APLIKASI WEB PENGHITUNG SKOR KELEMAHAN UMUM SOFTWARE MENGGUNAKAN SKOR CWSS (*COMMON WEAKNESS SCORING SYSTEM*)

## HALAMAN JUDUL

****

Disusun Oleh:

|  |  |
| --- | --- |
| N a m a  NIM | : Toni Prastiyo  : 15523204 |

**PROGRAM STUDI TEKNIK INFORMATIKA – PROGRAM SARJANA**

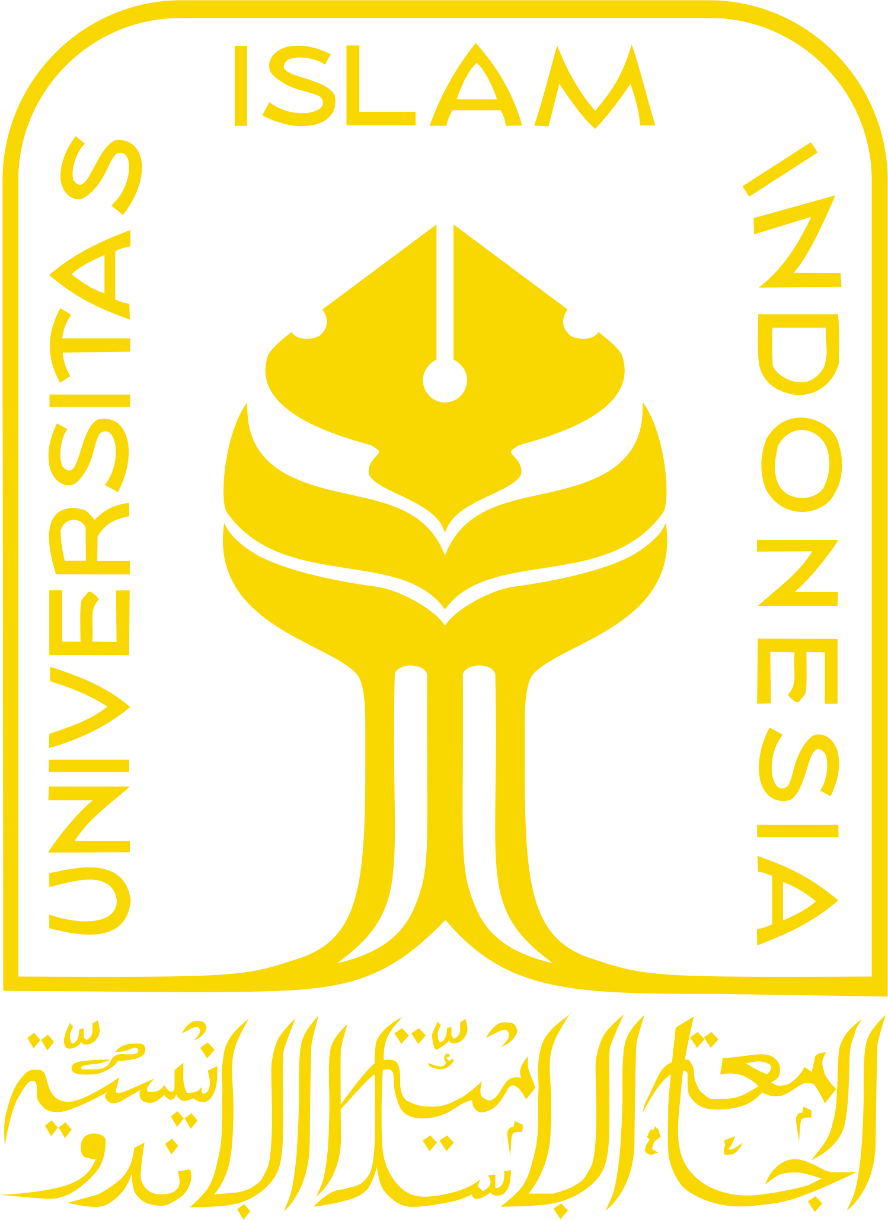
**FAKULTAS TEKNOLOGI INDUSTRI**

**UNIVERSITAS ISLAM INDONESIA**

**2019**

## HALAMAN PENGESAHAN DOSEN PEMBIMBING

APLIKASI WEB PENGHITUNG SKOR KELEMAHAN UMUM SOFTWARE MENGGUNAKAN SKOR CWSS (*COMMON WEAKNESS SCORING SYSTEM*)

TUGAS AKHIR

Disusun Oleh:

|  |  |
| --- | --- |
| N a m a  NIM | : Toni Prastiyo  : 15523204 |

Yogyakarta, 20 Oktober 2019

Pembimbing,

(Yudi Prayudi, S.Si, M.Kom)

## HALAMAN PENGESAHAN DOSEN PENGUJI

APLIKASI WEB PENGHITUNG SKOR KELEMAHAN UMUM SOFTWARE DENGAN MENGGUNAKAN SKOR CWSS (*COMMON WEAKNESS SCORING SYSTEM*)

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk   
memperoleh gelar Sarjana Komputer dari Program Studi Teknik Informatika

di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 4 Nopember 2019

|  |  |
| --- | --- |
| C:\Users\Windows\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Background Lembar Pengesahaan.pngTim Penguji | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Hendrik, S.T., M.Eng. |
| **Anggota 1** | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Dr. Raden Teduh Dirgahayu, S.T., M.Sc. |
| **Anggota 2** | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Dr. Mukhammad A Setiawan, S.T., M.Sc. |

Mengetahui,

Ketua Program Studi Teknik 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 : Toni Prastiyo

NIM : 15523204

Tugas akhir dengan judul:

APLIKASI WEB PENGHITUNG SKOR KELEMAHAN UMUM SOFTWARE MENGGUNAKAN SKOR CWSS (*COMMON WEAKNESS SCORING SYSTEM*)

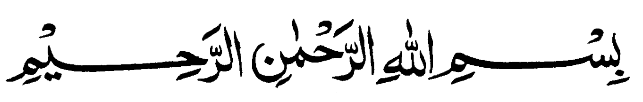
Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila dikemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung resiko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

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

Yogyakarta, 20 Oktober 2019

(Toni Prastiyo)

## HALAMAN PERSEMBAHAN



Dengan Rahmat Allah yang Maha Pengasih lagi Maha Penyayang, saya persembahkan karya tulis ini kepada:

1. Kedua orang tua tercinta, Ayahanda Andri Purwanto dan Ibunda Surani yang telah sabar mendidik saya, selalu memberikan dukungan moral dan selalu mendo’akan yang terbaik untuk saya.
2. Adik saya tersayang, Asyifa Rizki A yang selalu setia mengingatkan dan memberikan semangat kepada saya.
3. Keluarga besar, serta teman-teman yang selalu berbagi suka maupun duka, semoga kita bisa tetap menjaga tali silaturahmi.
4. Dosen pembimbing, Yudhi Prayudhi yang telah meluangkan waktu, serta tenaga, dan selalu sabar dalam membimbing saya selama mengerjakan Tugas Akhir ini.

## HALAMAN MOTO

*“Boleh jadi, kamu membenci sesuatu, padahal ia amat baik bagimu, dan boleh jadi (pula) kamu menyukai sesuatu, padahal ia amat buruk bagimu. Allah yang paling mengetahui, sedangkan kamu tidak mengetahui.”*

**(QS. Al-Baqarah: 216)**

“*Do what you can, with what you have, where you are*.”

**(Theodore Roosevelt)**

*“Kemarin saya pintar, jadi saya ingin mengubah dunia. Hari ini saya bijaksana, jadi saya mengubah diri saya sendiri*.”

**(Jalaluddin Rumi)**

*“Lakukan yang terbaik, sehingga aku tak akan menyalahkan diriku sendiri atas segalanya.”*

**(Magdalena Neuner)**

*“Hidup adalah kegelapan jika tanpa hasrat dan keinginan. Dan semua hasrat serta keinginan adalah buta, jika tidak disertai pengetahuan. Dan pengetahuan adalah hampa jika tidak diikuti pelajaran. Dan setiap pelajaran akan sia-sia jika tidak disertai cinta.”*

**(Kahlil Gibran)**

## KATA PENGANTAR

**Assalamu’alaikum Warahmatullahi Wabarakatuh**

Alhamdulillah, segenap puji dan syukur hanya bagi Allah Swt, Tuhan semesta alam yang selalu melimpahkan rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan penulisan laporan tugas akhir yang berjudul "Aplikasi Web Penghitung Kelemahan Umum Software Dengan Menggunakan Skor CWSS (*Common Weakness Scoring System*)".

Shalawat dan Salam senantiasa tercurah atas junjungan kita Nabi Muhammad Shallallahu ‘Alaihi Wasallam yang telah menuntun kita dari masa kegelapan ke masa terang-benderang dan semoga kita mendapatkan syafaatnya.

Laporan Tugas Akhir ini disusun sebagai salah satu syarat yang harus dipenuhi untuk menyelesaikan pendidikan pada jenjang Strata 1 (S1) Jurusan Teknik Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia. Penulis menyadari bahwa dalam penulisan dan penyusunan laporan ini tidak terlepas dari bantuan, bimbingan, serta dukungan dari berbagai pihak. Oleh karena itu, penulis ingin menyampaikan rasa terima kasih sebesar-besarnya kepada:

1. Allah subhanahu wa ta’ala, yang Maha Pengasih dan Maha Penolong yang selalu memberikan kemudahan dalam perkuliahan sehingga penulis dapat menyelesaikan laporan Tugas Akhir ini dengan baik dan lancar.
2. Orang tua dan keluarga penulis atas segala doa dan dukungan selama penulis melaksanakan penelitian untuk tugas akhir.
3. Bapak Fathul Wahid, S.T., M.Sc., Ph.D. selaku Rektor Universitas Islam Indonesia.
4. Bapak Prof. Dr. Ir. Hari Purnomo, M.T. selaku Dekan Fakultas Teknologi Industri Universitas Islam Indonesia.
5. Bapak Dr. Raden Teduh Dirgahayu, S.T., M.Sc. selaku Ketua Program Studi Teknik Informatika – Program Sarjana Fakultas Teknologi Industri Universitas Islam Indonesia.
6. Bapak Yudi Prayudi, S.Si, M.Si, selaku dosen pembimbing penulis yang telah banyak memberikan waktu luang serta arahan terkait pengerjaan tugas akhir ini.
7. Ibu Chanifah Indah Ratnasari, selaku Dosen Pembimbing Akademik yang telah membimbing penulis.
8. Bapak dan Ibu Dosen Jurusan Teknik Informatika yang telah memberikan ilmunya kepada penulis.
9. Segenap keluarga besar teman-teman di Fakultas Teknologi Industri Universitas Islam Indonesia yang telah memberikan bantuan dan dukungannya.
10. Semua pihak yang dicintai dan disayangi penulis, yang tidak dapat disebutkan satu persatu, terima kasih atas bantuan dan doanya

Rasa hormat dan terima kasih bagi semua pihak atas segala dukungan dan doanya semoga Allah Swt., membalas segala kebaikan yang telah mereka berikan kepada penulis. Amin.

Akhir kata penulis ucapkan terima kasih banyak kepada semua pihak yang telah membantu dan semoga Allah Swt melimpahkan karunianya dalam setiap amal kebaikan kita dan diberikan balasan. Amin.

*Wassalamualaikum Warrohmatullahi Wabarokatuh.*

Yogyakarta, 20 Oktober 2019

(Toni Prastiyo)

## SARI

Para *software developer* sering menghadapi ratusan atau ribuan laporan *bug* yang ditemukan dalam kode mereka. Dalam keadaan tertentu, kelemahan perangkat lunak dapat menyebabkan kerentanan yang dapat dieksploitasi. Karena tingginya volume kelemahan yang dilaporkan tersebut, para *stakeholder* sering dipaksa untuk memprioritaskan masalah mana yang harus mereka selidiki dan perbaiki terlebih dahulu.

CWE (*Common Weakness Enumeration*) merupakan komunitas yang mengembangkan daftar kelemahan umum keamanan perangkat lunak. CWE ini berfungsi sebagai tolak ukur untuk alat keamanan perangkat lunak, dan sebagai dasar untuk identifikasi kelemahan, mitigasi, dan upaya pencegahan.

CWE menyediakan metode *scoring* untuk kelemahan *software* salah satunya yaitu CWSS (*Common Weakness Scoring System*). CWSS memungkinkan organisasi untuk menilai tingkat keparahan kesalahan pengkodean perangkat lunak yang ditemukan dalam aplikasi perangkat lunak yang saat ini mereka gunakan, serta untuk mempengaruhi pembelian di masa depan. CWE telah menyediakan metode *scoring* CWSS, namun untuk saat ini pihak CWE belum menyediakan kalkulatornya.

Solusi yang dapat diberikan dari masalah yang ada yaitu dengan membangun aplikasi penghitung kelemahan umum *software* dengan menggunakan metode *scoring* CWSS berbasis *website* dengan pendekatan *self-assesment*. Aplikasi yang dibangun akan menghasilkan skor-skor kelemahan umum perangkat lunak sehingga bisa membantu para *software developer* ataupun *stakeholder* lainnya bisa mengidentifikasi, memprioritaskan kelemahan perangkat lunak dan bisa mengurangi kelemahan sedini mungkin

Kata kunci: CWE, CWSS, kelemahan, kalkulator, *Self Assesment*, *Software Developer, Scoring, Website.*

## GLOSARIUM

*Browser* Aplikasi yang digunakan untuk mengakses halaman web dan menampilkan konten web.

