Pengujian Keamanan Sistem Informasi Berbasis Web Berdasarkan Dokumen OWASP WSTG v4.2 (Studi Kasus: Sistem Informatics Expo Universitas Islam Indonesia)

Disusun Oleh:

N a m a : Dimmas Setyo Irawan
NIM : 15523004

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
2022

HALAMAN PENGESAHAN DOSEN PEMBIMBING

Pengujian Keamanan Sistem Informasi Berbasis Web Berdasarkan Dokumen OWASP WSTG v4.2 (Studi Kasus: Sistem Informatics Expo Universitas Islam Indonesia)

TUGAS AKHIR

Disusun Oleh:

N a m a : Dimmas Setyo Irawan

NIM : 15523004

Yogyakarta, Juni 2022

Pembimbing,

(Fayruz Rahma, S.T., M.Eng.)

HALAMAN PENGESAHAN DOSEN PENGUJI

Pengujian Keamanan Sistem Informasi Berbasis Web Berdasarkan Dokumen OWASP WSTG v4.2 (Studi Kasus: Sistem Informatics Expo Universitas Islam Indonesia)

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 21 Juli 2022

Tim Penguji

Fayruz Rahma, S.T., M.Eng. _____

Anggota 1

Ari Sujarwo, S.Kom., MIT. _____

Anggota 2

Dr. Raden Teduh Dirgahayu, S.T., M.Sc _____

Mengetahui,

Ketua Program Studi Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia

(Dr. Raden Teduh Dirgahayu, S.T., M.Sc.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Dimmas Setyo Irawan

NIM : 15523004

Tugas akhir dengan judul:

Pengujian Keamanan Sistem Informasi Berbasis Web Berdasarkan Dokumen OWASP WSTG v4.2 (Studi Kasus: Sistem Informatics Expo Universitas Islam Indonesia)

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 1 Juni 2022

(Dimmas Setyo Irawan)

HALAMAN PERSEMBAHAN

Alhamdulillahi Robbil ‘Alamin. Segala puji dan syukur atas ke hadirat Allah Subhana Wa Ta’ala yang telah memberikan nikmat dan hidayah-Nya sehingga kita masih diberi kesempatan untuk masih bisa bernafas dan memeluk Islam yang sebenar-benarnya. Shalawat serta salam kepada Nabi Muhammad Shallallahu ‘Alaihi Wasallam, sebagai pembawa risalah Allah terakhir dan penyempurna seluruh risalah-Nya yang telah membawa umatnya dari zaman yang gelap gulita ke zaman yang terang benderang, zaman yang kita nikmati saat ini. Tugas akhir ini kupersembahkan untuk semua orang yang aku cintai. Terutama untuk istriku tercinta Luzmi Istiqfara yang tiada hentinya memberikan semangat kepada suamimu ini agar selalu yakin akan menyelesaikan studi ini ditengah kesibukan duniawi lainya. Terima kasih atas omelanmu setiap hari dikala tidak ada semangat dikala tidak ada semangat yang dapat menjadikan cambuk bagiku untuk terus berusaha dengan sungguh-sungguh dalam menyelesaikan tugas akhir ini. Terima kasih untuk teman-teman yang terus memberikan semangat, arahan, dan doa yang tanpa kalian juga tak terasa lengkap dalam menjalani semua proses ini.

HALAMAN MOTO

“Yesterday is history, Tomorrow is a mystery, But today is a Gift”

(Bil Keane)

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah Subhanahu wa ta'ala atas nikmat dan hidayah-Nya, sehingga tugas akhir yang berjudul **"Pengujian Keamanan Sistem Informasi Berbasis Web Berdasarkan Dokumen OWASP WSTG v4.2 (Studi Kasus: Sistem Informatics Expo Universitas Islam Indonesia)"** dapat terselesaikan dengan baik. Shalawat serta salam tidak lupa senantiasa kita panjatkan kepada Nabi Muhammad Shallallahu 'Alaihi Wasallam, yang telah membawa Islam sampai ke kita dari zaman jahiliyah sampai zaman terang benderang seperti sekarang. Laporan tugas akhir ini sebagai salah satu syarat untuk menyelesaikan studi dan memperoleh gelar Sarjana Program Studi Informatika pada Fakultas Teknologi Industri, Universitas Islam Indonesia.

Penelitian Tugas Akhir ini tidak lepas dari bimbingan dan dukungan dari bapak/ibu dosen. Berbagai rintangan dan hambatan penulis temui dalam melakukan penelitian ini. Namun dengan bimbingan, motivasi, bantuan, dan do'a yang telah diberikan membuat penulis semangat dan yakin dalam melalui segala rintangan dan hambatan yang ditemui. Dengan penuh kesadaran dan kerendahan hati penulis ucapkan terima kasih dan penghargaan sebesar-besarnya kepada:

1. Kedua orang tua dan seluruh keluarga yang selalu memberi dukungan kepada penulis.
2. Bapak Hendrik, S.T., M.Eng, selaku Ketua Jurusan Informatika Fakultas Teknologi Industri Universitas Islam Indonesia.
3. Bapak Ahmad Munasir Raf'ie Pratama, S.T., MIT., Ph.D. selaku Sekretaris Jurusan Informatika.
4. Bapak Raden Teduh Dirgahayu, S.T., M.Sc. selaku Ketua Program Studi Informatika-Program Sarjana.
5. Bapak Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D. selaku Sekretaris Program Studi Informatika - Program Sarjana.
6. Ibu Fayruz Rahma, S.T., M.Eng. selaku Dosen Pembimbing Tugas Akhir yang selalu memberikan bimbingan dan arahan kepada penulis sehingga penelitian ini dapat terselesaikan dengan baik.
7. Luzmi Istiqfara, istri tercinta yang tiada henti memberikan semangat dan doa bagi penulis dalam melakukan penelitian ini.
8. M Kemal Abdan, teman seangkatan masuk dan seangkatan keluar banyak cerita yang telah kita ukir.

9. Mustafa, teman seperjuangan yang telah memberikan segala fasilitas bagi penulis dalam melakukan penelitian ini.
10. Kepada semua pihak yang telah membantu baik secara langsung maupun tidak langsung.

Semoga dengan selesainya penelitian tugas akhir ini dapat memberikan manfaat kepada semua pihak. Kebenaran datang dari Allah Subhanahu wa ta'ala dan kesalahan adalah murni dari penulis sebagai manusia jika ada kesalahan dalam penelitian ini penulis memohon maaf yang sebesar-besarnya.

Yogyakarta, 28 Juni 2022

(Dimmas Setyo I)

SARI

Keamanan merupakan aspek penting dalam segala hal terutama yang menyangkut tentang informasi. Perkembangan teknologi yang begitu pesat membuat perubahan dalam melakukan proses bisnis dari cara tradisional menjadi cara yang modern. Sistem informasi merupakan salah satu contoh penerapan teknologi dalam melakukan sebuah proses bisnis. Program Studi Informatika – Program Sarjana Universitas Islam Indonesia memiliki beberapa sistem informasi salah satunya dalah Informatics-Expo. Sistem ini memiliki informasi penting berkaitan dengan karya mahasiswa Informatika Universitas Islam Indonesia. Informasi di dalamnya merupakan hal yang berharga untuk dilindungi. Sebuah sistem informasi harus mampu melindungi berbagai informasi di dalamnya. Untuk mengetahui seberapa rentankah sebuah sistem informasi, perlu dilakukanya sebuah pengujian keamanan.

Dalam penelitian ini, dilakukan pengujian keamanan terhadap sistem Informatics-Expo berdasarkan *framework* WSTG v4.2. *Framework* ini merupakan salah satu produk dari organisasi nirlaba yang bergerak dalam keamanan sistem informasi berbasis web, OWASP. Proses pengujian dilakukan dengan menyimulasikan diri sebagai pihak penyerang yang ingin mendapatkan informasi dari sistem. *Framewok* WSTG v4.2 digunakan sebagai acuan agar pengujian lebih terstruktur. Dalam pengujian ini, ditemukan kerentanan yaitu pada poin WSTG-CLNT-07. Pada poin WSTG-CLNT lainya tidak ditemukan kerentanan pada pengujian ini.

Dari hasil pengujian, dilakukan juga sebuah analisis keamanan yang sudah berhasil diterapakan oleh sistem serta kerentanan yang ditemukan selama proses pengujian. Semoga dengan penelitian pengujian keamanan ini dapat menjadi acuan pengembangan sistem Informatics-Expo dalam melakukan upaya keamanan terhadap serangan di masa mendatang.

Kata kunci: *framework*, keamanan, sistem infromasi, OWASP, WSTG v4.2

GLOSARIUM

Burp Suite	Sebuah perangkat lunak untuk melakukan pengujian keamanan sebuah sistem berbasis web.
Browser	Merupakan sebuah perangkat lunak yang difungsikan untuk membuka atau mengakses sebuah sistem operasi berbasis web.
CORS	Sebuah metode konfigurasi sistem berbasis web untuk mendapatkan data dari web lain.
CSS	Sebuah bahasa pemrograman untuk mengatur tampilan elemen dalam sistem berbasis web.
DOM	Sebuah programming interface yang memungkinan pengembang web untuk dapat merubah tampilan sebuah halaman web tanpa perlu memuat ulang halaman tersebut.
Framework	Kerangka kerja dalam mengembangkan sebuah sistem operasi berbasis web
HTML	Sebuah bahasa pemrograman untuk membangun elemen dalam sistem berbasis web.
Javascript	Sebuah bahasa pemrograman dalam pengembangan sebuah sistem berbasis web.
JSONP	Metode pengiriman data JSON secara lintas situs.
Kali Linux	Sistem operasi berbasis Linux yang digunakan untuk pengujian keamanan jaringan.
Library	Kumpulan pustaka atau sumber layanan yang siap dipakai.
Path	Menjelaskan lokasi file dalam struktur folder pada sebuah sistem berbasi web.
Penetration testing	Proses pengujian keamanan yang dilakukan seseorang dengan menjadikannya sebagai pihak luar terhadap sebuah jaringan atau program.
Scanning	Proses pemindaian sebuah sistem.
Server	Sebuah sistem yang menyediakan jenis layanan terhadap jaringan komputer.
Software	Sebuah perangkat lunak yang mendukung kinerja dari perangkat komputer.
Tag	Merupakan penanda awalan dan akhiran dari sebuah elemen HTML.

Tools	Sebuah perangkat lunak dalam melakukan proses pengujian.
XSS	Sebuah serangan terhadap sistem berbasis web yang memanfaatkan skrip lintas situs.
Zap	Sebuah perangkat lunak untuk melakukan pengujian keamanan sebuah sistem berbasis web.

DAFTAR ISI

HALAMAN JUDUL i
HALAMAN PENGESAHAN DOSEN PEMBIMBING ii
HALAMAN PENGESAHAN DOSEN PENGUJI iii
HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR iv
HALAMAN PERSEMBAHAN v
HALAMAN MOTO vi
KATA PENGANTAR vii
SARI ix
GLOSARIUM x
DAFTAR ISI xii
DAFTAR TABEL xiv
DAFTAR GAMBAR xv
BAB I PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 2
1.4 Manfaat Penelitian 3
1.5 Tujuan Penelitian 3
1.6 Metodologi Penelitian 3
1.7 Sistematika Penulisan 4
BAB II LANDASAN TEORI 6
2.1 Sistem Informatics Expo 6
2.2 Teori Dasar 7
2.2.1 Keamanan Informasi 7
2.2.2 Vulnerability 8
2.2.3 Penetration Testing 8
2.2.4 *Black-Box Testing* 10
2.3 OWASP 10
2.3.1 WSTG 10
2.3.2 *Client-side Testing* 11
2.4 Penelitian Terkait 15
BAB III METODOLOGI 19
3.1 Metode Penelitian 19
3.2 Metode Pengumpulan Data 20
3.3 Alat Kebutuhan Penelitian 20
3.3.1 Perangkat Keras 20
3.3.2 Perangkat Lunak 21
3.4 Metode Pengujian Sistem 22
3.4.1 Alur Pengujian Sistem 22
3.4.2 Implementasi Pengujian 23
3.5 Metode Analisis Hasil 34
BAB IV HASIL DAN PEMBAHASAN 35
4.1 Hasil Pengujian 35
4.1.1 Hasil WSTG-CLNT-01 Testing for DOM-Based Cross Site Scripting 35
4.1.2 Hasil WSTG-CLNT-02 *Testing for* JavaScript *Execution* 37
4.1.3 Hasil WSTG-CLNT-03 *Testing for* HTML *Injection* 38
4.1.4 Hasil WSTG-CLNT-04 *Testing for Client-side* URL *Redirect* 39

4.1.5 Hasil WSTG-CLNT-05 *Testing for* CSS *Injection* 41
4.1.6 Hasil WSTG-CLNT-06 *Testing for Client-side Resource Manipulation* 41
4.1.7 Hasil WSTG-CLNT-07 *Testing Cross Origin Resource Sharing*.....43
4.1.8 Hasil WSTG-CLNT-08 *Testing for Cross Site Flashing* 44
4.1.9 Hasil WSTG-CLNT-09 *Testing for Clickjacking* 44
4.1.10 Hasil WSTG-CLNT-10 *Testing WebSockets* 46
4.1.11 Hasil WSTG-CLNT-11 *Testing Web Messaging*.....46
4.1.12 Hasil WSTG-CLNT-12 *Testing Browser storage*.....47
4.1.13 Hasil WSTG-CLNT-13 *Testing for Cross Site Script Inclusion*.....51
4.2 Analisis Hasil 52
BAB V KESIMPULAN DAN SARAN 59
5.1 Kesimpulan 59
5.2 Saran.....59
DAFTAR PUSTAKA.....60
LAMPIRAN 61

DAFTAR TABEL

Tabel 2.1 Teknologi sistem.....7
Tabel 2.2 Penelitian sebelumnya 17
Tabel 3.1 Spesifikasi Perangkat Keras.....21
Tabel 3.2 Spesifikasi Mesin Virtual.....21
Tabel 3.3 Identifikasi Poin.....23
Tabel 3.4 *Resource* dalam struktur web.....28
Tabel 4.1 Metode pengujian dan hasil 52
Tabel 4.2 Detail kerentanan WSTG-CLNT-07 *Testing Cross Origin Resource Sharing* 56

DAFTAR GAMBAR

Gambar 2.1 Sistem Informatics-expo 6
Gambar 2.2 Sistem Informatics-expo 7
Gambar 3.1 Bagan Alur Penelitian 20
Gambar 3.2 Bagan Alur Pengujian 22
Gambar 3.3 Contoh fungsi kerentanan DOM 25
Gambar 3.4 Contoh kode program serangan XSS 25
Gambar 3.5 Contoh fungsi Javascript 26
Gambar 3.6 Contoh kode program serangan *Javascript* 26
Gambar 3.7 Contoh fungsi perubahan elemen HTML pada DOM 26
Gambar 3.8 Contoh kode program serangan HTML *injection* 26
Gambar 3.9 Contoh fungsi 27
Gambar 3.10 Contoh penggunaan *tools* Burp Suite 27
Gambar 3.11 Contoh kode program serangan CSS *injection* 28
Gambar 3.12 Contoh penggunaan *tools* ZAP 29
Gambar 3.13 Contoh penggunaan *tools* Vais 30
Gambar 3.14 Contoh penggunaan *tools* CyberMoon 31
Gambar 3.15 Contoh penggunaan *tools* Clickjacking Tester 31
Gambar 3.16 Contoh penggunaan *tools* ZAP 32
Gambar 3.17 Contoh fungsi untuk Web Messaging 32
Gambar 3.18 Contoh penggunaan *tools* Burp Suite 33
Gambar 4.1 Hasil dari pencarian fungsi DOM 35
Gambar 4.2 Hasil dari percobaan injeksi kode 36
Gambar 4.3 Hasil dari scanning *tools* Acunetix 37
Gambar 4.4 Hasil dari pencarian fungsi Javascript 38
Gambar 4.5 Hasil dari pencarian fungsi HTML 39
Gambar 4.6 Hasil dari pencarian fungsi *redirect* 40
Gambar 4.7 Hasil dari pencarian *path* atau URL 40
Gambar 4.8 Hasil dari pencarian fungsi CSS 41
Gambar 4.9 Hasil dari pencarian *resource* 42
Gambar 4.10 *Resource* jenis *script* 42
Gambar 4.11 *Resource* jenis CSS 42
Gambar 4.12 *Resource* jenis Gambar 42

Gambar 4.13 Hasil *scanning tools* ZAP 43
Gambar 4.14 *Header* dari salah satu *response*.....43
Gambar 4.15 Hasil dari pencarian file SWF 44
Gambar 4.16 Hasil pengujian dengan *tools* CyberMoon.....45
Gambar 4.17 Hasil pengujian dengan *tools* Clickjacking Tester.....45
Gambar 4.18 Konfigurasi *Header*.....46
Gambar 4.19 Hasil dari perekaman lalu lintas Websockets 46
Gambar 4.20 Hasil pencarian fungsi *web messaging* 47
Gambar 4.21 *Browser storage* jenis Local *Storage*.....48
Gambar 4.22 *Browser storage* jenis *Session Storage* 48
Gambar 4.23 *Browser storage* jenis IndexedDB 49
Gambar 4.24 *Browser storage* jenis *WebSQL* 50
Gambar 4.25 *Browser storage* jenis Cookies 50
Gambar 4.26 *Browser storage* jenis *Cache Storage*.....51
Gambar 4.27 Hasil dari perekaman data menggunakan *tools* Burp Suite 52
Gambar 4.28 Pencarian informasi dengan *tools* BulitWith 54
Gambar 4.29 Pencarian informasi dengan *tools* BulitWith 55
Gambar 4.30 Bukti oleh *tools* ZAP.....58

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan sistem informasi adalah perkembangan yang menarik untuk diamati. Dalam berbagai bidang, sistem informasi sangat dibutuhkan karena sistem informasi adalah kombinasi dari orang-orang, perangkat keras, perangkat lunak, jaringan komunikasi, sumber daya data, dan kebijakan serta prosedur dalam menyimpan, mendapatkan kembali, mengubah, dan menyebarkan informasi dalam suatu organisasi (O'Brien & Marakas, 2010).

Dalam dunia pendidikan khususnya perguruan tinggi, sistem informasi sangat membantu *civitas* akademik dalam melakukan tugas-tugasnya. Sebuah sistem yang memiliki cakupan interaksi luas, sering kali membuat sistem menjadi rentan dari pihak-pihak yang tidak bertanggung jawab. Hal ini akan berdampak pada keamanan sistem tersebut. Masalah keamanan merupakan hal yang perlu dipertimbangkan dalam sebuah sistem informasi (Rahardjo, 2005). Masalah keamanan yang terjadi akan menimbulkan kerugian baik itu kerugian waktu yang terjadi ketika sistem tidak bisa diakses karena serangan ataupun perbaikan dari serangan dan kerugian data yang terjadi ketika data dalam sistem mengalami perubahan yang tidak semestinya maupun hilang.

Informatics Expo Universitas Islam Indonesia merupakan sistem informasi berbasis web yang dikelola oleh *Student Staff* Jurusan Informatika Universitas Islam Indonesia. Informatics Expo Universitas Islam Indonesia berfungsi untuk menghimpun karya-karya mahasiswa Jurusan Informatika Universitas Islam Indonesia dari berbagai macam mata kuliah yang nantinya akan mendapatkan penilaian dari dosen dan khalayak umum. Dalam sistem tersebut menyimpan informasi penting seperti data mahasiswa, data dosen, karya ilmiah, dan penilaian karya ilmiah. Dengan data yang tersimpan dalam tersebut berikut beberapa masalah yang bisa terjadi ketika sistem berhasil diretas

1. Perubahan data yang tidak semestinya akan berpengaruh terhadap reputasi dari pemilik data yang berubah.
2. Kehilangan data akan berpengaruh terhadap keberlangsungan proses bisnis selanjutnya pada data tersebut.

3. Tambahan waktu yang diperlukan ketika memperbaiki masalah yang diakibatkan dari serangan.

Maka dari itu, keamanan sistem ini akan menjadi hal yang perlu diperhatikan.

Sistem Informatics Expo Universitas Islam Indonesia belum pernah dilakukan pengujian keamanan sebelumnya. Melihat bahwa sistem ini memiliki informasi yang cukup penting maka masalah ini dapat menyebabkan kerugian bagi Jurusan Informatika Universitas Islam Indonesia dan pengguna sistem di dalamnya. Dalam mengatasi masalah ini, salah satu langkah yang dapat ditempuh adalah dengan cara melakukan analisis keamanan sistem. Penelitian ini berfokus pada pengumpulan informasi dan melakukan pengujian sistem berdasarkan salah satu kerangka kerja dari OWASP yaitu WSTG v4.2.

Open Web Application Security Project (OWASP) merupakan sebuah organisasi nirlaba yang bergerak dalam bidang keamanan sistem berbasis web. Sebagai organisasi nirlaba, OWASP telah menghasilkan artikel, metodologi, dokumentasi, alat dan teknologi yang dapat diakses dengan mudah dan gratis. *Web Security Testing Guide* (WSTG) merupakan kerangka kerja yang dirilis oleh OWASP. WSTG adalah sebuah kerangka kerja yang digunakan untuk melakukan analisis keamanan pada sistem berbasis web, OWASP sebagai pengembang kerangka kerja WSTG selalu memperbaharui versi WSTG. Sampai saat ini WSTG yang terbaru adalah WSTG v4.2.

Kemudian, dilakukannya pengujian keamanan tersebut akan menghasilkan analisis kerentanan yang berguna untuk mengetahui tingkat keamanan dan menjadikan saran bagi pengembang dalam meningkatkan keamanan sistem di masa mendatang.

1.2 Rumusan Masalah

Berdasarkan penjelasan pada latar belakang, maka rumusan masalah yang didapat adalah sebagai berikut:

a. Bagaimana cara melakukan pengujian keamanan sistem Informatics Expo berdasarkan *Framework* WSTG v4.2?
b. Bagaimana hasil dari analisis pengujian keamanan sistem Informatics Expo berdasarkan *Framework* WSTG v4.2?

