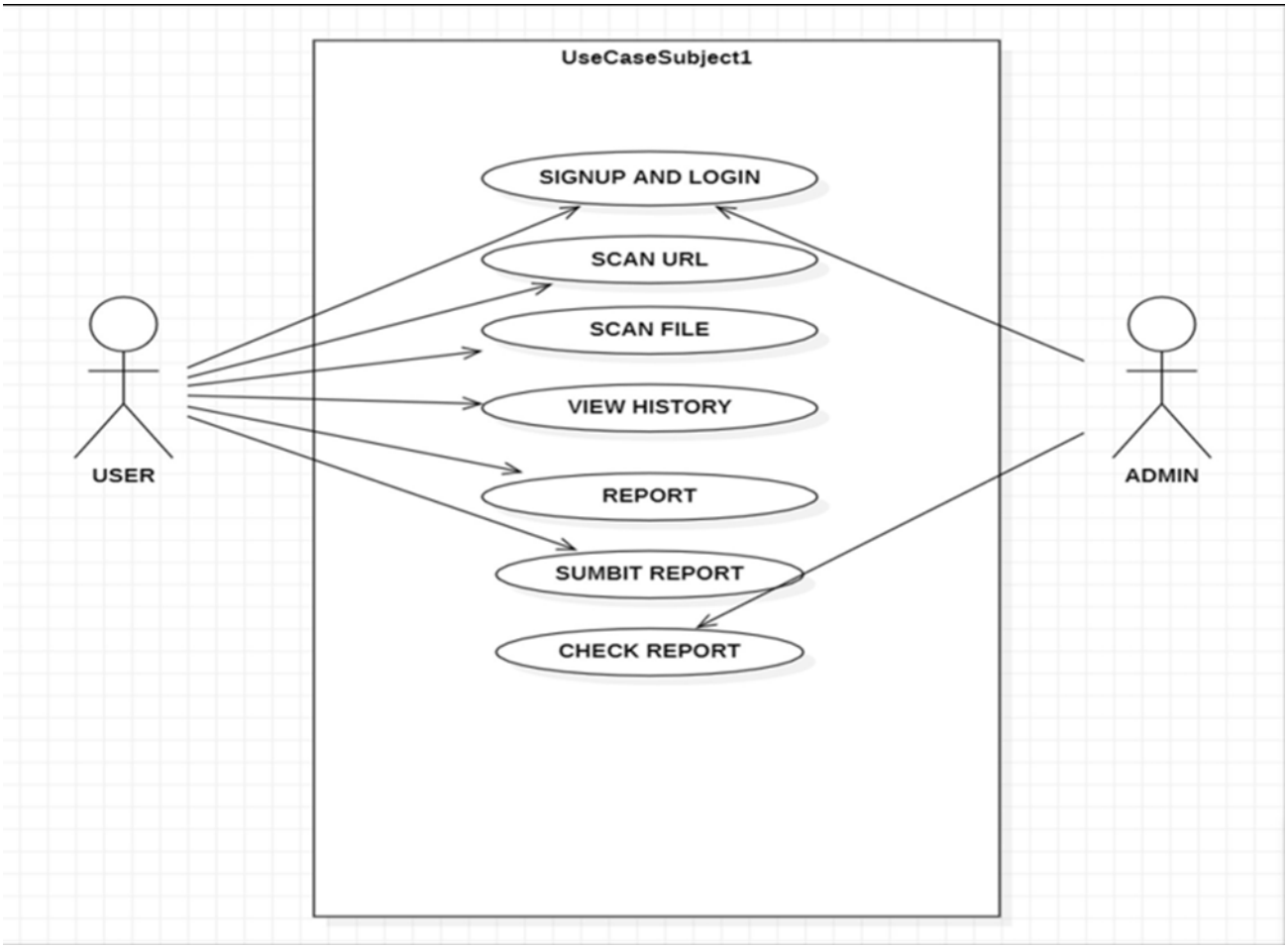


FIGURE 3: USE CASE DIAGRAM

5.3 Activity Diagram

The following diagram illustrates how scanning works. **User logs in → Sends scan → VirusTotal checks → results saved → user view results.** It describes the flow of actions.

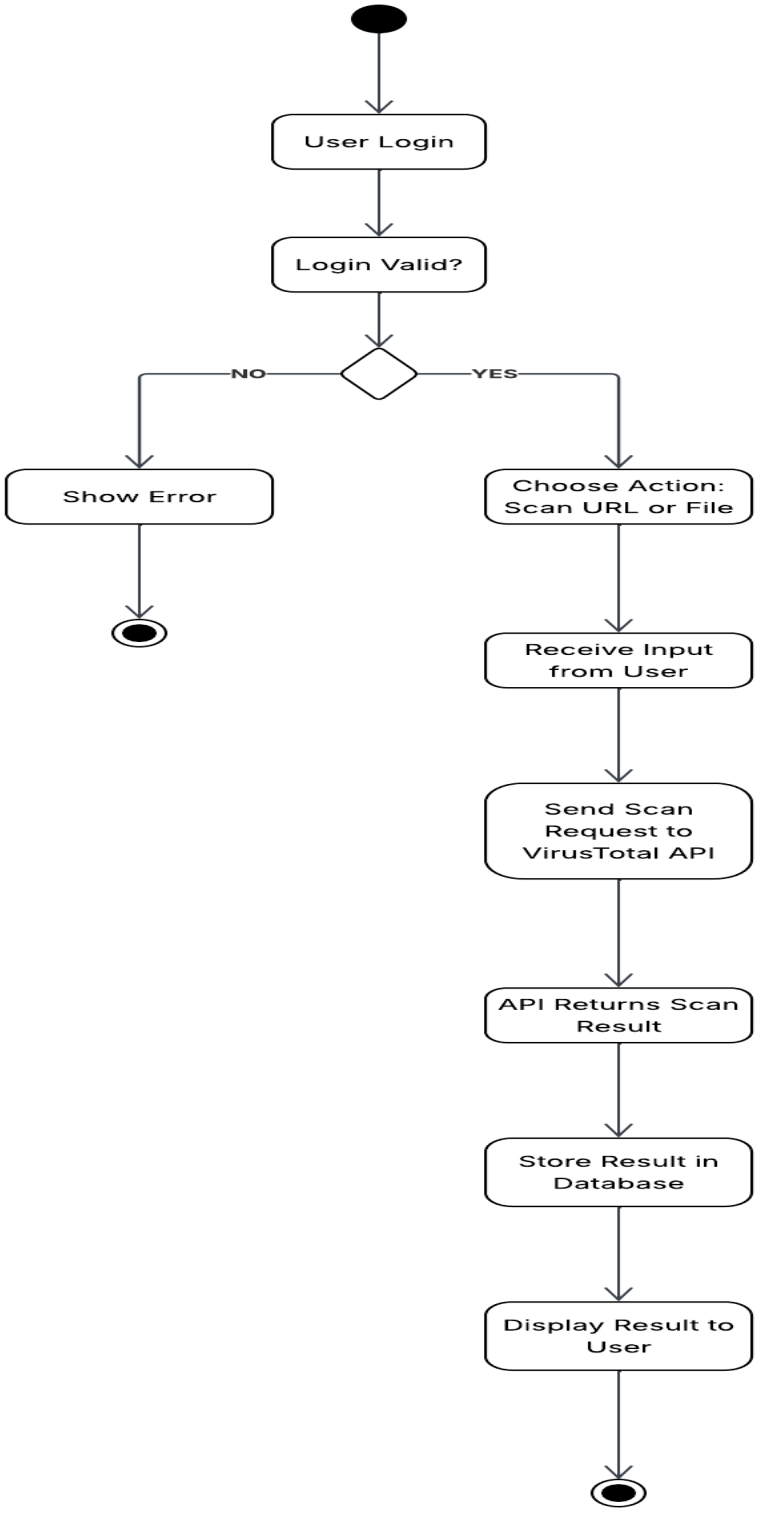


FIGURE 4: ACTIVITY DIAGRAM

5.4 Sequence Diagram

The following diagram illustrates the sequence of communication within the system components .**User → System → VirusTotal API → Database → back to User.** It demonstrates how messages move to accomplish task.