*CWE* (*Common Weakness Enumeration*) merupakan komunitas yang mengembangkan daftar kelemahan keamanan perangkat lunak umum.

*CWSS* (Common Weakness Scoring System) merupakan salah satu metode *scoring* dari CWE yang menyediakan mekanisme untuk memprioritaskan kelemahan perangkat lunak secara konsisten, fleksibel, dan terbuka, hal ini memungkinkan stakeholder untuk membuat keputusan yang lebih tepat ketika mencoba untuk mengurangi risiko yang disebabkan oleh kelemahan.

*Javascript* Bahasa permograman yang digunakan untuk mengembangkan aplikasi interaktif berbasis web.

*Self Assesment* Teknik penilaian dimana individu dapat menilai dirinya sendiri berkaitan dengan status, proses dan tingkat pencapaian [kompetensi](https://www.dictio.id/t/apa-yang-dimaksud-dengan-kompetensi/119787) yang dipelajarinya

*Website* Halaman informasi yang disediakan melalui jalur internet se-hingga bisa diakses di seluruh dunia selama terkoneksi dengan jaringan internet.

## DAFTAR ISI

[HALAMAN JUDUL i](#_Toc494572314)

[HALAMAN PENGESAHAN DOSEN PEMBIMBING ii](#_Toc494572315)

[HALAMAN PENGESAHAN DOSEN PENGUJI ii](#_Toc494572316)

[HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR ii](#_Toc494572317)

[HALAMAN PERSEMBAHAN ii](#_Toc494572318)

[HALAMAN MOTO ii](#_Toc494572319)

[KATA PENGANTAR ii](#_Toc494572320)

[SARI .. ii](#_Toc494572321)

[GLOSARIUM ii](#_Toc494572322)

[DAFTAR ISI ii](#_Toc494572323)

[DAFTAR TABEL ii](#_Toc494572324)

[DAFTAR GAMBAR ii](#_Toc494572325)

[BAB I TULISKAN JUDUL BAB DI BARIS INI 2](#_Toc494572326)

[1.1 Subbab **Error! Bookmark not defined.**](#_Toc494572327)

[1.1.1 Anak Subbab **Error! Bookmark not defined.**](#_Toc494572328)

[1.1.2 Anak Subbab **Error! Bookmark not defined.**](#_Toc494572329)

[1.2 Subbab **Error! Bookmark not defined.**](#_Toc494572330)

[1.3 Subbab **Error! Bookmark not defined.**](#_Toc494572331)

[1.4 Subbab **Error! Bookmark not defined.**](#_Toc494572332)

[BAB II TULISKAN JUDUL BAB DI BARIS INI 2](#_Toc494572333)

[2.1 Subbab 2](#_Toc494572334)

[2.2 Subbab **Error! Bookmark not defined.**](#_Toc494572335)

[BAB III TULISKAN JUDUL BAB DI BARIS INI 2](#_Toc494572336)

[3.1 Subbab **Error! Bookmark not defined.**](#_Toc494572337)

[3.2 Subbab **Error! Bookmark not defined.**](#_Toc494572338)

[3.3 Subbab **Error! Bookmark not defined.**](#_Toc494572339)

[BAB IV TULISKAN JUDUL BAB DI BARIS INI 2](#_Toc494572340)

[4.1 Subbab **Error! Bookmark not defined.**](#_Toc494572341)

[4.2 Subbab **Error! Bookmark not defined.**](#_Toc494572342)

[4.3 Subbab **Error! Bookmark not defined.**](#_Toc494572343)

[BAB V TULISKAN JUDUL BAB DI BARIS INI 2](#_Toc494572344)

[5.1 Subbab **Error! Bookmark not defined.**](#_Toc494572345)

[5.2 Subbab 2](#_Toc494572346)

[5.3 Subbab **Error! Bookmark not defined.**](#_Toc494572347)

[BAB VI TULISKAN JUDUL BAB DI BARIS INI 2](#_Toc494572348)

[6.1 Subbab **Error! Bookmark not defined.**](#_Toc494572349)

[6.2 Subbab **Error! Bookmark not defined.**](#_Toc494572350)

[6.3 Subbab **Error! Bookmark not defined.**](#_Toc494572351)

[DAFTAR PUSTAKA 2](#_Toc494572352)

[LAMPIRAN 2](#_Toc494572353)

## DAFTAR TABEL

[Tabel 2.1 Grup Metrik Base Finding Subscore 2](#_Toc22459424)

[Tabel 2.2 Grup Metrik Attack Surface Subscore 2](#_Toc22459425)

[Tabel 2.3 Grup Metrik Environmental Subscore 2](#_Toc22459426)

[Tabel 2.4 Nilai Ketidakpastian Dan Fleksibilitas 2](#_Toc22459427)

[Tabel 2.5 Technical Impact 2](#_Toc22459428)

[Tabel 2.6 Acquired Privilege 2](#_Toc22459429)

[Tabel 2.7 Acquired Privilege Layer 2](#_Toc22459430)

[Tabel 2.8 Internal Control Effectiveness 2](#_Toc22459431)

[Tabel 2.9 Finding Confidence 2](#_Toc22459432)

[Tabel 2.10 Required Privilege 2](#_Toc22459433)

[Tabel 2.11 Required Privilege Layer 2](#_Toc22459434)

[Tabel 2.12 Access Vector 2](#_Toc22459435)

[Tabel 2.13 Authentication Strength 2](#_Toc22459436)

[Tabel 2.14 Level Of Interaction 2](#_Toc22459437)

[Tabel 2.15 Deployment Scope 2](#_Toc22459438)

[Tabel 2.16 Business Impact 2](#_Toc22459439)

[Tabel 2.17 Likelihood Of Discovery 2](#_Toc22459440)

[Tabel 2.18 Likelihood Of Exploit 2](#_Toc22459441)

[Tabel 2.19 External Control Efectiveness 2](#_Toc22459442)

[Tabel 2.20 Prevalence 2](#_Toc22459443)

[Tabel 2.21 CWSS VS CVSS 2](#_Toc22459444)

[Tabel 3.1 Rancangan Skenario 1 Pengujian Skor 2](#_Toc22459445)

[Tabel 3.2 Rancangan Skenario 2 Pengujian Skor 2](#_Toc22459446)

[Tabel 3.3 Rancangan Kuesioner Pengujian 2](#_Toc22459447)

[Tabel 3.4 Kategori Penilaian Kuisioner 2](#_Toc22459448)

[Tabel 4.1 Hasil Pengujian Skor Skenario 1 2](#_Toc22459449)

[Tabel 4.2 Hasil Pengujian Skor Skenario 2 2](#_Toc22459450)

[Tabel 4.3 Hasil Kuesioner Pengujian Implementasi Sistem 2](#_Toc22459451)

## DAFTAR GAMBAR

[Gambar 2.1 Contoh Software Weakness 2](#_Toc22459392)

[Gambar 2.2 Model Waterfall 2](#_Toc22459393)

[Gambar 2.3 Cara Kerja CWSS 2](#_Toc22459394)

[Gambar 2.4 Grup Metric CWSS 2](#_Toc22459395)

[Gambar 2.5 Base Score Aplikasi CVSS 2](#_Toc22459396)

[Gambar 2.6 Temporal Score Aplikasi CVSS 2](#_Toc22459397)

[Gambar 2.7 Environmental Score Aplikasi CVSS 2](#_Toc22459398)

[Gambar 3.1 Use Case Diagram 2](#_Toc22459399)

[Gambar 3.2 Activity Diagram Lihat Halaman Home 2](#_Toc22459400)

[Gambar 3.3 Activity Diagram Lihat Halaman Deskripsi 2](#_Toc22459401)

[Gambar 3.4 Activity Diagram Melakukan Penilaian Di Halaman Assesment 2](#_Toc22459402)

[Gambar 3.5 Diagram HIPO 2](#_Toc22459403)

[Gambar 3.6 Perancangan Mockup Halaman Home 2](#_Toc22459404)

[Gambar 3.7 Perancangan Mockup Halaman Deskripsi 2](#_Toc22459405)

[Gambar 3.8 Perancangan Mockup Halaman Assesment 1 2](#_Toc22459406)

[Gambar 3.9 Perancangan Mockup Halaman Assesment 2 2](#_Toc22459407)

[Gambar 4.1 Halaman Home 2](#_Toc22459408)

[Gambar 4.2 Halaman Deskripsi 2](#_Toc22459409)

[Gambar 4.3 Halaman Pengertian CWSS Submenu Deskripsi 2](#_Toc22459410)

[Gambar 4.4 Halaman CWSS Formula Submenu Deskripsi 2](#_Toc22459411)

[Gambar 4.5 Halaman CWSS Metric Grup Submenu Deskripsi 2](#_Toc22459412)

[Gambar 4.6 Halaman Assesment Mengisi Identitas Diri 2](#_Toc22459413)

[Gambar 4.7 Halaman Assesment 2](#_Toc22459414)

[Gambar 4.8 Halaman Assesment Base Finding Score 2](#_Toc22459415)

[Gambar 4.9 *Source code* perhitungan base finding score 2](#_Toc22459416)

[Gambar 4.10 Halaman Assesment Attack Surface Score 2](#_Toc22459417)

[Gambar 4.11 *Source code* Perhitungan Attack Surface Score 2](#_Toc22459418)

[Gambar 4.12 Halaman Assesment Environmental Score 2](#_Toc22459419)

[Gambar 4.13 *Source code* Perhitungan Environmental Score 2](#_Toc22459420)

[Gambar 4.14 Halaman Assesment *click* CWSS Score 2](#_Toc22459421)

[Gambar 4.15 Halaman Assesment CWSS Score 2](#_Toc22459422)

[Gambar 4.16 *Source code* Perhitungan CWSS Score 2](#_Toc22459423)

## BAB I PENDAHULUAN

### Latar Belakang

Para *software developer* sering menghadapi ratusan atau ribuan laporan *bug* yang ditemukan dalam kode mereka. Dalam keadaan tertentu, kelemahan perangkat lunak dapat menyebabkan kerentanan yang dapat dieksploitasi. Karena tingginya volume kelemahan yang dilaporkan tersebut, para *stakeholder* sering dipaksa untuk memprioritaskan masalah mana yang harus mereka selidiki dan perbaiki terlebih dahulu, seringkali mereka menggunakan informasi yang tidak lengkap. Walaupun berbagai metode penilaian sudah digunakan saat ini, metode-metode tersebut tidak sesuai untuk aplikasi pada evaluasi keamanan perangkat lunak dan masih belum tepat penggunaanya (B. Martin & Coley, 2014)

*Software developer*, *manager, tester*, pembeli, vendor aplikasi, vendor keamanan dan peneliti harus mengidentifikasi dan menilai kelemahan dalam perangkat lunak yang mempunyai potensi sebagai kerentanan ketika perangkat lunak digunakan. Mereka harus dapat memprioritaskan kelemahan-kelemahan ini dan menentukan mana yang akan diperbaiki berdasarkan risiko terbesar.

Ketika ada begitu banyak kelemahan untuk diperbaiki, dengan masing-masing dinilai menggunakan skala yang berbeda dan sering beroperasi dengan informasi yang tidak lengkap, berbagai anggota komunitas, *manager*, penguji, pembeli, dan pengembang dibiarkan menggunakan metodologi mereka sendiri untuk menemukan cara membandingkan perbedaan yang berbeda (B. Martin & Coley, 2014).

CWE (*Common Weakness Enumeration*) merupakan komunitas yang mengembangkan daftar kelemahan umum keamanan perangkat lunak. CWE ini berfungsi sebagai tolak ukur untuk alat keamanan perangkat lunak, dan sebagai dasar untuk identifikasi kelemahan, mitigasi, dan upaya pencegahan.Tujuan utama dari CWE adalah untuk menghentikan kerentanan pada sumber dengan mendidik pengakuisisi perangkat lunak, arsitek, perancang, dan *programmer* tentang cara menghilangkan kesalahan paling umum sebelum perangkat lunak di kirimkan. CWE pada akhirnya membantu para *developer* mencegah jenis kerentanan keamanan yang mengganggu industri perangkat lunak dan membahayakan perusahaan. CWE terus berkembang sebagai upaya kolaboratif untuk mengisi repositori kesalahan perangkat lunak yang tersedia untuk umum dalam kode, desain, arsitektur, dan implementasi untuk pengembang dan praktisi keamanan yang juga dapat digunakan oleh masyarakat umum (R. A. Martin, 2007).

Berbagai sistem penilaian kelemahan telah digunakan atau diusulkan selama bertahun-tahun. Alat pemindai otomatis *source code* biasanya melakukan penilaian kustom sendiri. Sebagai hasilnya, banyak alat dapat menghasilkan skor yang tidak konsisten untuk kelemahan yang sama. CWE menyediakan metode *scoring* untuk kelemahan *software* salah satunya yaitu CWSS (*Common Weakness Scoring System*). CWSS memungkinkan organisasi untuk menilai tingkat keparahan kesalahan pengkodean perangkat lunak yang ditemukan dalam aplikasi perangkat lunak yang saat ini mereka gunakan serta untuk mempengaruhi pembelian di masa depan. CWSS menyediakan mekanisme memprioritaskan kelemahan perangkat lunak secara konsisten, fleksibel, dan terbuka. Ini adalah upaya kolaboratif berbasis komunitas yang menangani kebutuhan para *stakeholder* di seluruh pemerintahan, akademisi, dan industri. Walaupun CWE telah menyediakan metode *scoring* CWSS, namun untuk saat ini pihak CWE tidak menyediakan kalkulator CWSS (B. Martin & Coley, 2014).