1.3 Batasan Masalah

a. Pengujian dilakukan dengan beberapa *tools* keamanan pada sistem operasi Kali Linux

b. Pengujian dilakukan pada sistem *on production*
c. Pengujian dilakukan dengan metode *Blackbox*

1.4 Manfaat Penelitian

a. Bagi penulis
    1. Menambah pengetahuan penulis terkait standardisasi keamananan suatu sistem
b. Bagi pengembang sistem
    1. Mengetahui kerentanan pada sistem Informatics Expo Universitas Islam Indonesia
    2. Mempermudah perbaikan dan peningkatan keamanan pada sistem Informatics Expo Universitas Islam Indonesia
c. Bagi pengguna sistem
    1. Meningkatkan keamanan dan kenyamanan dalam menggunakan sistem Informatics Expo Universitas Islam Indonesia

1.5 Tujuan Penelitian

Untuk mengukur tingkat keamanan sistem Informatics Expo Universitas Islam Indonesia yang nantinya hasil dari pengukuran ini bisa digunakan untuk memetakan hal-hal yang perlu ditingkatkan.

1.6 Metodologi Penelitian

Berikut ini adalah tahapan-tahapan yang akan dikerjakan dalam penelitian ini:

a. Tinjauan Pustaka

    Tahap ini dilakukan untuk mencari sumber referensi yang berasal dari jurnal, paper, buku, maupun laporan hasil penelitian yang ada hubungannya dengan penelitian yang akan dikerjakan.

b. Wawancara

    Tahap ini dilakukan untuk mengetahui lebih dalam seputar informasi yang berkaitan dengan Sistem Informatics Expo. Wawancara dilakukan dengan salah satu Pengembang Sistem Informatics Expo sebagai narasumber.

c. Analisis Sistem

Fitur-fitur dan berbagai fungsionalitas sistem Informatics Expo akan dijadikan sumber referensi untuk kemudian dianalisis dalam mencari aspek sistem yang membutuhkan keamanan data yang lebih.

d. Analisis Masalah

Setelah permasalahan telah didapatkan pada tahap sebelumnya, kemudian permasalahan tersebut dianalisis pada tahap ini. Gunanya adalah agar dari permasalahan-permasalahan tersebut didapatkan solusi untuk masing-masing masalah tersebut.

e. Analisis Kebutuhan

Pada tahap ini analisis dilakukan untuk mengetahui apa saja yang dibutuhkan untuk melakukan pengujian sistem.

f. Implementasi

Pada tahap ini penulis mulai melakukan tahapan-tahapan pengujian dengan metode *Black-Box Testing* dan investigasi guna mendapatkan informasi kelemahan sistem.

g. Evaluasi Hasil

Merupakan langkah terakhir yang dilakukan dalam penelitian ini, di mana dilakukan evaluasi pada hasil dari implementasi sebelumnya dan mulai menyimpulkan hasil sebagai saran bagi para pengembang untuk meningkatkan keamanan sistem Informatics Expo ke depannya.

1.7 Sistematika Penulisan

Untuk memberikan Gambaran secara menyeluruh mengenai masalah yang akan dibahas dalam penulisan laporan tugas akhir ini, sistematika laporan ini dibagi menjadi lima bab. Adapun penjabarannya sebagai berikut:

BAB I PENDAHULUAN

Bab pendahuluan berisi tentang latar belakang masalah, rumusan masalah, batasan masalah, manfaat penelitian, tujuan penelitian, metodologi penelitian, dan sistematika penulisan laporan Pengujian Keamanan Sistem Informasi Berbasis Web berdasarkan *Framework* OWASP WSTG v4.2 sistem Informatics Expo UII.

BAB II LANDASAN TEORI

Bab ini berisi tinjauan penelitian serupa yang dijadikan acuan dalam melakukan penelitian ini, serta penjelasan teori yang dipakai dalam melakukan penelitian ini.

BAB III METODOLOGI

Bab ini membahas tentang metode apa saja yang dilakukan dalam penelitian. Dimulai dari tahapan awal dalam penelitian sampai tahap akhir penelitian ini. Bab ini juga membahas perancangan metode pengujian terhadap sistem dan analisis hasil.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi tentang detail pembahasan terkait dengan proses pengujian, temuan bukti-bukti pada setiap langkah pengujian ditemukan dalam bab ini.

BAB V KESIMPULAN DAN SARAN

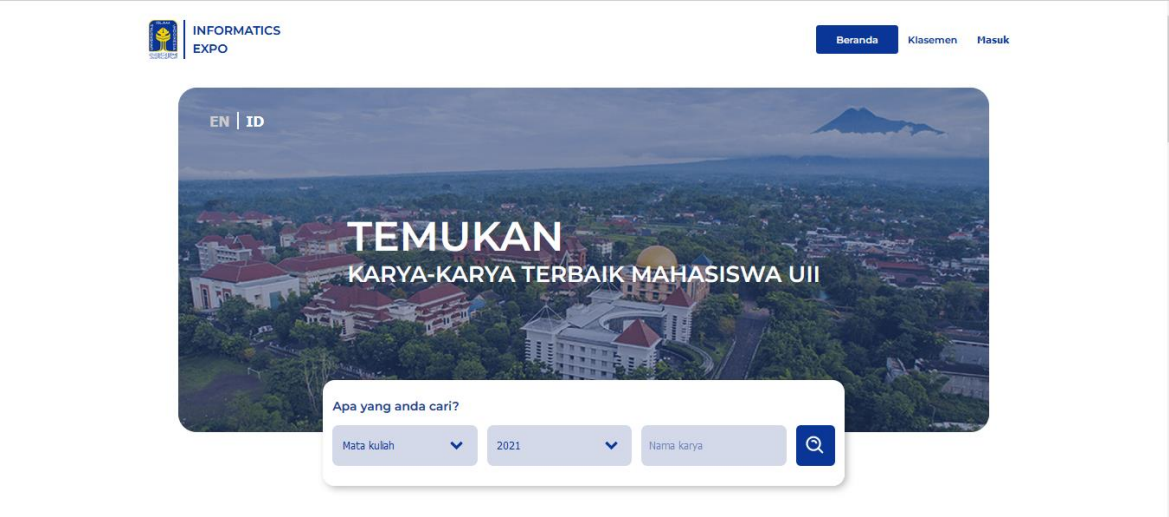
Bab ini berisi penutup yang meliputi kesimpulan-kesimpulan dari hasil pengujian yang telah dilakukan sebelumnya, dan terdapat saran-saran dari hasil keseluruhan penelitian yang telah dilakukan.

BAB II
LANDASAN TEORI

Sebelum penelitian dilakukan, dibutuhkan pengetahuan seputar keamanan sistem. Penelitian yang telah dilakukan dapat menjadi acuan sebelum melakukan penelitian berikutnya. Serta teori pengetahuan dasar tentang keamanan sistem. Dengan tercapainya kebutuhan ini dapat memberikan gambaran yang lebih jelas dalam melakukan penelitian.

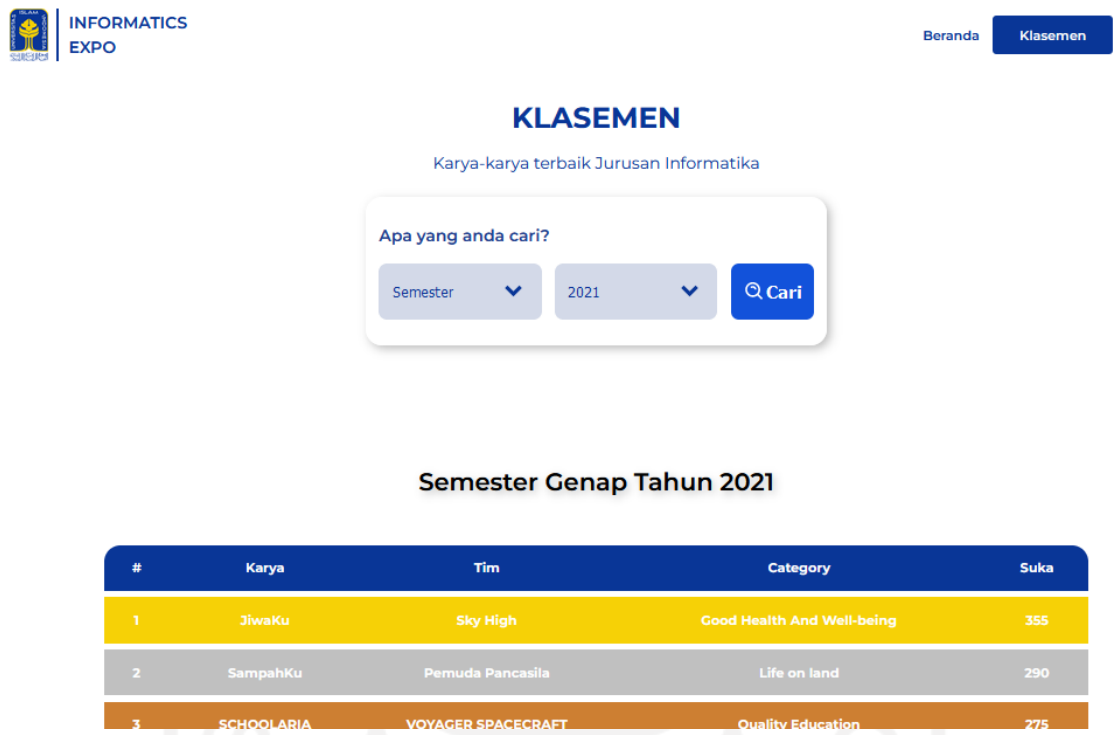
2.1 Sistem Informatics Expo

Sistem Informatics Expo adalah sebuah sistem yang dibuat untuk agenda pameran Program Studi Informatika Program Sarjana yang digelar tiap akhir semester. Agenda ini menjadi wadah yang mewujudkan gagasan mahasiswa secara tim. Karya yang dipamerkan di Informatics Expo merupakan keluaran dari mata kuliah yang di*expo*-kan. Berikut Gambar 2.1 merupakan halaman awal sistem Infromatics Expo



Gambar 2.1 Sistem Informatics-expo

Karya para mahasiswa dapat dilihat dalam sistem ini oleh siapapun. Pada saat masa Expo, karya-karya ini akan dilombakan. Gambar 2.2 merupakan hasil klasemen pada masa Expo



Gambar 2.2 Sistem Informatics-expo

Beberapa teknologi yang diterapkan oleh sistem ini terlihat pada Tabel 2.1 berikut:

Tabel 2.1 Teknologi sistem

Jenis Teknologi	Penyedia Teknologi
Webserver	Cloudflare
Hosting	Cloudflare
Bahasa Pemrograman	PHP
Web *Framework*	Laravel
UI *Framework*	React

2.2 Teori Dasar

2.2.1 Keamanan Informasi

Keamanan informasi merupakan upaya perlindungan informasi dari ancaman untuk meminimalisir risiko bisnis, menjamin kelanjutan proses bisnis dan meningkatkan investasi. Jadi dapat diartikan bahwa keamanan informasi merupakan sebuah perlindungan dari ancaman yang dapat menggangu proses bisnis dalam sebuah sistem ataupun membahayakan kebocoran informasi dari sistem tersebut.

Menurut (de Lange et al., 2016) keamanan informasi merupakan komponen yang krusial dalam mencapai kesuksesan organisasi, terlepas dari bidang atau fungsi organisasi tersebut. Dalam menerapkan Keamanan Informasi, perusahaan/organiasasi harus memperhatikan tiga aspek yaitu *Confidentially*, *Integrity*, dan *Availability* (CIA). Pada umumnya, ketika serangan, masalah, risiko yang mengancam keamanan informasi muncul, setidaknya terdapat salah satu dari aspek CIA yang akan menjadi target dari serangan tersebut.

a. *Confidentiality* (kerahasiaan) merupakan karakteristik dari sebuah informasi di mana hanya orang yang mempunyai hak yang dapat mengakses informasi tersebut.
b. *Integrity* (integritas) merupakan jaminan bahwa data tersebut tetap utuh tidak berubah tanpa izin dari pemegang data tersebut.
c. *Availability* (ketersediaan) merupakan jaminan bahwa data tersebut tersimpan dengan aman dan dapat tersedia saat pemegang izin data tersebut membutuhkannya.

2.2.2 Vulnerability

Vulnerability atau kerentanan adalah suatu poin kelemahan di mana suatu sistem rentan terhadap serangan (Juardi, 2017). Setiap sistem akan selalu memiliki suatu kerentanan, terutama pada masalah keamanan karena pada dasarnya suatu sistem akan memiliki data atau informasi dengan tingkat keamanan tertentu. Semakin tinggi tingkat keamanan maka seharusnya data atau informasi yang disimpan akan semakin penting. Semakin penting data atau informasi maka akan semakin menarik minat pihak yang tidak bertanggung jawab untuk mencoba mengambilnya.

2.2.3 Penetration Testing

Penetration testing adalah sebuah pengujian terhadap sistem yang bertujuan untuk mengevaluasi keamanan suatu sistem. Pengujian dilakukan dengan sebuah simulasi serangan terhadap sistem untuk menemukan celah keamanan yang dapat terjadi dari konfigurasi yang tidak benar, kelemahan dalam proses teknis, dan kelemahan lain yang dapat menjadi celah keamanan pada sistem tersebut. Hasil dari *Penetration testing* dapat menjadi sebuah laporan yang bisa menjadi masukan kepada pemilik sistem tentang celah keamanan terhadap sistem yang mereka miliki kemudian digunakan untuk proses evaluasi guna melakukan penambalan

terhadap kebocoran celah yang ditemukan. Berdasarkan objeknya, *Pentetration testing* terdapat enam jenis:

a. *Network Service*

Objek yang diuji dalam *network service penetration test* adalah infrastruktur jaringan. Tujuan utama dari jenis *penetration test* ini untuk mengidentifikasi kelemahan pada objek-objek *network service* seperti *server, firewall, switch, router, printer, workstation,* dan lain-lain.

b. *Web Application*

Web application penetration test digunakan untuk menemukan kerentanan dan kelemahan keamanan pada aplikasi berbasis web. *Penetration test* ini menggunakan beberapa teknik dan serangan yang tujuannya untuk menembus keamanan suatu web *application*. Beberapa elemen yang dipindai dalam upaya *penetration test* jenis ini antara lain web *based application, browser* dan komponen-komponen lainnya. Cara melakukan *penetration test* jenis ini mengalami perubahan dari waktu ke waktu. Seiring perkembangan teknologi yang kian pesat, tingkat ancaman juga ikut berkembang.

c. *Client Side*

Client side penetration test digunakan untuk menemukan kelemahan keamanan pada *client side application*. Beberapa program atau aplikasi yang termasuk *client side application* antara lain *Putty, email clients,* web *browsers, Macromedia Flash,* dan lain-lain.

d. *Wireless*

Wireless penetration test melibatkan identifikasi dan inspeksi koneksi yang menghubungkan perangkat-perangkat dalam satu jaringan *wifi* perusahaan. Beberapa perangkat yang menjadi objek *pentest* jenis ini antara lain, *desktop,* laptop, *tablet, smartphones,* dan *Internet of Things* (IoT).

e. *Social Engineering*

Social penetration test merupakan sebuah upaya untuk membujuk atau menebar trik kepada *user* untuk memberikan informasi sensitif. Beberapa data yang kerap menjadi sasaran upaya ini antara lain *username* dan *password.* Serangan *cyber social engineering* yang biasa dilakukan oleh *pentester* antara lain:

Physing, Tailgating, Imposter, Name Dropping, Pre-texting, Dumpster Diving, Eaves Dropping, dan *Gifts.*

f. Physical

Physical penetration test merupakan upaya dari *pentester* untuk menembus hambatan fisik dari infrastruktur, bangunan, sistem, bahkan staf dari sebuah perusahaan.

2.2.4 *Black-Box Testing*

Blackbox testing merupakan pengujian perangkat lunak tanpa menguji desain dan kode. pengujian hanya berfokus kepada spesifikasi fungsionalitas (Ariani Sukamto & Shalahuddin, 2016). Metode ini dapat dijadikan pengujian setelah tahap akhir dari sebuah sistem ataupun sistem yang sudah berjalan. Pada pengujian keamanan sebuah sistem metode ini digunakan untuk menguji sistem dari sudut pandang penyerang yang tidak mengetahui struktur kode ataupun infrastruktur jaringan pada sistem. Penguji yang menggunakan metode ini memposisikan diri sebagai penyerang atau *hacker* yang akan mengeksploitasi sistem untuk mencari celah kerentanan yang dapat diretas.

2.3 OWASP

OWASP (*Open Web Application Security Porject*) adalah komunitas terbuka yang berdedikasi untuk membuat sebuah organisasi yang bertujuan untuk mengembangkan keamanan aplikasi berbasis web. OWASP membuat proyek-proyek gratis yang dapat diakses siapapun untuk keperluan pengembangan perangkat lunak berbasis web. Selain pengembangan aplikasi keamanan, OWASP juga membuat panduan-panduan pengembangan serta pemeliharaan sebuah sistem. OWASP tidak terafiliasi dengan perusahaan teknologi komersil, namun OWASP juga tetap mendukung perusahaan komersil yang bergerak dalam bidang teknologi. Beberapa proyek panduan pengembangan aplikasi berbasis web seperti WSTG (*Web Security Testing Guide*).

2.3.1 WSTG

WSTG merupakan salah satu proyek dari OWASP yang berisi tentang tahapan-tahapan pengujian kemanan sistem berbasis web. WSTG sendiri dikembangkan oleh para profesional dalam bidang keamanan sistem berbasis web yang tergabung dalam komunitas OWASP.

Seiring pesatnya kemajuan teknologi dan diikuti dengan semakin kompleks kerentanan terhadap sebuah sistem yang dapat dimanfaatkan oleh pihak yang tidak bertanggung jawab, WSTG juga selalu dikembangkan. Sampai saat ini WSTG telah mencapai versi 4.2. Dalam WSTG terdapat beberapa bab yang menjelaskan tentang panduan pengembangan sistem sampai dengan pengujian sistem seperti berikut:

1. OWASP *Testing Framework*

   Merupakan bab yang berisi tentang panduan langkah-langkah pengembangan sebuah sistem seperti proses sebelum mengembangkan sistem, proses pengembangan sistem, dan perawatan setelah sistem berhasil dikembangkan.

2. Web *Application Security Testing*

   Merupakan bab yang berisi tentang poin-poin pengujian keamanan sebuah sistem berserta dengan penjelasan langkah-langkahnya. Berikut beberapa poin pengujian keamanan yang disebutkan pada bab ini:

   a. Information Gathering.
   b. Configuration and Deployment Management Testing.
   c. Identity Management Testing.
   d. Authentication Testing.
   e. Authorization Testing.
   f. Session Management Testing.
   g. Input Validation Testing.
   h. Testing for Error Handling.
   i. Testing for Weak Cryptography.
   j. Business Logic Testing.
   k. Client-Side Testing.

3. *Reporting*

   Merupakan bab yang menjelaskan tentang panduan pembuatan laporan hasil dari proses pengujian keamanan.

2.3.2 *Client-side Testing*

Merupakan bagian dari WSTG yang berupa panduan untuk melakukan pengujian sebuah sistem berbasis web pada sisi pengguna. Pada *Client-side Testing* terdapat tiga belas poin yang perlu diuji yaitu:

1. *Testing for DOM-Based Cross Site Scripting*

Cross Site Scripting sering disebut dengan XSS merupakan sebuah serangan kepada sistem berbasis web dengan cara memasukan sebuah kode program berbahaya. Umumnya kode program berbahaya ini akan dimasukan oleh penyerang melalui kolom input pada sistem tersebut ataupun pada kolom *url*. Serangan bisa ditujukan untuk mencuri data penting seperti *cookie*, domain, dan bahkan dapat mengambil alih akun seseorang. XSS berbasis DOM merupakan sebuah serangan ketika terdapat aliran data pada DOM yang berhasil dimodifikasi oleh penyerang dengan menyisipkan sebuah kode program berbahaya. Pengujian pada poin ini bertujuan untuk mengetahui keamanan sistem terhadap serangan XSS berbasis DOM.

2. *Testing for JavaScript Execution*

JavasScript merupakan sebuah bahasa pemrograman yang digunakan untuk membuat sebuah sistem berbasis web menjadi lebih dinamis dan interaktif. *JavaScript* juga digunakan untuk proses logika data sehingga penggunaan *JavaScript* tidak hanya untuk masalah tampilan antarmuka namun dapat digunakan untuk proses pengiriman data. Ketika sebuah sistem berbasis web tidak memiliki fitur keamanan terhadap XSS, penyerang akan dengan mudah memanipulasi sebuah kode program berbahaya seperti *JavaScript* untuk tujuan yang dapat merugikan. Pengujian pada poin ini bertujuan untuk mengetahui keamanan sistem terhadap serangan injeksi kode program menggunakan *JavaScript.*

3. *Testing for HTML Injection*

Hypertext Markup Language atau HTML adalah bahasa *markup* standar yang digunakan untuk membuat halaman web. Dirilis pada tahun 1991 dan terus dikembangkan dengan versi berikutnya, HTML mendapatkan popularitas yang cukup banyak HTML menjadi bahasa *markup* standar web resmi. Sampai pada skripsi ini ditulis HTML telah mencapai versi HTML5 yang diperkenalkan tahun 2014. Melihat kembali pada pengertian tentang XSS, ketika penyerang dapat menyisipkan sebuah kode program berbahaya, hal itu dapat menjadi sebuah ancaman yang berbahaya. Kode program tersebut juga tidak menutup kemungkinan dari sebuah *tag* HTML yang dimodifikasi untuk mencuri data penting. Pengujian pada poin ini bertujuan untuk mengetahui keamanan sistem terhadap serangan injeksi kode program menggunakan HTML.

4. *Testing for Client-side URL Redirect*

Uniform Resource Locator atau URL merupakan sebuah alamat yang digunakan untuk mengakses sebuah web pada *browser*. Sebuah web harus memiliki URL untuk dapat diakses oleh penggunanya. Pengguna cukup memasukkan URL tersebut pada *browser* agar kemudian

browser dapat menampilkan halaman web tersebut. Pada sebuah web tidak menutup kemungkinan ketika pengguna mengakses suatu fitur tertentu, pengguna akan diarahkan ke URL lain. Hal ini bisa menjadi celah keamanan ketika penyerang dapat menemukan celah ketika URL yang seharusnya dituju dapat dimodifikasi sehingga pengguna dapat diarahkan ke web lain untuk tujuan yang merugikan. Pengujian pada poin bertujuan untuk mengetahui keamanan sistem terhadap serangan manipulasi pada saat proses *redirect* berlangsung.

