Perancangan Aplikasi Deteksi Kerentanan *Sql Injection* Dan *Cross-Site Script* Pada Aplikasi Berbasis *Website* Menggunakan Metode *Waterfall*

Herwanto Patah[1], Mustofa Zaid[2]
Program Studi Teknik Informatika
STMIK Indonesia Mandiri, Jl.Jakarta No.79 Bandung
Email : Penulis[1], zaidmustofa@stmik-im.ac.id[2]

ABSTRAK

Tingginya akses terhadap *website* ini diiringi juga dengan tingginya pula tingkat keamanan nya. Menguji sistem keamanan sangat penting dari sekian faktor penyebab kurangnya keamanan *website* salah satunya adalah kesalahan penulisan kode program. Berbagai jenis serangan biasanya digunakan untuk menemukan celah keamanan seperti *SQL Injection* dan *Cross-site Script*.

Perancangan aplikasi menggunakan pendekatan *waterfall* dimulai dengan mengalisis masalah pada penelitian sebelumnya yaitu terdapat kekurangan pada metode *crawling* sehingga proses pencarian kerentanan menjadi kurang efektif. Dengan demikain membuat penulis tertarik dalam melakukan penelitian untuk merancang aplikasi untuk mendeteksi kerentanan pada jenis serangan *SQL injection* dan *Cross-Site Script* dan menganalisis kebutuhan untuk perancangan aplikasi yang akan dibangun.

Aplikasi yang dirancang akan memudahkan bagi para pengembang *website* sehingga dapat menekan terjadinya serangan *SQL injection* dan *Cross-Site Script*. Pada tahap pengujian aplikasi berhasil mendeteksi kerentanan terhadap *SQL Injection* dan *Cross-Site Script* pada *website* yang telah ditentukan sebelumnya.

Kata kunci : *Website,* Kerentanan, Keamanan. *Sql Injection, Cross-Site Script*

ABSTARCT

High access to this website is also accompanied by a high level of security. Testing the security system is very important. Of the many factors that cause the lack of website security. One of which is the error in writing program code. Various types of attacks are commonly used to find vulnerabilities such as SQL Injection and Cross-site Script.

Designing applications using the waterfall approach begins with analyzing the problem in previous research, namely that there are deficiencies in the crawling method so that the vulnerability search process becomes less effective. Thus making the author interested in conducting research to design applications to detect vulnerabilities in the type of SQL injection and Cross-Site Script attacks and analyze the need for designing applications to be built.

This application is designed to make it easier for website developers so that they can suppress SQL injection and Cross-Site Script attacks. At the testing stage the application successfully detected vulnerabilities to SQL Injection and Cross-Site Script

*on predetermined websites.**Keyword :** vulnerability, security, website, Sql Injection, Cross-Site Script*

***Keyword :** vulnerability, security, website, Sql Injection, Cross-Site Script*

1. PENDAHULUAN

Di era new normal sekarang ini, semua orang bergantung pada internet. kehidupan berubah menjadi serba online. Sebagai contoh di sektor Pendidikan metode pembelajaran dialihkan menjadi secara online. Tingginya akses terhadap *website* ini diiringi juga dengan tingginya pula tingkat keamanan nya.

Berdasarkan informasi dari Pusat Opersi Keamanan Siber Nasional (Pusopskamsinas) dan Badan Siber Sandi Negara (BSSN) Tercatat 88.414.296 serangan siber yang terjadi dari 1 Januari hingga 12 April 2020. Terpantau 25.224.811 serangan di bulan Januari, kemudian tercatat 29.188.645 serangan di bulan Februari, lalu 26.423.989 serangan terjadi di bulan Maret hingga 12 April 2020. 7.576.851 serangan. (BSSN. 2020)

Menguji sistem keamanan sangat penting kurangnya keamanan website salah satunya adalah kesalahan penulisan kode program. Kesalahan saat menulis kode program dalam pembuatan *website* adalah sesuatu yang sering digunakan penyerang. Berbagai teknik serangan biasanya digunakan untuk menemukan celah keamanan. Jenis serangan ini termasuk *SQL Injection, Cross-Site Script*.