Seperti yang penulis sudah dijelaskan diatas, sebenarnya sudah ada berbagai metode yang digunakan untuk mengukur kelemahan suatu perangkat lunak tetapi masih menggunakan metode yang belum tepat dalam penggunaannya dan hasilnya tidak konsisten pada kelemahan yang sama saat diuji. CWE menyediakan metode *scoring* kelemahan CWSS (*Common Weakness Scoring System*) namun mereka belum menyediakan kalkulatornya, sehingga penulis bertujuan membangun aplikasi penghitung kelemahan umum *software* dengan menggunakan metode *scoring* CWSS berbasis *website* dengan pendekatan *self-assesment*. Aplikasi yang dibangun akan menghasilkan skor-skor kelemahan umum perangkat lunak yang diuji sehingga bisa membantu para *software developer* ataupun *stakeholder* lainnya bisa mengidentifikasi, memprioritaskan kelemahan perangkat lunak dan bisa mengurangi kelemahan sedini mungkin

### Rumusan Masalah

Berdasarkan latar belakang diatas dapat diperoleh rumusan masalah yaitu bagaimana membangun suatu sistem yang dapat digunakan untuk menghitung skor kelemahan umum pada *software* sehingga para *software developer* bisa memprioritaskan kelemahan pada *software* yang kelak akan menimbulkan kerentanan dan harus segera diperbaiki.

### Batasan Masalah

Untuk menjaga fokus penelitian dalam TA ini, ada beberapa batasan yang perlu diperhatikan:

1. Penerapan aplikasi perhitungan skor kelemahan menggunakan formula dari CWSS (*Common Weakness Scoring System*) versi 1.0.1
2. Aplikasi yang dibangun berbasis *website*
3. Aplikasi yang dibangun menggunakan pendekatan *self-assessment*

### Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah membuat aplikasi penghitung skor kelemahan umum *software* berbasis websitedengan menggunakan metode *scoring* CWSS (*Common weakness Scoring System*)

### Manfaat Penelitian

Manfaat yang didapat pada penelitian ini yaitu:

* 1. Membantu *developer, researcher, end user,* praktisi keamanan meminimalkan kelemahan dalam perangkat lunak mereka sedini mungkin dan membantu mereka dalam memprioritaskan kelemahan- kelemahan pada *software*.
  2. Membantu *Stakeholder* untuk membuat keputusan yang lebih tepat ketika mencoba untuk mengurangi risiko yang disebabkan oleh kelemahan

### Metodologi Penelitian

Adapun metodologi penelitian yang digunakan dalam pembuatan aplikasi:

* + - * 1. Studi pustaka

Mengumpulkan data sesuai topik dengan mencari bahan dari internet, artikel, jurnal ilmiah, maupun tugas akhir.

* + - * 1. Analisis kebutuhan

Pada tahapan ini dilakukan analisis terhadap kebutuhan apa saja yang diperlukan dalam penelitian ini, seperti identifikasi masalah, usulan solusi dan kebutuhan sistem.

* 1. Desain perancangan sistem

Pada tahapan ini dilakukan pembuatan rancang desain dari apa yang sudah didapatkan dari hasil analisis pada tahapan sebelumnya, yang nantinya akan dipakai pada proses implementasi.

1. Implementasi

Pada tahapan ini dilakukan implementasi dari hasil perancangan sistem yang sebelumnya telah dibuat. Aplikasi penghitung skor kelemahan umum *software* ini akan diimplementasikan dalam bentuk aplikasi web.

1. Pengujian

Pada tahapan ini dilakukan pengujian terhadap sistem yang telah dikembangkan. Pengujian dilakukan untuk memeriksa atau melakukan perbaikan apabila terdapat kesalahan

### Sistematika Penulisan

Sistematika penulisan ditujukan untuk memudahkan dalam melakukan pembahasan tugas akhir ini. Secara garis besar, sistematika penulisan laporan ini terbagi menjadi lima bab. Adapun uraian dari masing-masing bab tersebut yaitu:

* + - * 1. Bab I Pendahuluan

Pada bab ini berisi latar belakang, batasan masalah, rumusan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian, dan sistematika penulisan.

* + - * 1. Bab II Landasan Teori

Pada bab ini berisi teori-teori yang berkaitan dan menjadi dasar dalam penelitian. Adapun teori-teori tersebut tentang metode *Scoring* CWSS (*Common Weakness Scoring System*), *Software Weakness*, *Self Asessment* dan membahas peninjauan sistem sejenis yang akan menjadi dasar dalam pengembangan aplikasi penghitung skor kelemahan umum *software*.

* + - * 1. Bab III Metodologi

Pada bab ini berisi identifikasi masalah, gambaran umum sistem, solusi penyelesaian masalah, analisis kebutuhan, perancangan sistem, dan desain pengujian yang diperlukan dalam pengembangan sistem.

* + - * 1. Bab IV Hasil dan Pembahasan

Pada bab ini berisi hasil implementasi dan rancangan yang telah dibuat serta hasil pengujian dari sistem yang telah dibuat terhadap calon pengguna.

* + - * 1. Bab V Kesimpulan dan Saran

Pada bab ini berisi kesimpulan dari penelitian yang telah dilakukan dan saran untuk pengembang sistem ini kedepannya.

.

## BAB II LANDASAN TEORI

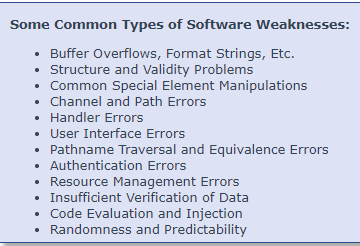
### Self-Assesment

*Self-assesment* merupakan teknik penilaian di mana individu dapat menilai dirinya sendiri berkaitan dengan status, proses dan tingkat pencapaian [kompetensi](https://www.dictio.id/t/apa-yang-dimaksud-dengan-kompetensi/119787) yang dipelajarinya (Mimin, 2007). Adapun menurut (Andrade & Du, 2007) *self-assesment* adalah proses penilaian formatif dimana individu merenungkan, mengevaluasi kualitas mereka dan menilai sejauh mana kualitas yang dimiliki kemudian memperbaiki kekurangan yang ada dikemudian hari.

Dari pengertian pakar diatas dapat ditarik sebuah kesimpulan bahwa self-assessmentadalah teknik penilaian yang dilakukan oleh individu dalam mengevaluasi dan mengemukakan kelebihan maupun kekurangan dirinya dalam berbagai hal, serta mampu untuk menindaklanjuti dan memperbaiki segala kekurangan yang ada.

### Software Weakness

*Sofware weakness* atau kelemahan perangkat lunak adalah suatu kecacatan, kerentanan, ataupun kesalahan lain dalam implementasi kode, desain, atau arsitektur perangkat lunak yang jika tidak ditangani dapat menyebabkan sistem dan jaringan rentan terhadap serangan (R. A. Martin, 2007). Beberapa jenis kelemahan perangkat lunak yang umum bisa dilihat pada Gambar 2.1.



Gambar 2.1 Contoh Software Weakness

**Perbedaan Software Weakness dan Software Vulnerability**

*Software weakness* adalah kesalahan yang dapat menyebabkan kerentanan perangkat lunak, sedangkan *Software vulnerability* adalah kesalahan dalam perangkat lunak yang dapat langsung digunakan oleh peretas untuk mendapatkan akses ke sistem atau jaringan (R. A. Martin, 2007).

### Aplikasi Web

Aplikasi web adalah aplikasi yang telah dirancang khusus untuk dijalankan di lingkungan berbasis web (Finkelstein, Kappel, & Retschitzegger, 2002). Aplikasi web memungkinkan fungsi pemrosesan informasi dimulai dari *browser* dan dieksekusi sebagian pada *web server*, *application server* dan *database server* (Tam, 2005).

Aplikasi web biasanya dikodekan dalam bahasa yang didukung *browser* seperti JavaScript dan HTML. Beberapa aplikasi bersifat dinamis, membutuhkan pemrosesan di sisi server, yang lain sepenuhnya statis tanpa perlu pemrosesan di server. Aplikasi web memerlukan *web server* untuk mengelola permintaan dari klien, server aplikasi untuk melakukan tugas yang diminta, dan kadang-kadang *database* untuk menyimpan informasi. Teknologi application server berkisar seperti ASP.NET, ASP dan ColdFusion, hingga PHP dan JSP (Gibb, 2016).

Keuntungan aplikasi web:

1. Aplikasi web dapat berjalan pada berbagai *platform*
2. Aplikasi web tidak diinstal pada *harddrive*, sehingga menghilangkan keterbatasan ruang
3. Aplikasi web mengurangi pembajakan perangkat lunak
4. Aplikasi web mudah dikembangkan
5. Aplikasi web mudah diakses

### Javascript

JavaScript adalah bahasa pemrograman pada website. Mayoritas situs web modern menggunakan JavaScript. JavaScript merupakan bahasa pemrograman paling umum dalam sejarah. JavaScript adalah bagian dari triad teknologi yang harus dipelajari semua pengembang Web. HTML untuk menentukan konten halaman web, CSS untuk menentukan presentasi halaman web, dan JavaScript untuk menentukan perilaku halaman web (Flanagan, 2006).

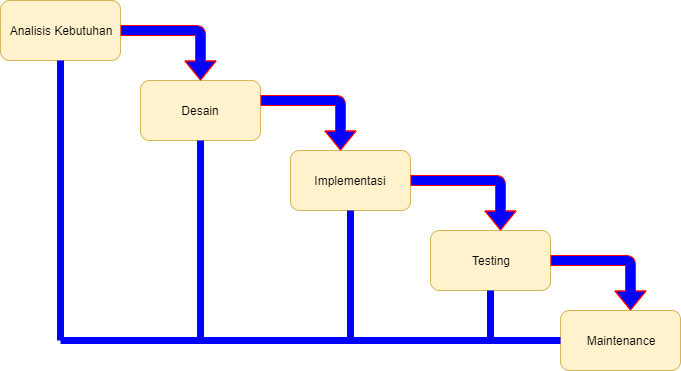
JavaScript populer di [internet](https://id.wikipedia.org/wiki/Internet) dan dapat bekerja di sebagian besar [penjelajah web](https://id.wikipedia.org/wiki/Penjelajah_web) populer seperti Google Chrome, [Internet Explorer](https://id.wikipedia.org/wiki/Internet_Explorer) (IE), [Mozilla Firefox](https://id.wikipedia.org/wiki/Mozilla_Firefox), [Netscape](https://id.wikipedia.org/wiki/Netscape_Navigator) dan [Opera](https://id.wikipedia.org/wiki/Opera). [Kode](https://id.wikipedia.org/wiki/Kode) JavaScript dapat disisipkan dalam [halaman](https://id.wikipedia.org/wiki/Halaman) [web](https://id.wikipedia.org/wiki/Web) menggunakan [tag](https://id.wikipedia.org/wiki/Tag) SCRIPT. JavaScript merupakan salah satu teknologi inti [World Wide Web](https://id.wikipedia.org/wiki/World_Wide_Web) selain [HTML](https://id.wikipedia.org/wiki/HTML) dan [CSS](https://id.wikipedia.org/wiki/Cascading_Style_Sheets). JavaScript membantu membuat halaman web interaktif dan merupakan bagian aplikasi web yang esensial (Sunyoto & Kom, 2007).

Beberapa hal tentang javascript menurut (Sunyoto & Kom, 2007):

1. Javascript didesain agar website menjadi interaktif
2. Javascript merupakan sebuah bahasa scripting yakni bahasa pemrograman yang cenderung ringan
3. Javascript berisi baris kode yang dijalankan di web browser
4. Javascript biasanya disisipkan dalam halaman HTML
5. Setiap orang dapat menggunakan javascript tanpa membayar lisensi

### Metode Waterfall

Model waterfall adalah model klasik yang bersifat sistematis, berurutan dalam membangun software (Pressman, 2012). Model waterfall diagram prosesnya mirip dengan air terjun yang bertingkat. Model ini termasuk ke dalam model yang paling banyak dipakai dalam *Software Engineering* (SE). Tahapan-tahapan pada metode waterfall dimulai dari melakukan analisis kebutuhan, desain sistem, implementasi kode pemrograman, pengujian, dan maintenance dapat dilihat pada Gambar 2.2. Penulis memilih menggunakan metode pengembangan waterfall dalam mengembangkan aplikasi web penghitung skor kelemahan umum menggunakan skor CWSS (*Common Weakness Scoring System*) supaya kebutuhan sistem dapat didokumentasikan dengan lengkap dan proses pengembangannya dapat dilaksanakan tepat waktu.



Gambar 2.2 Model Waterfall

Dalam pengembangannya metode waterfall memiliki beberapa tahapan sebagai berikut :

1. Analisis Kebutuhan

Tahap ini pengembang sistem menganalisis permasalahan yang dihadapi dan mengumpulkan data-data yang diperlukan, serta membantu mendefinisikan fitur dan fungsi software. Pengumpulan data-data tambahan bisa juga diambil dari jurnal, artikel, dan internet.