5. *Testing for CSS Injection*

Tampilan utama sebuah web ditulis dalam bahasa markup HTML sedangkan CSS atau *Cascading Style Sheet* adalah sebuah bahasa untuk mengatur elemen pada bahasa *markup*. Ketika elemen *markup* dapat diatur dalam sebuah bahasa pemrograman sendiri maka akan mudah dalam mengatur tampilan font, ukuran baris paragraf, warna *background* dan lain sebagainya. Karena CSS merupakan sebuah bahasa yang dapat dijalankan oleh web, XSS dapat dijalankan dengan memanfaatkan bahasa CSS untuk penyisipan kode berbahaya. Pengujian pada poin ini bertujuan untuk mengetahui keamanan sistem terhadap serangan injeksi kode program menggunakan CSS.

6. *Testing for Client-side Resource Manipulation*

Resource merupakan sumber daya yang menyediakan kebutuhan. Sebuah web memiliki *resource* dalam menjalankan proses bisnisnya. Seperti saat pengguna mengakses data pada web, web akan mengirim data pada *resource* sisi *server* kepada pengguna tersebut. Celah keamanan pada *Resource Manipulation* terjadi ketika penyerang dapat menemukan celah untuk memodifikasi sebuah *Request* yang akan dikirimkan oleh pengguna. Hal ini dapat juga digunakan untuk melakukan serangan XSS. Pengujian pada poin ini bertujuan untuk mengetahui keamanan sistem terhadap serangan manipulasi *resource* yang terhubung dengan sistem.

7. *Testing Cross Origin Resource Sharing*

Ketika sebuah web mencoba untuk mengambil *resource* dari web lain, hal tersebut tidak diperbolehkan sebab melanggar *Same Origin Policy*. Karena, semakin terbukanya pintu untuk mengakses *resource* web lain ataupun diaksesnya *resource*, akan semakin berisiko untuk terbukanya celah keamanan. Pada kasus tertentu akan dibutuhkan keadaan ketika sebuah web memerlukan *resource* dari web lain, *Cross Origin Resource Sharing* atau CORS merupakan sebuah mekanisme untuk mengizinkan akses *resource* web dari web lain. Dengan adanya CORS, sebuah web dapat diatur untuk sejauh mana sebuah *resource* dapat diakses oleh web lain. Celah keamanan umumnya terjadi pada kesalahan dalam konfigurasi CORS sehingga

dapat membuka celah untuk data penting diakses oleh penyerang. Pengujian pada poin ini bertujuan untuk memastikan konfigurasi CORS yang diterapkan pada sistem

8. *Testing for Cross Site Flashing*

Flash Player adalah sebuah *software* yang digunakan untuk memainkan konten multimedia, *Rich Internet Applications*, dan *streaming* audio video. Untuk lebih memanjakan pengalaman pengguna sebuah web terkadang menggunakan animasi interkaktif untuk berkomunikasi dengan pengguna. Di sinilah *Flash Player* dibutuhkan. Kerentanan dapat terjadi ketika konten yang dimuat oleh *Flash Player* dapat dimodifikasi oleh penyerang. Pengujian pada poin ini bertujuan untuk mengetahui keamanan *file flash player* yang digunakan oleh sistem terhadap serangan yang memanfaatkan *file* tersebut.

9. *Testing for Clickjacking*

Ketika *resource* web dapat diakses dengan mudah tanpa adanya konfigurasi CORS yang benar, kerentanan yang timbul dapat terjadi seperti *Clickjacking*. *Clickjacking* memanfaatkan sebuah halaman web yang dapat termuat ke dalam halaman web lainya secara utuh, dengan cara menyisipkan tautan berbahaya di atas tautan aslinya sehingga korban yang tidak tahu akan hal ini menganggap bahwa halaman web tersebut adalah asli. Penyerang dapat dengan mudah mendapatkan informasi korban. Pengujian pada poin ini bertujuan untuk mengetahui keamanan sistem terhadap serangan *Clickjacking* dengan memuat sistem pada domain lainya.

10. *Testing WebSockets*

Merupakan sebuah protokol komunikasi yang menyediakan saluran *dupleks* secara penuh melalui koneksi TCP tunggal. *WebSockets* merupakan standar komunikasi baru yang menyediakan komunikasi secara *realtime*. Sebelum *WebSockets*, terdapat solusi terdahulu seperti *polling* dan *long-polling*. *WebSockets* lebih unggul karena dapat memangkas *latency* dan mengurangi beban jaringan yang tidak perlu. Kerentanan terjadi ketika validasi *origin* tidak dilakukan oleh *server Websockets*. Hal ini akan mempermudah penyerang untuk masuk ke dalam koneksi *Websockets* dari *origin* manapun. Kerentanan ini dapat diperluas untuk serangan lain seperti CSRF (*Cross Site Request Forgery*). Pengujian pada poin ini bertujuan untuk mengetahui keamanan sistem terhadap serangan yang memanipulasi lalulintas *Websockets* pada sistem.

11. *Testing Web Messaging*

Merupakan sebuah API (*Application Programming Interface*) yang diperkenalkan dalam draf WHATWG (*Web Hypertext Application Technology Working Group*) HMTL5 sebuah komunitas yang tertarik untuk mengembangkan dan memajukan HTML dan teknologi terkait,

dikenal juga dengan nama *cross-document messaging*. API ini memungkinkan dokumen berkomunikasi satu sama lain dari berbagai *origin* atau domain asal. Sama seperti pada *Websockets*, kerentanan terjadi ketika *origin* tidak tervalidasi dengan benar maka akan menimbulkan kerentanan. Pengujian pada poin ini bertujuan untuk memastikan keamanan sistem terhadap serangan yang memanfaatkan validasi yang tidak benar dari *origin* lainya.

12. *Testing Browser storage*

Sebelum dikenalkanya HTML5, data aplikasi harus disimpan dalam *cookie*, termasuk dalam setiap *server request*. Dengan Web *Storage*, data aplikasi dapat tersimpan secara lokal di dalam *browser* pengguna. Jumlah dari penyimpanan juga jauh lebih besar dari *cookie*. Beberapa mekanisme yang disediakan *browser* untuk menyimpan dan mengambil data antara lain:

a. *Local Storage*
b. *Session Storage*
c. *IndexedDB*
d. *Web SQL (Deprecated)*
e. *Cookies*
f. *Cache Storage*

Pengujian pada poin ini bertujuan untuk memastikan bahwa data yang tersimpan ke dalam *Browser Storage* telah organisir dengan baik.

13. *Testing for Cross-Site Script Inclusion*

Cross-Site Script Inclusion (XSSI) merupakan salah satu dari XSS. Serangan ini mirip dengan *Cross Site Request Forgecy* (CSRF) namun dengan tujuan yang berbeda. Jika CSRF menggunakan autentikasi untuk menjalankan tindakanya, XSSI menggunakan *Javascript* untuk membocorkan data sensitif. Kedua serangan ini berjalan pada sisi pengguna. Pengujian pada poin ini bertujuan untuk memastikan bahwa pengiriman data dari *server* menuju *client* telah terjaga dengan aman.

2.4 Penelitian Terkait

Penelitian tentang topik keamanan sistem berbasis web tentunya sudah banyak dilakukan sebelumnya. Dalam melakukan penelitian ini beberapa peneltian sebelumnya yang berasal dari berbagai sumber digunakan sebagai acuan. Beberapa penelitian yang dipilih memiliki topik yang sama dengan beberapa metode yang berbeda.

Berikut adalah lima penelitian yang dipilih yang digunakan sebagai acuan dalam penelitian ini:

a. Peneltian Cunong, Saputra, & Puspitasari (2020) melakukan uji keamanan dari sebuah web penanaman modal dan pelayanan terpadu satu pemerintah XYZ. Pengujian ini diawali dengan melakukan *Vulnerability Assessment* dilanjutan dengan melakukan *Penetration Testing* dengan standar *Penetration Testing Execution Standard* (PTES). Kemudian dihasilkan sebuah analisis hasil kerentanan yang ditemukan pada web tersebut. Dari penelitian ini diketahui bahwa dalam melakukan *Penetration Testing* yang lebih terstruktur dapat menggunakan standar PTES.
b. Penelitian Zen, Gultom, & Reksoprodjo (2020) melakukan uji keamanan *webserver* dengan melakukan *security assessment* menggunakan beberapa tahapan seperti *scanning vulnerability* standar OWASP, *Common Vulnerability Scoring System* yang kemudian akan menghasilkan sebuah penilaian kelayakan pada suatu sistem. Dari penelitian ini diketahui bahwa *Common Vulnerability Scoring System* cukup bagus sebagai penilaian kerentanan dari sebuah sistem karena terus diperbaharui dan dikembangkan.
c. Penelitian Yunanri, Riadi, & Yudhana (2016) melakukan uji keamanan sebuah *webserver* dengan metode *Penetration Testing*. Pada penelitian ini, dijelaskan lebih tentang metode *Penetration Testing* seperti fase yaitu fase persiapan, fase tes, dan fase analisis. Kemudian langkah-langkah juga dijelaskan dalam penelitian ini yaitu pengumpulan informasi, analisis kerentanan, dan eksploitasi kerentanan. Dari penelitian ini diketahui bahwa dalam melakukan Penetration Testing harus melewati fase persiapan dengan mengumpulkan segala sesatu yang berkaitan dengan kebutuhan pengujian kemudian fase tes yaitu fase pengujian sistem dan yang terakhir adalah fase analisis yaitu fase penyusunan hasil pengujian yang akan dianalisis lebih lanjut.
d. Penelitian Andria (2020) melakukan uji keamanan pada sebuah web menggunakan *tools* WEBPWN3R, merupakan sebuah *tools* yang berjalan pada sistem operasi Kali Linux yang dapat menganalisis dan mendeteksi sebuah kerentanan pada web. Penelitian ini menghasilkan sebuah analisis hasil kerentanan yang dapat dijadikan saran perbaikan yang tepat berdasarkan hasil kerentanan yang ditemukan. Dari penelitian ini diketahui bahwa sistem operasi Kali Linux merupakan sistem operasi yang dirancang untuk kepentingan keamanan jaringan dengan disediakan dan dikembangkan beberapa *tools* dengan kemampuan masing-masing.

e. Penelitian Tenggono, Purnama, & Budi (2018) uji keamanan pada *webserver* palcomtech, pengujian *webserver* tersebut dilakukan dengan tiga tahpan yaitu diagnosa, *action planing*, dan *action taking*. Dari pengujian tersebut dihasilkan beberapa celah keamanan yang dapat menjadi rekomendasi perbaikan. Pada pengujian ini diketahui bahwa pemakaian *framework* dalam pengembangan sistem juga akan mempengaruhi metode pengujian yang akan dilakukan. Seperti penggunaan *tools* WPScan pada penelitian ini untuk menguji sistem yang dikembangkan menggunakan *framework* WordPress.

Tabel 2.2 Penelitian sebelumnya

No	Peneliti	Objek Penelitian	Kerangka Kerja	*Tools*	Metode *Pentest*
1	Penelitian ini	Sistem Informatics Expo Universitas Islam Indonesia	OWASP WSTG v4.2	OWASP ZAP, Burp Suite, Zap, Acunetix, Vais, Clickjacking-Tester	*Black-Box testing*
2	Denis Nigel Cunong, Muhardi Saputra, Warih Puspitasari (2020)	*Website* Dinas Penanaman Modal dan Pelayanan Terpadu Satu Pintu Pemerintahan XYZ	OWASP	OWASP ZAP, Nikto, Zenmap	-
3	Bita Parga Zen, Rudy A.G Gultom, Agus H.S Reksoprodjo (2020)	*Webserver* di lingkungan labolatorium Universitas Pertahanan Indonesia	CVSS, OWASP TOP 10	OWASP ZAP,	*Black-Box testing*
4	Yunanri W, Imam Riadi, Anton	Sebuah *webserver* (tidak disebutkan *webserver*nya)	-	Beberapa *tools* di BackTrack Linux	*Black-Box testing*

	Yudhana (2016)				
5	Andria (2020)	Beberapa *website* (tidak disebutkan *website*nya)	-	WEBPWN3R	*Black-Box testing*
6	Alfred Tenggono, Tegar Purnama, Andi Setia Budi (2018)	*Webserver* www.palcomtech.com	-	WPScan	-

Penelitian yang dijadikan acuan dalam melakukan penelitian ini lebih berfokus terhadap keamanan *webserver* atau lebih kepada sisi *server*. Pada penelitian ini dilakukan pengujian keamanan kepada sisi pengguna dengan menggunakan salah satu *framework* dari OWASP yaitu WSTG v4.2. Dengan metode penelitian *blackbox testing*, beberapa tools dipilih guna melakukan pengujian seperti ZAP, Burp Suite, Acunetix, dan lain sebagainya sesuai dengan kemampuan tools masing-masing. Pemilihan tools dilakukan setelah mengetahui tingkat kemampuan tools dari penelitian yang dilakukan sebelumnya.

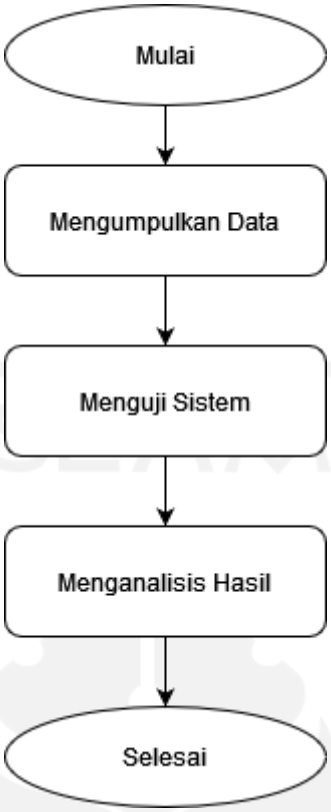
BAB III
METODOLOGI

Dalam melakukan penelitian, dibutuhkan metodologi sebagai tata cara yang digunakan untuk mendapatkan hasil dari penelitian. Tata cara tersebut disususn secara sistematis dan terstruktur agar proses penelitian tetap berfokus kepada tujuannya. Tata cara diperoleh dari acuan dan landasan teori yang telah didapatkan dari proses sebelumnya.

3.1 Metode Penelitian

Dalam melakukan penelitian pengujian keamanan sistem Informatics Expo ini terdapat tahapan pengumpulan data. Pada tahapan pengumpulan data ini terdapat dari beberapa sumber seperti melalui literatur jurnal, paper ilmiah, tugas akhir, buku, termasuk juga media digital seperti internet. Selain itu, informasi didapatkan melalui diskusi dengan pengembang sistem Informatics Expo.

Kemudian informasi tersebut digunakan untuk menentukan jenis pengujian yang cocok terhadap sistem. Beberapa *tools* dipakai dalam melakukan pengujian sistem. Jenis *tools* yang dipakai disesuaikan dengan beberapa informasi yang didapat. Kemudian hasil yang diperoleh dari *tools* apakah ada celah keamanan ataupun tidak akan didokumentasikan. Setelah proses pengujian telah dilakukan, dilakukan analisis hasil. Analisis hasil dilakukan untuk menentukan tingkat keamanan sistem tersebut dari hasil pengujian dan diharapkan juga dari analisis hasil ini bisa untuk menjadi saran perbaikan sistem. Berikut merupakan bagan alur proses penelitian seperti pada Gambar 3.1



Gambar 3.1 Bagan Alur Penelitian

3.2 Metode Pengumpulan Data

Setelah memiliki acuan pengujian sistem dari studi literatur yang telah dilakukan. Kemudian dilakukan wawancara dengan pengembang Informatics Expo didapatkan bahwa sampai saat penelitian ini dibuat sistem belum pernah dilakukan pengujian keamanan. Sistem juga belum pernah mengalami masalah terkait keamanan. Ditemukan juga bahwa sistem belum memiliki sistem *staging*. Dalam pengembangan sistem juga menggunakan beberapa teknologi yang telah diterapkan. Data yang telah terkumpul ini kemudian dijadikan referensi untuk langkah pengujian yang akan dilakukan selanjutnya terkait dengan kebutuhan yang diperlukan.

3.3 Alat Kebutuhan Penelitian

Dalam melakukan penelitian ini, alat kebutuhan dibagi menjadi dua kategori yaitu perangkat lunak dan perangkat keras.

3.3.1 Perangkat Keras

Perangkat keras yang digunakan adalah sebuah laptop. Adapun spesifikasi minimun dan spesifikasi yang digunakan seperti pada Tabel 3.1 berikut:

Tabel 3.1 Spesifikasi Perangkat Keras

Komponen	Spesifikasi Minimum	Spesifikasi yang digunakan
CPU	1Ghz or *faster*	Intel Core i3 6006U (3M *Cache* 2.3Ghz)
RAM	2GB	4GB DDR4 (2400Mhz *Dual Channel*)
Storage	20GB	SSD 512GB
VGA	Nvidia GT 9600 512MB or better	Nvidia GTX 920MX 2GB

3.3.2 Perangkat Lunak

Dalam penelitian ini, digunakan dua jenis sistem operasi yaitu berbasis Windows dan Kali Linux. Pada masing-masing sistem operasi, terdapat *tools* yang dipilih untuk melakukan penelitian. Berikut *tools* yang dipilih berdasarkan basis sistem operasinya.

A. Sistem operasi Windows

1. Microsoft Word, berguna untuk penulisan dan penyusunan penelitian.
2. Vmware, berguna sebagai virtual mesin untuk menjalankan sistem operasi kali linux dengan spesifikasi pada tabel 3.2 berikut:

Tabel 3.2 Spesifikasi Mesin Virtual

Komponen	Spesifikasi Minimum	Spesifikasi yang digunakan
Processor	1 *Core or Better*	2 *Core*
RAM	2 GB	2 GB
Storage	15 GB	20 GB

3. Mozilla Firefox dan Google Chrome, berguna untuk membuka sistem Informatics Expo dan menjalankan beberapa poin pengujian.
4. BulitWith, merupakan *addon* pada *browser* untuk mengetahui beberapa teknologi yang diterapkan oleh sistem.
5. *Clickjacking* Cyber Moon, sebuah web yang menyediakan tempat untuk pengecekan kerentanan *Clickjacking*.
6. ZAP *tools*, merupakan sebuah *tools* penetration testing resmi dari OWASP. Digunakan untuk proses *scanning* celah keamanan.

7. Acunetix, merupakan *tools vulnerability scanning* yang dapat memilih beberapa poin pengujian.

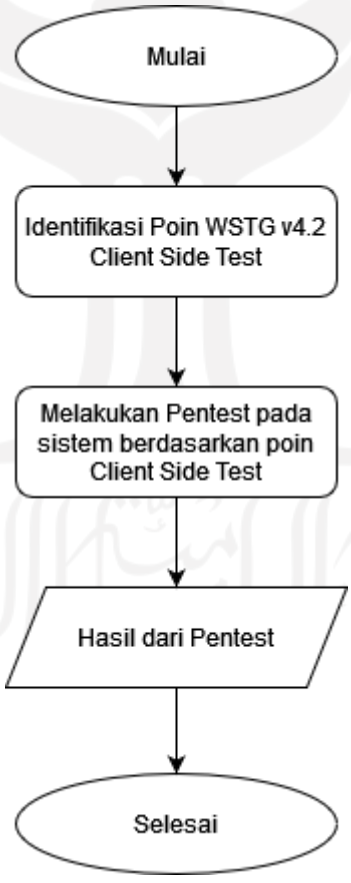
B. Sistem Operasi Kali Linux

1. Wafwoof, berguna untuk pemeriksaan *firewall* pada suatu sistem.
2. *Clickjacking*-Tester, berguna untuk melakukan pemeriksaan celah keamanan terhadap serangan *Clickjacking.*
3. Burp Suite, untuk melakukan *penetration testing* .
4. Vais, berguna untuk melakukan pengecekan *file* SWF pada sistem.

3.4 Metode Pengujian Sistem

3.4.1 Alur Pengujian Sistem

Dalam melakukan pengujian sistem, terdapat beberapa langkah yang dilakukan, seperti pada Gambar 3.2



Gambar 3.2 Bagan Alur Pengujian

Berikut penjelasan dari bagan alur di atas :

a. Identifikasi poin WSTG *Client-side Testing* yang memungkinkan untuk dilakukan pengujian dengan *penetration testing* menggunakan *blackbox.*
b. Kemudian setelah semua poin diidentifikasi, dilakukan uji *Penetration testing* terhadap sistem berdasarkan poin WSTG *Client-side Testing*. Pada tahapan ini berbagai *tools* digunakan sesuai dengan fungsinya masing-masing.
c. Setelah dilakukan pengujian, akan didapatkan sebuah hasil pengujian berdasarkan WSTG *Client-side Testing*. Hasil ini akan dianalisis lebih lanjut.

3.4.2 Implementasi Pengujian

Pada tahap pengujian terdapat beberapa langkah yang dilakukan. Berikut merupakan beberapa langkah yang dilakukan dalam melakukan proses pengujian:

a. Identifikasi poin WSTG *Client-side Testing*

Pada tahap ini dilakukan identifikasi poin WSTG *Client-side Testing*. identifikasi tersebut kemudian dikelompokkan ke dalam Tabel 3.3 berikut:

Tabel 3.3 Identifikasi Poin

WSTG Client-side Testing	Langkah Pengujian	Metode *Blackbox*	Target Serangan pada Sistem	Jenis Eksploitasi
WSTG-CLNT-01	Identifikasi *sinks*, Menyusun kode serangan	✓	Fitur Pencarian	Injeksi kode program, Pengujian dengan *tools*
WSTG-CLNT-02	Identifikasi *sinks*, Menyusun kode serangan	✓	Fitur Pencarian	Injeksi kode program
WSTG-CLNT-03	Identifikasi *sinks*, Menyusun kode serangan	✓	Fitur Pencarian	Injeksi kode program
WSTG-CLNT-04	Identifikasi *Injection Points*, Penilaian	✓	*Request header*	Penangkapan data dengan *tools*
WSTG-CLNT-05	Identifikasi *Injection Points*, Penilaian	✓	Fitur Pencarian	Injeksi kode program
WSTG-CLNT-06	Identifikasi *sinks*, Menyusun kode serangan	✓	*Request header*	Perubahan paket data dengan *tools*

WSTG-CLNT-07	Identifikasi *Endpoints* dan Analisis konfigurasi Header	✓	*Request header*	Perubahan paket data dengan *tools*
WSTG-CLNT-08	*Decompile* dan Analisis *file* SWF	✓	*File* SWF pada sistem	Pengujian menggunakan *tools*
WSTG-CLNT-09	Identifikasi penerapan keamanan pada *header*	✓	Halaman sistem	Pengujian menggunakan *tools*
WSTG-CLNT-10	Identifikasi fitur, Penilaian implementasi	✓	Lalu lintas Websockets	Pengujian menggunakan *tools*
WSTG-CLNT-11	Penilaian dan Validasi Metode *Messaging*	✓	Fungsi API *WebMessaging*	Pencarian fungsi API untuk *WebMessaging*
WSTG-CLNT-12	Identifikasi dan penanganan *storing* data	✓	*Browser storage*	Pencarian direktori yang terbuka pada sistem
WSTG-CLNT-13	Identifikasi fungsi yang menggunakan JSONP	✓	Request Header	Pencarian fungsi yang terbuka pada proses pengiriman data

Pengelompokan poin tersebut bertujuan untuk pelaksanaan *penetration testing* yang lebih terstruktur sesuai dengan Framework WSTG v4.2 poin *Client-side Testing*.

b. Penetration Testing

Pada langkah ini, penguji bertindak sebagai pengunjung biasa yang tidak memiliki akun pada sistem. Penguji menyimulasikan diri sebagai penyerang yang sebelumnya tidak ada pengetahuan mendalam tentang sistem ini. Sistem ini bersifat terbuka untuk umum, pengguna yang tidak memiliki akun bisa melihat beberapa informasi dan bisa memanfaatkan beberapa fitur.