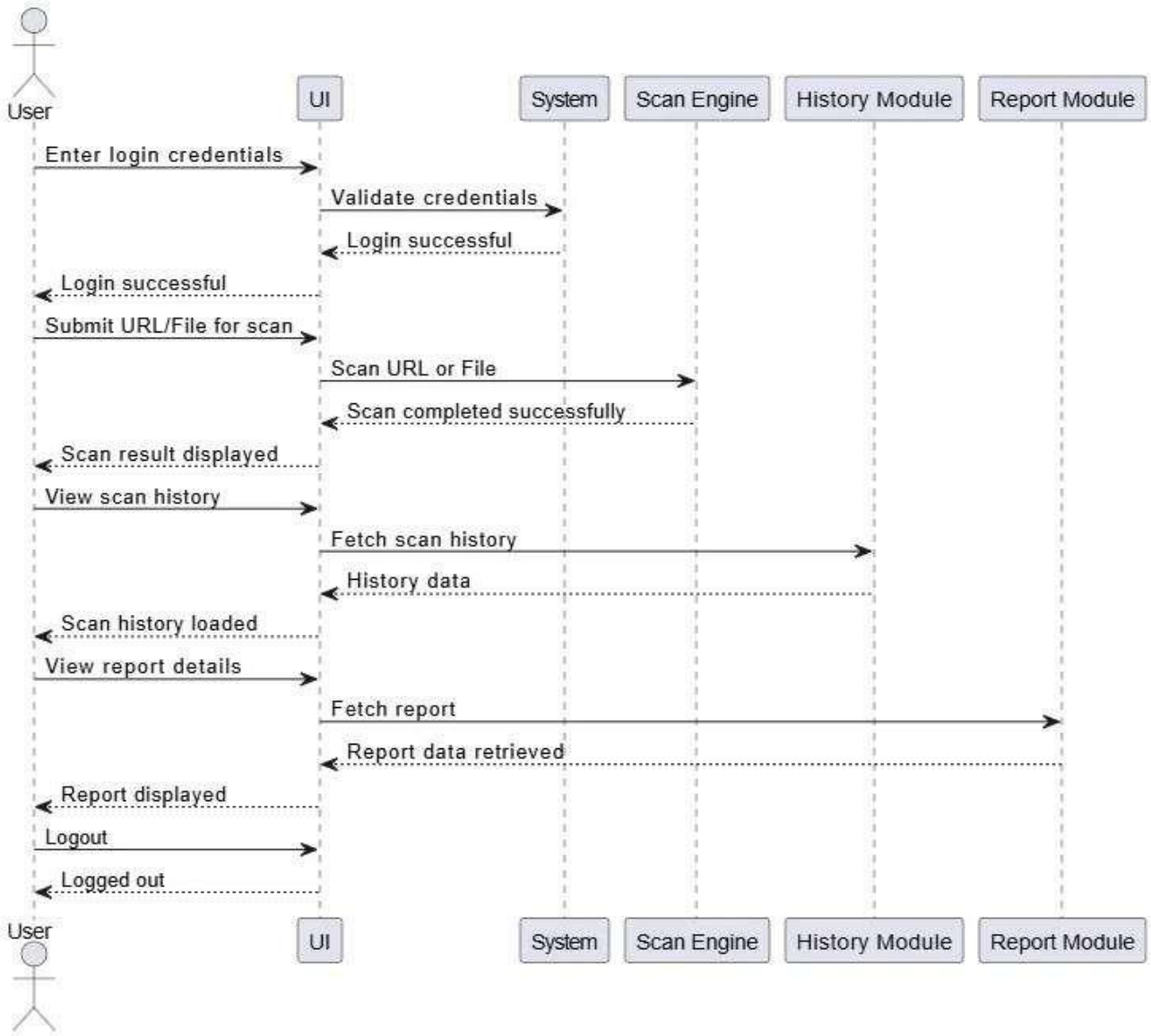


FIGURE 5: SEQUENCE DIAGRAM

6. DATABASE DESIGN

6.1 ER Diagram

The ERD displays the main database tables that we are using in our system and how they relate to one another.

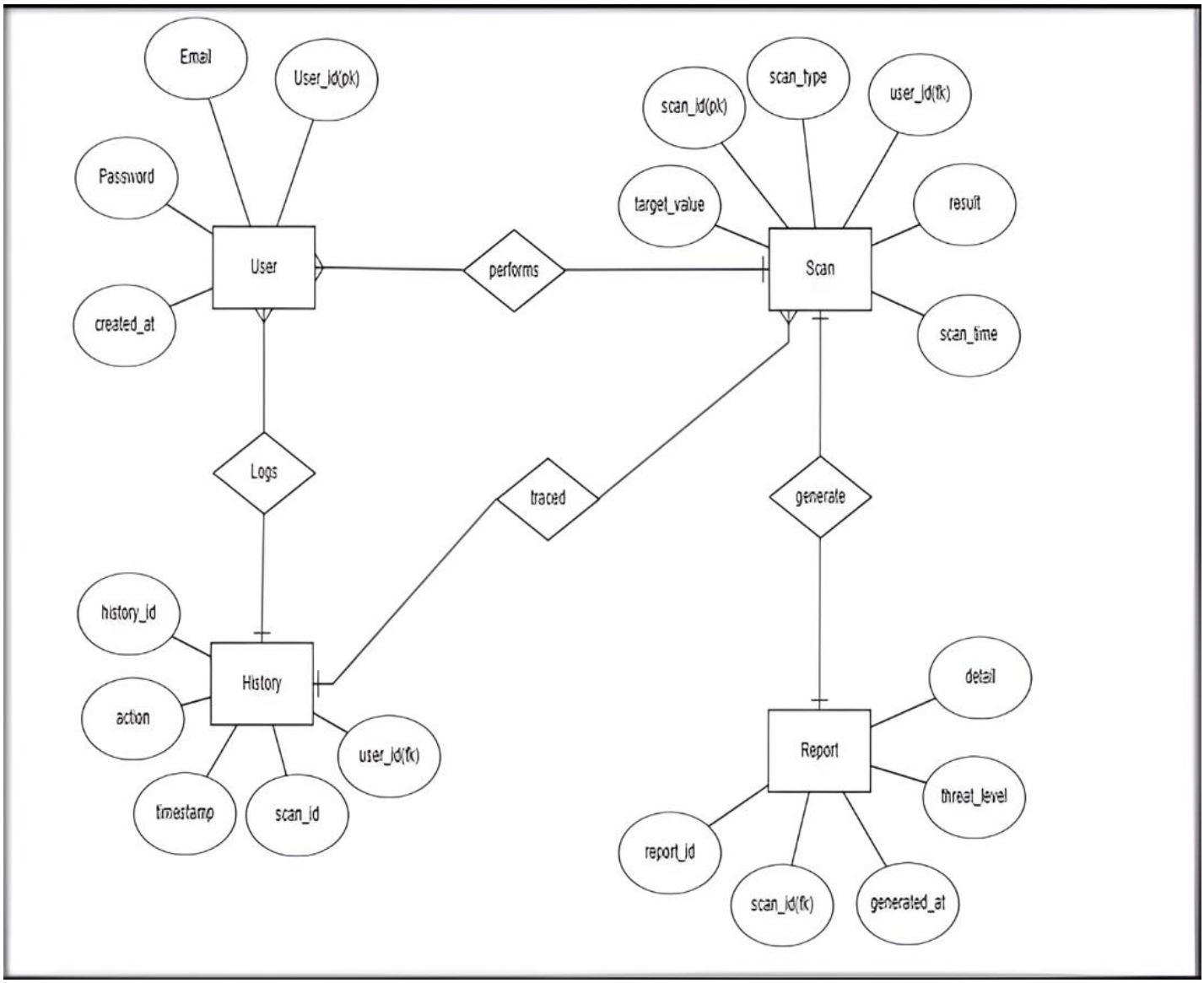


FIGURE 6 : ERD DIAGRAM

6.2 Database Tables Description

The database in the system contains four tables:

Table Name	Purpose	Fields
User	Stores login information of each user.	created_at, password, email, user id.
Scan	Stores information of each file or URL scanned.	scan_id, scan_type, target_value, user_id, result, scan_time.
History	Logs activities that have been done by users (like scanning).	history_id, action, timestamp, user_id, scan_id.
Report	Stores more elaborate results per scan if need needed.	report_id, scan_id, detail, threat_level, generated_at

TABLE 1 : DATABASE TABLES WITH THEIR PURPOSE AND FIELDS

7. IMPLEMENTATION

7.1 Technologies Used

In order to create ViroScan, we used:

- **Python Flask :** Connecting to **VirusTotal** for the backend.
- **HTML, CSS, and JavaScript:** The website pages are designed and constructed.
- **SQLite :** To store user information and scan history
- **VirusTotal API :** To check files and URLs for threats.
- **Visual Studio Code :** The primary tool for writing code .

These tools were appropriate for developing a web security system and were simple to learn.

7.2 Key Features

The following features are offered by ViroScan:

- Use URL scanning to determine whether a website is dangerous or safe.
- To check for malware, upload files.
- Examine every scan on a history page.
- View a detailed report for every scan.
- To secure access, the user must log in.
- A straightforward and user-friendly interface

7.3 Architectural Pattern

A **client-server architecture** was used. The main work is performed by the server which includes scanning and storage of results, and the user is the one who works with the web site (client). The results are returned to the client after the communication with the **VirusTotal** database and API. This tendency simplifies the process of managing the system, securing it and updating it.

7.4 Data Structures & Algorithms

Python We primarily used lists and dictionaries to manage and store data. These made it easier for us to format and temporarily store scan results before storing them in the database.

When displaying scan history or verifying values returned from the **API, basic algorithms like searching, sorting, and filtering** were employed. These minor tasks keep the data organized and speed up the system.

7.5 API Integration (VirusTotal)

Our system is linked to **VirusTotal API** to determine whether the files or URLs are safe. **VirusTotal** checks numerous antivirus engines hence we do not have to develop a scanner. This contributes to the improved precision of the results and makes the project more safe and professional.