Berdasarkan *report* OWASP 2003 sampai 2017 tentang 10 risiko keamanan aplikasi bersis web paling kritis adalah serangan *SQL Injection* dan *Cross-Site Script* berada di urutan teratas

2. METODE PENELITIAN

2.1.1 Studi Pustaka

(Elu, A, M. 2013) Melakukan penelitian tentang perancangan aplikasi untuk mendeteksi kerentanan *SQL Injection* yang bernama *SVSQL Injection.* Aplikasi ini digunakan untuk mendeteksi keamanan *website* dari serangan *SQL Injection*. Aplikasi yang dirancang hanya menggunakan satu jenis serangan yaitu *SQL Injection.*

(Yudha, F. Panji, A, M. 2018) juga melakukan penelitian tentang perancangan aplikasi untuk menguji keamanan *website* terhadap kerentanan *Phising Cross-Site Scripting, SQL Injection,,* aplikasi yang dirancang menggunakan Bahasa *python*. Terdapat kekurangan pada teknik *crawling* hanya bisa menelusuri halaman depan dan metode *SQL Injection* dan *Cross-Site Script* pada *website* yang terdapat *page* "*id*" saja sedangkan ketika *website* tidak memiliki *page "id"* nya di anggap tidak rentan sehingga kurang efektif untuk mendeteksi kerentanan.

2.1.1.1 SQL Injection

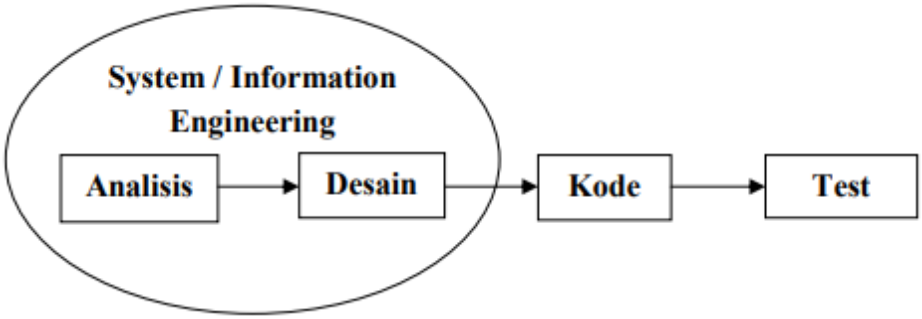
menurut Justin Clarke (2009) *SQL Injection* adalah teknik serangan yang mengeksploitasi kode dengan memodifikasi backend SQL dengan Memasukkan kalimat manipulasi. . Dengan mencoba manipulasi parameter pada url target dan menginputkan tanda petik satu (') pada akhiran url atau pada *form* jika *website* target menampikan peringantan kesalahan database

2.1.1.2 Cross-Site Script

Menurut (Seth Fogie. 2007) XSS adalah teknik serangan yang memaksa situs Web untuk menampilkan kode berbahaya, yang kemudian dijalankan di browser Web pengguna. Sekali penyerang memiliki rangkaian kontrol di browser Web pengguna, mereka dapat melakukan banyak tindakan jahat termasuk pembajakan akun, perekaman penekanan tombol, peretasan intranet, riwayat pencurian, dan sebagainya. Bagian ini menjelaskan berbagai cara yang mungkin dilakukan pengguna menjadi XSS dan mengontrak muatan malware JavaScript

2.1.2 Metode Pengembangan Perangkat Lunak

Pengembangan aplikasi yang dirancang menggunakan pendekatan *waterfall*



Gambar 1 Model *Waterfall*

1. Analisis

   Untuk menganalisis permasalahan pada penelitian sebelumnya dan apa saja yang dibutuhkan untuk merancang sebuah aplikasi.
2. Desain

   Tahap perancangan meliputi perancangan antarmuka dan rancangan komponen antarmuka, serta alur program perancangan
3. Kode

   Pada tahap ini, hasil desain di implementasikan sebagai baris kode program yang dapat dipahami oleh komputer.
4. Test

   Tahap pengujian dilakukan untuk menguji semua fungsi dari aplikasi yang telah di rancang