Tidak adanya *staging* menjadikan batasan bagi penguji untuk melakukan pengujian di sisi pengguna yang telah terdaftar, karena di sisi pengguna tedaftar terdapat beberapa proses bisnis yang berhubungan dengan *database*. Dikhawatirkan jika pengujian juga dilakukan di sisi pengguna terdaftar akan berpengaruh pada sistem yang sedang berjalan. Berikut merupakan poin pengujian berdasarkan WSTG *Client-side Testing*:

1. WSTG-CLNT-01 *Testing* for DOM-*Based Cross Site Scripting*

Pada bagian ini langkah pertama adalah dilakukan pengujian pada fitur pencarian yang terdapat pada sistem. Langkah pertama adalah dengan mengidentifikasi fungsi yang rentan akan serangan XSS dengan mencari fungsi tersebut menggunakan *tools developer* pada *browser*. Mengacu pada pedoman WSTG *Client-side Testing* ada beberapa fungsi yang rentan terhadap serangan XSS seperti pada Gambar 3.3 berikut:

```
<script>
var pos=document.URL.indexOf("message=")+5;
document.write(document.URL.substring(pos,document.URL.length));
</script>
```

Gambar 3.3 Contoh fungsi kerentanan DOM

Kemudian langkah selanjutnya adalah membuat kode program *Javascript* yang telah disesuaikan dengan fungsi yang telah ditemukan. Berikut contoh kode program yang bisa dijadikan serangan seperti pada Gambar 3.4:

```
<script>alert(“XSS”)</script>
```

Gambar 3.4 Contoh kode program serangan XSS

Kode program pada Gambar 3.4 akan diinjeksikan ke dalam fitur pencarian yang akan memanfaatkan fungsi pada Gambar 3.3. Jika sistem tidak memiliki pertahanan terhadap jenis serangan ini, maka kode program dapat dijalankan dan akan memunculkan sebuah jendela notifikasi yang bertuliskan XSS pada sistem.

2. WSTG-CLNT-02 *Testing for JavaScript Execution*

Pada bagian ini dilakukan pengujian pada bagian fitur pencarian atau kolom URL yang terdapat pada web. Langkah pertama adalah identifikasi fungsi yang dapat menjadi celah keamanan dengan mencari fungsi tersebut menggunakan *tools developer* pada *browser*. Mengacu pada WSTG-CLNT-02 contoh fungsi tersebut seperti pada Gambar 3.5 berikut:

```
<script>
function loadObj(){
    var cc=eval('('+aMess+')');
    document.getElementById('mess').textContent=cc.message;
}

if(window.location.hash.indexOf('message')==-1) {
    var aMess='({"message":"Hello User!"})';
} else {
    var
aMess=location.hash.substr(window.location.hash.indexOf('message=
')+8)
}
```

```
</script>
```

Gambar 3.5 Contoh fungsi Javascript

Langkah selanjutnya membuat sebuah kode program untuk dimasukkan ke fungsi tersebut. Berikut pada Gambar 3.6 merupakan contoh kode program yang dapat dijadikan serangan pada fungsi tersebut:

```
<img src=1 href=1 onerror="javascript:alert(1)"></img>
```

Gambar 3.6 Contoh kode program serangan *Javascript*

Kode program pada Gambar 3.6 akan dimasukan ke dalam fitur pencarian yang akan memanfaatkan fungsi pada Gambar 3.5. Jika sistem tidak memiliki pertahanan terhadap jenis serangan ini, maka kode program dapat dijalankan dan akan memunculkan sebuah jendela notifikasi yang bertuliskan angka 1 pada sistem.

3. WSTG-CLNT-03 *Testing for* HTML *Injection*

Pada bagian ini dilakukan pengujian pada fitur pencarian. Sama halnya dengan DOM XSS, pengujian dilakukan dengan menginjeksikan kode program berbahaya yang sebelumnya telah disesuaikan dengan fungsi yang ditemukan. Berikut merupakan contoh fungsi yang rentan untuk serangan HTML Injection seperti pada Gambar 3.7:

```
<script>
var userposition=location.href.indexOf("user=");
var user=location.href.substring(userposition+5);
document.write("<h1>Hello, " + user +"</h1>");
</script>
```

Gambar 3.7 Contoh fungsi perubahan elemen HTML pada DOM

Kode program yang diinjeksikan merupakan *Javascript* yang dikemas dengan *tag* HTML. Jika sistem dapat menjalankan *tag* tersebut, kode *Javascript* di dalamnya juga akan ikut tereksekusi. Berikut contoh kode program yang dapat dijadikan serangan HTML *injection* seperti pada Gambar 3.8 berikut:

```
<img%20src='coba'%20onerror=alert(1)>
```

Gambar 3.8 Contoh kode program serangan HTML *injection*

Jika sistem tidak memiliki pertahanan terhadap jenis serangan ini, maka kode program dapat dijalankan dan akan memunculkan sebuah jendela notifikasi yang bertuliskan angka 1 pada sistem.

4. WSTG-CLNT-04 *Testing for Client-side* URL *Redirect*

Pada bagian ini dilakukan pengujian dengan cara memanipulasi fungsi *redirect* URL. Tahap pertama yang dilakukan adalah identifikasi fungsi yang ditemukan pada sistem.

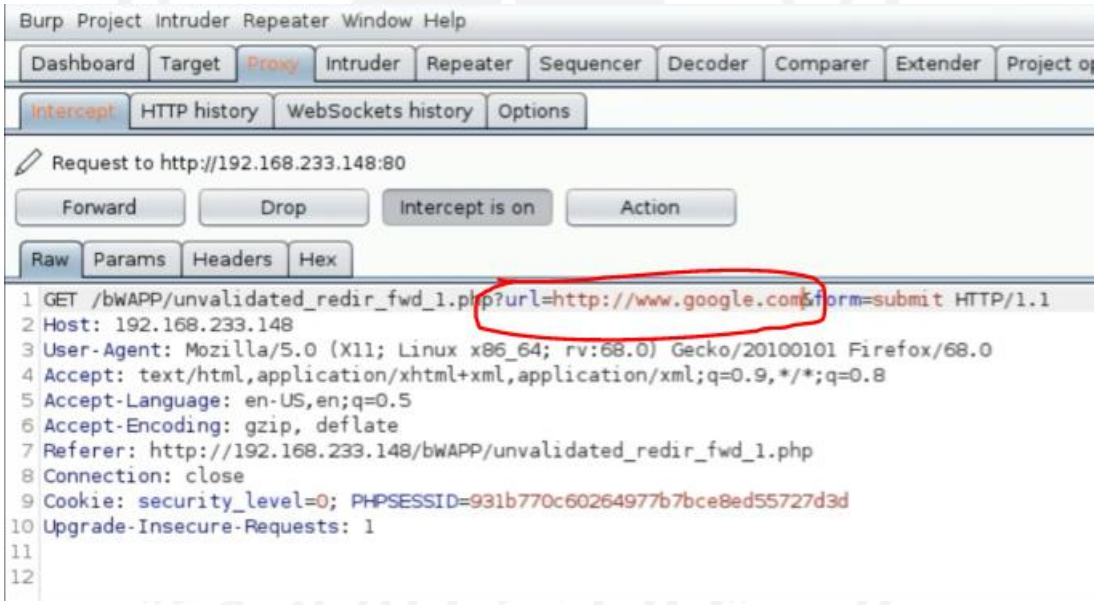
Berikut adalah contoh fungsi yang memiliki kerentanan terhadap *redirect* seperti pada Gambar 3.9:

```
var redir = location.hash.substring(1);
if (redir) {
    window.location='http://'+decodeURIComponent(redir);
}
```

Gambar 3.9 Contoh fungsi

Jika fungsi pada Gambar 3.9 ditemukan, tahap kedua adalah melakukan manipulasi URL, manipulasi ini memiliki tujuan untuk mengarahkan pengguna ke alamat URL yang telah disiapkan oleh penyerang.

Selain itu juga dilakukan identifikasi URL tujuan menggunakan *tools* Burp Suite. *Tools* akan menangkap *request get* yang mengarahkan ke URL tertentu. Kemudian akan diidentifikasi apakah URL tujuan terlihat dalam sebuah *request get* tersebut atau tidak. Berikut Gambar 3.10 contoh penggunaan *tools* Burp Suite :



Gambar 3.10 Contoh penggunaan *tools* Burp Suite

Jika fungsi dan *path redirect* berhasil ditemukan serta dapat dimanipulasi, maka dapat dipastikan bahwa sistem memiliki kerentanan pada poin pengujian ini.

5. WSTG-CLNT-05 *Testing for* CSS *Injection*

Pada bagian ini dilakukan pengujian pada fitur pencarian. Sama halnya dengan DOM XSS pengujian dilakukan dengan menginjeksikan kode program berbahaya. Kode program yang diinjeksikan merupakan *Javascript* yang dikemas dengan *tag* CSS. Jika sistem dapat menjalankan *tag* tersebut, kode *Javascript* di dalamnya juga akan ikut

tereksekusi. Berikut contoh kode program yang dapat dijadikan serangan CSS *injection* seperti pada Gambar 3.11 :

```
red;-o-link:'<javascript:alert(1)>';-o-link-source:current;
red;-:expression(alert(URL=1));
red;-moz-binding:url(victim/page?par=val#checkbox);
```

Gambar 3.11 Contoh kode program serangan CSS *injection*

Jika sistem tidak memiliki pertahanan terhadap jenis serangan ini, maka kode program dapat dijalankan dan akan memunculkan sebuah jendela notifikasi yang bertuliskan angka 1.

6. WSTG-CLNT-06 *Testing for Client-side Resource Manipulation*

   Pada bagian ini dilakukan pengujian dengan cara identifikasi *resource* yang terhubung dengan sistem. *Resource* pada sistem dapat dilihat dengan menggunakan *tools developer* pada *browser*. *Resource* yang telah didapatkan kemudian akan dianalisis apakah mengandung kerentanan atau tidak sesuai dengan pedoman WSTG. Berikut Tabel 3.4 merupakan macam *resource* dalam struktur web menurut pedoman WSTG:

Tabel 3.4 *Resource* dalam struktur web

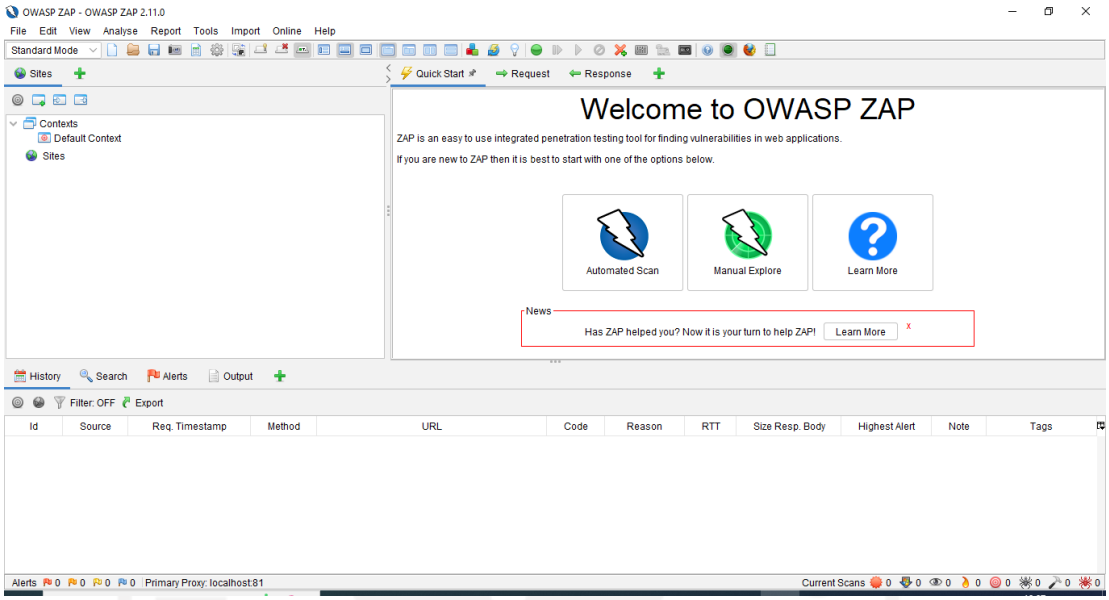
Tipe *resource*	*Tag/Method*	*Sink*
Frame	`iframe`	`src`
Tautan	`a`	`href`
AJAX *Request*	`xhr.open(method, [url], true);`	`URL`
CSS	`link`	`href`
Gambar	`img`	`src`
Objek	`object`	`data`
Script	`script`	`src`

Jika pada poin pengujian ini ditemukan resource yang tidak divalidasi dengan baik maka dipastikan bahwa sistem ini memiliki kerentanan pada pengujian ini.

7. WSTG-CLNT-07 *Testing Cross Origin Resource Sharing*

   Pada bagian ini dilakukan pengujian terhadap konfigurasi *Cross Origin Resource Sharing*. Penggunaan konfigurasi *Cross Origin Resource Sharing* yang kurang tepat dapat menjadi celah untuk serangan ke dalam sistem. Pengujian ini menggunakan *tools* ZAP untuk pengujiannya. *Tools* ZAP akan otomatis mencari kerentanan *Cross Origin Resource Sharing*. *Tools* ini juga akan menampilkan letak kerentanan tersebut serta

penyebab kerentanan terjadi dan rekomendasi solusi untuk kerentanan tersebut. Berikut contoh penggunaan *tools* ZAP seperti pada Gambar 3.12:

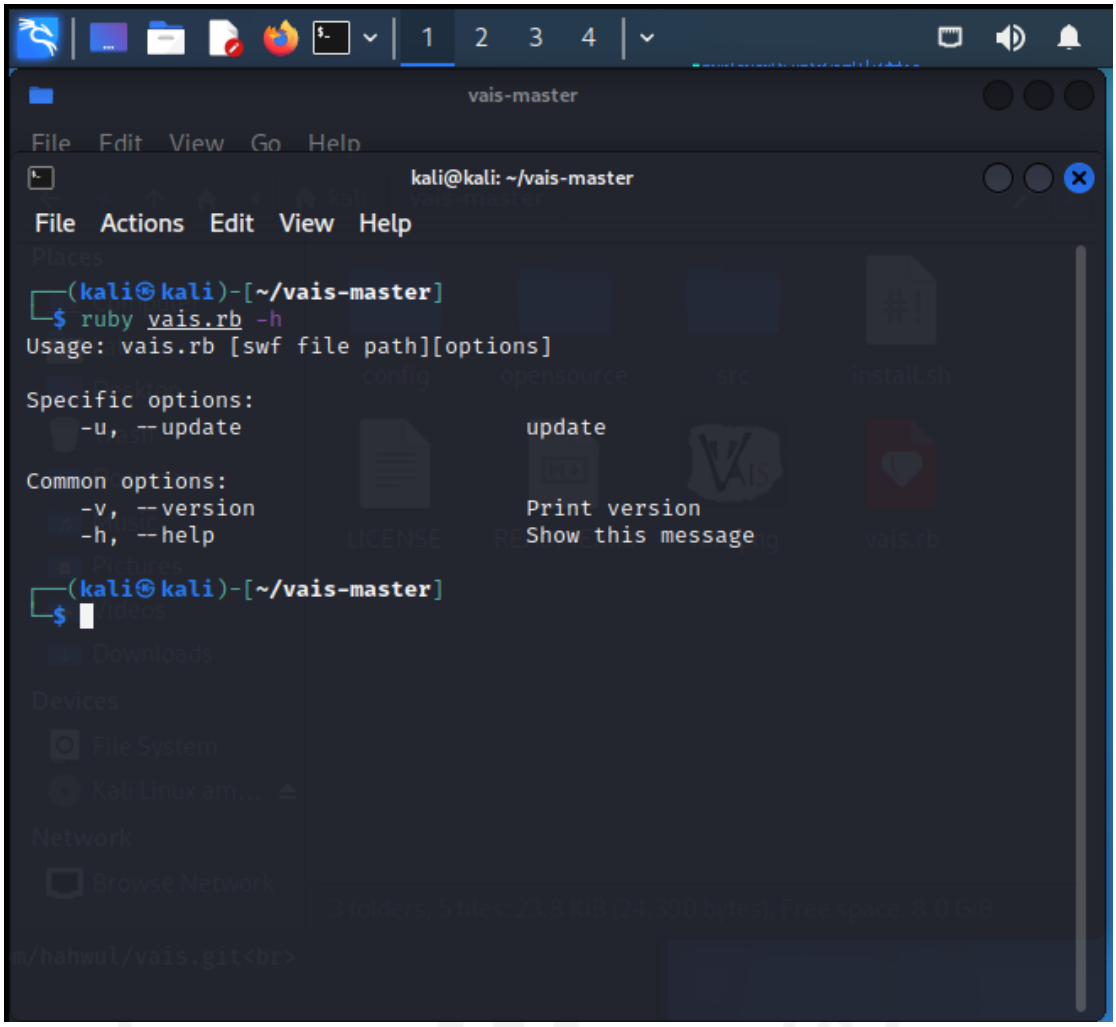


Gambar 3.12 Contoh penggunaan *tools* ZAP

Jika sistem memiliki kerentanan pada poin ini, maka *tools* akan mendeteksi kerentanan tersebut dan menampilkannya pada *tab Alerts* beserta dengan jumlah dan detail letak kerentanan yang terjadi pada sistem.

8. WSTG-CLNT-08 *Testing for Cross Site Flashing*

Pada bagian ini dilakukan pengujian pada file SWF pada sistem. Langkah pertama yang dilakukan adalah dengan cara melakukan penelusuran file dengan format SWF pada sistem menggunakan *tools developer* pada *browser*. Kemudian langkah selanjutnya file swf yang ditemukan dilakukan proses *decompile* menggunakan *tools* Vais. *Tools* Vais akan melakukan uji *flashing* secara otomatis. Berikut contoh penggunaan *tools* Vais seperti pada Gambar 3.13:



Gambar 3.13 Contoh penggunaan *tools* Vais

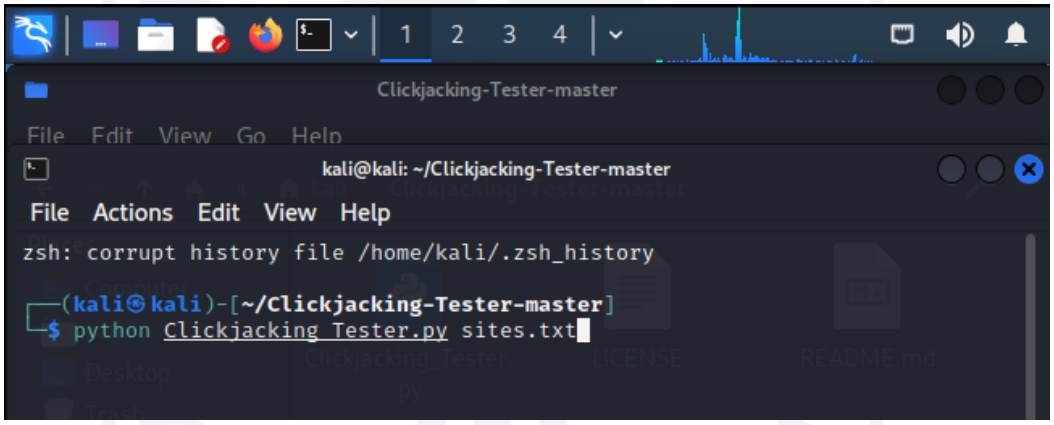
Jika *file* SWF ditemukan pada sistem dan berhasil diuji *flashing* pada *tools*, maka dapat dipastikan bahwa sistem memiliki kerentanan pada poin pengujian ini.

9. WSTG-CLNT-09 *Testing for Clickjacking*

Pada bagian ini dilakukan pengujian menggunakan *tools Clickjacking* Cyber Moon dan Clickjacing-Tester untuk memastikan sistem aman dari serangan *Clickjacking* atau tidak. Sesuai dengan pedoman WSTG-CLNT-09, sebuah sistem akan rentan jika berhasil dimuat ke dalam halaman web lainya. Dengan memasukkan URL sistem ke dalam *tools Clickjacking* Cyber Moon dan *Clickjacking-Tester*, akan terlihat jika sistem akan termuat atau tidak. Berikut contoh penggunaan *tools Clickjacking* Cyber Moon dan *Clickjacking-Tester* seperti pada Gambar 3.14 dan Gambar 3.15:

Gambar 3.14 Contoh penggunaan *tools* CyberMoon

Pada *tools Clickjacking* Cyber Moon jika sistem tidak memiliki pertahanan terhadap jenis serangan *Clickjacking* maka sistem akan termuat kedalam *tools*.