1. Desain

Tahap ini adalah tahap perancangan dan permodelan arsitektur sistem yang berfokus pada perancangan struktur data, arsitektur software, tampilan interface, dan algoritma program. Tujuannya untuk lebih memahami gambaran besar dari apa yang akan dikerjakan..

1. Implementasi

Tahap ini pengembang akan mengubah hasil desain tersebut menjadi kode program dan modul-modul yang nantinya akan dilakukan integrasi menjadi sebuah sistem aplikasi yang utuh dan juga dilakukan pengujian terhadap fungsionalitas yang telah dikembangkan.

1. Pengujian

Proses ini akan menguji kode program yang telah dibuat dengan memfokuskan pada bagian dalam piranti lunak. Tujuannya untuk memastikan bahwa semua pernyataan telah diuji dan memastikan juga bahwa input yang digunakan akan menghasilkan output yang sesuai.

1. Maintenance

Tahap akhir dalam model waterfall. Perangkat lunak yang sudah jadi dijalankan serta dilakukan pemeliharaan. Pemeliharaan termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya. Perbaikan implementasi unit sistem dan peningkatan jasa sistem sebagai kebutuhan baru.

### Common Weakness Enumeration (CWE)

CWE (*Common Weakness Enumeration*) merupakan komunitas yang mengembangkan daftar kelemahan umum keamanan perangkat lunak. CWE ini berfungsi sebagai bahasa umum yang menggambarkan kelemahan keamanan perangkat lunak dalam arsitektur, desain, atau kode. CWE juga berfungsi sebagai standar perbandingan untuk alat keamanan perangkat lunak yang menargetkan kelemahan. Selain itu CWE memberikan dasar standar umum untuk identifikasi kelemahan, mitigasi, dan upaya pencegahan (R. A. Martin, 2007).

CWE bebas untuk digunakan oleh organisasi atau individu apapun untuk tujuan penelitian, pengembangan, dan / atau komersial sesuai dengan ketentuan. CWE ini dikelola dan didukung oleh MITRE agar memungkinan adanya kolaborasi dengan *stakeholder* lain. MITRE adalah perusahaan nirlaba yang bekerja untuk kepentingan umum serta membahas isu-isu penting nasional secara kritis, menggabungkan rekayasa sistem dan teknologi informasi untuk mengembangkan solusi inovatif yang membuat perbedaan (R. A. Martin, 2007).

CWE menyediakan beberapa metode *scoring* kelemahan *software* yaitu:

1. Memprioritaskan kelemahan berdasarkan misi organisasi

Proyek CWE menawarkan beberapa pendekatan untuk memprioritaskan kelemahan sehingga *developer* dapat fokus pada subset yang sesuai untuk kebutuhan organisasi. Metode ini digunakan untuk mendapatkan manfaat langsung dari peningkatan besar pada ketahanan, keandalan, dan integritas perangkat lunak sesegera mungkin.

1. Common Weakness Scoring System (CWSS™)

CWSS menyediakan mekanisme menilai kelemahan secara konsisten, fleksibel, terbuka sambil mengakomodasi konteks untuk berbagai domain bisnis. CWSS juga dapat digunakan oleh masing-masing pengembang untuk memprioritaskan kelemahan yang tidak tetap dalam perangkat lunak mereka sendiri.

1. Common Weakness Risk Analysis Framework (CWRAF™)

Metode CWRAF ini digunakan bersama dengan CWSS, akan memberi organisasi daftar kelemahan "umum XX" yang dirancang khusus. .

1. CWE/SANS Top 25 Most Dangerous Software Errors

*The CWE/SANS Top 25 Most Dangerous Software Errors* adalah daftar terbaru kelemahan perangkat lunak yang paling umum dan mudah dieksploitasi yang dinilai oleh lebih dari 20 pakar industri

### CWSS (Common Weakness Scoring System)

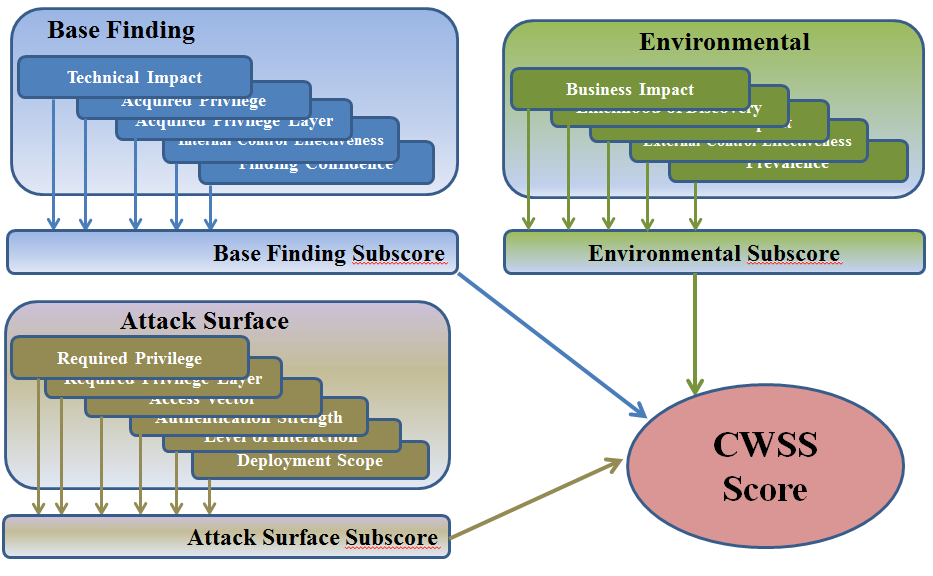
CWSS (*Common Weakness Scoring System*) adalah bagian dari proyek *Common Weakness Enumeration* (CWE) yang menyediakan mekanisme untuk memprioritaskan kelemahan perangkat lunak secara konsisten, fleksibel, dan terbuka, hal ini memungkinkan *stakeholder* untuk membuat keputusan yang lebih tepat ketika mencoba untuk mengurangi risiko yang disebabkan oleh kelemahan.

Cwss menawarkan:

1. *Quantitative measurement*: CWSS memberikan pengukuran kelemahan secara kuantitatif yang tidak tetap yang ada dalam aplikasi perangkat lunak.
2. *Common framework*: CWSS menyediakan kerangka kerja umum untuk memprioritaskan kesalahan keamanan (“kelemahan”) yang ditemukan dalam aplikasi perangkat lunak
3. *Customized Prioritization*: CWSS dapat digunakan oleh konsumen untuk mengidentifikasi jenis-jenis kelemahan terpenting yang sesuai dengan domain bisnis mereka, untuk menginformasikan kegiatan akuisisi dan perlindungan mereka sebagai salah satu bagian dari yang lebih besar.

#### Cara Kerja CWSS (Common Weaknesss Scoring System)

Setiap faktor dalam grup *Base Finding metric group* diberi nilai (value). Nilai-nilai ini dikonversikan ke bobot terkait. *Base finding Subscore* dapat berkisar antara 0 dan 100. Metode yang sama juga diterapkan pada *Attack Surface* dan *Environmental metric group*. *Subscore* mereka berkisar antara 0 dan 100 yang mana tiga *subscore* dikalikan dan menghasilkan skor CWSS antara 0 dan 100 (B. Martin & Coley, 2014). Perhitungan skor CWSS dapat dilihat pada Gambar 2.3.



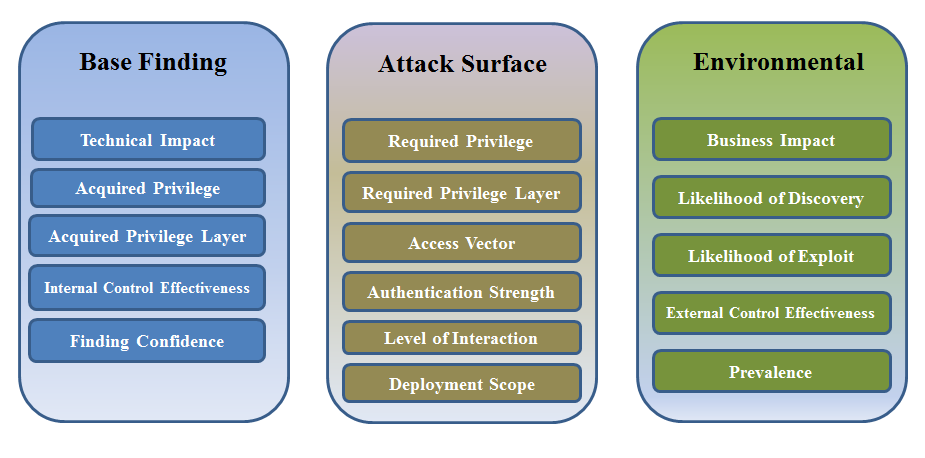
Gambar 2.3 Cara Kerja CWSS

#### Grup Metrik CWSS

CWSS ini tersusun dari tiga grup metrik:

1. *Base finding metric group*: menangkap resiko yang berkaitan dengan kelemahan, kepercayaan pada keakuratan temuan dan kekuatan mengontrol aplikasi dari internal.
2. *Attack surface metric group*: hambatan yang harus diatasi penyerang untuk mengeksploitasi kelemahan
3. *Environmental Metric group*: suatu karakteristik kelemahan yang spesifik dalam konteks operasional tertentu.

Gambar grup metric bisa dilihat pada Gambar 2.4



Gambar 2.4 Grup Metric CWSS

CWSS disusun berdasarkan beberapa grup metrik dan bisa dilihat pada Tabel 2.1,

Tabel 2.2, Tabel 2.3.

Tabel 2.1 Grup Metrik Base Finding Subscore

|  |  |  |
| --- | --- | --- |
| Group | Nama | Summary |
| Base finding | Technical Impact (TI) | Dampak teknis yang diakibatkan oleh kelemahan, dengan asumsi bahwa kelemahan tersebut berhasil dieksploitasi |
| Base finding | Acquired Privilage (AP) | Hak istimewa yang diperoleh penyerang untuk mengeksploitasi kelemahan |
| Base finding | Acquired Privileged Layer (AL) | Lapisan operasional dimana penyerang mendapatkan hak istimewa dengan berhasil mengeksploitasi kelemahan |
| Base finding | Internal Control Effectiveness (IC) | Kemampuan mengontrol kelemahan dari internal agar tidak dapat diekploitasi oleh penyerang |
| Base finding | Finding Confidence (FC) | Keyakinan bahwa masalah yang dilaporkan adalah kelemahan yang dapat dimanfaatkan oleh penyerang |

Tabel 2.2 Grup Metrik Attack Surface Subscore

|  |  |  |
| --- | --- | --- |
| Group | Nama | Summary |
| Attack surface | Required Privileged (RP) | Jenis hak istimewa yang harus dimiliki penyerang untuk mencapai kode/ fungsi yang mengandung kelemahan |
| Attack surface | Required Privilaged Layer (RL) | Lapisan operasional dimana penyerang harus memiliki hak istimewa untuk mencoba menyerang kelemahan |
| Attack surface | Acces Vector (AV) | Saluran dimana penyerang harus berkomunikasi untuk mencapai kode atau fungsionalitas yang mengandung kelemahan |
| Attack surface | Authentication Strength (AS) | Kemampuan autentikasi yang melindungi kode / fungsionalitas yang mengandung kelemahan |
| Attack surface | Level of Interaction (IN) | Tindakan yang diperlukan oleh korban untuk kemungkinan serangan yang berhasil terjadi |
| Attack surface | Deployment Scope (SC) | Apakah kelemahan yang hadir dalam semua instance perangkat lunak dapat digunakan atau jika terbatas pada subset platform dan atau konfigurasi |

Tabel 2.3 Grup Metrik Environmental Subscore

|  |  |  |
| --- | --- | --- |
| Group | Nama | Summary |
| Environmental | Business Impact (BI) | Dampak secara bisnis atau tujuan jika kelemahan berhasil dieksploitasi |
| Environmental | Likelihood of Discovery (DI) | Kemungkinan dimana penyerang dapat menemukan kelemahannya |
| Environmental | Likelihood of Exploir (EX) | Kemungkinan bahwa, jika kelemahan ditemukan, penyerang dengan hak istimewa/ memiliki autentikasi/ akses syang diperlukan dapat berhasil mengeksploitasinya |
| Environmental | External Control Effectiveness (EC) | Kemampuan kontrol atau mitigasi di luar perangkat lunak yang dapat membuat kelemahan lebih sulit bagi penyerang untuk mencapai dan memicu perangkat lunak |
| Environmental | Prevelance (P) | Seberapa sering jenis kelemahan ini muncul dalam perangkat lunak |

**Nilai untuk Ketidakpastian dan Fleksibilitas**

CWSS dapat digunakan dalam kasus di mana ada sedikit informasi pada awalnya, tetapi kualitas informasi dapat meningkatkan seiring berjalannya waktu. Hal ini diantisipasi bahwa dalam banyak penggunaan-kasus, Skor CWSS menemukan kelemahan individu dapat sering berubah, karena informasi lebih lanjut ditemukan.

Entitas yang berbeda dapat menentukan faktor terpisah pada titik waktu yang berbeda. Dengan demikian, setiap faktor CWSS secara efektif memiliki "lingkungan " atau "temporal " karakteristik, sehingga tidak terlalu berguna untuk mengadopsi jenis yang sama kelompok metrik seperti yang digunakan dalam CVSS. Nilai ketidakpastian dan fleksibilitas ini bisa dilihat pada Tabel 2.4.