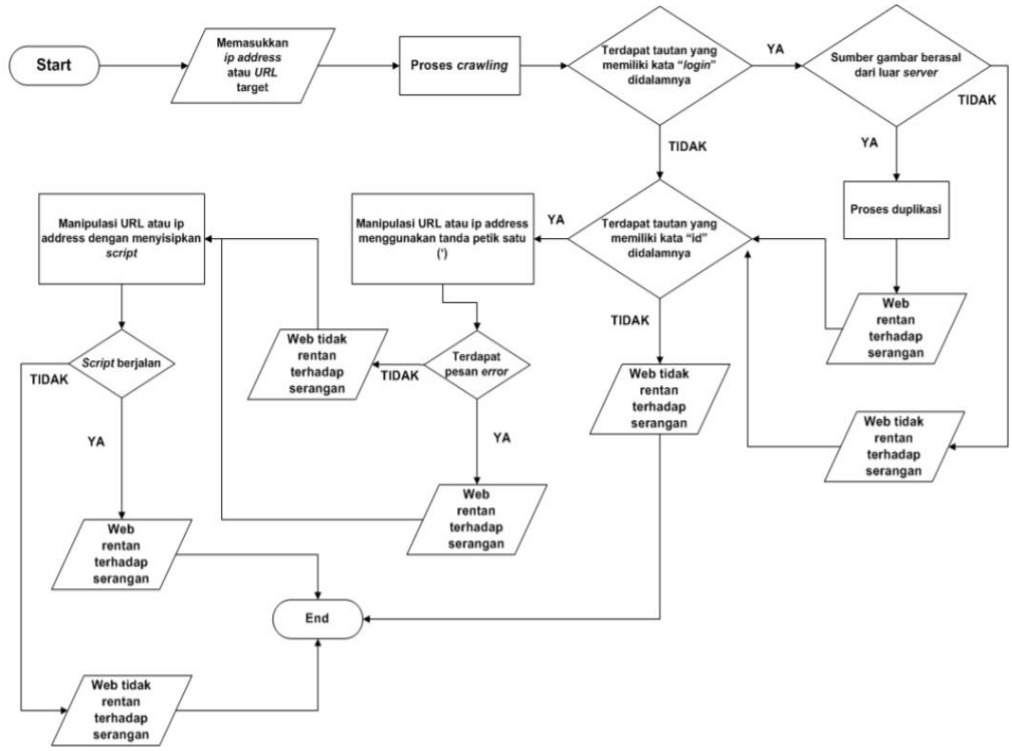
3. HASIL DAN PEMBAHASAN

3.1 Analisis Pada Penelitian Sebelumnya

Pada penelitian sebelumnya yaitu (Yudha. F., Panji. A.M. 2018) melakukan Perancangan Aplikasi Pengujian Celah Keamanan Pada Aplikasi Berbasis *Web*, aplikasi yang dibangun untuk mendeteksi celah keamanan pada jenis serangan *SQL Injection, Cross-Site Script* dan *phising* tetapi masih terdapat kekurangan pada metode pendeteksian celah keamanan yaitu :

1. Metode *crawling* hanya bisa menelusuri pada halaman depan
2. Metode *SQL Injection* dan *Cross-Site Script* menguji pada website yang terdapat page "*id*"