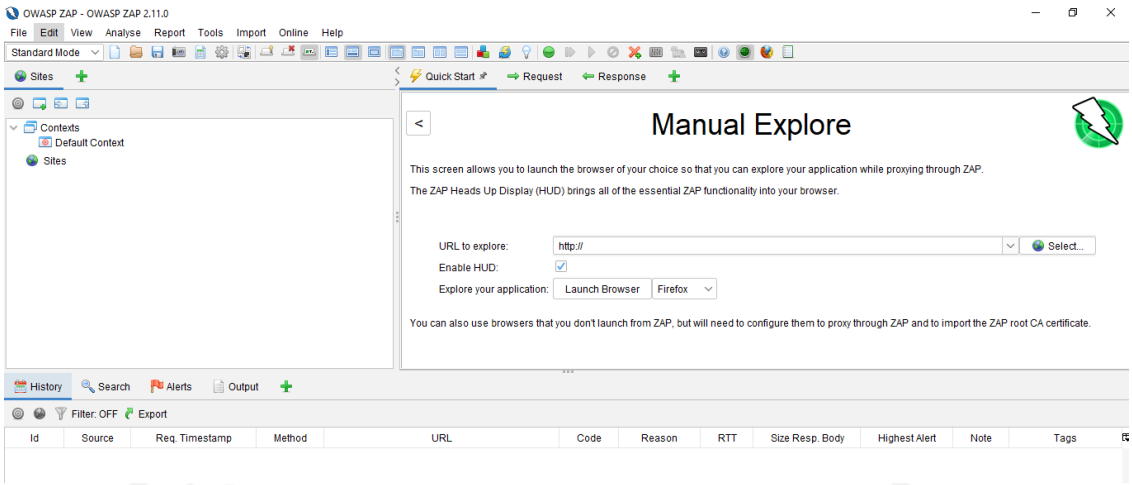


Gambar 3.15 Contoh penggunaan *tools* Clickjacking Tester

Pada *tools Clickjacking-Tester* jika sistem tidak memiliki pertahanan pada serangan *Clickjacking* maka akan menghasilkan status ditemukan kerentanan.

10. WSTG-CLNT-10 *Testing Websockets*

Pada pengujian ini dilakukan pengujian dengan cara mengidentifikasi *Websockets* pada sistem. Identifikasi *Websockets* menggunakan *tools* ZAP. Semua fitur pada sistem akan dicoba kemudian *tools* akan merekam lalu lintas websockets pada sistem. Daftar dari lalu lintas websocket yang terekam pada *tools* akan dianalisis apakah terdapat kerentanan sesuai dengan pedoman WSTG-CLNT-10 atau tidak. Berikut contoh penggunaan *tools* ZAP untuk *Testing Websockets* seperti pada Gambar 3.16:



Gambar 3.16 Contoh penggunaan *tools* ZAP

Jika lalu lintas *Websockets* ditemukan dan hasil analisis menunjukan adanya validasi yang kurang aman, maka dapat dipastikan sistem memiliki kerentanan terhadap serangan pada poin ini.

11. WSTG-CLNT-11 *Testing Web Messaging*

Pada pengujian ini dilakukan pengujian dengan cara mengidentifikasi fungsi untuk messaging API. identifikasi fungsi ini dilakukan dengan *tools developer* pada *browser*. Berikut contoh fungsi yang diidentifikasi seperti pada Gambar 3.17:

```
postMessage(message, targetOrigin)
postMessage(message, targetOrigin, transfer)
```

Gambar 3.17 Contoh fungsi untuk Web Messaging

Jika fungsi pada Gambar 3.17 ditemukan, pengujian selanjutnya adalah dengan menganalisis fungsi tersebut berdasarkan keamanan origin, validasi input, dan analisis kode statis. Pada pengujian ini sistem memiliki kerentanan jika fungsi API Web Messaging dapat terekspos dan tidak memiliki validasi yang aman.

12. WSTG-CLNT-12 *Testing Browser storage*

Pada pengujian ini dilakukan analisis terhadap penanganan sistem dalam menyimpan data yang dibutuhkan ke dalam *browser storage*. Analisis dilakukan dengan cara penelusuran data ketika data tersebut masuk ke dalam *browser storage* dengan bantuan *tools developer* pada *browser*. Berikut merupakan jenis *browser storage* yang menjadi tempat sistem memasukkan datanya ke dalam *browser*:

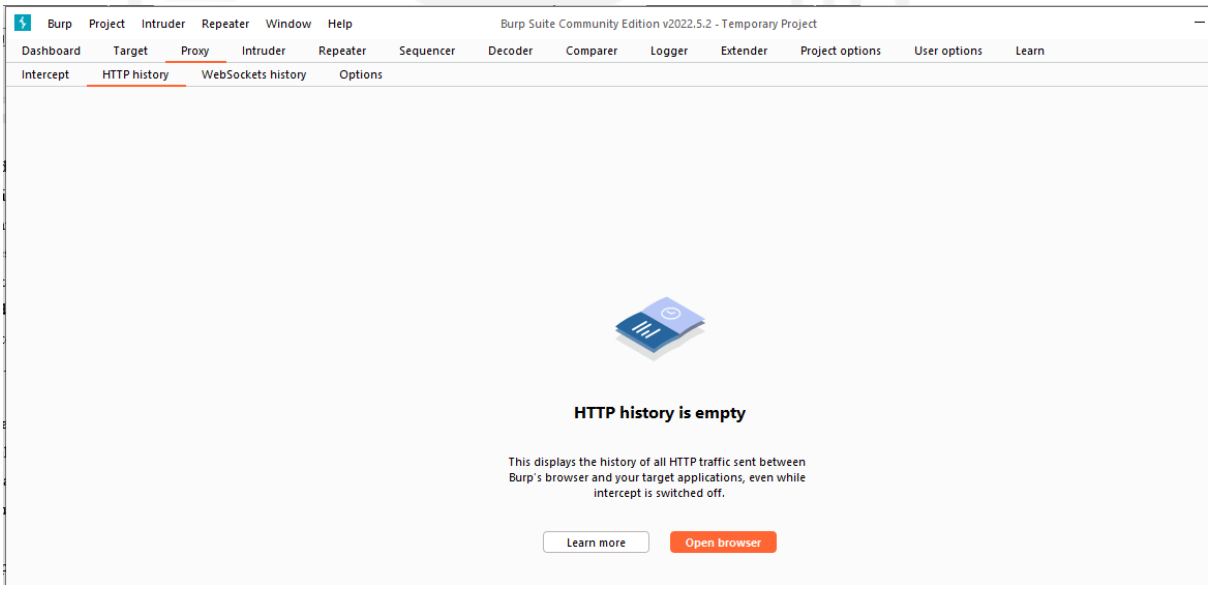
1. Local Sotrage
2. *Session Storage*
3. IndexedDB
4. Web SQL

5. Cookies
6. Cache *Storage*

Data pada masing-masing jenis *browser storage* tersebut dicek dan dipastikan bahwa setiap parameter seperti *value*, *path*, dan parameter lainnya tidak terlihat secara utuh. Untuk Web SQL, dipastikan juga tidak diterapkan lagi karena jenis *browser storage* ini sudah tidak digunakan lagi sejak tahun 2010. Pada poin ini sistem dipastikan aman jika sistem mampu mengirimkan beberapa data yang dibutuhkan oleh *browser storage* sesuai dengan jenisnya dan pengamanan data seperti enskripsi berhasil dilakukan.

13. WSTG-CLNT-13 *Testing for Cross Site Script Inclusion*

Pengujian ini dilakukan dengan cara mengidentifikasi teknik JSONP dalam melakukan pertukaran data. Identifikasi dilakukan dengan menggunakan *tools* Burp Suite. *Tools* akan merekam pertukaran data pada sistem. Kemudian data yang telah terekam akan diidentifikasi apakah menggunakan teknik JSONP atau tidak. Kemudian teknik JSONP yang ditemukan akan dianalisis apakah ada data sensitif yang dapat dibocorkan atau tidak melalui kerentanan *Cross Site Script Inclusion*. Berikut contoh penggunaan *tools* Burp Suite untuk *Testing for Cross Site Script Inclusion* seperti pada Gambar 3.18:



Gambar 3.18 Contoh penggunaan *tools* Burp Suite

Pada poin pengujian ini jika ditemukan pertukaran data menggunakan teknik JSONP ditemukan dan dari hasil analisis data tersebut dapat dibocorkan, maka dapat dipastikan bahwa sistem memiliki kerentanan pada poin ini.

3.5 Metode Analisis Hasil

Hasil dari penetration testing yang telah dilakukan kemudian akan dianalisis berdasarkan masalah dan hasil yang didapatkan selama proses penetration testing.

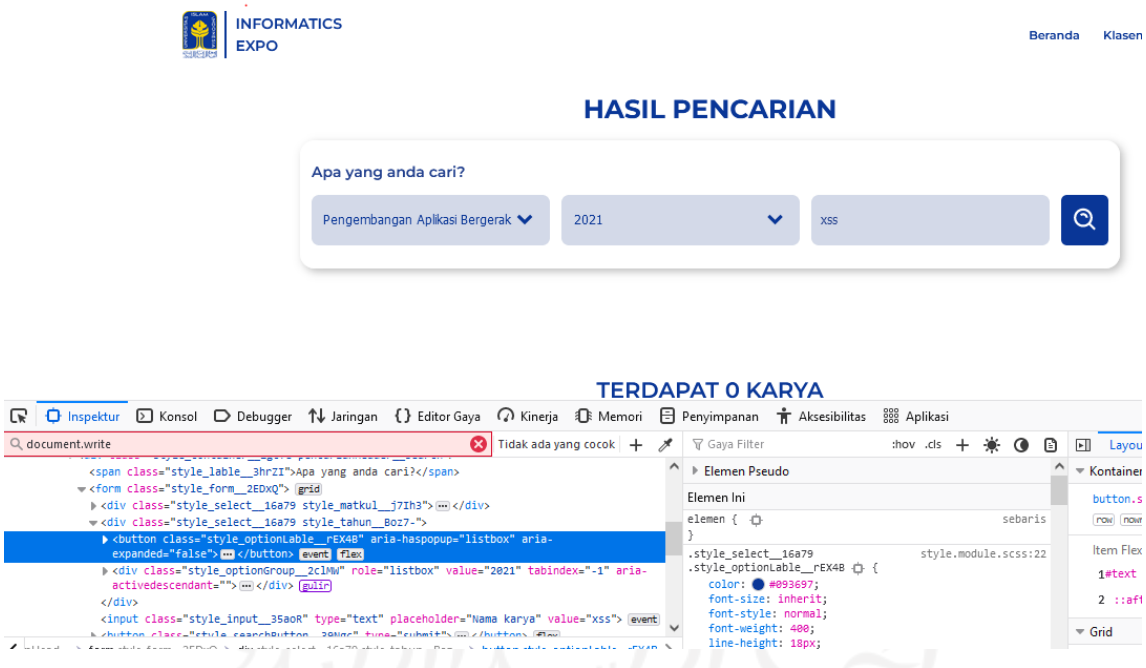
BAB IV
HASIL DAN PEMBAHASAN

Tahap ini merupakan tahap akhir dari penelitian ini. Pada proses pertama adalah penguatan tentang landasan teroi dan acuan penelitian, kemudian dilanjut dengan perancangan tata cara yang terstruktur berdasarkan landasan teori, dan pada tahap ini merupakan pelaksana dari tata cara yang telah dirancang sebelumnya. Pada tahap ini juga akan menghasilkan analisis dari penelitian yang telah dilakukan.

4.1 Hasil Pengujian

4.1.1 Hasil WSTG-CLNT-01 Testing for DOM-Based Cross Site Scripting

Tahap pertama pengujian ini adalah mencari fungsi yang mengubah elemen DOM. Pengujian dilakukan dengan memasukkan sebuah input ke dalam fitur pencarian seperti pada Gambar 4.1 berikut:

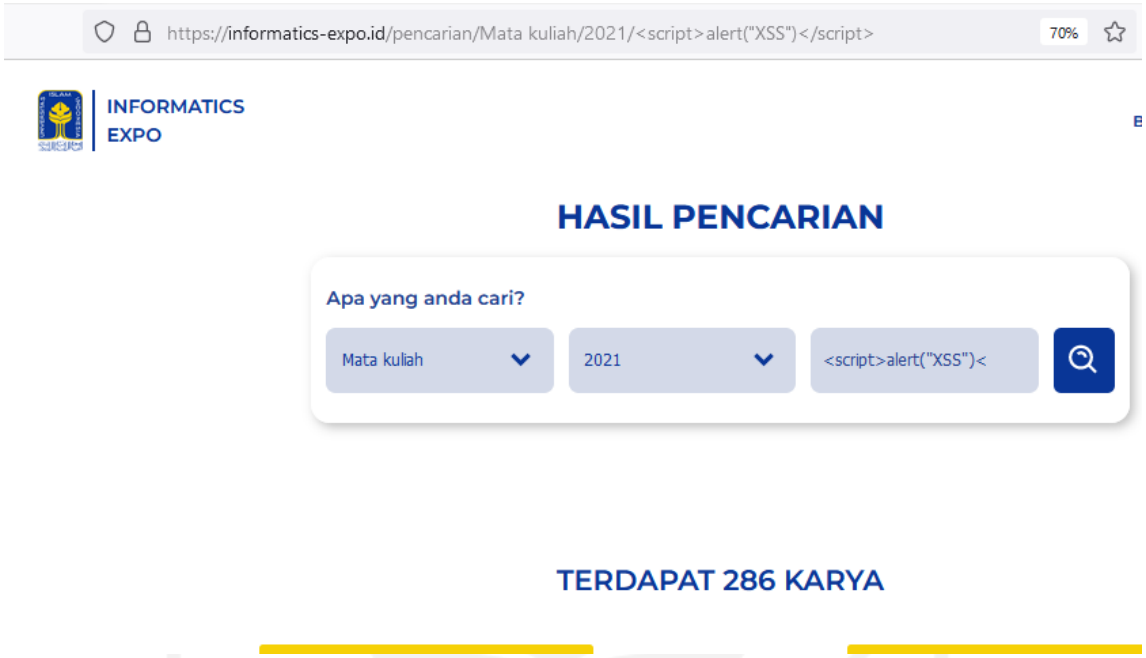


Gambar 4.1 Hasil dari pencarian fungsi DOM

Dari Gambar 4.1 berikut ditemukan bahwa hasil input dari pencarian tidak terefleksikan kembali dan tidak ditemukannya fungsi perubahan DOM yang rentan akan serangan. Dengan tidak ditemukannya fungsi, penyusunan kode serangan tidak bisa dilakukan.

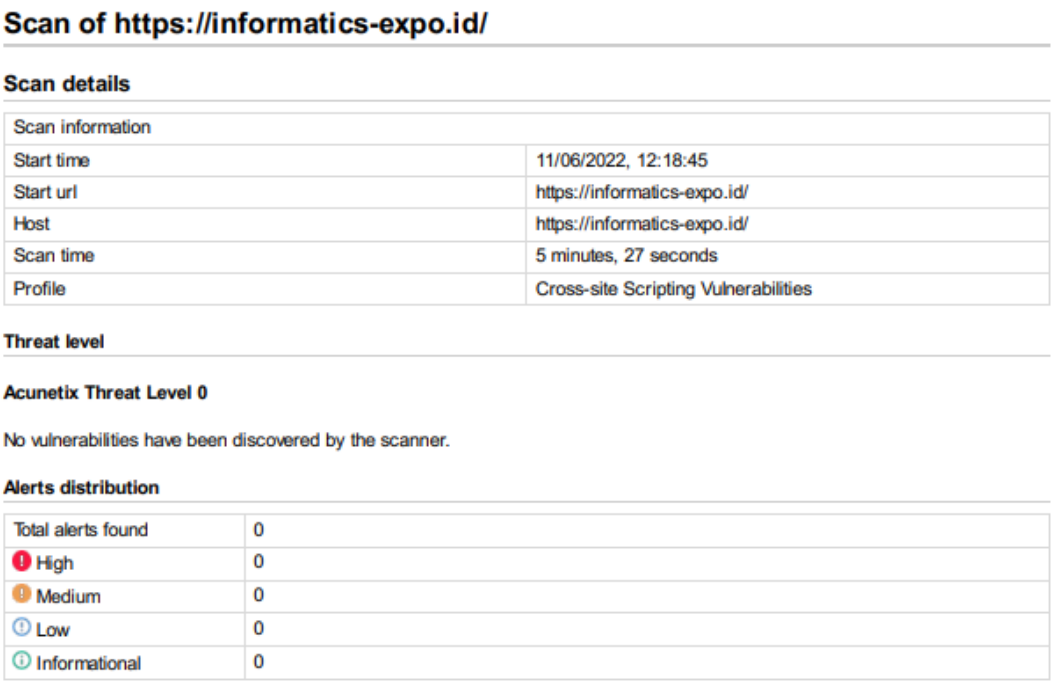
Selanjutnya injeksi kode program dilakukan dengan kode program XSS yang akan menampilkan *pop-up* notifikasi jika berhasil dijalankan. Kode program ini tidak disusun

sesuai dengan fungsi sistem karena fungsi sistem tidak dapat teridentifikasi. Kode program yang telah dimasukan tidak dapat tereksekusi, sistem membaca kode program sebagai input biasa, terbukti dengan munculnya data karya berdasarkan karakter yang terbaca. Berikut proses injeksi kode program tersebut seperti pada Gambar 4.2:



Gambar 4.2 Hasil dari percobaan injeksi kode

Untuk lebih memastikan pengujian, dilakukan uji menggunakan *tools* Acunetix. Untuk lebih mempersingkat waktu pengujian, konfigurasi pada *tools* ini dipilih hanya untuk *scanning* pada kerentanan XSS. Berikut hasil *scanning* menggunakan Acunetix pada Gambar 4.3:



Scan of https://informatics-expo.id/

Scan details

Scan information	
Start time	11/06/2022, 12:18:45
Start url	https://informatics-expo.id/
Host	https://informatics-expo.id/
Scan time	5 minutes, 27 seconds
Profile	Cross-site Scripting Vulnerabilities

Threat level

Acunetix Threat Level 0

No vulnerabilities have been discovered by the scanner.

Alerts distribution

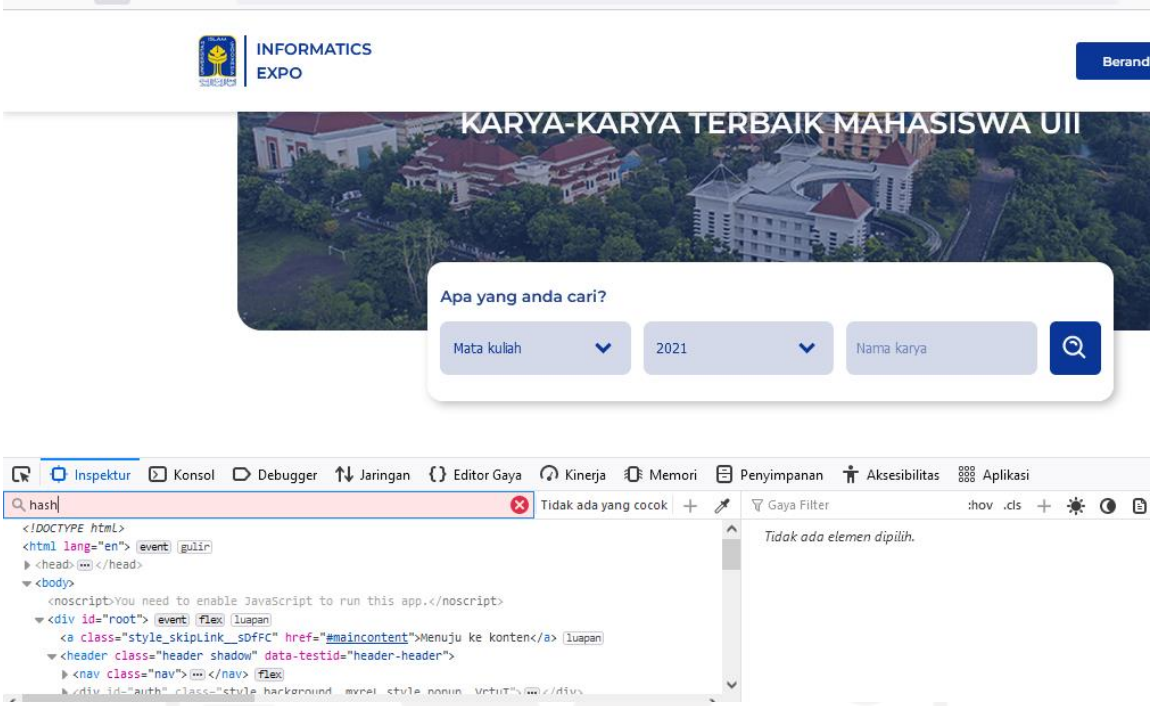
Total alerts found	0
High	0
Medium	0
Low	0
Informational	0

Gambar 4.3 Hasil dari scanning *tools* Acunetix

Pada Gambar diketahui bahwa hasil *scanning* tidak menunjukan adanya kerentanan. Dari pengujian WSTG-CLNT-01 *Testing* for DOM-*Based Cross Site Scripting* tidak ditemukan kerentanan.