7.6 Security Measures

We have implemented a number of safety techniques:

- The purpose of hashing real passwords is to avoid storing passwords.
- Blocking malicious data entry.
- Protection of data on transfer through HTTPS.
- Protect API key so that nobody abuses the service.
- User authentication mechanism to limit functionality.

Such measures prevent the death of user data and cyber attacks.

8.TESTING & RESULTS

8.1 Testing Strategy

The three kinds of testing we conducted to test the system are:

1. **Unit Testing :**

   We tried the features individually such as **login, file scan and URL scan** to ensure that they were functioning properly.

2. **Integration Testing :**

   We had tested how the **web page, server, database, and VirusTotal API** work with no issues.

3. **Manual Testing :**

   We tested the system on the browser to see whether the buttons, forms and results are presented correctly.

Testing Goal :

So that the system provides the right results, does not crash and that the user data remains safe.

8.2 Test Cases & Outcomes

Test Case	Expected Output	Result	Status
URL Scan	Show safe/malicious result	Correct result displayed	Pass
File Upload Scan	VirusTotal report returned	Correct report & counts shown	Pass
Login	Only valid users allowed	Invalid users blocked	Pass
History View	All scans stored and shown	Table displayed correctly	Pass
Empty Input Error	Warning should appear	Alert shown correctly	Pass
Database Insert	Scan stored properly	Data saved without error	Pass
External API	API should respond	Successful response	Pass

TABLE 2 : TEST CASES WITH EXPECTED OUTPUT, RESULT, AND STATUS

8.3 Testing Results

Every test was successfully performed. The scan findings were accurate and timely. The system never failed and all information was duly stored in the database. There were error messages when necessary and the VirusTotal connection was effective.

9. CONCLUSION

ViroScan is efficient in **scanning file and URLs with VirusTotal**. It displays the definite results and stores each scan in the database. The system is easy to operate and enhances the safety of users over the internet. More features can be added to the application in the future.

10. REFERENCES

- VirusTotal (2024) *VirusTotal API Documentation*. Available at: https://developers.virustotal.com (Accessed: 07 December 2025).
- Flask (2024) *Flask Documentation*. Available at: https://flask.palletsprojects.com (Accessed: 07 December 2025).
- SQLite (2024) *SQLite Documentation*. Available at: https://www.sqlite.org/docs.html (Accessed: 07 December 2025).