Tabel 2.4 Nilai Ketidakpastian Dan Fleksibilitas

|  |  |
| --- | --- |
| Value | Penggunaan |
| Unknown | Entitas menghitung skor yang tidak memiliki cukup informasi untuk memberikan nilai untuk faktor. Hal ini dapat menjadi sinyal untuk penyelidikan lebih lanjut. Misalnya, pemindai kode otomatis mungkin dapat menemukan kelemahan tertentu, namun tidak dapat mendeteksi apakah ada mekanisme autentikasi yang berada di tempatnya.  Penggunaan "Unknown” menekankan bahwa skor tidak lengkap atau diperkirakan, dan analisis lebih lanjut mungkin diperlukan. Hal ini membuat lebih mudah untuk model informasi yang tidak lengkap, dan untuk bisnis nilai konteks untuk mempengaruhi skor akhir yang dihasilkan dengan menggunakan informasi yang tidak lengkap.  Bobot untuk nilai ini adalah 0,5 untuk semua faktor, yang umumnya menghasilkan skor yang lebih rendah. Penambahan informasi baru mengubah beberapa faktor dari “Unknown” ke nilai lain kemudian akan menyesuaikan skor ke atas atau ke bawah berdasarkan informasi baru. |
| Not Applicable | Faktor yang secara eksplisit diabaikan dalam perhitungan skor. Hal ini secara efektif memungkinkan konteks nilai bisnis untuk mendikte apakah suatu faktor relevan dengan skor akhir. Misalnya, metode penilaian CWSS yang berfokus pada pelanggan mungkin mengabaikan upaya remediasi, dan lingkungan dengan jaminan tinggi mungkin memerlukan investigasi atas semua temuan yang dilaporkan, bahkan jika ada keyakinan rendah terhadap akurasinya.  Untuk satu set kelemahan temuan untuk paket perangkat lunak individu, diharapkan bahwa semua temuan akan sama "tidak berlaku” nilai untuk faktor yang sedang diabaikan. |
| Quantified | Faktor ini dapat dibobot dengan menggunakan kuantifikasi, terus menerus rentang 0, 0 melalui 1, 0, bukan didefinisikan faktor set nilai diskrit. Tidak semua faktor dapat dikuantifikasi dengan cara ini, tetapi memungkinkan penyesuaian metrik tambahan. |
| Default | Bobot faktor tersebut dapat diatur ke nilai default. Pelabelan faktor sebagai default memungkinkan untuk penyelidikan dan kemungkinan modifikasi di lain waktu |

##### Base Finding Metric Grup

Base finding metric grup terdiri dari faktor – faktor berikut:

1. Technical impact (TI)
2. Acquired privilege (AP)
3. Acquired privilege layer (AL)
4. Internal control effectiveness (IC)
5. Finding confidence (FC)

**Technical Impact (TI)**

Technical impact (TI) adalah dampak teknis yang dapat dihasilkan oleh kelemahan, dengan asumsi bahwa kelemahan tersebut berhasil dieksploitasi. Dampak teknis ini bisa dilihat pada Tabel 2.5.

Tabel 2.5 Technical Impact

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| Critical | C | 1.0 | Kontrol penuh atas perangkat lunak yang dianalisis ke titik di mana operasi sedang berlangsung |
| High | H | 0.9 | Kontrol tinggi atas perangkat lunak yang dianalisis, atau akses ke informasi penting dapat diperoleh. |
| Medium | M | 0.6 | Kontrol menengah atas perangkat lunak yang dianalisis, atau akses ke informasi yang cukup penting dapat diperoleh |
| Low | L | 0.3 | Kontrol minimal atas perangkat lunak yang dianalisis, atau hanya akses ke informasi yang relatif tidak penting dapat diperoleh. |
| None | N | 0.0 | Tidak ada dampak teknis terhadap perangkat lunak yang dianalisis sama sekali. Dengan kata lain, ini tidak menyebabkan kerentanan. |
| Default | D | 0.6 | Bobot median dari bobot Critical, High, Medium, Low, dan none. |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan. Faktor ini mungkin tidak berlaku di lingkungan dengan persyaratan jaminan yang tinggi; pengguna mungkin ingin menyelidiki setiap kelemahan menemukan bunga, terlepas dari keyakinan. |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot kustom. |

**Acquired Privilage (AP)**

The Acquired Privilege mengidentifikasi jenis hak istimewa yang diperoleh oleh seorang penyerang yang dapat berhasil mengeksploitasi kelemahan. Penjelasan bisa dilihat pada Tabel 2.6.

Tabel 2.6 Acquired Privilege

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| Administrator | A | 1.0 | Penyerang mendapatkan akses ke entitas dengan administrator, root, sistem, atau hak istimewa setara yang menyiratkan kontrol penuh atas perangkat lunak dalam analisis; atau, penyerang dapat meningkatkan hak mereka sendiri (lebih rendah) untuk administrator. |
| Partially Privileged user | P | 0.9 | Penyerang mendapatkan akses ke entitas dengan beberapa hak istimewa, tetapi tidak setara dengan administrator; atau, penyerang dapat meningkatkan hak mereka sendiri (lebih rendah) untuk pengguna yang sebagian-istimewa. Sebagai contoh, pengguna mungkin memiliki hak untuk membuat cadangan, tetapi tidak untuk mengubah konfigurasi perangkat lunak atau menginstal pembaruan. |
| Regular user | RU | 0.7 | Penyerang mendapatkan akses ke entitas yang merupakan pengguna biasa yang tidak memiliki hak istimewa; atau, penyerang dapat meningkatkan hak mereka sendiri (lebih rendah) untuk pengguna biasa. |
| Limited/ guest | L | 0.6 | Penyerang mendapatkan akses ke entitas dengan hak istimewa atau "tamu" keistimewaan yang dapat secara signifikan membatasi aktivitas yang diizinkan; atau, penyerang dapat meningkatkan hak mereka sendiri (lebih rendah) untuk tamu. Catatan: nilai ini tidak mengacu pada konsep "Guest Operating System " di host virtual |
| None | N | 0.1 | Penyerang tidak dapat memperoleh akses ke hak tambahan apa pun selain yang telah tersedia untuk penyerang. (Perhatikan bahwa nilai ini dapat berguna dalam situasi terbatas di mana penyerang dapat melarikan diri dari Sandbox atau lingkungan membatasi lainnya tetapi masih tidak dapat memperoleh hak tambahan, atau mendapatkan akses sebagai pengguna lain. |
| Default | D | 0.7 | Bobot median untuk None, Guest, Regular User, Partially-Privileged User, and Administrator |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor. |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan. |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot kustom. Perhatikan bahwa nilai quantified didukung untuk kelengkapan; Namun, karena hak istimewa dan pengguna adalah entitas diskrit, mungkin ada situasi terbatas di mana model yang diukur akan berguna. |

**Acquired Privilaged Layer(AL)**

The Acquired Privilege layer mengidentifikasi lapisan operasional dimana penyerang mendapatkan hak istimewa dengan berhasil mengeksploitasi kelemahan. Penjelasan bisa dilihat pada Tabel 2.7.

Tabel 2.7 Acquired Privilege Layer

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| Application | A | 1.0 | Penyerang memperoleh hak istimewa yang didukung dalam perangkat lunak di bawah analisis itu sendiri. (Jika perangkat lunak dalam analisis adalah bagian penting dari sistem yang mendasari, seperti sistem operasi kernel, maka nilai sistem mungkin lebih tepat.) |
| System | S | 0.9 | Penyerang memperoleh hak istimewa untuk sistem dasar atau host fisik yang digunakan untuk menjalankan perangkat lunak dalam analisis. |
| Network | N | 0.7 | Penyerang memperoleh hak istimewa untuk mengakses jaringan. |
| Enterprise infrastructure | E | 1.0 | Penyerang memperoleh hak istimewa untuk mengakses infrastruktur perusahaan, seperti router, switch, DNS, pengontrol domain, firewall, server identitas, dll. |
| Default | D | 0.9 | Bobot median Application, System, Network, and Enterprise Infrastructure. |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor. |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan.  Faktor ini mungkin tidak berlaku di lingkungan dengan persyaratan jaminan tinggi yang ingin penegakan ketat pemisahan hak istimewa, bahkan antara pengguna yang sudah istimewa. |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot kustom. Perhatikan bahwa nilai quantified didukung untuk kelengkapan; Namun, karena lapisan hak adalah entitas diskrit, mungkin ada situasi terbatas di mana model yang diukur akan berguna. |

**Internal Control Effectiveness (IC)**

Kontrol internal adalah kontrol, mekanisme perlindungan, atau mitigasi yang telah secara eksplisit dibangun ke dalam perangkat lunak (baik melalui arsitektur, Desain, atau implementasi). Efektivitas pengendalian internal mengukur kemampuan kontrol untuk merender kelemahan yang tidak dapat dimanfaatkan oleh penyerang. Sebagai contoh, rutin validasi input yang membatasi panjang input 15 karakter mungkin cukup efektif terhadap serangan XSS dengan mengurangi ukuran eksploitasi XSS yang dapat dicoba.

Bila ada beberapa kontrol internal, atau beberapa jalur kode yang dapat mencapai kelemahan yang sama, maka panduan berikut berlaku:

1. Untuk setiap jalur kode, analisis setiap kontrol internal yang ada di sepanjang jalur kode, dan pilih nilai dengan bobot terendah (yaitu kontrol internal terkuat di sepanjang jalur kode). Ini disebut nilai jalur kode.
2. Mengumpulkan semua kode path nilai.
3. Pilih nilai jalur kode yang memiliki bobot tertinggi (yaitu, adalah kontrol yang paling lemah).

Metode ini mengevaluasi setiap jalur kode dalam hal kontrol terkuat jalur kode (karena penyerang harus melewati kontrol itu), lalu memilih jalur kode terlemah (yaitu rute termudah bagi penyerang untuk mengambil). Penjelasan bisa dilihat pada Tabel 2.8.

Tabel 2.8 Internal Control Effectiveness

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| None | N | 1.0 | Tidak ada control |
| Limited | L | 0.9 | Ada metode sederhana atau pembatasan disengaja yang mungkin mencegah penyerang mengeksploitasi masalah. |
| Moderate | M | 0.7 | Mekanisme perlindungan biasanya digunakan tetapi telah diketahui keterbatasan yang mungkin dilewati dengan beberapa upaya oleh penyerang berpengetahuan. Sebagai contoh, penggunaan pengkodean entitas HTML untuk mencegah serangan XSS mungkin dilewati ketika output ditempatkan ke konteks lain seperti Cascading Style Sheet atau HTML Tag atribut. |
| Indirect (defense in depth ) | I | 0.5 | Kontrol tidak secara khusus melindungi terhadap eksploitasi kelemahan, tetapi secara tidak langsung mengurangi dampak ketika serangan berhasil diluncurkan, atau sebaliknya membuatnya lebih sulit untuk membangun eksploitasi fungsional. Sebagai contoh, rutin validasi mungkin secara tidak langsung membatasi ukuran input, yang mungkin membuat sulit bagi penyerang untuk membangun muatan untuk XSS atau SQL Injeksi serangan. |
| Best available | B | 0.3 | Kontrol terbaik yang dimiliki, meskipun mungkin memiliki beberapa keterbatasan yang dapat diatasi oleh penyerang terampil. Sebagai contoh, metode submit ganda untuk perlindungan CSRF dianggap salah satu yang terkuat yang tersedia, tetapi dapat dikalahkan dalam hubungannya dengan perilaku fungsi tertentu yang dapat membaca header HTTP mentah. |
| Complete | C | 0.0 | Kontrol yang sepenuhnya efektif terhadap kelemahan, yaitu, tidak ada bug atau kerentanan, dan tidak ada konsekuensi yang merugikan dari eksploitasi masalah. Sebagai contoh, operasi buffer salinan yang memastikan bahwa buffer tujuan selalu lebih besar daripada sumber (ditambah ekspansi tidak langsung dari ukuran sumber asli) tidak akan menyebabkan overflow. |
| Default | D | 0.6 | Bobot median dari Complete, Best-Available, Indirect, Moderate, Limited, and None. |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor. |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan. |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot kustom. |

**Finding Confidence (FC)**

Finding Confidence adalah keyakinan bahwa masalah yang dilaporkan adalah kelemahan yang dapat dimanfaatkan oleh penyerang. Penjelasan bisa dilihat pada Tabel 2.9.

Tabel 2.9 Finding Confidence

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| Proven true | T | 1.0 | Kelemahan dapat dicapai oleh penyerang. |
| Proven locally true | LT | 0.8 | Kelemahan terjadi dalam fungsi individu atau komponen desain yang bergantung pada fungsi itu, tetapi penyerang memliki keterbatasan menjangkau fungsi yang tidak diketahui atau tidak hadir. Sebagai contoh, fungsi utilitas mungkin membangun query database tanpa encoding input, tetapi jika hanya disebut dengan string konstan, temuan secara lokal benar |
| Proven false | F | 0.0 | Temuan ini keliru (yaitu temuan positif palsu dan tidak ada kelemahan), dan/atau tidak ada peran penyerang mungkin. |
| Default | D | 0.8 | Bobot median dari Proven True, Proven Locally True, and Proven False. |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor. |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan. |
| Quantified |  |  | Faktor ini mungkin tidak berlaku di lingkungan dengan persyaratan jaminan yang tinggi; pengguna mungkin ingin menyelidiki setiap kelemahan menemukan bunga, terlepas dari keyakinan |

##### Attack Surface Metric Grup

The Attack Surface metric group terdiri dari faktor berikut:

1. Required Privilege (RP)
2. Required Privilege Layer (RL)
3. Access Vector (AV)
4. Authentication Strength (AS)
5. Level of Interaction (IN)
6. Deployment Scope (SC)
7. **Required Privilege (RP)**

The Required Privilege mengidentifikasi jenis hak istimewa yang harus dimiliki penyerang untuk mencapai kode/ fungsi yang mengandung kelemahan. Penjelasan bisa dilihat pada Tabel 2.10.