4.1.2 Hasil WSTG-CLNT-02 *Testing for* JavaScript *Execution*

Tahap pertama adalah mengidentifikasi fungsi *Javascript* yang dapat menjadi celah kerentanan. Identifikasi fungsi ini bertujuan untuk penyusunan kode program serangan. Berikut hasil dari identifikasi fungsi menggunakan *tools* pada *browser* seperti pada Gambar 4.4:

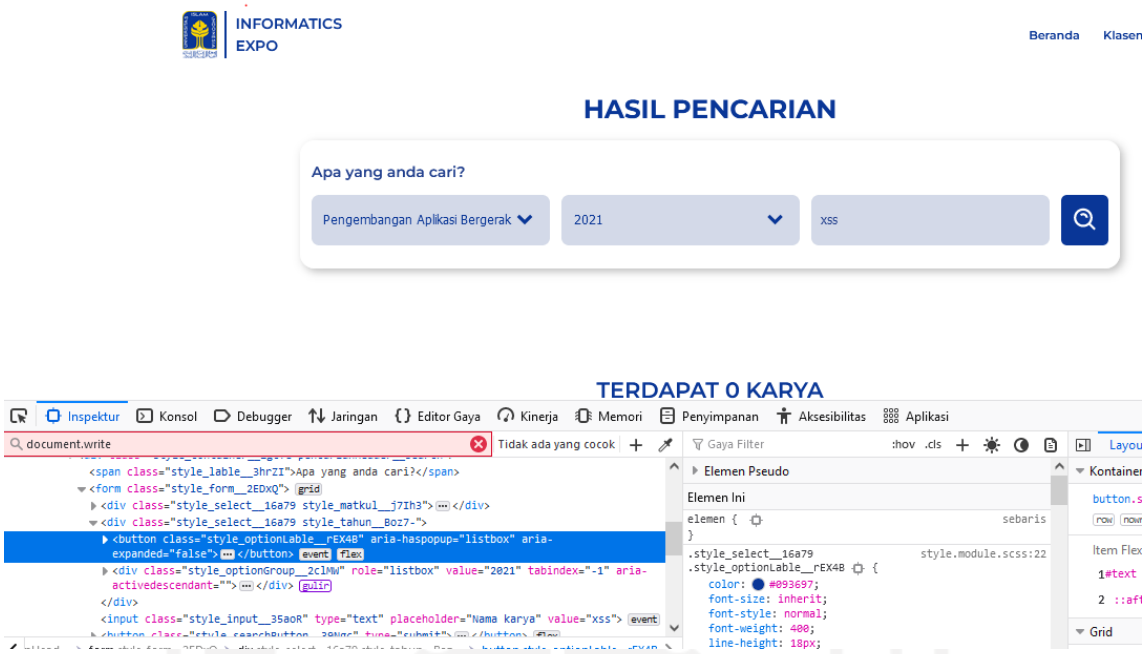


Gambar 4.4 Hasil dari pencarian fungsi Javascript

Fungsi yang dapat menjadi celah kerentanan tidak ditemukan, sehingga penyusunan kode program serangan yang kemudian akan diinjeksikan tidak dapat dilakukan. Dengan demikian, berdasarkan hasil dari pengujian WSTG-CLNT-02 *Testing for* JavaScript *Execution*, tidak ditemukan kerentanan pada sistem.

4.1.3 Hasil WSTG-CLNT-03 *Testing for* HTML *Injection*

Pengujian dilakukan dengan identifikasi fungsi yang dapat menjadi kerentanan. Berikut Gambar 4.5 menunjukan hasil identifikasi fungsi:

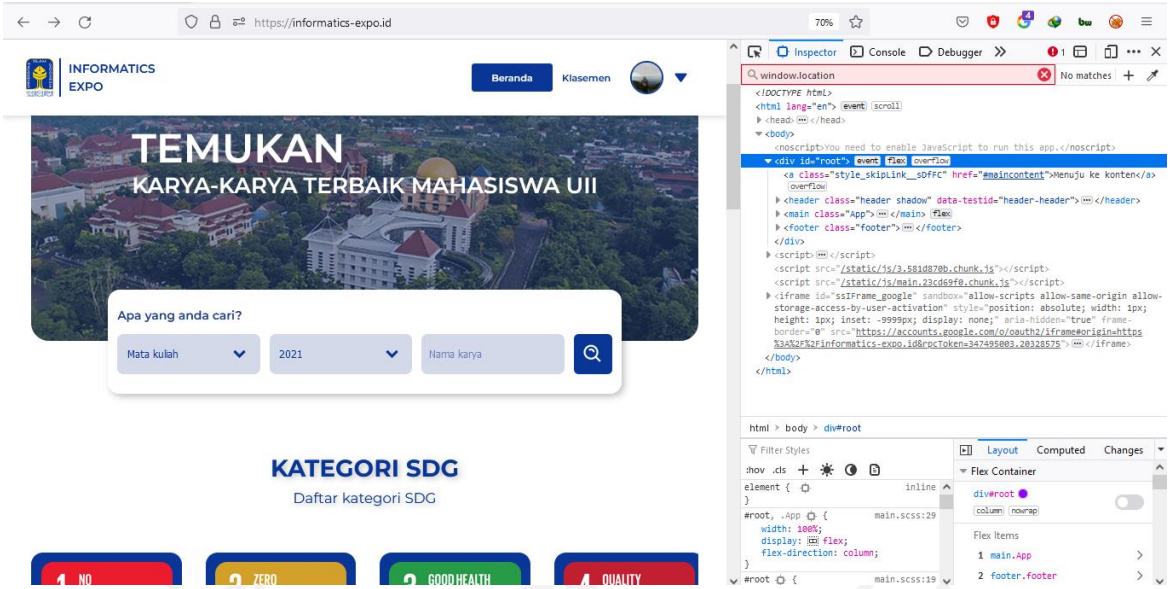


Gambar 4.5 Hasil dari pencarian fungsi HTML

Tidak terdapat fungsi yang bisa menjadi celah kerentanan pada sistem. Proses selanjutnya yaitu penyusunan kode program serangan berdasarkan fungsi yang ditemukan tidak dapat dilanjutkan. Dengan demikian, berdasarkan hasil dari pengujian WSTG-CLNT-03 *Testing for* HTML *Injection* tidak ditemukan kerentanan.

4.1.4 Hasil WSTG-CLNT-04 *Testing for Client-side* URL *Redirect*

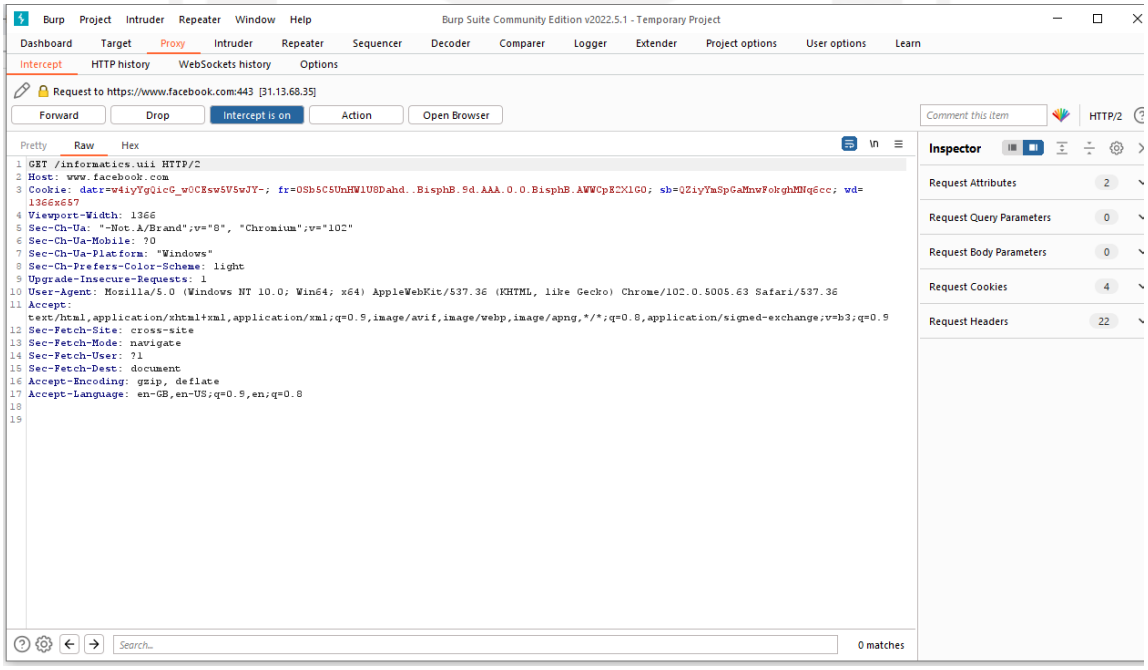
Pengujian dilakukan dengan mengidentifikasi fungsi yang bisa menjadi celah kerentanan Client-side URL *Redirect*. Fungsi yang bisa menjadi celah kerentanan merupakan fungsi untuk *redirect* ke sebuah alamat web tertentu. Berikut Gambar 4.6 menunujukan hasil identifikasi fungsi *redirect*:



Gambar 4.6 Hasil dari pencarian fungsi *redirect*

Pada Gambar 4.6 diketahui bahwa fungsi *redirect* yaitu window.location tidak ditemukan. Proses selanjutnya yaitu eksploitasi menggunakan *script* tidak dapat dilakukan.

Kemudian dilakukan pengujian dengan menggunakan *tools* Burp Suite untuk memastikan apakah *path* atau URL menuju alamat tertentu dapat terlihat oleh *tools*. Berikut Gambar 4.7 yang menunjukan hasil dari *tools* Burp Suite:



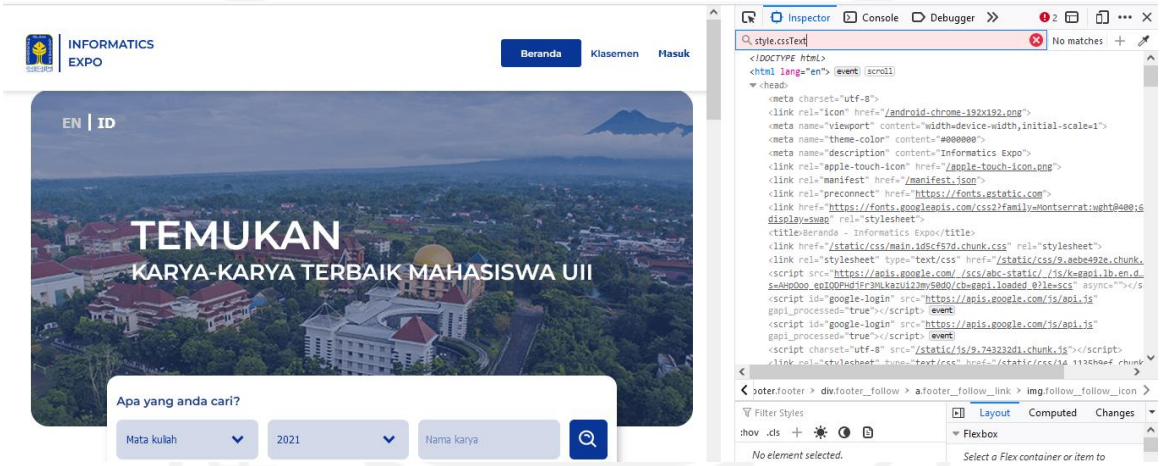
Gambar 4.7 Hasil dari pencarian *path* atau URL

Dalam Gambar 4.7 tersebut diketahui bahwa ketika sistem melakukan *redirect* menuju facebook, path atau URL tidak dapat muncul dalam *tools*. Ini meenunjukkan bahwa tidak bisa dilakukannya perubahan path atau URL ketika proses *redirect* terjadi. Dengan Demikian,

berdasarkan hasil pengujian *Testing for Client-side* URL *Redirect* tidak ditemukannya kerentanan.

4.1.5 Hasil WSTG-CLNT-05 *Testing for* CSS *Injection*

Pengujian dilakukan dengan mengidentifikasi fungsi yang bisa menjadi celah kerentanan CSS *injection*. Berikut hasil dari pencarian fungsi CSS yang dapat menjadi kerentanan seperti pada Gambar 4.8:

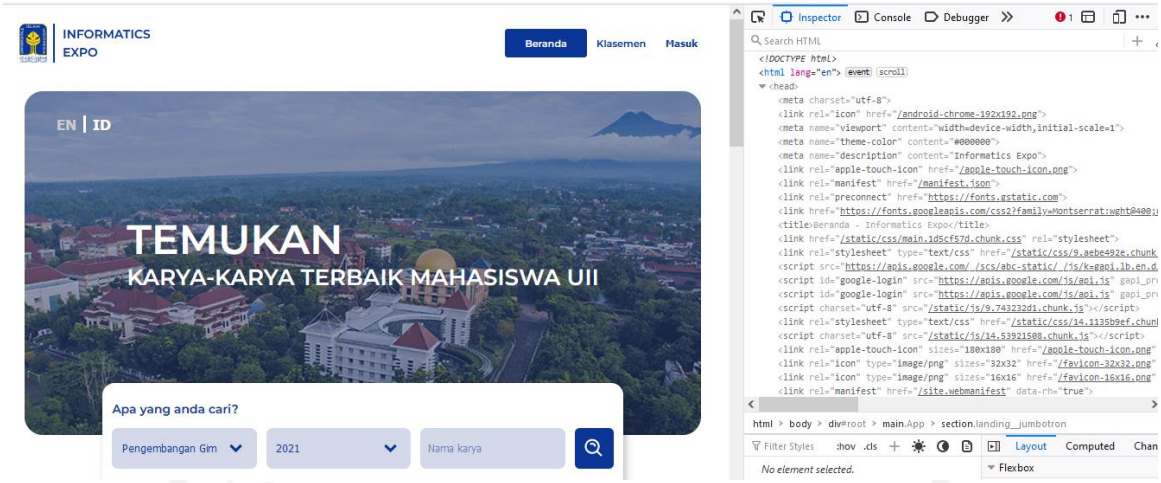


Gambar 4.8 Hasil dari pencarian fungsi CSS

Pada Gambar 4.8 fungsi CSS tidak ditemukan. Proses selanjutnya yaitu menyusun kode serangan untuk memanipulasi location.hash tidak bisa dilakukan karena tidak ditemukanya fungsi yang dapat dimodifikasi untuk memanfaatkan location.hash sebagai celah kerentanan. Dengan demikian, berdasarkan hasil dari pengujian WSTG-CLNT-05 *Testing for* CSS *Injection* tidak ditemukan kerentanan.

4.1.6 Hasil WSTG-CLNT-06 *Testing for Client-side Resource Manipulation*

Pada pengujian ini dilakukan dengan mengidentifikasi *resource* yang terhubung dengan sistem. Berikut Gambar 4.9 hasil dari identifikasi *resource* pada sistem:



Gambar 4.9 Hasil dari pencarian *resource*

Gambar 4.9 menunjukan hasil dari identifikasi yang menemukan beberapa *resource* yang terhubung dengan sistem. berikut beberapa contoh *resource* yang digolongkan berdasarkan jenisnya:

A. Script

Contoh *resource* jenis script yang terhubung dengan sistem seperti pada Gambar 4.10:

```
<link href="/static/css/main.1d5cf57d.chunk.css" rel="stylesheet">
<script id="google-login" src="https://apis.google.com/js/api.js"
gapi_processed="true"></script>
<script charset="utf-8"
src="/static/js/9.743232d1.chunk.js"></script>
```

Gambar 4.10 *Resource* jenis *script*

B. CSS

Contoh *resource* jenis CSS yang terhubung dengan sistem seperti pada Gambar 4.11:

```
<link href="/static/css/main.1d5cf57d.chunk.css" rel="stylesheet">
<link
href="https://fonts.googleapis.com/css2?family=Montserrat:wght@400;6
00;700&amp;display=swap" rel="stylesheet">
```

Gambar 4.11 *Resource* jenis CSS

C. Gambar

Contoh *resource* jenis Gambar yang terhubung dengan sistem seperti pada Gambar 4.12:

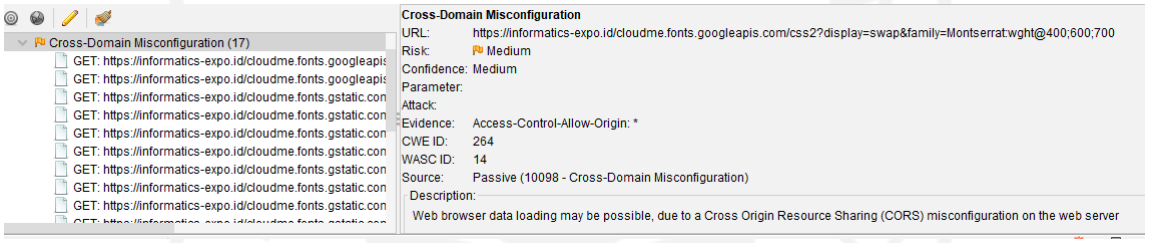
```
<img src="/static/media/UII-Biru.83dd5fbe.png" alt="Logo UII">
```

Gambar 4.12 *Resource* jenis Gambar

Resource yang ditemukan tergolong aman karena *resource* yang ditemukan merupakan *resource* yang terhubung dengan sistem tanpa melalui fungsi yang dapat dimanupulasi untuk perubahan *resource*-nya. Dengan demikian, berdasarkan hasil pengujian WSTG-CLNT-06 *Testing for Client-side Resource Manipulation* tidak ditemukan kerentanan.

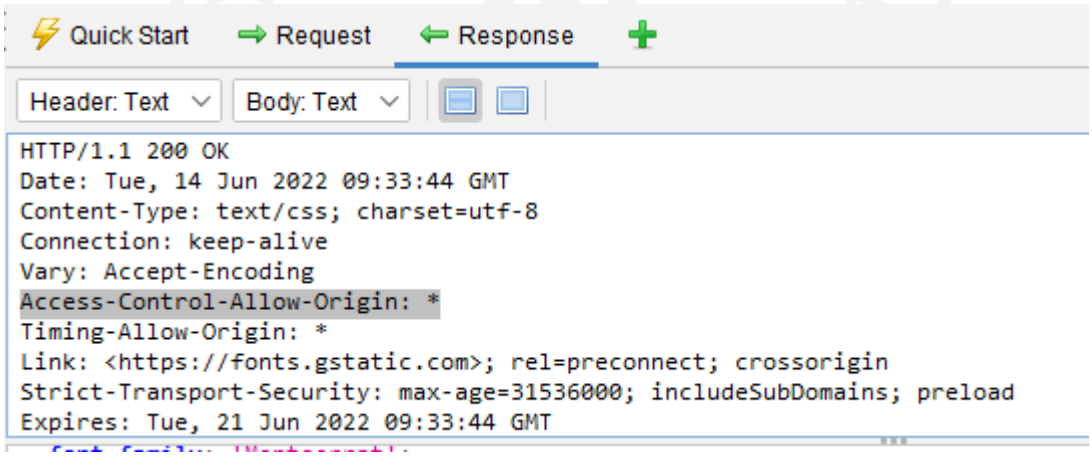
4.1.7 Hasil WSTG-CLNT-07 *Testing Cross Origin Resource Sharing*

Pengujian dilakukan dengan *scanning* kerentanan *Cross Origin Resource Sharing* menggunakan *tools* ZAP. *tools* ini dapat mencakup pengujian terkait Cross *Origin Resource Sharing*. Dari hasil *scanning* menggunakan *tools* ZAP ditemukan kerentanan pada sistem ini. Berikut hasil dari *scanning* menggunakan *tools* ZAP pada Gambar 4.13 :



Gambar 4.13 Hasil *scanning tools* ZAP

Pada Gambar 4.13 menunjukkan bahwa terdapat tujuh belas halaman yang mengandung kerentanan pada sistem. Kerentanan tersebut karena pada *header response Access-Control-Origin* diset * atau *wildcard* yang menjadikan semua domain dapat mengakses halaman tersebut. Gambar 4.14 berikut merupakan salah satu *header response* yang berhasil di-*scan* oleh *tools* ZAP:

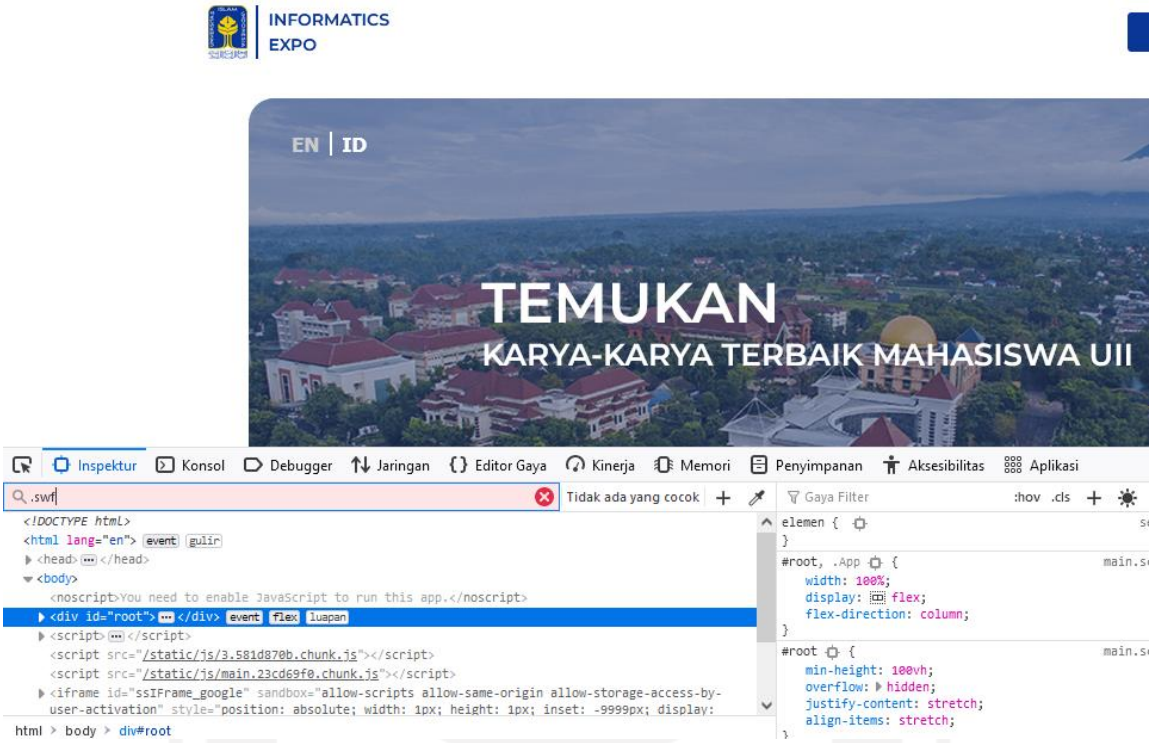


Gambar 4.14 *Header* dari salah satu *response*

Dengan Demikian, berdasarkan hasil dari pengujian *Testing Cross Origin Resource Sharing* ditemukan adanya kerentanan.