11. APPENDIX

11.1 System Screenshots

1. New users are required to create an account in order to use the system.

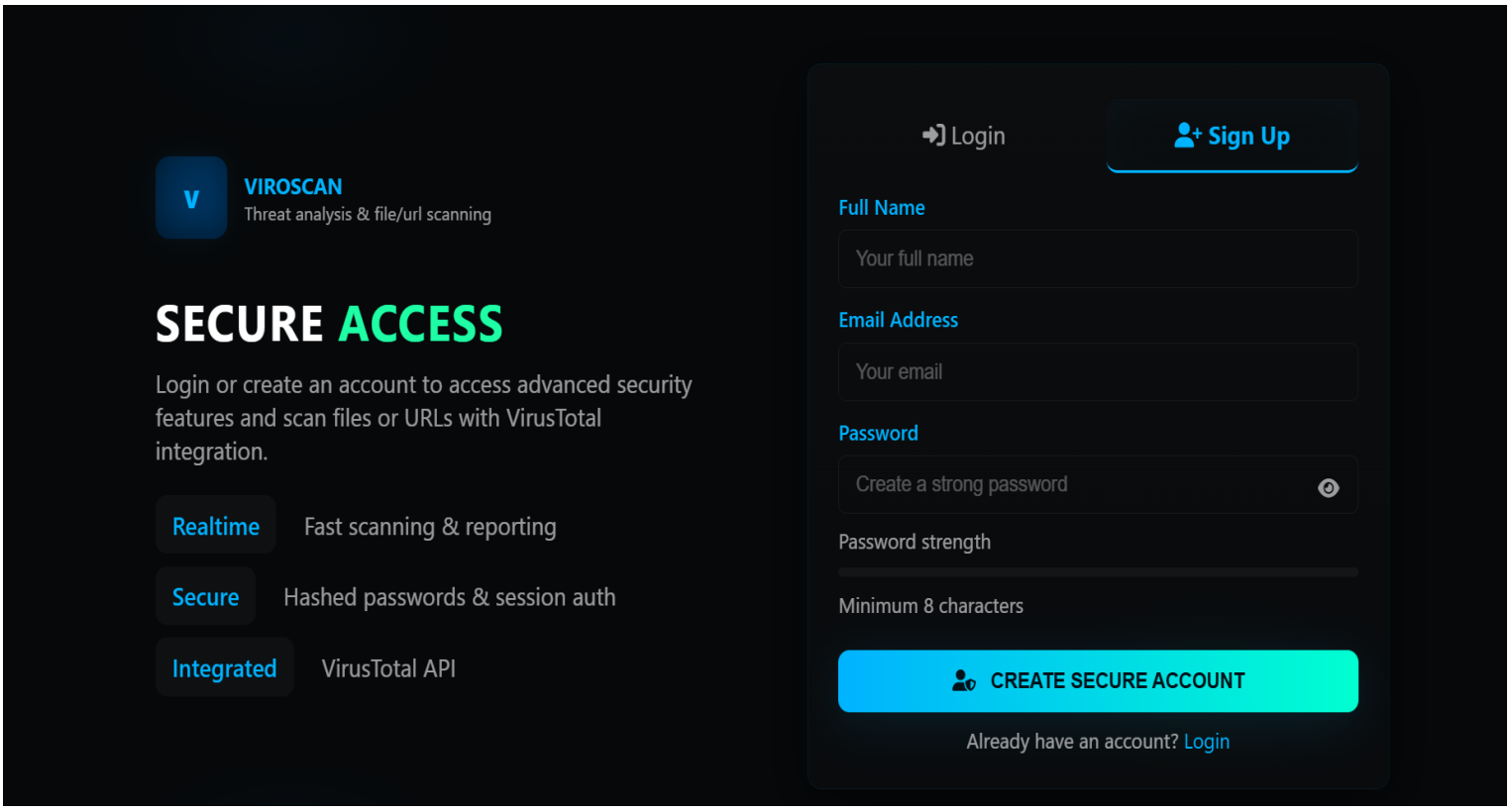


FIGURE 1: LOGIN PAGE

2. To use the system, new users must first create an account.

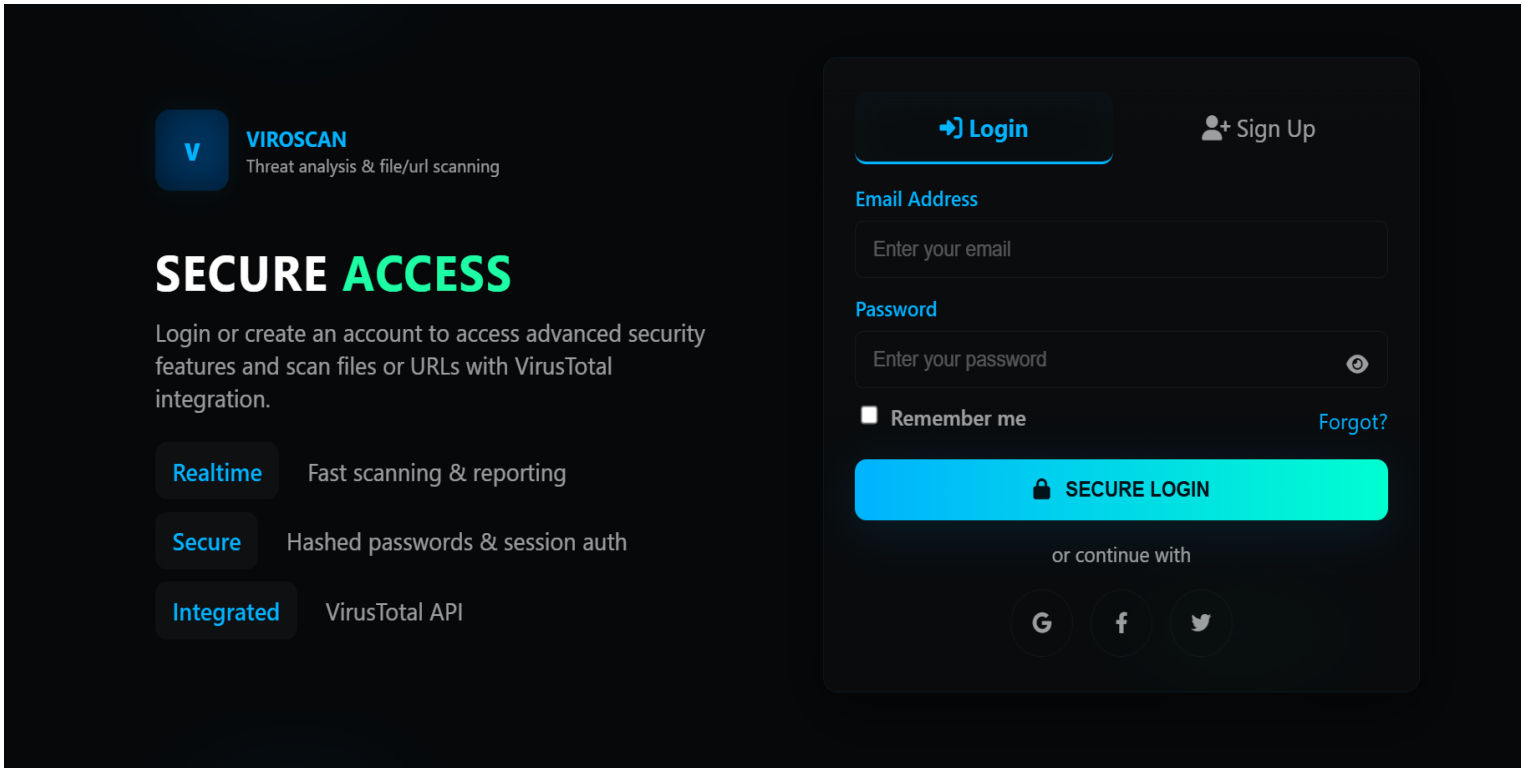


FIGURE 2: SIGNUP PAGE

3. Users are able to enter a website link in order to ascertain whether the site is secure or unsafe.