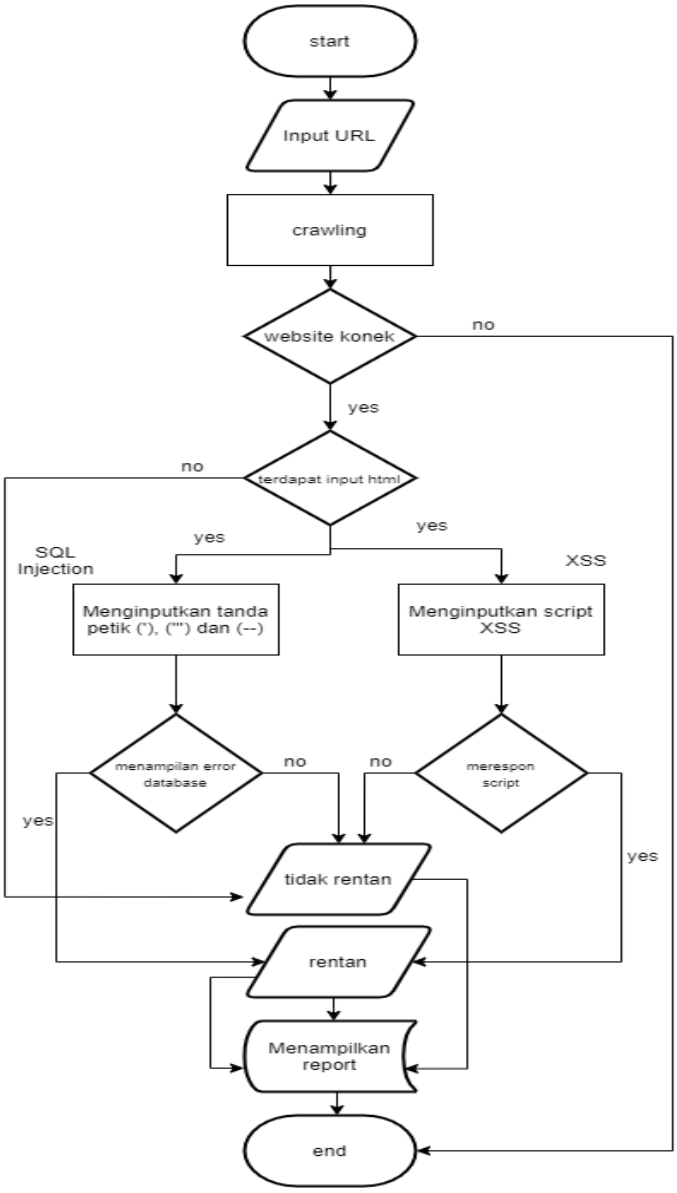
Sehingga menjadi kurang efektif dalam pendeteksian kerentanan pada *website* yang akan diuji



Gambar 3.1 Flowchart metode scan (Yudha. F., Panji. A.M. 2018)

3.2 Rancangan Sistem Yang Akan Dibangun

Berdasarkan analisis pada penelitian sebelumnya maka dibuatlah aplikasi untuk menutupi kekurangan tersebut dengan menggunakan Bahasa pemrograman php. Proses *phpcrawling* mampu merayapi semua konten yang ada di dalam website target dengan mencari *form input html.* Pada gambar 3.2 *flowchart* alur aplikasi untuk mendeteksi keamanan.



Gambar 3.2 *Flowchart* alur deteksi keamanan

Aplikasi yang dibangun memiliki 2 pengujian keamanan yaitu *Sql Injection* dan *Cross-Site Script.* Dengan dimulai dengan menginputkan alamat url target aplikasi akan melakukan *crawling* jika respon *website* tidak konek proses scanner akan berhenti, jika bukan maka selanjutnya aplikasi akan mencoba melakukan scan dengan metode yang ada yaitu *SQL Injection*, *Cross-Site Script*.