Tabel 2.10 Required Privilege

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| None | N | 1.0 | Tidak ada hak istimewa yang diperlukan. Misalnya, mesin telusur berbasis web mungkin tidak memerlukan hak istimewa apa pun untuk entitas untuk memasukkan istilah penelusuran dan melihat hasil. |
| Limited / guest | L | 0.9 | Entitas telah dibatasi atau "tamu" hak istimewa yang dapat secara signifikan membatasi aktivitas yang diizinkan; entitas mungkin dapat mendaftar atau membuat akun baru tanpa persyaratan khusus atau bukti identitas. Misalnya, blog web dapat memungkinkan peserta membuat nama pengguna dan mengirimkan alamat email yang valid sebelum memasukkan komentar. Catatan: nilai ini tidak mengacu pada konsep “Guest Operating System “dalam host virtual. |
| Regular user | RU | 0.7 | Entitas adalah pengguna biasa yang tidak memiliki hak istimewa. |
| Partially privileged use | P | 0.6 | Entitas adalah pengguna yang valid dengan beberapa hak istimewa, namun tidak setara dengan administrator. Sebagai contoh, pengguna mungkin memiliki hak untuk membuat cadangan, tetapi tidak untuk mengubah konfigurasi perangkat lunak atau menginstal pemutakhiran. |
| Administrator | A | 0.1 | Entitas memiliki administrator, root, sistem, atau hak istimewa setara yang menyiratkan kontrol penuh atas perangkat lunak atau OS yang mendasari. |
| Default | D | 0.7 | Bobot median dari None, Limited, Regular User, Partially-Privileged User, and Administrator |
| Uknown | UK | 0.5 | Tidak ada, terbatas, pengguna biasa, pengguna sebagian-istimewa, dan administrator |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan.  Faktor ini mungkin tidak berlaku di lingkungan dengan persyaratan jaminan tinggi yang ingin penegakan ketat pemisahan hak istimewa, bahkan antara pengguna yang sudah istimewa. |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot kustom. Perhatikan bahwa nilai quantified didukung untuk kelengkapan; Namun, karena hak istimewa dan pengguna adalah entitas diskrit, mungkin ada situasi terbatas di mana model yang diukur akan berguna. |

1. **Required Privilege Layer (RL)**

The Required Privilege Layer mengidentifikasi lapisan operasional dimana penyerang harus memiliki hak istimewa untuk mencoba menyerang kelemahan. Penjelasan bisa dilihat pada Tabel 2.11.

Tabel 2.11 Required Privilege Layer

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| Application | A | 1.0 | Penyerang harus memiliki hak yang didukung dalam perangkat lunak di bawah analisis itu sendiri. (Jika perangkat lunak dalam analisis adalah bagian penting dari sistem yang mendasari, seperti sistem operasi kernel, maka nilai sistem mungkin lebih tepat.) |
| System | S | 0.9 | Penyerang harus memiliki hak untuk sistem dasar atau host fisik yang digunakan untuk menjalankan perangkat lunak di bawah analisis |
| Network | N | 0.7 | Penyerang harus memiliki hak istimewa untuk mengakses jaringan. |
| Enterprise infrastructure | E | 1.0 | Penyerang harus memiliki hak istimewa pada bagian penting dari infrastruktur perusahaan, seperti router, switch, DNS, pengontrol domain, firewall, server identitas, dll. |
| Default | D | 0.9 | Bobot median dari Application, System, Network, and Enterprise Infrastructure. |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor. |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan. Faktor ini mungkin tidak berlaku di lingkungan dengan persyaratan jaminan tinggi yang ingin penegakan ketat pemisahan hak istimewa, bahkan antara pengguna yang sudah istimewa. |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot kustom. Perhatikan bahwa nilai quantified didukung untuk kelengkapan; Namun, karena lapisan hak adalah entitas diskrit, mungkin ada situasi terbatas di mana model yang diukur akan berguna |

1. **Acces Vector (AV)**

Faktor ini dapat diukur dengan bobot kustom. Perhatikan bahwa nilai quantified didukung untuk kelengkapan; Namun, karena lapisan hak adalah entitas diskrit, mungkin ada situasi terbatas di mana model yang Vektor akses mengidentifikasi saluran melalui mana penyerang harus berkomunikasi untuk mencapai kode atau fungsionalitas yang berisi kelemahan. Perhatikan bahwa nilai ini sangat mirip dengan yang digunakan dalam CVSS, kecuali CWSS membedakan antara akses fisik dan akses lokal (Shell/account).

Meskipun ada hubungan dekat antara Access Vector dan required Privilege Layer, keduanya berbeda. Sebagai contoh, penyerang dengan “fisik " akses ke router mungkin dapat mempengaruhi layer Network atau Enterprise. Penjelasan bisa dilihat pada Tabel 2.12.

Tabel 2.12 Access Vector

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| Internet | I | 1.0 | Penyerang harus memiliki akses ke internet untuk mencapai kelemahan. |
| Intranet | R | 0.8 | Penyerang harus memiliki akses ke intranet perusahaan yang terlindung dari akses langsung dari internet, misalnya dengan menggunakan firewall, tetapi jika intranet dapat diakses oleh sebagian besar anggota perusahaan. |
| Private network | V | 0.8 | Penyerang harus memiliki akses ke jaringan pribadi yang hanya dapat diakses oleh pihak terpercaya |
| Adjacent network | A | 0.7 | Penyerang harus memiliki akses ke antarmuka fisik ke jaringan. Contoh jaringan lokal mencakup subnet IP lokal, Bluetooth, IEEE 802, 11, dan segmen Ethernet lokal. |
| Local | L | 0.5 | Penyerang harus memiliki akun interaktif, antarmuka local (shell) secara langsung dengan sistem operasi yang mendasari. |
| Physical | P | 0.2 | Penyerang harus memiliki akses fisik ke sistem yang menjalankan perangkat lunak, atau dapat berinteraksi dengan sistem melalui antarmuka seperti USB, CD, keyboard, mouse, dll. |
| Default | D | 0.75 | Bobot median dari nilai yang relevan |
| Uknown | U | 0.5 |  |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan. |
| Quantified | Q |  | Faktor-nya dapat diukur dengan bobot kustom. Perhatikan bahwa nilai quantified didukung untuk kelengkapan; Namun, karena vektor akses adalah entitas diskrit, mungkin ada situasi terbatas di mana model yang diukur akan berguna. |

1. **Authentication Strength (AS)**

Kekuatan otentikasi mencakup kekuatan rutin otentikasi yang melindungi kode/fungsionalitas yang berisi kelemahan. Penjelasan bisa dilihat pada Tabel 2.13.

Ketika ada beberapa otentikasi rutin, atau beberapa jalur kode yang dapat mencapai kelemahan yang sama, maka panduan berikut berlaku:

1. Untuk setiap jalur kode, analisis setiap rutinitas autentikasi yang ada di sepanjang jalur kode, dan pilih nilai dengan bobot terendah (yaitu, rutinitas autentikasi terkuat di sepanjang jalur kode). Ini disebut nilai jalur kode.
2. Mengumpulkan semua kode path nilai.
3. Pilih nilai jalur kode yang memiliki bobot tertinggi (yaitu, berisi rutinitas yang paling lemah).

Tabel 2.13 Authentication Strength

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| Strong | S | 0.7 | Kelemahan membutuhkan metode yang paling kuat yang tersedia untuk mengikat entitas ke identitas dunia nyata, seperti token berbasis perangkat keras, dan/atau autentikasi multifaktor. |
| Moderate | M | 0.8 | Kelemahan memerlukan otentikasi menggunakan metode yang cukup kuat, seperti penggunaan sertifikat dari otoritas yang tidak tepercaya, otentikasi berbasis pengetahuan, atau password satu kali. |
| Weak | W | 0.9 | Kelemahan membutuhkan metode otentikasi sederhana dan lemah yang mudah dikompromikan dengan menggunakan spoofing, Dictionary, atau serangan replay, seperti password statis |
| None | N | 1.0 | Kelemahan tidak memerlukan otentikasi sama sekali. |
| Default | D | 0.85 | Nilai median untuk Strong, Moderate, Weak, and None |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor. |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan. Ini mungkin tidak berlaku di lingkungan dengan persyaratan jaminan tinggi yang berusaha untuk menghilangkan semua kelemahan. |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot kustom. |

1. **Level Of Interaction (IN)**

Level interaksi mencakup tindakan yang diperlukan oleh korban manusia untuk memungkinkan serangan yang sukses terjadi. Penjelasan bisa dilihat pada Tabel 2.14.

Tabel 2.14 Level Of Interaction

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| Automated | A | 1.0 | Tidak ada interaksi manusia yang diperlukan |
| Typical/limited | T | 0.9 | Penyerang harus meyakinkan pengguna untuk melakukan tindakan yang umum atau dianggap sebagai "normal" dalam operasi produk khas. Misalnya, mengeklik tautan di laman web, atau melihat pratinjau badan email, adalah perilaku yang umum. |
| Moderate | M | 0.8 | Penyerang harus meyakinkan pengguna untuk melakukan tindakan yang mungkin tampak mencurigakan untuk pengguna yang berhati-hati dan berpengetahuan. Misalnya: pengguna harus menerima peringatan yang menunjukkan muatan penyerang mungkin berisi konten berbahaya. |
| Opportunistic | O | 0.3 | Penyerang tidak dapat secara langsung mengendalikan atau mempengaruhi korban, dan hanya dapat secara pasif memanfaatkan kesalahan atau tindakan orang lain. |
| High | H | 0.1 | Sejumlah besar rekayasa sosial diperlukan, mungkin termasuk ketidaktahuan atau kelalaian pada bagian dari korban. |
| No interaction | NI | 0.0 | Tidak ada interaksi yang mungkin, bahkan tidak oistically; ini biasanya akan membuat kelemahan sebagai "bug" bukannya mengarah ke kerentanan. Karena CWSS adalah untuk keamanan, berat adalah 0. |
| Default | D | 0.55 | Nilai median dari Automated, Limited, Moderate, Opportunistic, High, and No interaction. |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor. |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan. |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot kustom. |

1. **Deployment Scope (SC)**

Deployment scope mengidentifikasi apakah kelemahan hadir dalam semua pengembangan perangkat lunak, atau jika dibatasi untuk subset dari platform dan/atau konfigurasi. Penjelasan bisa dilihat pada Tabel 2.15.

Tabel 2.15 Deployment Scope

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| All | A | 1.0 | Tersedia di semua platform atau konfigurasi |
| Moderate | M | 0.9 | Tersedia dalam platform umum atau konfigurasi |
| Rare | R | 0.5 | Hanya ada di platform atau konfigurasi langka |
| Potentially reachable | P | 0.1 | Berpotensi dicapai tetapi semua jalur kode yang saat ini aman, dan/atau kelemahan dalam kode mati |
| Default | D | 0.7 | Bobot median untuk nilai RAMP (Rare, All, Moderate, Potentially Reachable). |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor. |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan. |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot kustom. Pengguna mungkin tahu berapa persentase perangkat lunak yang dikirim (atau didukung) berisi bug ini. |

##### Environmental Metric Grup

The Environmental metric grup terdiri dari beberapa faktor berikut:

1. Business Impact (BI)
2. Likelihood of Discovery (DI)
3. Likelihood of Exploit (EX)
4. External Control Effectiveness (EC)
5. Prevalence (P)
6. **Business Impact (BI)**

Business impact menggambarkan dampak potensial terhadap bisnis atau misi jika kelemahan dapat berhasil dieksploitasi. Penjelasan bisa dilihat pada Tabel 2.16.

Tabel 2.16 Business Impact

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| Critical | C | 1.0 | Bisnis/misi bisa gagal. |
| High | H | 0.9 | Operasi bisnis/misi akan terpengaruh secara signifikan. |
| Medium | M | 0.6 | Bisnis/misi akan terpengaruh, tetapi tanpa kerusakan yang luas untuk operasi rutin. |
| Low | L | 0.3 | Dampak minimal pada bisnis/misi. |
| None | N | 0.0 | Tidak ada dampak. |
| Default | D | 0.6 | Bobot median untuk Critical, High, Medium, Low, and None. |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor. |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan. Faktor ini mungkin tidak berlaku dalam konteks di mana dampak bisnis tidak relevan, atau jika dampaknya sedang dinilai dan dipertimbangkan dalam proses analisis yang berada di luar Skor CWSS itu sendiri |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot kustom. Beberapa organisasi mungkin memiliki pengukuran spesifik untuk nilai bisnis aset, misalnya, yang dapat diintegrasikan ke dalam pengukuran ini. |

1. **Likelihood Of Discovery (DI)**

Likelihood of discover adalah kemungkinan bahwa seorang penyerang dapat menemukan kelemahan. Penjelasan bisa dilihat pada Tabel 2.17.