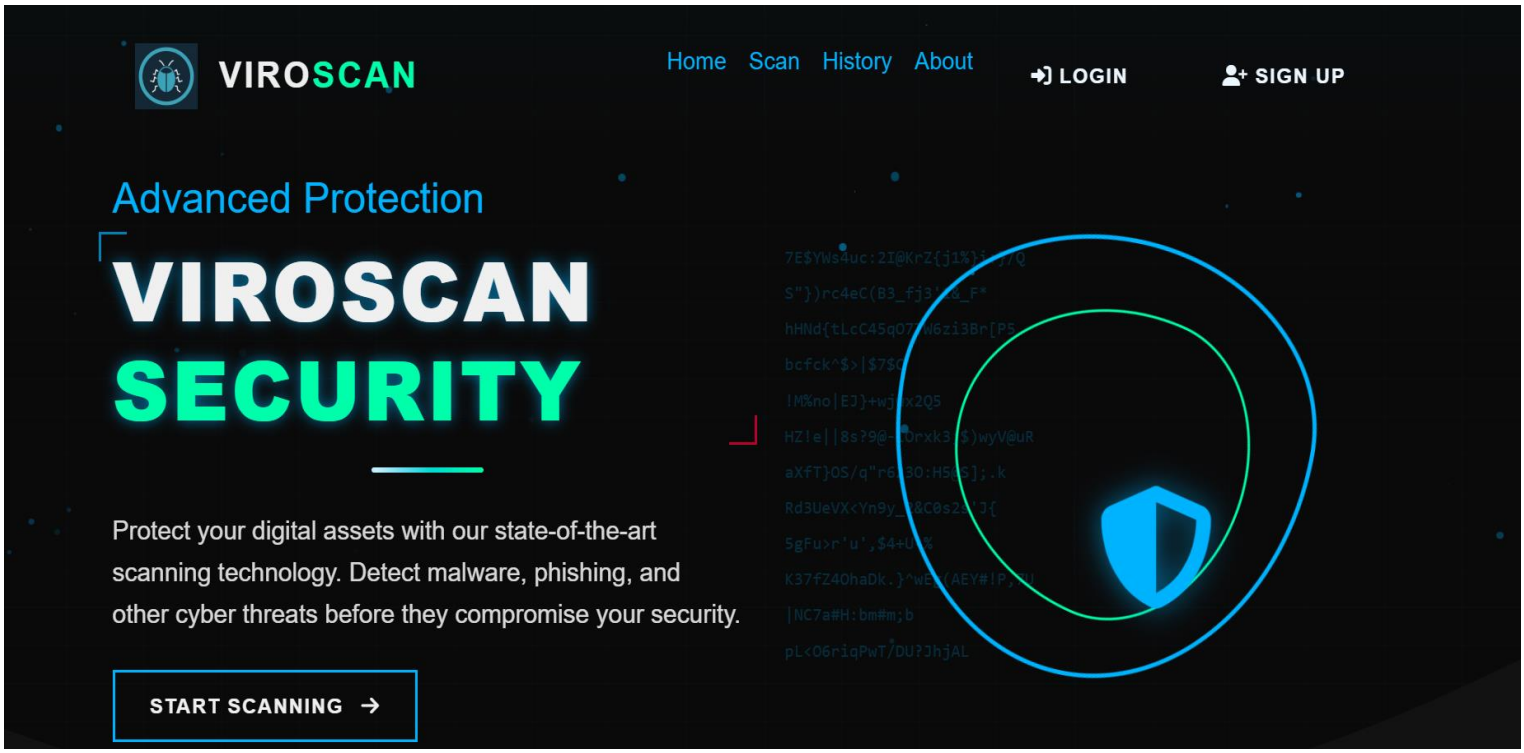


FIGURE 3: DASHBOARD / HOME PAGE

4. To determine whether a website link is safe or dangerous, users can enter it.

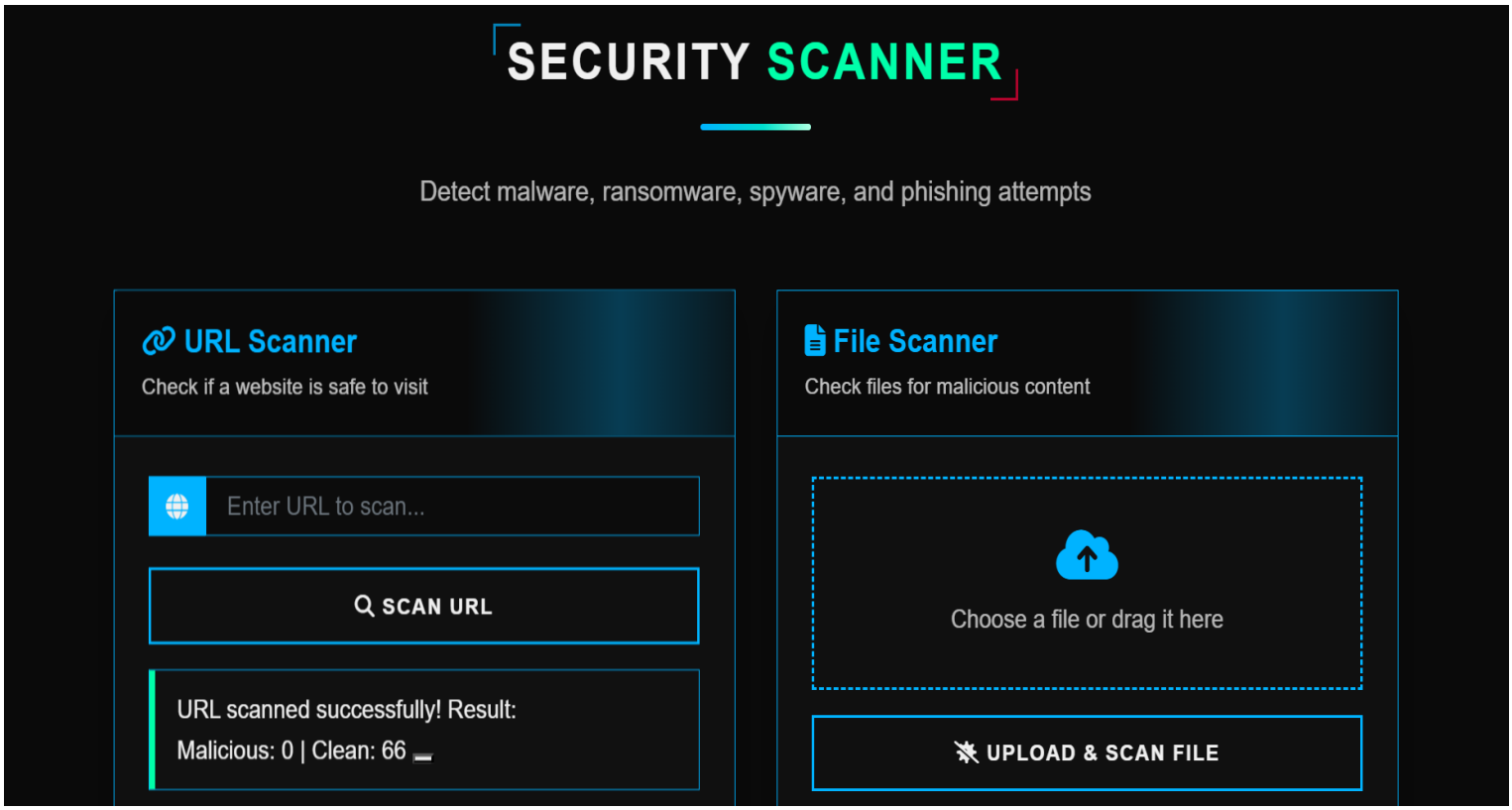


FIGURE 4: URL SCAN SCREEN

5. To check for malware, users can upload a file.

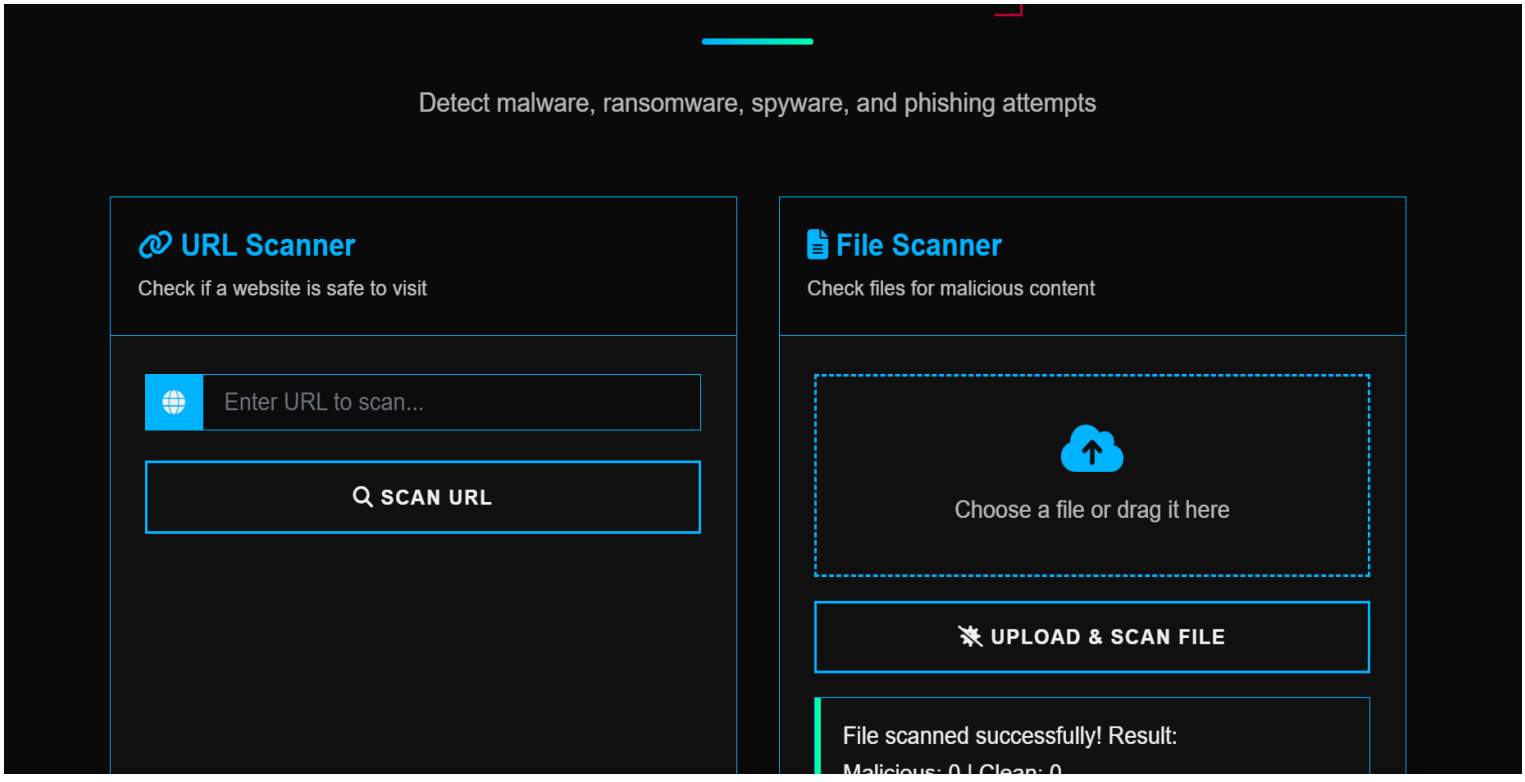
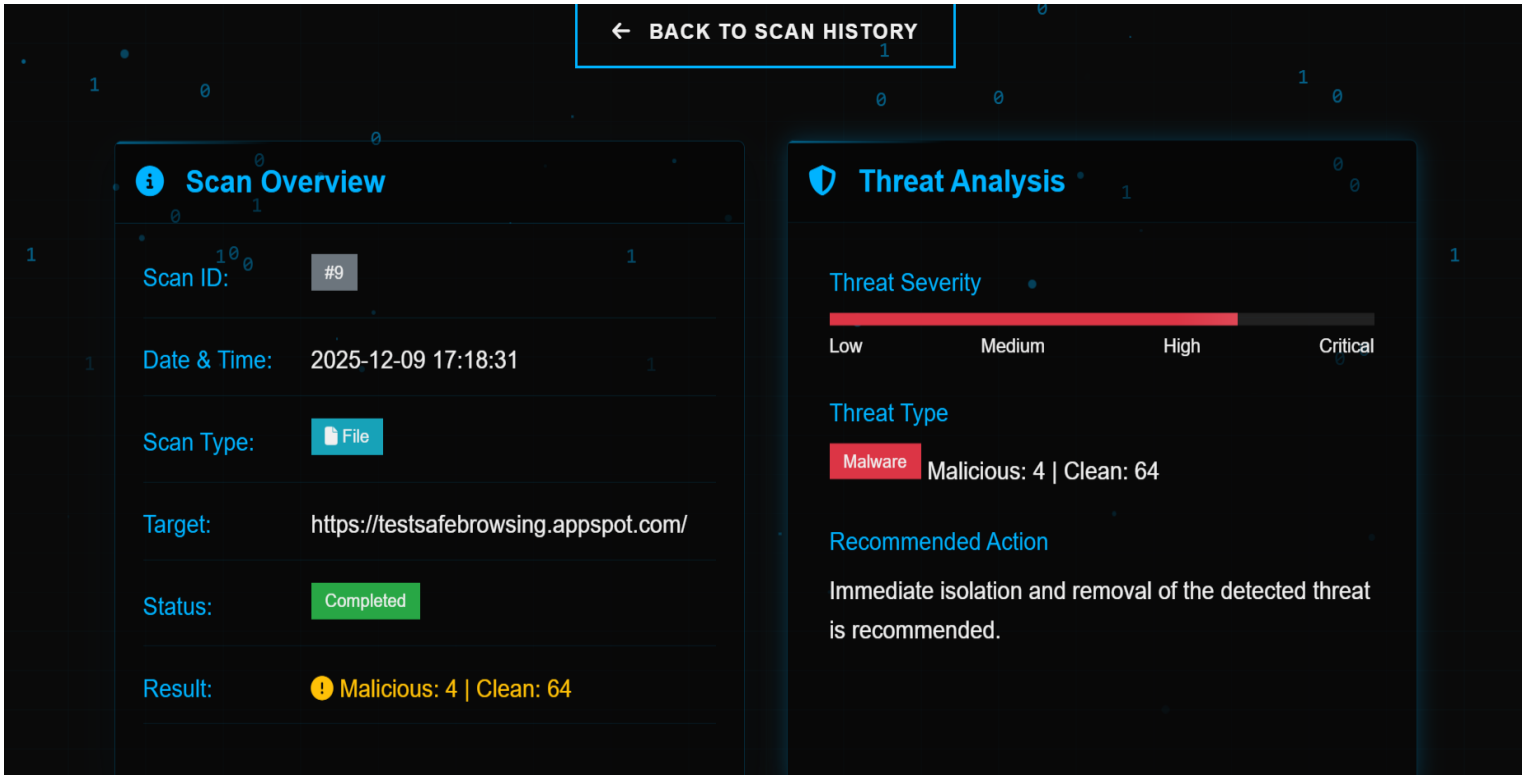