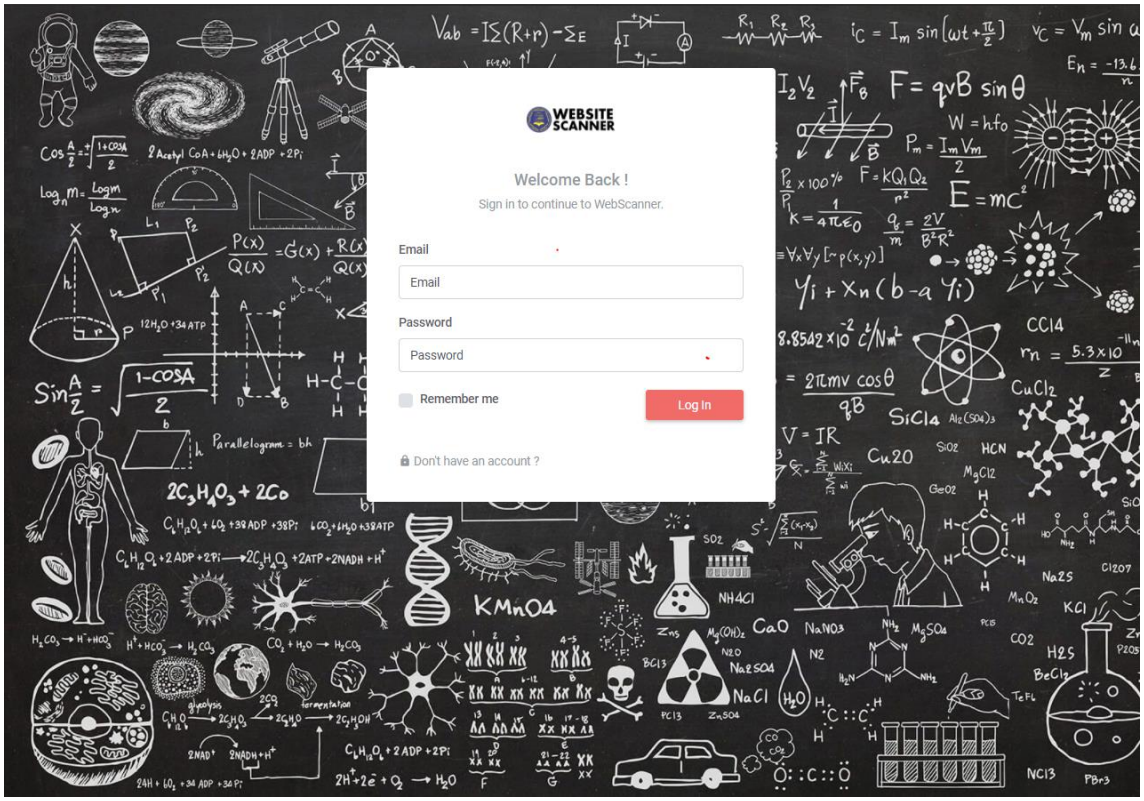
Pada pengujian *SQL Injection* aplikasi akan melakukan *crawling* terhadap semua konten website dan mencari halaman yang memiliki *form input html* lalu mencoba meninputkan tanda petik satu (‘) pada form tersebut dan mendeteksi pesan kesalahan

SQL error. Jika dalam hal Ada pesan kesalahan *SQL* di halaman, Kemudian aplikasi akan membaca dan memberi Informasi bahwa website target rentan serangan *SQL Injection*. Sebaliknya jika Pesan kesalahan *SQL* tidak terdeteksi, aplikasi akan menampilakn informasi bahwa *website* target tidak rentan terhadap serangan *SQL Injection.*

Berikutnya pada pengujian *Cross-Site Script* aplikasi akan melakukan *crawling* terhadap semua konten website dan mencari halaman yang memiliki *form input html* lalu mencoba meninputkan *script XSS* pada form tersebut dan mendeteksi respon *website*. Jika target website merespon kode dengan menampilakan *script* di halaman, kemudian aplikasi akan membaca dan memberi Informasi bahwa website target rentan serangan *Cross-Site Site*. Sebaliknya jika Pesan kesalahan *SQL* tidak terdeteksi, aplikasi akan menampilakn informasi bahwa *website* target tidak rentan terhadap serangan *SQL Injection.*