4.1.8 Hasil WSTG-CLNT-08 *Testing for Cross Site Flashing*

Pengujian dilakukan dengan mencari file SWF pada sistem menggunakan *tools* pada *browser*. Berikut adalah Gambar 4.15 hasil dari pencarian file SWF:



Gambar 4.15 Hasil dari pencarian file SWF

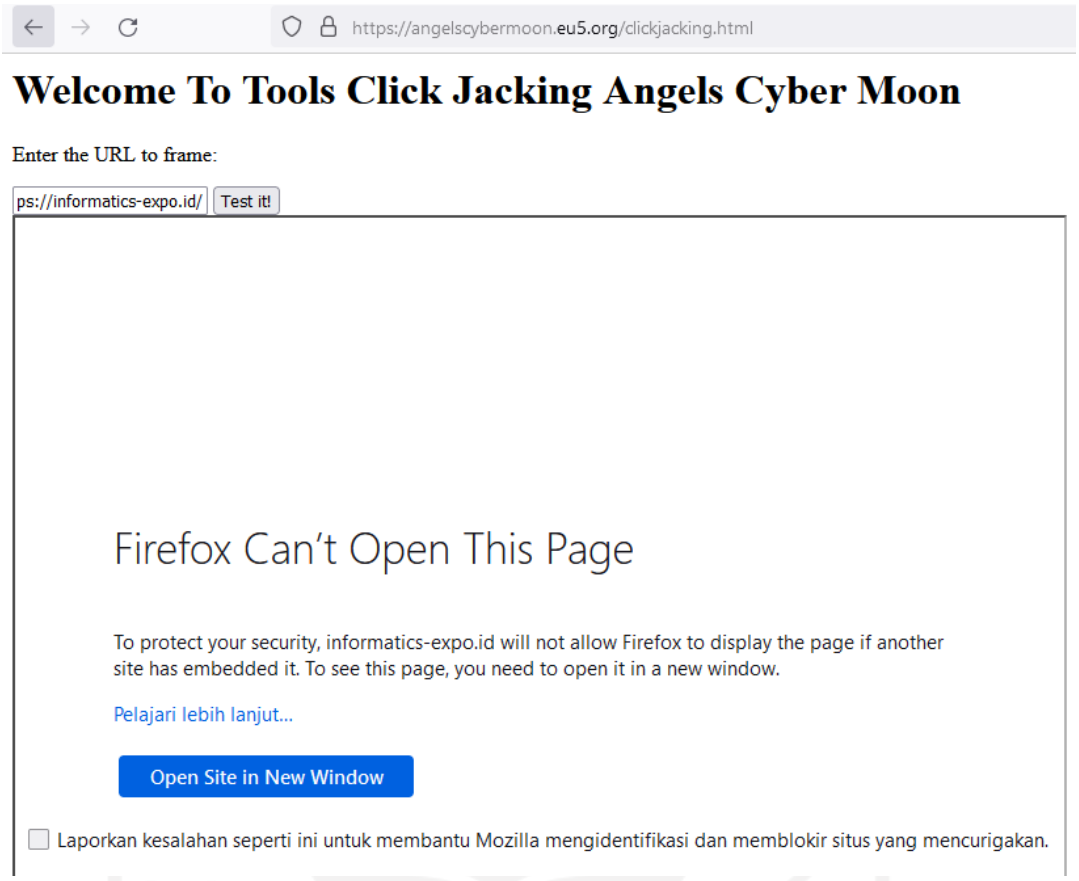
Dari Gambar 4.15 diketahui bahwa sistem tidak memiliki file SWF. Dengan tidak ditemukannya file SWF pada sistem, proses pengujian dengan *tools* Vais tidak dapat dilakukan. Berdasarkan hasil dari pengujian Hasil WSTG-CLNT-08 *Testing for Cross Site Flashing* tidak ditemukan kerentanan.

4.1.9 Hasil WSTG-CLNT-09 *Testing for Clickjacking*

Pengujian dilakukan menggunakan dua *tools* berbeda untuk lebih memantapkan hasil yang diperoleh. Berikut dua *tools* yang digunakan:

1. Cyber Moon *Clickjacking*

   Merupakan *tools* berbasis web yang menyediakan fitur pemuatan pada halaman web lainnya. Sesuai dengan pedoman WSTG-CLNT-09 *Testing for Clickjacking*, kerentanan terjadi ketika sistem termuat di halaman web lain. Berikut hasil dari pengujian menggunakan *tools* Cyber Moon seperti pada Gambar 4.16:

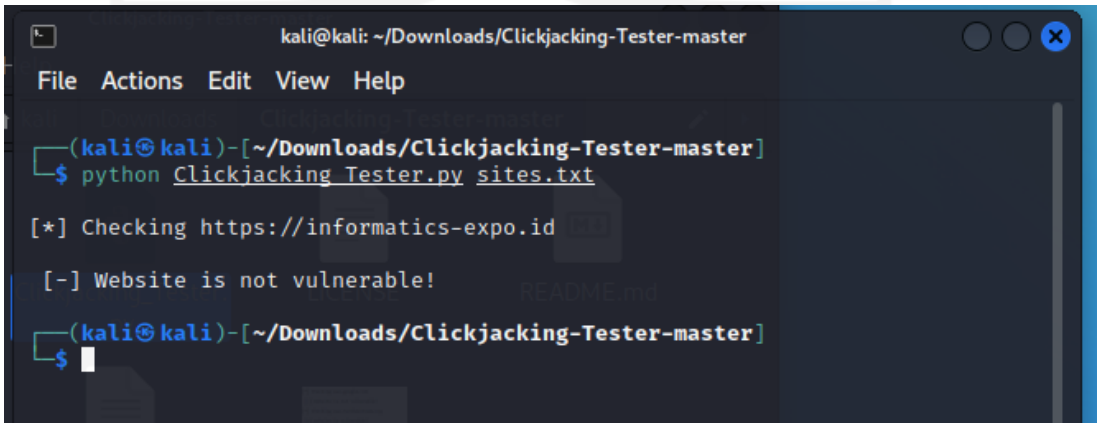


Gambar 4.16 Hasil pengujian dengan *tools* CyberMoon

Berdasarkan Gambar 4.16, sistem tidak dapat termuat dalam *tools* CyberMoon.

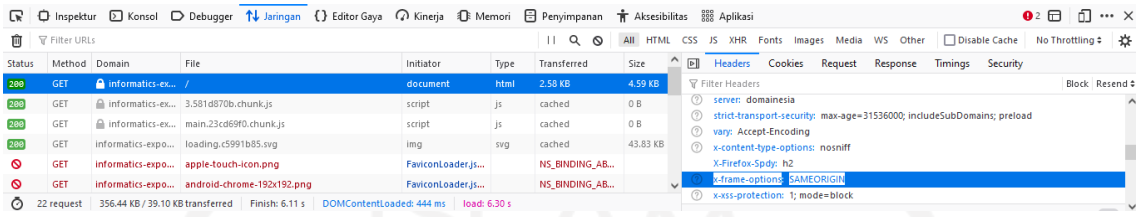
2. *Clickjacking*-Tester

Sebuah *tools* untuk pengujian *Clickjacking* yang berjalan pada sistem operasi Kali Linux. *Tools* ini akan mencari otomatis kerentanan pada *Clickjacking*. Berikut hasil dari *tools Clickjacking*-Tester seperti pada Gambar 4.17:



Gambar 4.17 Hasil pengujian dengan *tools* Clickjacking Tester

Dengan Demikian, berdasarkan Gambar 4.17 *tools* Clickjacking-Tester tidak mendeteksi adanya kerentanan pada sistem. Kemudian dilakukan pengecekan pada *header* untuk melihat konfigurasi yang diterapkan terkait kerentanan *Clickjacking*, berikut Gambar 4.18 menunjukkan konfigurasi *header* menggunakan *tools developer* pada *browser* :

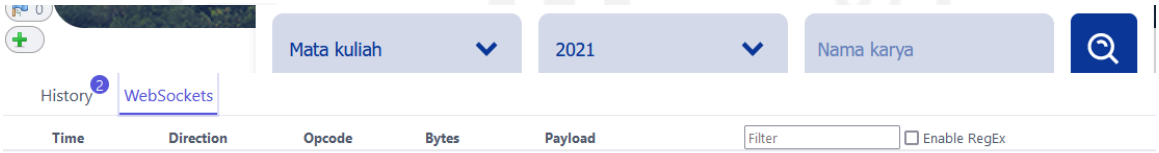


Gambar 4.18 Konfigurasi *Header*

Dari Gambar 4.18 diketahui bahwa konfigurasi *header* memakai *X-Frame-Option SAMEORIGIN* yang berarti sistem tidak bisa termuat dalam halaman web lain kecuali dengan domain yang sama. Berdasarkan hasil pengujian WSTG-CLNT-09 *Testing for Clickjacking* dapat disimpulkan bahwa sistem tidak ada kerentanan.

4.1.10 Hasil WSTG-CLNT-10 *Testing WebSockets*

Pengujian dilakukan dengan mengidentifikasi penerapan *Websockets* pada sistem. Proses ini menggunakan *tools* ZAP dengan metode manual *explore*. ZAP akan menangkap lalu lintas *Websockets* jika ditemukan. Berikut proses pencarian lalu lintas *Websockets* menggunakan *tools* ZAP metode manual *explore* seperti pada Gambar 4.19:

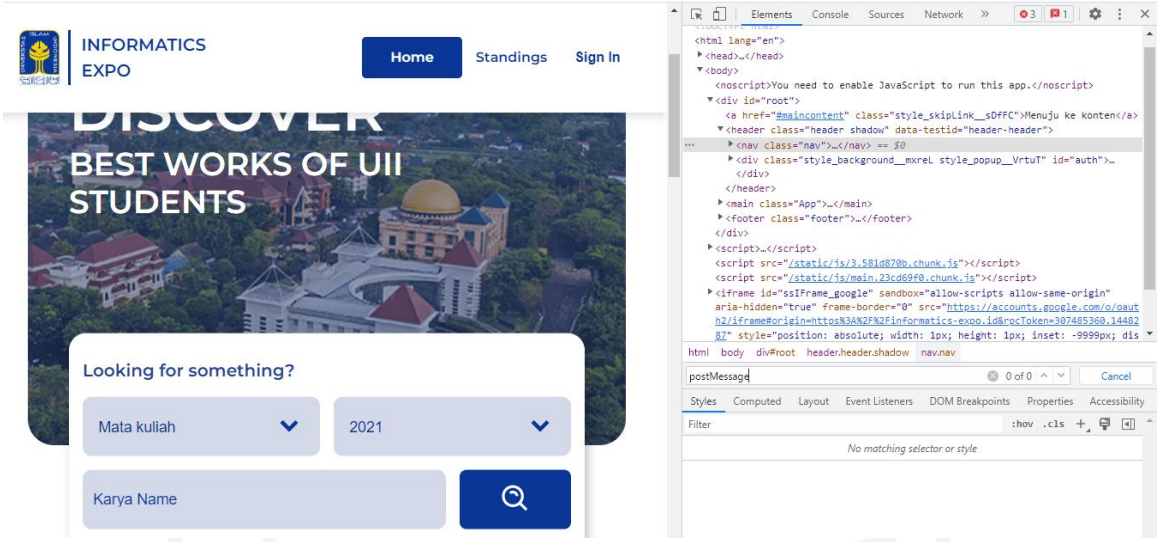


Gambar 4.19 Hasil dari perekaman lalu lintas Websockets

Proses pencarian lalu lintas *Webscokets* dilakukan dengan membuka seluruh halaman sistem beserta fitur-fitur didalamnya. Pada Gambar 4.19 menunjukkan bahwa tidak menunjukkan adanya lalu lintas *Websockets*. Dengan demikian proses pengujian tidak dapat dilanjutkan. Dengan demikian, berdasarkan hasil pengujian dari WSTG-CLNT-10 *Testing WebSockets* sistem tidak menggunakan fitur tersebut.

4.1.11 Hasil WSTG-CLNT-11 *Testing Web Messaging*

Pengujian dilakukan dengan mengidentifikasi fungsi `postMessage()` pada sistem dengan *tools* pada *browser*. Berikut Gambar 4.20 yang menunjukkan hasil dari proses pencarian fungsi:



Gambar 4.20 Hasil pencarian fungsi *web messaging*

Fungsi `postMessage()` tidak ditemukan, dengan demikian proses selanjutnya yaitu analisis fungsi tidak bisa dilakukan.

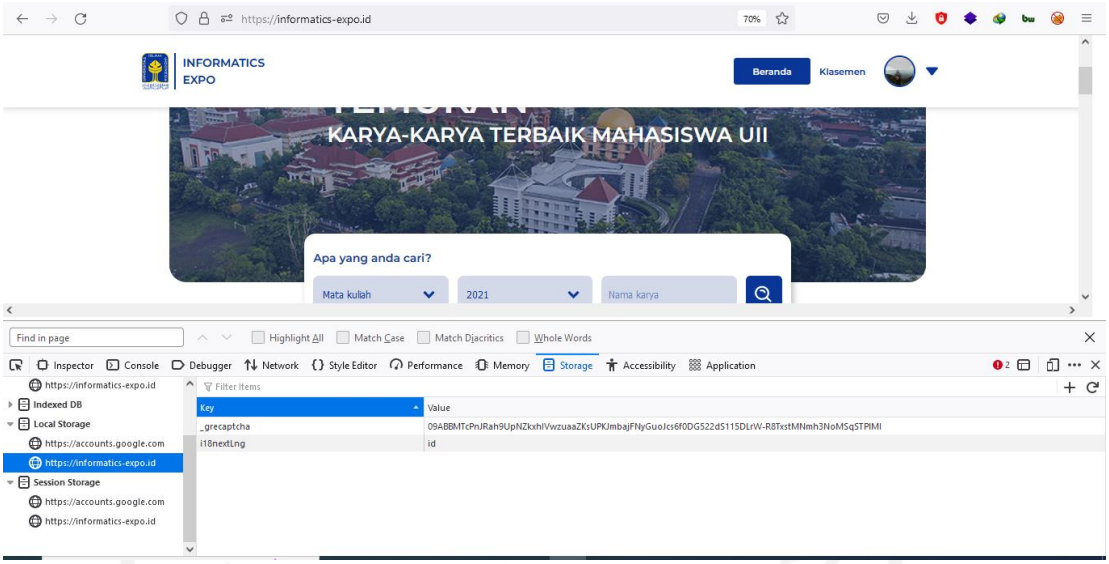
Berdasarkan hasil pengujian WSTG-CLNT-11 *Testing Web Messaging* tidak ditemukan kerentanan.

4.1.12 Hasil WSTG-CLNT-12 *Testing Browser storage*

Pengujian dilakukan dengan melakukan penelusuran pada *tools browser* bagian *storage*. Pengujian ini melakukan analisis terhadap data yang berhasil disimpan pada *browser storage*. Analisis dilakukan untuk memastikan bahwa data yang tersimpan telah aman dan tidak bisa digunakan untuk hal yang berbahaya. Berikut analisis data berdasarkan jenis *browser storage*:

1. *Local Storage*

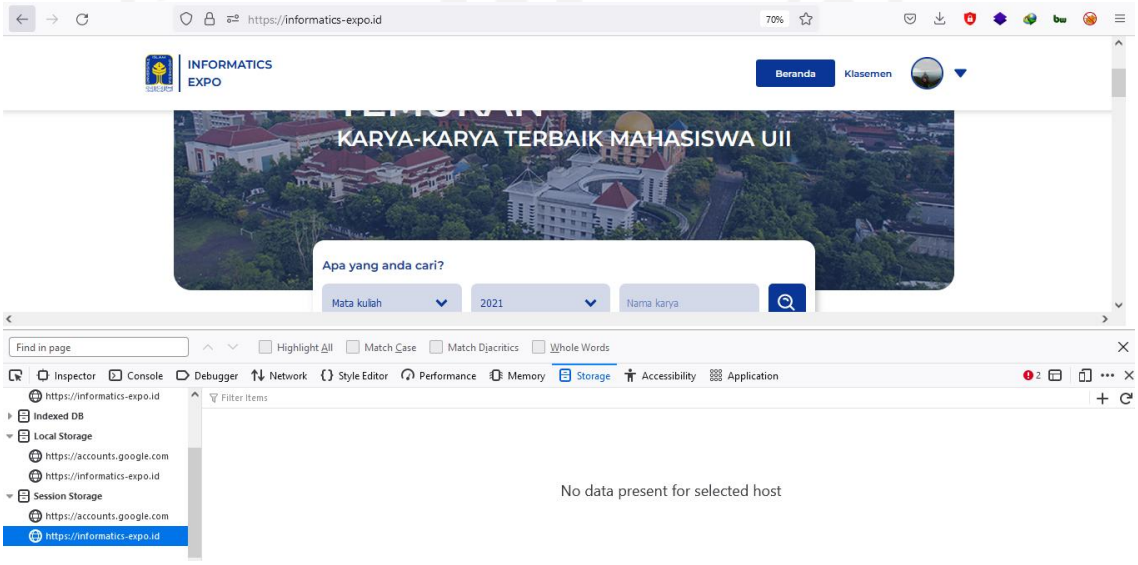
   Pada jenis ini ditemukan data yang tersimpan pada *browser storage*. Data yang tersimpan memiliki parameter yang telah terenkripsi. Berikut adalah Gambar 4.21 hasil penelusuran pada jenis *Local Storage*:



Gambar 4.21 *Browser storage* jenis Local *Storage*

2. *Session Storage*

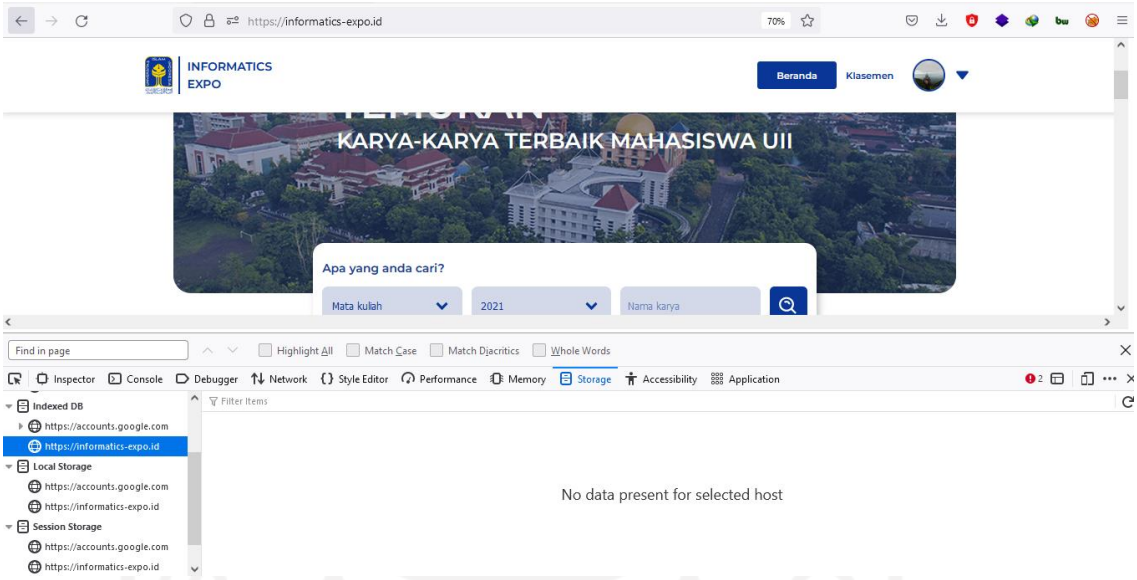
Pada jenis ini tidak ditemukan data yang tersimpan pada *browser storage*. *Login* akun dilakukan dengan harapan akan ada *session* yang masuk ke dalam *browser*, akan tetapi *session* tidak tersimpan. Berikut adalah Gambar 4.22 hasil penelusuran pada jenis *Session Storage*:



Gambar 4.22 *Browser storage* jenis *Session Storage*

3. *IndexedDB*

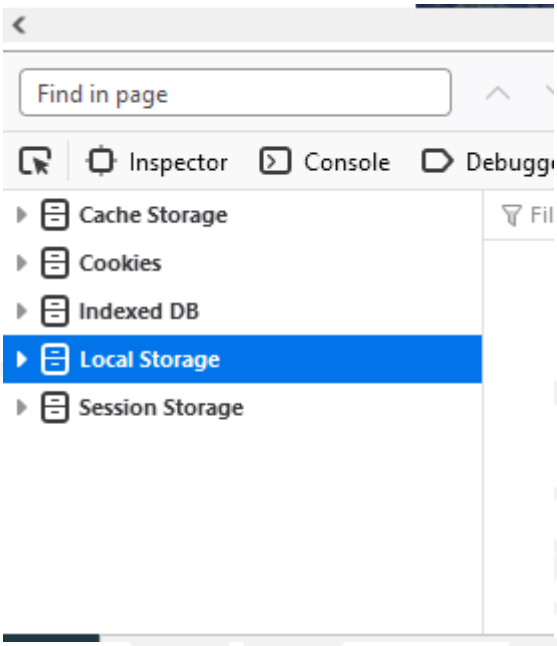
Pada jenis ini tidak ditemukan data yang tersimpan pada *browser storage*. Berikut adalah Gambar 4.23 hasil penelusuran pada jenis *IndexedDB*:



Gambar 4.23 *Browser storage* jenis IndexedDB

4. *Web SQL*

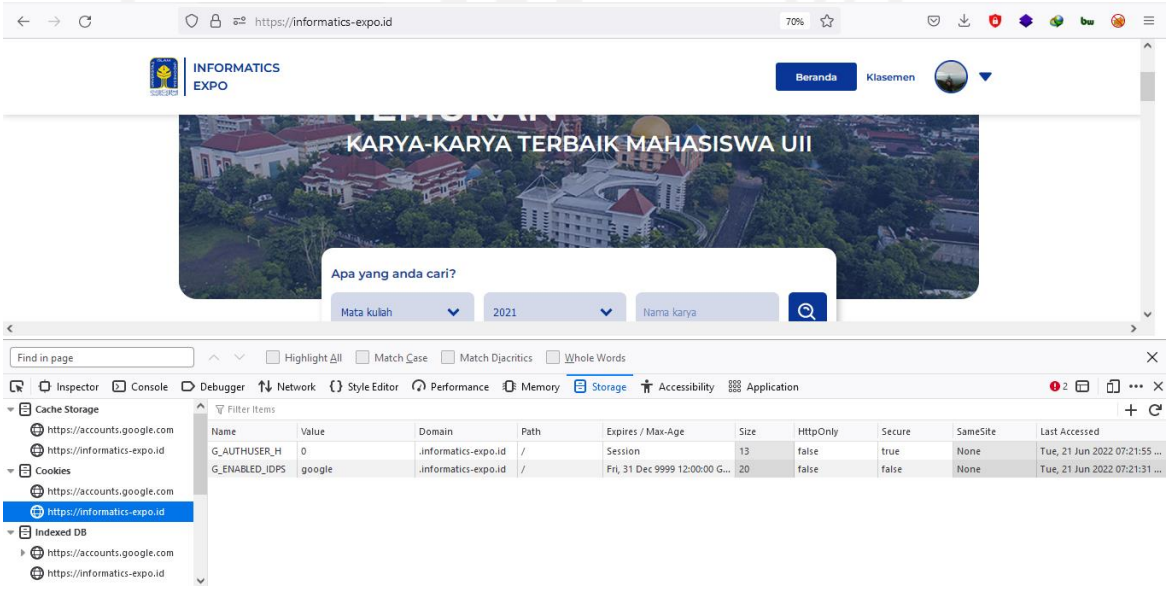
Web SQL tidak ditemukan pada *browser storage*, menandakan bahwa *storage* jenis ini sudah tidak digunakan lagi. Berikut adalah Gambar 4.24 hasil penelusuran pada jenis *Web SQL*:



Gambar 4.24 *Browser storage* jenis *WebSQL*

5. *Cookies*

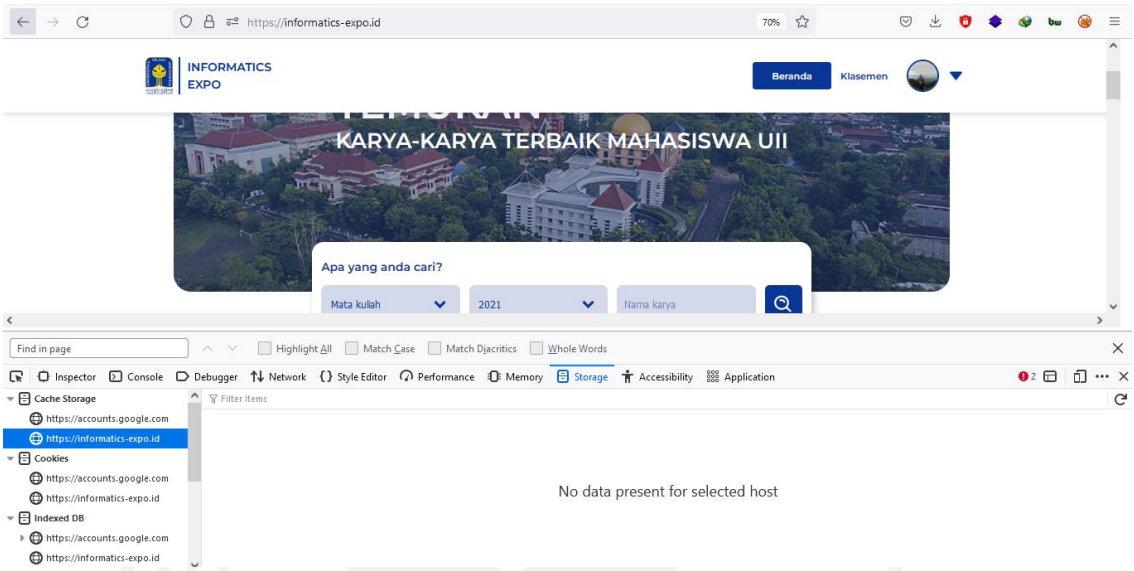
Pada jenis ini ditemukan data yang tersimpan pada *browser storage*. Tidak ditemukan data yang terbuka secara langsung. Data pada jenis ini merupakan bagian penyimpanan sesi pengguna dalam *browser*. Berikut adalah Gambar 4.25 hasil penelusuran pada jenis *Cookies*:



Gambar 4.25 *Browser storage* jenis Cookies

6. *Cache Storage*

Pada jenis ini tidak ditemukan data yang tersimpan pada *browser storage*. Berikut adalah Gambar 4.26 hasil penelusuran pada jenis *Cache Storage*:

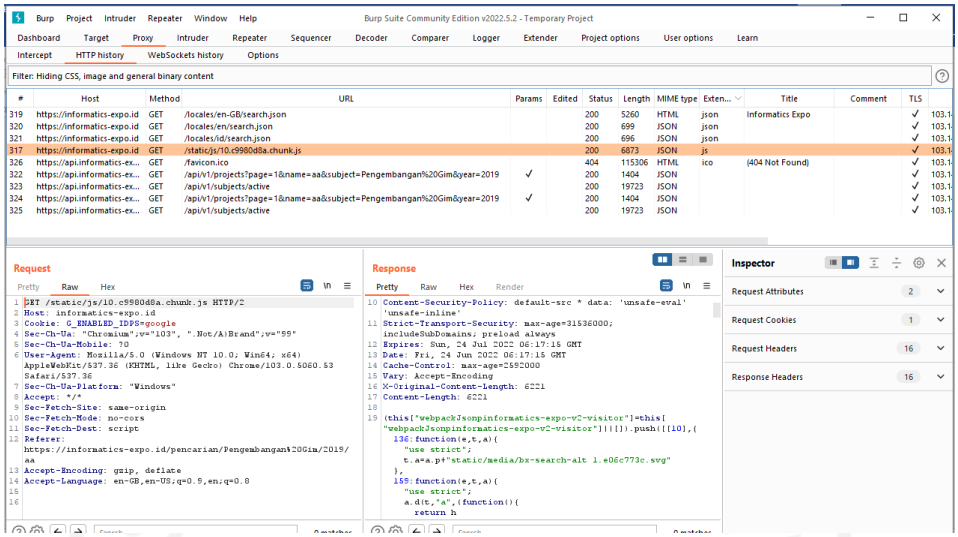


Gambar 4.26 *Browser storage* jenis *Cache Storage*

Dengan demikian, berdasarkan hasil pengujian WSTG-CLNT-12 *Testing Browser storage* tidak ditemukan kerentanan.

4.1.13 Hasil WSTG-CLNT-13 *Testing for Cross Site Script Inclusion*

Pengujian ini dilakukan menggunakan *tools* Burp Suite untuk mengidentifikasi sistem menggunakan JSONP atau tidak dalam melakukan pengiriman data. Pengujian ini dilakukan dengan merekam aliran data pada sistem menggunakan *tools* Burp Suite. Berikut hasil dari sebuah *request* yang ditangkap oleh *tools* Burp Suite seperti pada Gambar 4.27:



Gambar 4.27 Hasil dari perekaman data menggunakan *tools* Burp Suite

Dari hasil perekaman data pada Gambar 4.27 ditemukan beberapa pengiriman data yang menggunakan teknik JSONP. Namun dalam JSONP tersebut tidak mengandung fungsi yang bisa menjadi celah kerentanan. Dengan demikian, berdasarkan hasil dari pengujian WSTG-CLNT-13 *Testing for Cross Site Script Inclusion* tidak ditemukan kerentanan.

4.2 Analisis Hasil

Dari pengujian yang telah dilakukan dengan beberapa metode seperti identifikasi fungsi, identifikasi fitur, percobaan injeksi kode, dan pengujian menggunakan *tools*, dibuatlah matriks ke dalam Tabel 4.1 berikut:

Tabel 4.1 Metode pengujian dan hasil

Poin WSTG	Identifikasi fungsi	Identifikasi fitur	Injeksi kode	Pengujian dengan *tools*	Kerentanan
WSTG-CLNT-01	Tidak ditemukan		Tidak berhasil	Tidak ditemukan kerentanan	Tidak ditemukan
WSTG-CLNT-02	Tidak ditemukan		Tidak berhasil		Tidak ditemukan
WSTG-CLNT-03	Tidak ditemukan		Tidak berhasil		Tidak ditemukan
WSTG-CLNT-04	Tidak ditemukan				Tidak ditemukan

WSTG-CLNT-05	Tidak ditemukan		Tidak berhasil		Tidak ditemukan
WSTG-CLNT-06	Tidak ditemukan				Tidak ditemukan
WSTG-CLNT-07				Ditemukan kerentanan	Ditemukan
WSTG-CLNT-08		Tidak ditemukan			Tidak ditemukan
WSTG-CLNT-09				Tidak ditemukan kerentanan	Tidak ditemukan
WSTG-CLNT-10		Tidak ditemukan			Tidak ditemukan
WSTG-CLNT-11	Tidak ditemukan				Tidak ditemukan
WSTG-CLNT-12		Ditemukan dan aman			Tidak ditemukan
WSTG-CLNT-13					Tidak ditemukan

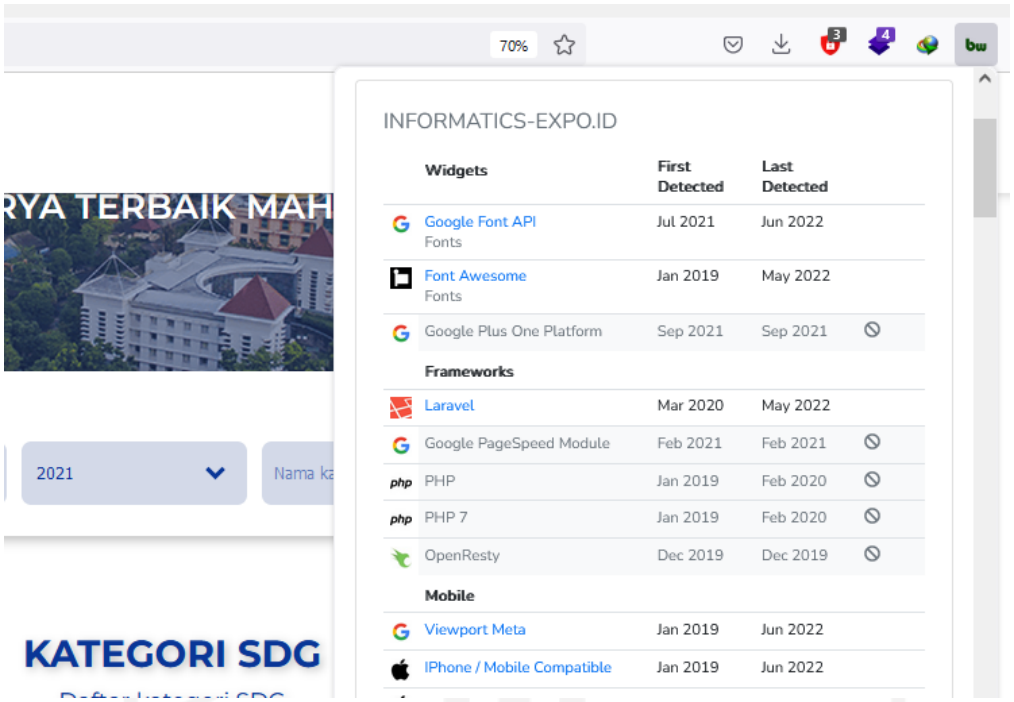
Berikut penjelasan dari tabel 4.1

A. Tidak ditemukan kerentanan

Dapat diketahui bahwa tujuh poin dalam pengujian ditemukan keamanan yang sama yaitu tidak ditemukanya fungsi yang dapat dimanipulasi sehingga penyusunan kode program serangan ke dalam sistem tidak dapat dilakukan. Berikut tujuh poin pengujian yang memiliki keamanan yang sama:

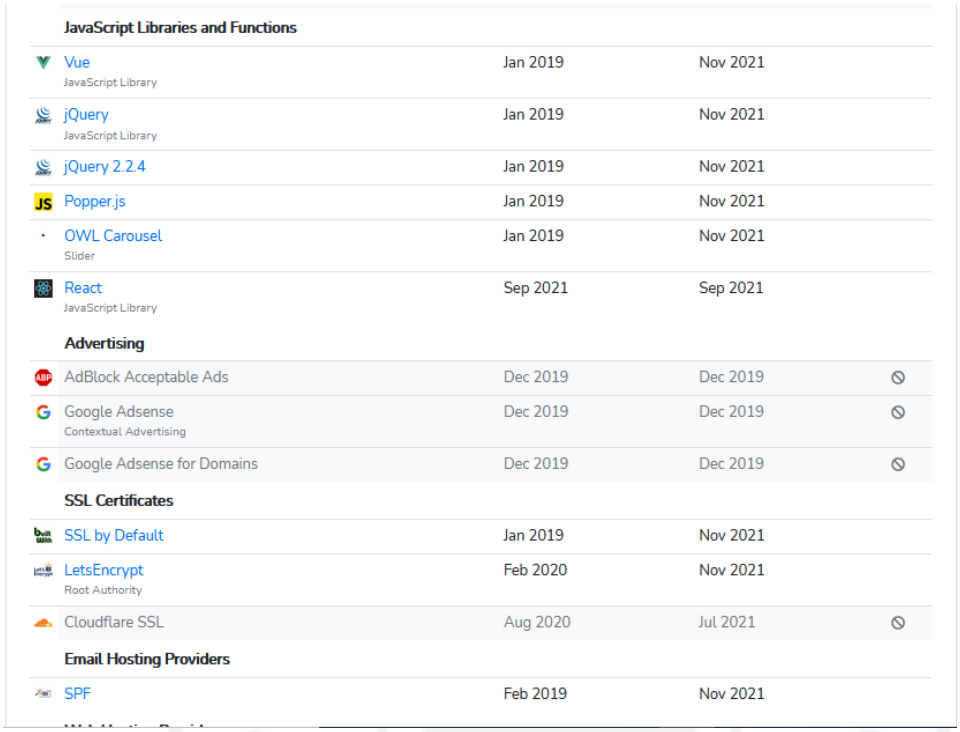
1. WSTG-CLNT-01 *Testing for DOM-Based Cross Site Scripting*
2. WSTG-CLNT-02 *Testing for Javascript Execution*
3. WSTG-CLNT-03 *Testing for HTML Injection*
4. WSTG-CLNT-04 *Testing for Client-side URL Redirect*
5. WSTG-CLNT-05 *Testing for CSS Injection*
6. WSTG-CLNT-06 *Testing for Client-side Resource Manipulation*
7. WSTG-CLNT-11 *Testing for Web Messaging*

Dari tidak bisa teridentifikasinya fungsi yang dipakai oleh sistem ini karena sistem telah menerapkan beberapa teknologi terkait pengembangan web. Pencarian informasi tentang teknologi yang diterapkan oleh sistem menggunakan *tools* BulitWith. Berikut Gambar 4.28 menunjukan bahwa sistem telah menerapakan *framework* php sebagai dasar pengembangan sisi *server*:



Gambar 4.28 Pencarian informasi dengan *tools* BulitWith

Dari Gambar 4.28 tersebut diketahui bahwa sistem menerapkan *framework* Laravel sebagai kerangka kerja untuk sisi bagian *server*. Dengan penggunaan *framework* akan meningkatkan keamanan dalam pengembangan web. Diketahui juga bahwa sistem menggunakan *library* untuk pengembangan web pada sisi *client*. Berikut Gambar 4.29 yang menunjukkan penggunaan *library* untuk pengembangan web pada sisi *client*:



JavaScript Libraries and Functions		
Vue JavaScript Library	Jan 2019	Nov 2021
jQuery JavaScript Library	Jan 2019	Nov 2021
jQuery 2.2.4	Jan 2019	Nov 2021
Popper.js	Jan 2019	Nov 2021
OWL Carousel Slider	Jan 2019	Nov 2021
React JavaScript Library	Sep 2021	Sep 2021
Advertising		
AdBlock Acceptable Ads	Dec 2019	Dec 2019
Google Adsense Contextual Advertising	Dec 2019	Dec 2019
Google Adsense for Domains	Dec 2019	Dec 2019
SSL Certificates		
SSL by Default	Jan 2019	Nov 2021
LetsEncrypt Root Authority	Feb 2020	Nov 2021
Cloudflare SSL	Aug 2020	Jul 2021
Email Hosting Providers		
SPF	Feb 2019	Nov 2021

Gambar 4.29 Pencarian informasi dengan *tools* BulitWith

Dari Gambar 4.29 ditemukan bahwa sistem menggunakan beberapa *library* yang berhubungan dengan sisi *client* seperti React dan Vue. Penggunaan *library* tentunya akan membuat sistem menjadi lebih aman pada sisi pengguna. Berdasarkan bukti yang ditemukan, keamanan pada tujuh poin pengujian yang sama karena sistem telah menerapkan beberapa teknologi yang membantu dalam keamanan sistem. Beberapa poin lain yang tidak ditemukan kerentanan dengan pengujian menggunakan *tools* dan identifikasi fitur adalah:

1. WSTG-CLNT-08 *Testing for Cross Site Flashing*
2. WSTG-CLNT-09 *Testing for Clickjacking*
3. WSTG-CLNT-10 *Testing for Websockets*
4. WSTG-CLNT-12 *Testing for Browser storage*
5. WSTG-CLNT-13 *Testing for Cross Site Script Inclusion*

B. Ditemukan kerentanan

Kemudian untuk kerentanan yang ditemukan adalah pada poin WSTG-CLNT-07 *Testing Cross Origin Resource Sharing.* Berikut Tabel 4.2 detail kerentanan pada poin ini:

Tabel 4.2 Detail kerentanan WSTG-CLNT-07 *Testing Cross Origin Resource Sharing*

Kerentanan	*Path*
WSTG-CLNT-07 *Testing Cross Origin Resource Sharing*	https://informatics-expo.id/cloudme.fonts.googleapis.com/css2?display=swap&family=Montserrat:wght@400;600;700
	https://informatics-expo.id/cloudme.fonts.googleapis.com/css2?family=Montserrat:wght@400;600;700&display=swap
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCtr6Hw0aXpsog.woff2
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCtr6Hw2aXpsog.woff2
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCtr6Hw3aXpsog.woff2
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCtr6Hw5aXo.woff2
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCtr6Hw9aXpsog.woff2
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCu173w0aXpsog.woff2
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCu173w2aXpsog.woff2

	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCu173w3aXpsog.woff2
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCu173w5aXo.woff2
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCu173w9aXpsog.woff2
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCuM73w0aXpsog.woff2
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCuM73w2aXpsog.woff2
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCuM73w3aXpsog.woff2
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCuM73w5aXo.woff2
	https://informatics-expo.id/cloudme.fonts.gstatic.com/s/montserrat/v24/JTUHjIg1_i6t8kCHKm4532VJOt5-QNFgpCuM73w9aXpsog.woff2

Kerentanan pada poin ini ditemukan ketika sistem mencoba untuk mengakses data dari domain lain. Seperti pada tabel 4.2 *path* yang dideteksi rentan oleh *tools* ZAP adalah ketika sistem meminta data *font* pada domain cloudme.fonts.gstatic.com. Gambar 4.30 menunjukkan bukti yang dilaporkan oleh *tools* ZAP:

Gambar 4.30 Bukti oleh *tools* ZAP

Bukti pada Gambar 4.30 menunjukan bahwa konfigurasi untuk Acces-Control-Allow-Origin diset "*" atau *wildcard*. Ini menjadikan *path* tersebut dapat diakses oleh domain manapun. Meskipun data yang terkandung dalam *path* tersebut bukanlah data sensitif konfigurasi *Acces-Control-Allow-Origin* tetap perlu diperhatikan karena menurut standar WSTG-CLNT-07 penggunaan konfigurasi "*" atau *Wildcard* akan sangat berisiko.

C. Saran perbaikan

Saran perbaikan terkait kerentanan yang ditemukan pada poin WSTG-CLNT-07 *Testing Cross Origin Resource* adalah dengan mengonfigurasi ulang *Acces-Control-Allow-Origin*, konfigurasi sebaiknya lebih dispesifikkan lagi untuk domain yang dapat mengakses data tersebut.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Pengujian keamanan menggunakan metode WSTG v4.2 bertujuan untuk menguji tingkat keamanan sistem Informatic-Expo pada sisi *client-side.* Berdasarkan dari seluruh kegiatan yang dilakukan, dapat diperoleh beberapa kesimpulan sebagai berikut:

1. Metode pengujian dengan WSTG v4.2 cocok dalam pengujian sistem terutama dalam sisi pengguna karena poin yang diuji mencakup semua fitur yang ada dalam sebuah sistem.
2. Sistem Informatics-Expo tergolong cukup aman karena celah yang bisa menjadi kerentanan tidak terekspos dengan mudah.
3. Berdasarkan hasil analisis, kerentanan yang ditemukan hanya dari konfigurasi yang kurang tepat. Dan tidak ditemukan kerentanan lain karena sistem telah berhasil menerapkan beberapa fitur keamanan.

5.2 Saran

Berdasarkan penelitian yang sudah dilakukan terdapat beberapa saran yang dapat dilakukan dalam pengembangan sistem Informatics-Expo dan sebagai studi kasus yang bisa diterapkan dalam melakukan penelitian terkait, sebagai berikut:

1. Perlunya dilakukan pengujian terhadap semua *role* pengguna untuk lebih memaksimalkan pengujian fitur sistem.
2. Perlunya sebuah sistem s*tag*ing yang dapat menjadi objek pengujian agar sistem berjalan tidak terganggu.
3. Perlunya dilakukan pengujian dengan beberapa metode yang berbeda.

DAFTAR PUSTAKA

Alfred Tenggono, Tegar Purnama, & Andi Setia Budi. (2018). Audit keamanan webserver pada website "www.palcomtech.com." *Knsi 2018*.

Ariani Sukamto, R., & Shalahuddin, M. (2016). Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek. Bandung : Informatika. In *Jurnal Pilar Nusa Mandiri*.

Cunong, D. N., Saputra, M., & Puspitasari, W. (2020). Analisis Resiko Keamanan Terhadap Website Dinas Penanaman Modal Dan Pelayanan Terpadu Satu Pintu Pemerintahan Xyz Menggunakan Standar Penetration Testing Execution Standard (Ptes). *Journal of Chemical Information and Modeling*, *53*(9).

de Lange, J., von Solms, R., & Gerber, M. (2016). Information security management in local government. *2016 IST-Africa Conference, IST-Africa 2016*. https://doi.org/10.1109/ISTAFRICA.2016.7530584

Juardi, D. (2017). Kajian Vulnerability Keamanan Jaringan Internet Menggunakan Nessus. *Syntax Jurnal Informatika*, *6*(1).

O'Brien, J. A., & Marakas, G. M. (2010). Intoduction To Information System. *McGraw Hill*, *15*.

Rahardjo, B. (2005). Keamanan Sistem Informasi Berbasis Internet. In *PT Insan Infonesia* (Vol. 0).

W, Yunanri., Riadi, I., & Yudhana, A. (2018). Analisis Deteksi Vulnerability Pada Web Server Open Journal System Menggunakan OWASP Scanner. *Jurnal Rekayasa Teknologi Informasi (JURTI)*, *2*(1). https://doi.org/10.30872/jurti.v2i1.1319

Zen, B. P., Gultom, R. A. G., & Reksoprodjo, A. H. S. (2020). Analisis Security Assessment Menggunakan Metode Penetration Testing dalam Menjaga Kapabilitas Keamanan Teknologi Informasi Pertahanan Negara. *Jurnal Teknologi Penginderaan, 2*(1).

LAMPIRAN