FIGURE 5: FILE SCAN SCREEN

6. The scan results are in a clear manner thus showing whether they are malicious or safe.



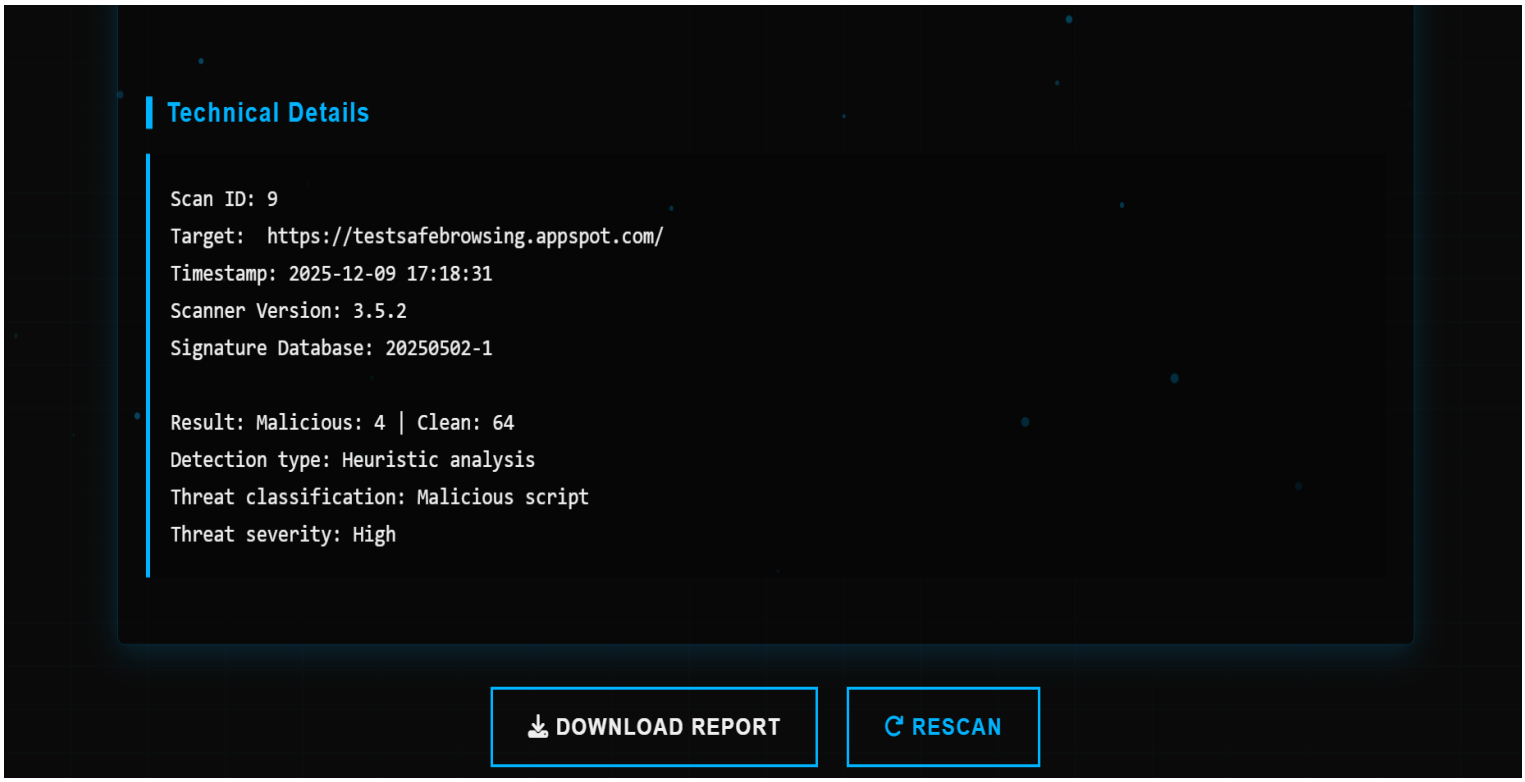


FIGURE 6: SCAN RESULTS PAGE

7. Every prior scan is stored and displayed for the user to examine at a later time.

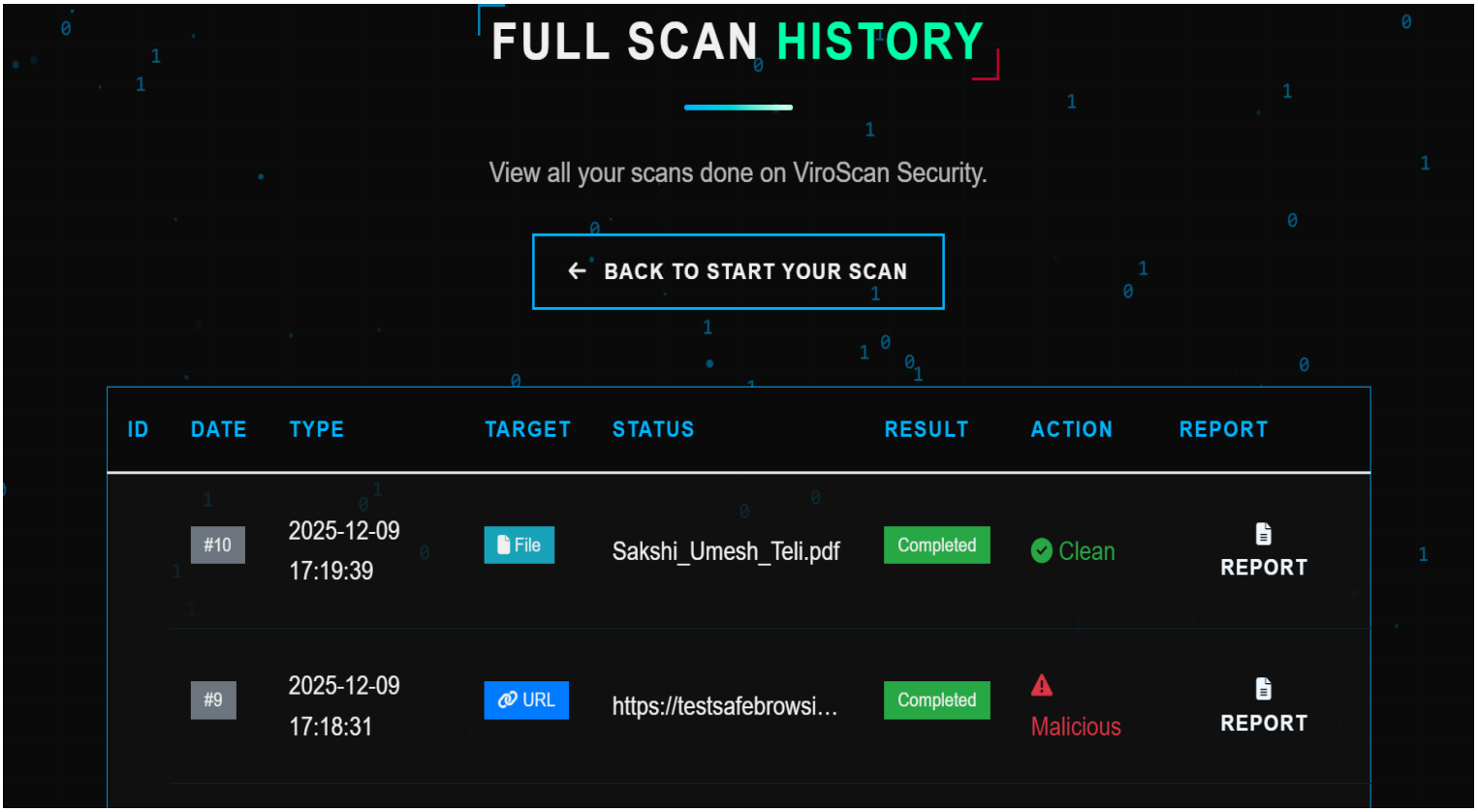
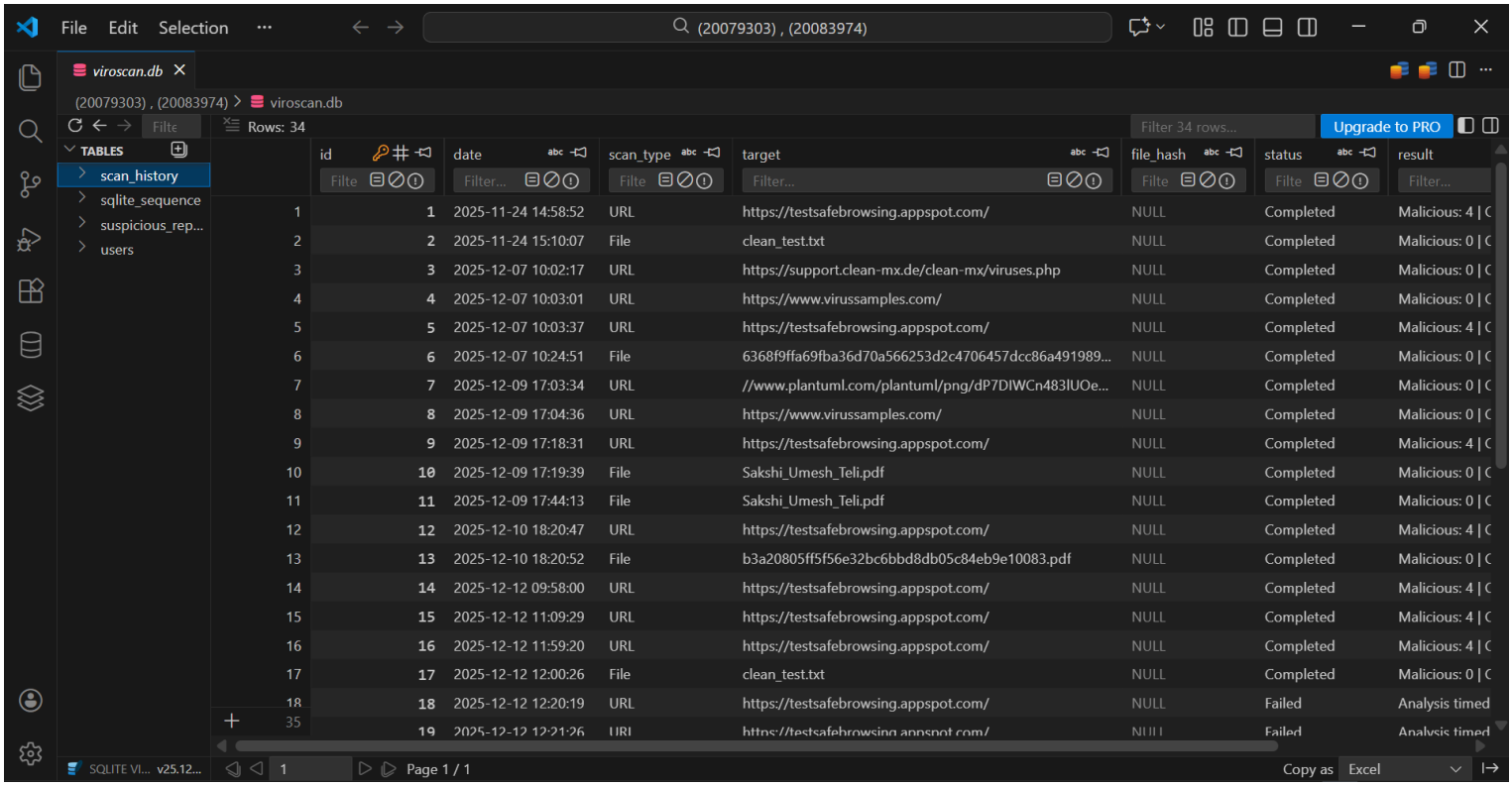