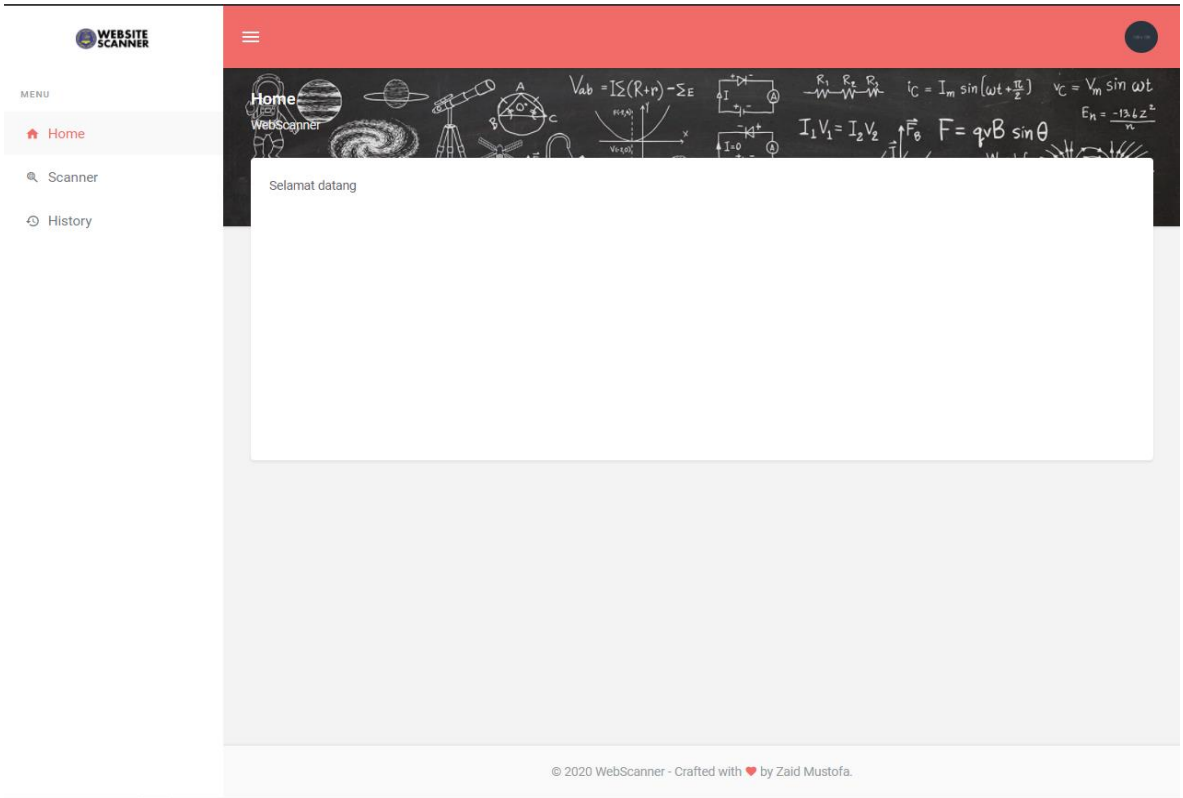
3.3 Implementasi Antarmuka Aplikasi

1. Menu Login



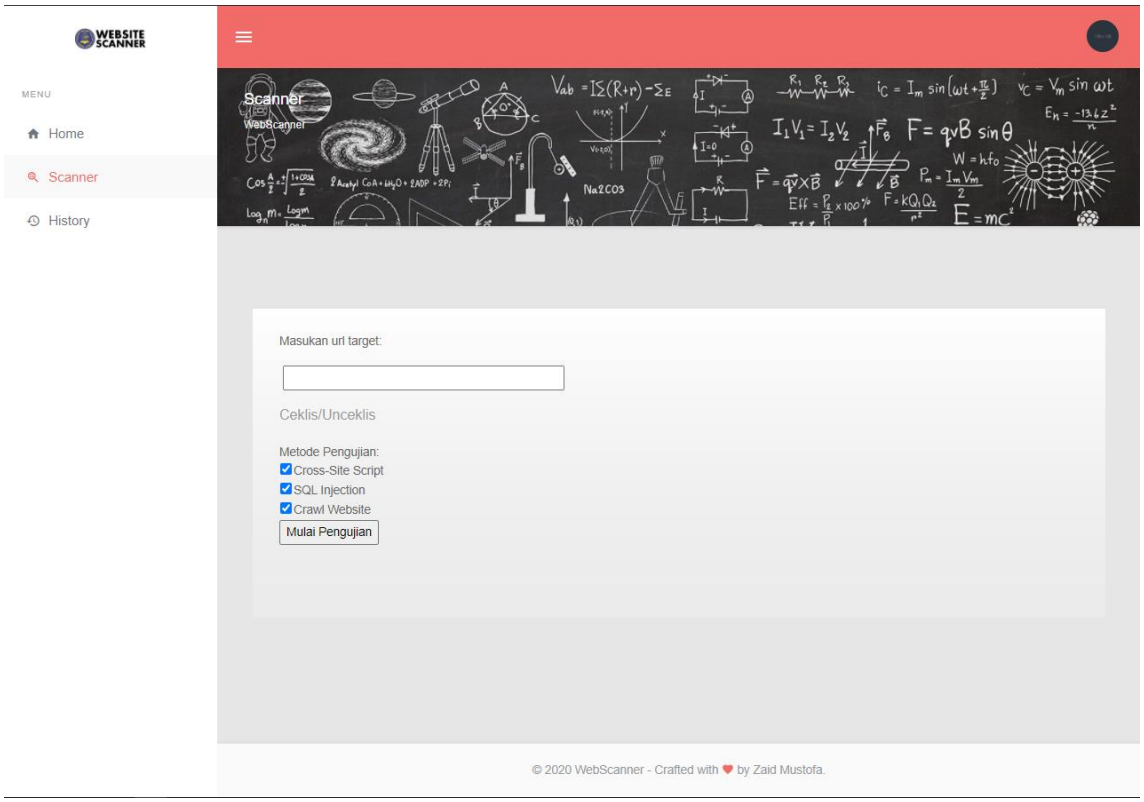
Gambar 3.3 Menu Login

2. Menu Home



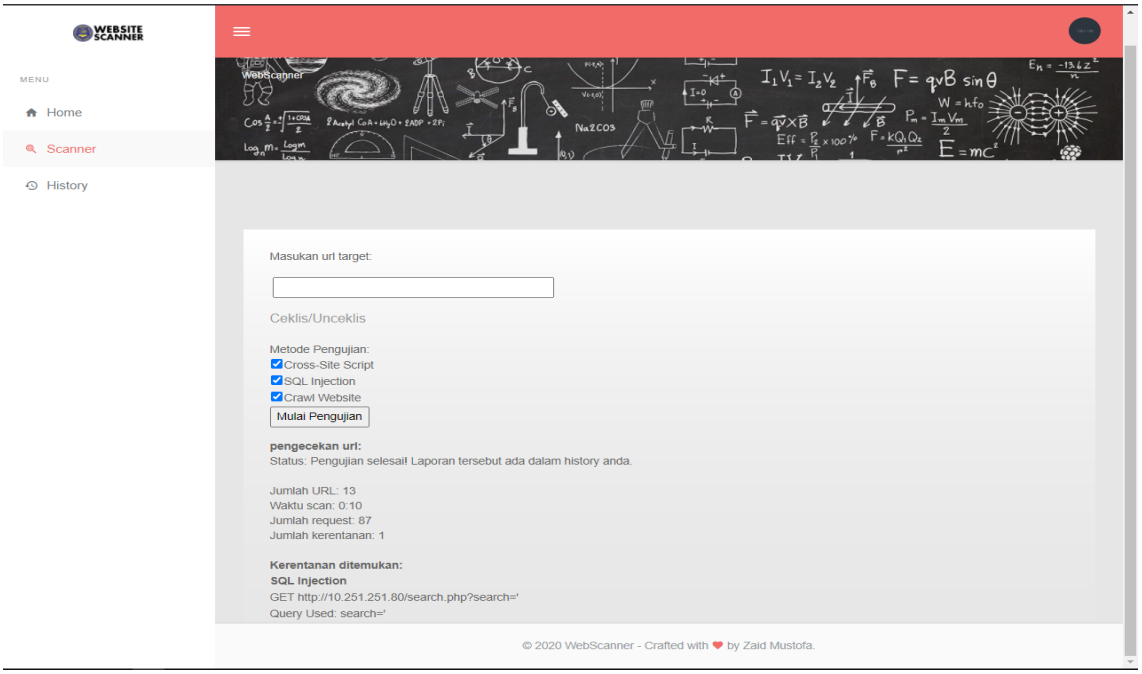
Gambar 3.4 Menu Home

3. Menu scanner

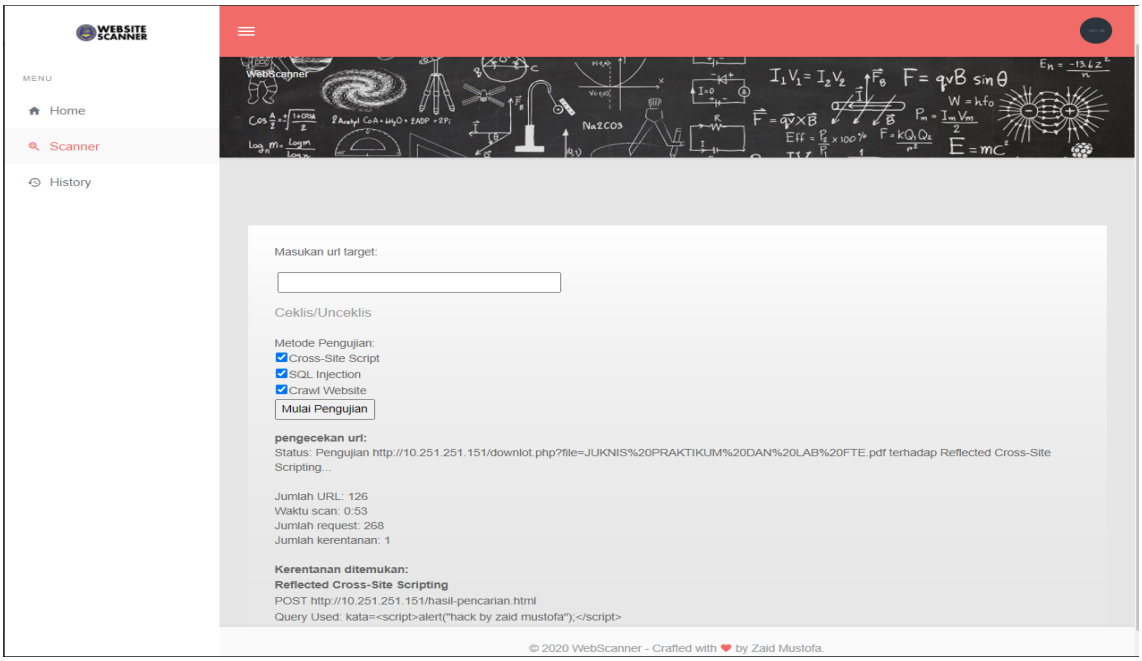


Gambar 3.5 Menu Scanner

Pada menu scanner untuk melakukan pengujian keamanan *website* .User hanya menginputkan alamat target dan menekan tombil mulai pengujian. Ketika aplikasi mendeteksi adanya kerentanan aplikasi akan menampilakan letak kerentananya seperti pada gambar 3.6 dan 3.7



Gambar 3.6 deteksi *SQL Injection*