Tabel 2.17 Likelihood Of Discovery

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| High | H | 1.0 | Sangat mungkin bahwa seorang penyerang dapat menemukan kelemahan dengan cepat dan dengan sedikit usaha menggunakan teknik sederhana, tanpa akses ke kode sumber atau artefak lain yang menyederhanakan deteksi kelemahan. |
| Medium | M | 0.6 | Seorang penyerang mungkin dapat menemukan kelemahan, tetapi akan memerlukan keterampilan tertentu untuk melakukannya, mungkin memerlukan akses kode sumber atau pengetahuan reverse engineering. Ini mungkin memerlukan beberapa waktu investasi untuk menemukan masalah. |
| Low | L | 0.2 | Seorang penyerang tidak mungkin untuk menemukan kelemahan tanpa keterampilan yang sangat khusus, akses ke kode sumber (atau setara), dan investasi waktu yang besar. |
| Default | D | 0.6 | Nilai median dari High, Medium, and Low values. |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan. Ini mungkin tidak berlaku ketika pencetak gol mengasumsikan bahwa semua kelemahan akan ditemukan oleh penyerang. |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot kustom. |

1. **Likelihood Of Exploit (EX)**

Likelihood of exploit adalah kemungkinan bahwa, jika kelemahan ditemukan, seorang penyerang dengan hak istimewa/otentikasi/akses yang diperlukan akan dapat berhasil mengeksploitasi itu. Penjelasan bisa dilihat pada Tabel 2.18.

Tabel 2.18 Likelihood Of Exploit

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| High | H | 1.0 | Sangat mungkin bahwa seorang penyerang akan menargetkan keberhasilan mengeksploitasi dengan mudah. |
| Medium | M | 0.6 | Seorang penyerang mungkin akan menargetkan kelemahan ini berhasil, tetapi kemungkinan keberhasilan dapat bervariasi, atau memerlukan beberapa upaya untuk berhasil. |
| Low | L | 0.2 | Seorang penyerang mungkin tidak akan menargetkan kelemahan ini, atau bisa memiliki kemungkinan sangat terbatas pada keberhasilan. |
| None | N | 0.0 | Seorang penyerang tidak memiliki kesempatan untuk sukses; yaitu, masalahnya adalah "bug" karena tidak ada peran penyerang, dan tidak ada manfaat bagi penyerang. |
| Default | D | 0.6 | Nilai median dari High, Medium, and Low. “none” diabaikan dengan harapan bahwa beberapa kelemahan temuan akan dicetak dengan menggunakan nilai, dan termasuk dalam perhitungan median akan mengurangi berat ke tingkat non-intuitif. |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor. |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan.  Sebagai contoh, pencetak gol mungkin ingin berasumsi bahwa penyerang dapat mengeksploitasi kelemahan apa pun yang dapat mereka temukan, atau bersedia menginvestasikan sumber daya yang signifikan untuk mengatasi kemungkinan hambatan untuk mengeksploitasi keberhasilan |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot khusus |

1. **External Control Efectiveness (EC)**

Efektivitas kontrol eksternal adalah kemampuan kontrol atau Mitigations di luar perangkat lunak yang dapat membuat kelemahan lebih sulit bagi seorang penyerang untuk mencapai dan/atau memicu. Sebagai contoh, alamat Randomisasi tata letak ruang (ASLR) dan teknologi serupa mengurangi, tetapi tidak menghilangkan, kemungkinan keberhasilan untuk serangan kelebihan buffer. Namun, aslr tidak langsung dipakai dalam perangkat lunak itu sendiri.

Bila ada beberapa kontrol eksternal, atau beberapa jalur kode yang dapat mencapai kelemahan yang sama, maka panduan berikut berlaku:

1. Untuk setiap jalur kode, menganalisa setiap kontrol eksternal yang ada di sepanjang jalur kode, dan memilih nilai dengan bobot terendah (yaitu, kontrol eksternal terkuat di sepanjang jalur kode). Ini disebut nilai jalur kode.
2. Mengumpulkan semua kode path nilai.
3. Pilih nilai jalur kode yang memiliki bobot tertinggi (yaitu, adalah kontrol yang paling lemah). Pilih nilai jalur kode yang memiliki bobot tertinggi (yaitu, adalah kontrol yang paling lemah).

Metode ini mengevaluasi setiap jalur kode dalam hal kontrol terkuat jalur kode (karena penyerang harus melewati kontrol itu), lalu memilih jalur kode terlemah (yaitu rute termudah bagi penyerang untuk mengambil). Penjelasan bisa dilihat pada Tabel 2.19.

Tabel 2.19 External Control Efectiveness

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| None | N | 1.0 | Tidak ada kontrol |
| Limited | L | 0.9 | Ada metode sederhana atau pembatasan yang mungkin mencegah penyerang mengeksploitasi masalah. |
| Moderate | M | 0.7 | Mekanisme perlindungan biasanya digunakan tetapi telah diketahui batasannya yang mungkin dilewati dengan beberapa upaya oleh penyerang berpengetahuan. |
| Indirect(defense in depth ) | I | 0.5 | Kontrol tidak secara khusus melindungi terhadap eksploitasi kelemahan, tetapi secara tidak langsung mengurangi dampak ketika serangan berhasil diluncurkan, atau sebaliknya membuatnya lebih sulit untuk membangun eksploitasi fungsional.  Sebagai contoh, alamat Randomisasi tata letak ruang (ASLR) dan teknologi serupa mengurangi, tetapi tidak menghilangkan, kemungkinan keberhasilan dalam serangan kelebihan buffer. Karena respons biasanya untuk keluar dari proses, hasilnya masih merupakan penyangkalan layanan. |
| Best available | B | 0.3 | Kontrol terbaik saat ini, meskipun mungkin memiliki beberapa keterbatasan yang dapat diatasi oleh penyerang terampil, ditentukan, mungkin memerlukan kehadiran kelemahan lain. Misalnya, Transport Layer Security (TLS)/SSL 3 beroperasi di sebagian besar web, dan metode yang lebih kuat umumnya tidak tersedia karena masalah kompatibilitas. |
| Complete | C | 0.1 | Kontrol sepenuhnya efektif terhadap kelemahan, yaitu, tidak ada bug atau kerentanan, dan tidak ada konsekuensi yang merugikan dari eksploitasi masalah. Misalnya, lingkungan sandbox mungkin membatasi operasi akses file ke satu direktori kerja, yang akan melindungi dari eksploitasi jalur traversal. Bobot bukan nol digunakan untuk mencerminkan kemungkinan bahwa kontrol eksternal dapat dihapus secara tidak sengaja di masa mendatang, misalnya jika perubahan lingkungan perangkat lunak. |
| Default | D | 0.6 | Median dari Complete, Best-Available, Indirect, Moderate, Limited, and None. |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot kustom. |

1. **Prevalence (P)**

Prevalensi dari temuan mengidentifikasi seberapa sering kelemahan jenis ini muncul dalam perangkat lunak. Penjelasan bisa dilihat pada Tabel 2.20.

Faktor ini dimaksudkan untuk digunakan dalam Skor Generalized kelas kelemahan, seperti pengembangan kustom Top-N daftar kelemahan

Tabel 2.20 Prevalence

|  |  |  |  |
| --- | --- | --- | --- |
| **Nilai** | **Kode** | **Weight** | **Deskripsi** |
| Widespread | W | 1.0 | Kelemahan yang ditemukan di sebagian besar atau semua perangkat lunak dalam lingkungan terkait, dan mungkin terjadi beberapa kali dalam paket perangkat lunak yang sama. |
| High | H | 0.9 | Kelemahan yang dihadapi sangat sering, tetapi tidak luas. |
| Common | C | 0.8 | Kelemahan yang dihadapi secara berkala. |
| Limited | L | 0.7 | Kelemahan yang dihadapi jarang, atau tidak pernah. |
| Default | D | 0.85 | Median dari Limited, Common, High, and Widespread. |
| Uknown | UK | 0.5 | Tidak ada informasi yang cukup untuk memberikan nilai untuk faktor ini. Analisis lebih lanjut mungkin diperlukan. Di masa depan, nilai yang berbeda mungkin dipilih, yang dapat mempengaruhi Skor. |
| Not Applicable | NA | 1.0 | Faktor ini secara sengaja diabaikan dalam perhitungan Skor karena tidak relevan dengan bagaimana skorer memprioritaskan kelemahan. Ketika melakukan penilaian yang ditargetkan terhadap temuan spesifik kelemahan dalam aplikasi, prevalensi biasanya diharapkan tidak relevan, karena aplikasi individu dan teknik analisis menentukan seberapa sering kelemahan terjadi, dan banyak metode penilaian agregat akan menghasilkan Skor yang lebih besar jika ada lebih banyak kelemahan. |
| Quantified | Q |  | Faktor ini dapat diukur dengan bobot kustom. Data prevalensi yang tepat mungkin tersedia dalam kasus penggunaan terbatas, asalkan pengguna melacak data kelemahan pada tingkat granularity yang rendah. Misalnya, pengembang mungkin melacak kelemahan di seluruh rangkaian produk, atau vendor audit kode dapat mengukur prevalensi dari perangkat lunak yang dianalisis di seluruh basis pelanggan. Dalam versi CWSS sebelumnya, prevalensi dihitung berdasarkan data pemungutan suara mentah yang dikumpulkan untuk 2010 Top 25, yang menggunakan nilai diskrit (kisaran 1 sampai 4) yang kemudian disesuaikan dengan kisaran 1-ke-10. |

#### Formula Skor CWSS

Skor CWSS versi 1.0 berkisar antara 0 dan 100. Formula skor CWSS bisa dilihat pada persamaan

##### Base Finding Subscore

Base finding Subscore dapat dilihat pada persamaan

Definisi f (Technical impact) digunakan untuk memastikan bahwa jika Technical Impact adalah 0, bahwa faktor tambahan lainnya tidak secara sengaja menghasilkan skor bukan nol. Potensi maksimum Base finding subscore adalah 100.

##### Attack Surface Subsccore

Attack surface subscore dapat dilihat pada persamaan

Kombinasi dari required privilege / acces membuat 60 % dari Attack surface subscore, deployment scope 20 %, interaction 15 % dan authentication 5%. Authentication tidak diberikan banyak focus, dengan asumsi bahwa bukti identitas yang kuat tidak akan secara signifikan menghalangi penyerang untuk mencoba mengeksploitasi kerentanan. Attack surface menghasilkan kisaran nilai antara 0 dan 10, yang kemudian dibagi dengan 100.

##### Environmental Subscore

Environmental subscore dihitung sebagai berikut:

Definisi f (Business impact) digunakan untuk memastikan bahwa jika Business Impact adalah 0, bahwa faktor tambahan lainnya tidak secara sengaja menghasilkan skor bukan nol.

### Peninjauan Aplikasi Sejenis

Peninjauan aplikasi sejenis dilakukan untuk membandingkan aplikasi yang sedang dibangun dengan aplikasi sejenis lainnya dengan harapan dilakukannya perbandingan dapat menjadikan aplikasi yang hendak dibangun menjadi lebih baik.

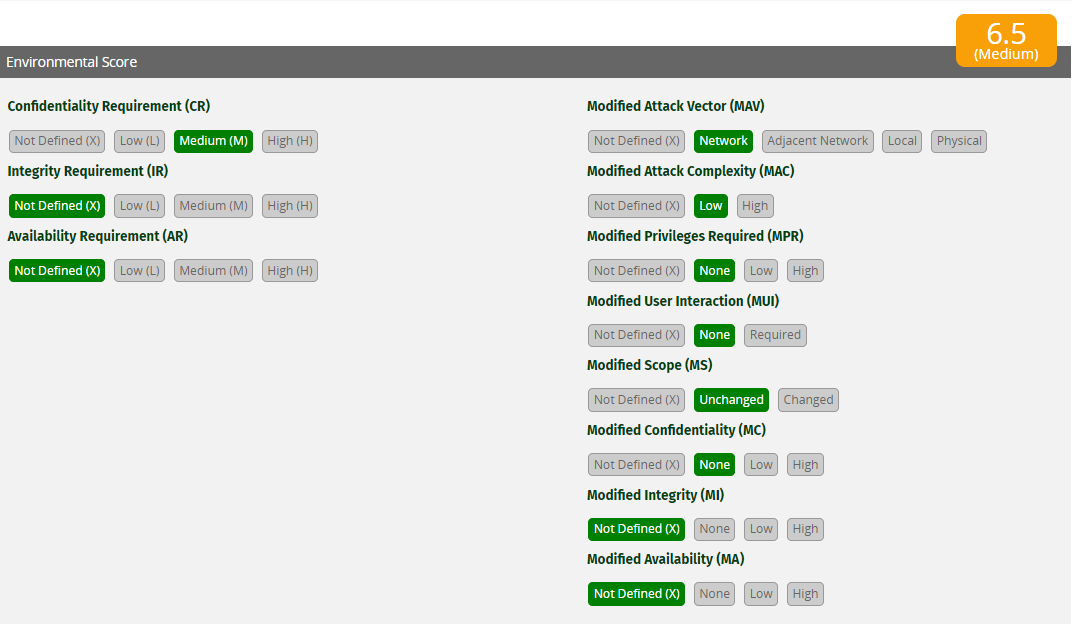
Aplikasi penghitung kelemahan umum software CWSS ini setelah penulis cari belum menemukan penelitian yang sama, tetapi penelitian terkait aplikasi yang sejenis pernah dilakukan (Pawelana, Juliharta, & Suwirmayanti, 2015), pada penelitian tersebut membuat aplikasi perhitungan kerentanan menggunakan metode CVSS (Common Vulnerability Scoring System) Version 2 berbasis desktop dengan tools visual basic 2008. Penelitian tersebut diharapkan supaya tim manajemen IT dapat lebih mudah mengambil keputusan untuk mengatasi kerentanan yang ditemukan, baik dari platform software maupun hardware yang berbeda-beda. Selain itu FIRST.Org,Inc selaku organisasi nirlaba yang mengelola CVSS juga menyediakan aplikasi CVSS berbasis website. Aplikasi CVSS berbasis web ini bisa dilihat pada Gambar 2.5, Gambar 2.6, Gambar 2.7.