FIGURE 7: SCAN HISTORY PAGE

8. Stores user and scan data in a structured manner.



viroscan.db

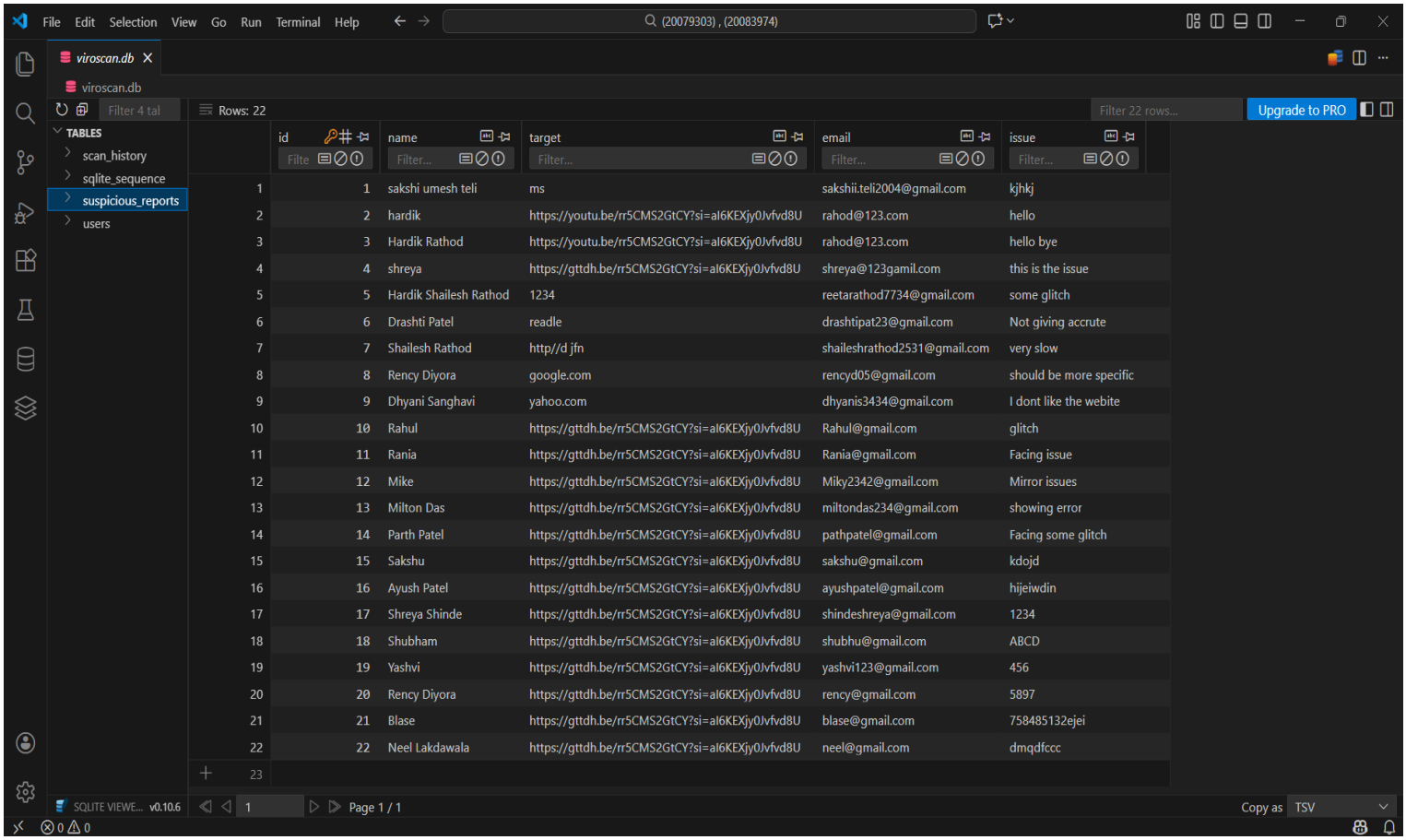
Rows: 34

TABLES: scan_history, sqlite_sequence, suspicious_rep..., users

	id	date	scan_type	target	file_hash	status	result
1	1	2025-11-24 14:58:52	URL	https://testsafebrowsing.appspot.com/	NULL	Completed	Malicious: 4 | C
2	2	2025-11-24 15:10:07	File	clean_test.txt	NULL	Completed	Malicious: 0 | C
3	3	2025-12-07 10:02:17	URL	https://support.clean-mx.de/clean-mx/viruses.php	NULL	Completed	Malicious: 0 | C
4	4	2025-12-07 10:03:01	URL	https://www.virussamples.com/	NULL	Completed	Malicious: 0 | C
5	5	2025-12-07 10:03:37	URL	https://testsafebrowsing.appspot.com/	NULL	Completed	Malicious: 4 | C
6	6	2025-12-07 10:24:51	File	6368f9ffa69fba36d70a566253d2c4706457dcc86a491989...	NULL	Completed	Malicious: 0 | C
7	7	2025-12-09 17:03:34	URL	//www.plantuml.com/plantuml/png/dP7DIWCn483lUOe...	NULL	Completed	Malicious: 0 | C
8	8	2025-12-09 17:04:36	URL	https://www.virussamples.com/	NULL	Completed	Malicious: 0 | C
9	9	2025-12-09 17:18:31	URL	https://testsafebrowsing.appspot.com/	NULL	Completed	Malicious: 4 | C
10	10	2025-12-09 17:19:39	File	Sakshi_Umesh_Teli.pdf	NULL	Completed	Malicious: 0 | C
11	11	2025-12-09 17:44:13	File	Sakshi_Umesh_Teli.pdf	NULL	Completed	Malicious: 0 | C
12	12	2025-12-10 18:20:47	URL	https://testsafebrowsing.appspot.com/	NULL	Completed	Malicious: 4 | C
13	13	2025-12-10 18:20:52	File	b3a20805ff5f56e32bc6bbd8db05c84eb9e10083.pdf	NULL	Completed	Malicious: 0 | C
14	14	2025-12-12 09:58:00	URL	https://testsafebrowsing.appspot.com/	NULL	Completed	Malicious: 4 | C
15	15	2025-12-12 11:09:29	URL	https://testsafebrowsing.appspot.com/	NULL	Completed	Malicious: 4 | C
16	16	2025-12-12 11:59:20	URL	https://testsafebrowsing.appspot.com/	NULL	Completed	Malicious: 4 | C
17	17	2025-12-12 12:00:26	File	clean_test.txt	NULL	Completed	Malicious: 0 | C
18	18	2025-12-12 12:20:19	URL	https://testsafebrowsing.appspot.com/	NULL	Failed	Analysis timed
19	19	2025-12-12 12:21:26	URL	https://testsafebrowsing.appspot.com/	NULL	Failed	Analysis timed

Page 1 / 1 — Copy as Excel

FIGURE 8 : STRUCTURE OF THE SCAN_HISTORY TABLE



viroscan.db

Rows: 22

TABLES: scan_history, sqlite_sequence, suspicious_reports, users

	id	name	target	email	issue
1	1	sakshi umesh teli	ms	sakshii.teli2004@gmail.com	kjhkj
2	2	hardik	https://youtu.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	rahod@123.com	hello
3	3	Hardik Rathod	https://youtu.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	rahod@123.com	hello bye
4	4	shreya	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	shreya@123gamil.com	this is the issue
5	5	Hardik Shailesh Rathod	1234	reetarathod7734@gmail.com	some glitch
6	6	Drashti Patel	readle	drashtipat23@gmail.com	Not giving accrute
7	7	Shailesh Rathod	http//d jfn	shaileshrathod2531@gmail.com	very slow
8	8	Rency Diyora	google.com	rencyd05@gmail.com	should be more specific
9	9	Dhyani Sanghavi	yahoo.com	dhyanis3434@gmail.com	I dont like the webite
10	10	Rahul	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	Rahul@gmail.com	glitch
11	11	Rania	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	Rania@gmail.com	Facing issue
12	12	Mike	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	Miky2342@gmail.com	Mirror issues
13	13	Milton Das	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	miltondas234@gmail.com	showing error
14	14	Parth Patel	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	pathpatel@gmail.com	Facing some glitch
15	15	Sakshu	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	sakshu@gmail.com	kdojd
16	16	Ayush Patel	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	ayushpatel@gmail.com	hijeiwdin
17	17	Shreya Shinde	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	shindeshreya@gmail.com	1234
18	18	Shubham	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	shubhu@gmail.com	ABCD
19	19	Yashvi	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	yashvi123@gmail.com	456
20	20	Rency Diyora	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	rency@gmail.com	5897
21	21	Blase	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	blase@gmail.com	758485132ejei
22	22	Neel Lakdawala	https://gttdh.be/rr5CMS2GtCY?si=aI6KEXjy0Jvfvd8U	neel@gmail.com	dmqdfccc