Gambar 3.7 Deteksi *Cross-Site Script*

4. SIMPULAN

Berdasarkan hasil yang telah dijelaskan maka dapat disimpulkan bahwa aplikasi yang dirancang mampu mendeteksi kemanan *website* pada kerentanan *SQL Injection* dan *Cross-Site Script* dan mempermudah dalam proses pengujian keamanan pada *website.*

5. DAFTAR PUSTAKA

BSSN. 2020. Rekap Serangan Siber (Januari – April 2020). https://www.kominfo.go.id/content/detail/3434/open-source-di-kominfo/0/program_prioritas#:~:text=Sumber%20terbuka%20(open%20source)%20adalah,biasanya%20menggunakan%20fasilitas%20komunikasi%20internet). diakses 10 desember 2020.

Elu A, M. 2013. Rancang Bangun Aplikasi Pendeteksian *Vulnerability Structured Query Language (Sql) Injection* Untuk Keamanan Website

Justin Clarke. 2009. *SQL Injection Attack and Defense*. Burlington, MA 01803. Syngress Publishing, Inc., Elsevier, Inc.

Seth Fogie. 2007. *XSS Attacks*. Burlington, MA 01803. Syngress Publishing, Inc., Elsevier, Inc.

Varacode .*Application Security Vulnerability: Code Flaws, Insecure Code*. https://www.veracode.com/security/application-security-vulnerability-code-flaws-insecure-code. diakses tanggal 16 desember 2020

Yudha, F. Panji A, M. 2018. Perancangan Aplikasi Pengujian Celah Keamanan Pada Aplikasi Berbasis Web