Page 1 / 1 — Copy as TSV

FIGURE 9 : STRUCTURE OF THE SUSPICIOUS_REPORTS TABLE

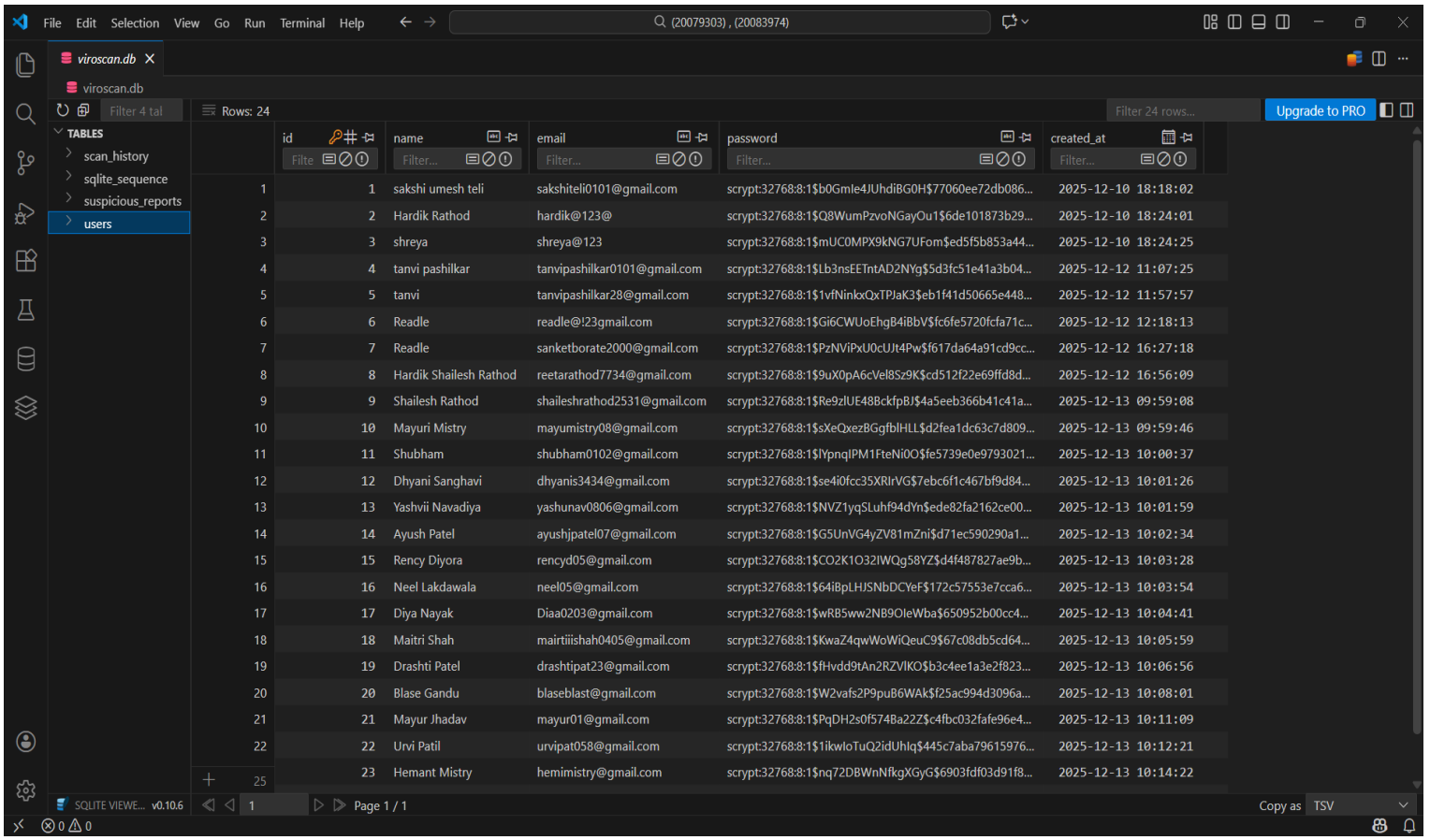


FIGURE 10 : STRUCTURE OF THE USERS TABLE

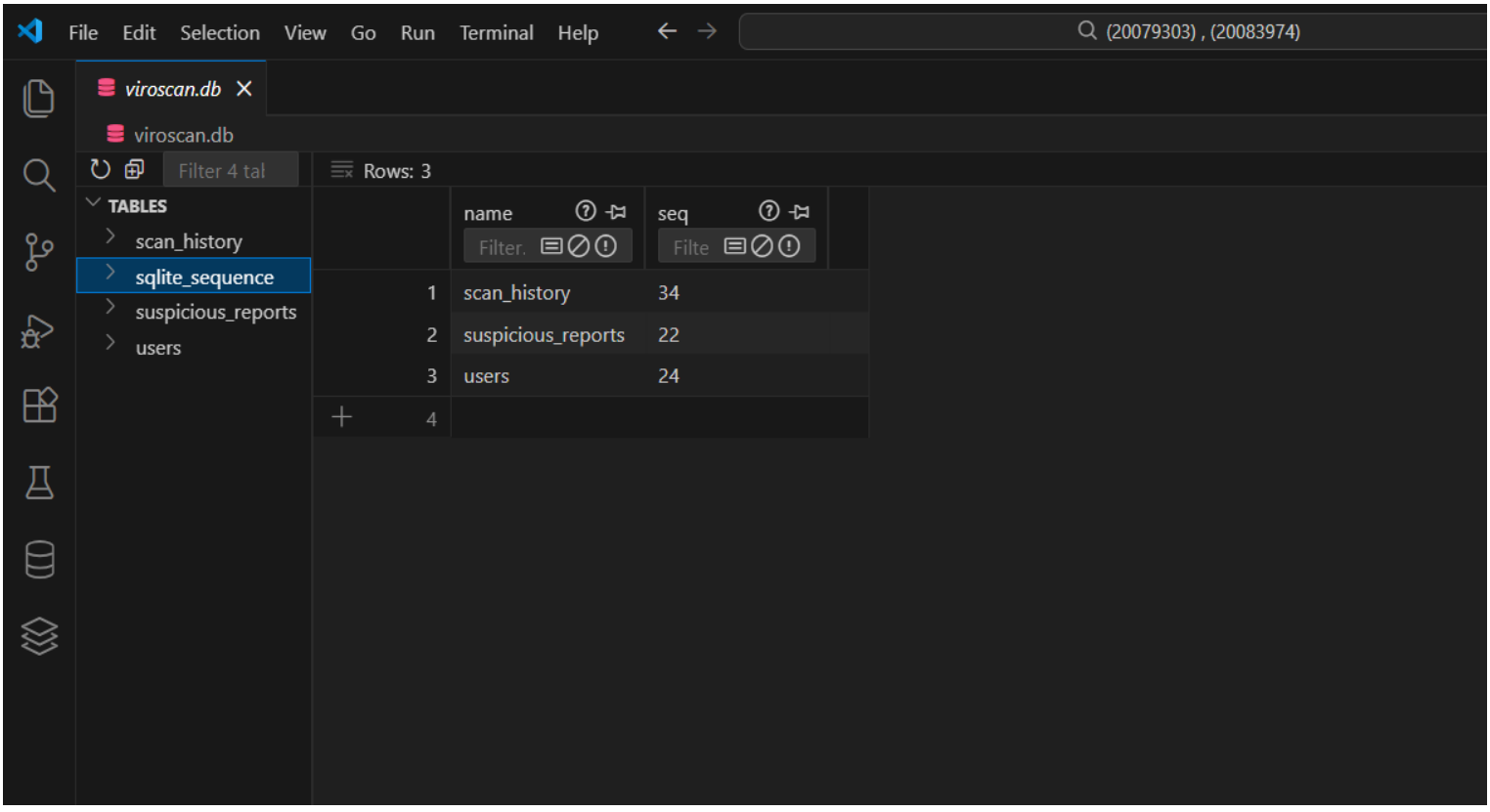


FIGURE 11: STRUCTURE OF THE SQLITE SEQUENCE SYSTEM TABLE

11.2 AI Assistance Statement

For this assignment, ChatGPT was a useful tool. It made it easier for us to comprehend technical concepts, write explanations, and deal with code errors. Before incorporating it into our project, we reviewed and revised everything. We complete the last piece of work. As requested, chat transcripts are provided.

11.3 Team Contribution Statement

To finish this project, both team members collaborated. We fairly divided up the tasks:

Sakshi Umesh Teli (20079303): Wrote the majority of the report and worked on the user interface, login system, and VirusTotal API integration.

Hardik Shailesh Rathod (20083974): Helped in provision of the database, testing, capturing screen shots, and preparing diagrams of the report.

The project was completed together via discussion and each member contributed equally to the final product.

11.4 GitHub Repository Link

https://github.com/sakshi220122/viroScan.git

11.5 Video Link

https://youtu.be/ydDqTS--PSY