Gambar 2.5 Base Score Aplikasi CVSS



Gambar 2.6 Temporal Score Aplikasi CVSS



Gambar 2.7 Environmental Score Aplikasi CVSS

CWSS berbeda tetapi bukan pesaing sistem penilaian kerentanan umum CVSS (Common Vulnerability Scoring System). Mereka memiliki peran yang berbeda, dan dapat dimanfaatkan bersama.

Skor CWSS dapat dihitung dalam skenario awal, informasi rendah, banyak faktor bersifat "temporal", terlepas dari kelompok mana mereka berada juga, skor ini cenderung berubah karena analisis lebih lanjut menghasilkan lebih banyak informasi tentang kelemahan. Perbedaan penggunaan CWSS dengan CVSS bisa dilihat pada Tabel 2.21.

Tabel 2.21 CWSS VS CVSS

|  |  |
| --- | --- |
| **CWSS** | **CVSS** |
| CWSS dapat diterapkan lebih awal dalam proses, sebelum kerentanan ditemukan | CVSS mengasumsikan bahwa kerentanan telah ditemukan dan diverifikasi. |
| Penilaian CWSS memiliki dukungan bawaan untuk informasi yang tidak lengkap. Di dalam CWSS, penilaian mungkin diperlukan sebelum kelemahan bahkan diketahui berkontribusi terhadap kerentanan | Penilaian CVSS tidak memperhitungkan informasi yang tidak lengkap |
| CWSS menghitung skor kelemahan umum pada software | CVSS menghitung skor vulnerability pada software |
| Rentang skor CWSS antara 0 - 100 | Rentang skor CVSS antara 0 -10 |
| Tidak menyediakan rating skor | Menyediakan rating skor |

## BAB III METODOLOGI PENELTIAN

### Analisis Kebutuhan Sistem

Dalam pembuatan “Aplikasi Web Penghitung Skor Kelemahan Umum Software Menggunakan Skor CWSS (Common Weakness Scoring System)” terdapat beberapa analisis kebutuhan, yaitu analisis kebutuhan perangkat lunak, analisis kebutuhan perangkat keras, analisis kebutuhan masukan, analisis kebutuhan proses, analisis kebutuhan keluaran dan analisis kebutuhan antarmuka.

#### Analisis Kebutuhan Perangkat Lunak

Perangkat lunak yang dibutuhkan pada pembangunan aplikasi agar dapat berjalan dengan baik, diantaranya sebagai berikut:

1. Sistem operasi : Windows 10 (64 bit)
2. Browser : Opera versi 63.0.3368.71, Microsoft Edge versi 44.18362.329.0 dan Google Chrome versi 76.0.3809.132
3. Text editor : Sublime Text 3 (64 bit)
4. Tools : Snipping Tool, Balsamiq Mockup 3
5. Web Server : Xampp

#### Analisis Kebutuhan Perangkat Keras

Selain menggunakan perangkat lunak tentunya juga dibutuhkan perangkat keras untuk mendukung kinerja dari perangkat lunak tersebut. Adapun perangkat keras yang digunakan untuk pembuatan aplikasi ini adalah sebagai berikut:

1. Laptop HP dengan prosesor Intel Core i5
2. Ram 4 GB
3. VGA AMD Radeon
4. Keyboard
5. Mouse
6. Hardisk 500 GB

#### Analisis Kebutuhan Masukan

Analisis kebutuhan masukan dilakukan untuk mengidentifikasi masukan-masukan apa saja yang diperlukan sistem untuk menjalankan fungsionalitas sistem sebagaimana mestinya. Kebutuhan masukan yang digunakan oleh pengguna di dalam sistem adalah:

1. Pengguna memilih menu pada aplikasi
2. Pengguna mengisi dan memilih jawaban pada halaman penilaian aplikasi

#### Analisis Kebutuhan Proses

Analisis kebutuhan proses merupakan analisis untuk mengetahui proses-proses apa saja yang diperlukan di dalam sistem. Berikut merupakan perincian kebutuhan proses yang terdapat di dalam sistem:

1. Proses perhitungan penilaian skor kelemahan umum software
2. Proses mengubah hasil menjadi laporan PDF

#### Analisis Kebutuhan keluaran

Analisis kebutuhan keluaran dilakukan untuk mengidentifikasi keluaran-keluaran apa saja yang akan ditampilkan kepada pengguna. Berikut merupakan perincian kebutuhan keluaran dari aplikasi web penghitung kelemahan umum software menggunakan skor CWSS (Common Weakness Scoring System):

1. Informasi skor penilaian kelemahan umum software
2. Informasi laporan hasil penilaian kelemahan umum software dalam bentuk PDF

#### Analisis Kebutuhan Antarmuka

Analisis kebutuhan antarmuka dilakukan untuk mengetahui kebutuhan antarmuka dari sistem yang akan dibangun. Berikut merupakan perincian kebutuhan antarmuka di dalam sistem:

1. Antarmuka halaman Home

Halaman ini yang muncul saat pengguna menjalankan aplikasi. Halaman ini berisi informasi tentang gambaran umum CWE (Common Weakness Enumeration) selaku penyedia metode soring yang dipakai dalam penelitian.

1. Antarmuka halaman Deskripsi

Halaman ini berisi informasi detail tentang metode scoring yang dipakai dalam penelitian.

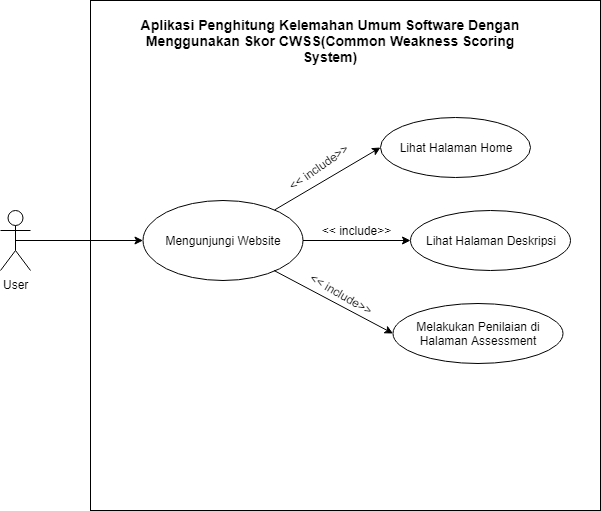
1. Antarmuka halaman Assessment

Halaman ini berisi form penilaian yang harus diisi oleh pengguna untuk mendapat skor kelemahan umum software. Pada halaman ini pengguna bisa melihat laporan hasil skor yang dapat diunduh dalam bentuk PDF

### Use Case Diagram

Use Case Diagram merupakan rangkaian tindakan yang dilakukan oleh sistem, aktor mewakili user atau sistem lain yang berinteraksi dengan sistem yang dimodelkan (Satzinger, Jackson, & Burd, 2011). Adanya use case ini dapat memberikan informasi mengenai fungsi-sfungsi apa saja yang terdapat pada sistem yang dibuat.

Diagram use case digunakan untuk mendeskripsikan jenis fungsionalitas dan aktor yang berperan dalam sistem. Dalam use case sistem ini terdapat satu aktor dalam sistem ini yaitu user/pengguna, yang berperan sebagai developer. Pengguna sistem dapat melakukan akses melihat halaman Home, Deskripsi dan melakukan penilaian. Berikut adalah use case diagram aplikasi web penghitung kelemahan umum *software* dapat dilihat pada **Error! Reference source not found.**.

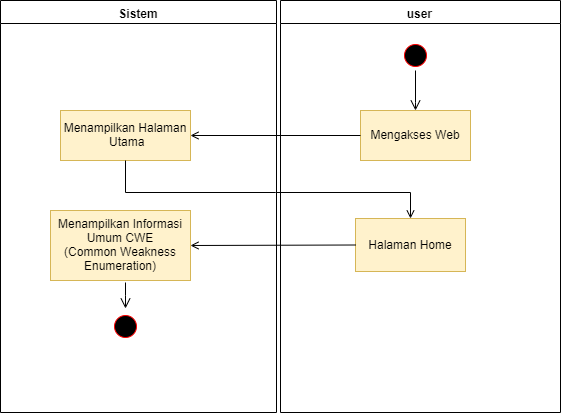
  
Gambar 3.1 Use Case Diagram

### Activity Diagram

Menurut (Rosa & Shalahuddin, 2011) diagram aktivitas atau activity diagram menggambarkan work flow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Activity Diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang. Diagram ini menunjukan langkah-langkah dalam proses kerja sistem yang dibuat.

#### Activity Diagram Lihat Halaman Home

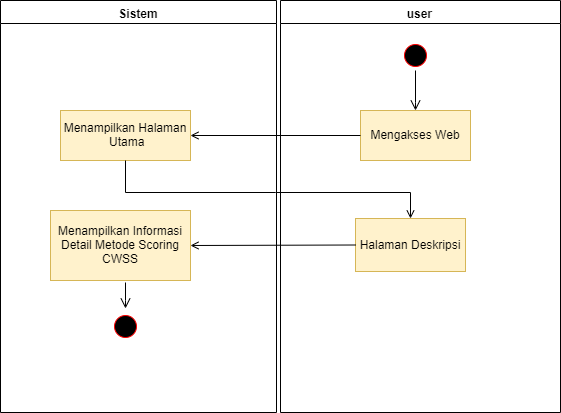
Diagram ini menggambarkan proses dimana user mengakses web kemudian sistem menampilkan halaman utama yang berisi informasi umum tentang CWE (Common Weakness Enumeration), selaku komunitas penyedia metode scoring yang dipakai pada penelitian ini. Berikut adalah activity diagram lihat halaman home pada Gambar 3.2.



Gambar 3.2 Activity Diagram Lihat Halaman Home

#### Activity Diagram Lihat Halaman Deskripsi

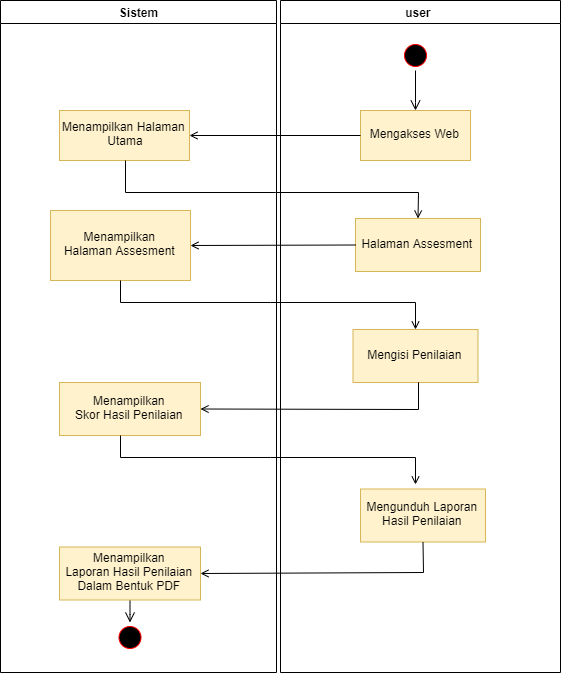
Diagram ini menggambarkan proses saat user ingin melihat halaman deskripsi. User mengakses web kemudian pilih menu deskripsi kemudian sistem akan menampilkan halaman yang berisi detail informasi mengenai metode scoring CWSS yang dipakai dalam penelitian ini. Berikut adalah activity diagram lihat halaman deskripsi pada Gambar 3.3.



Gambar 3.3 Activity Diagram Lihat Halaman Deskripsi

#### Activity Diagram Melakukan Penilaian Di Halaman Assesment

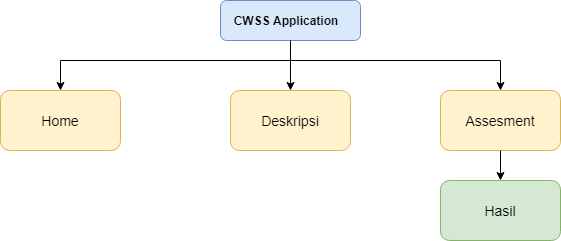
Diagram ini menggambarkan proses user melakukan penilaian kelemahan umum software. User mengakses web kemudian pilih menu assessment kemudian sistem akan menampilkan halaman berisi penilaian yang harus diisi oleh user, sistem akan menampilkan skor hasil penilaian dan laporan hasil penilaian dalam file bentuk PDF. Berikut adalah activity diagram melakukan penilaian di halaman assessment pada Gambar 3.4.



Gambar 3.4 Activity Diagram Melakukan Penilaian Di Halaman Assesment

### Diagram HIPO (Hirarchy Plus Input Process Output)

HIPO (Hirarchy Plus Input Process Output) adalah alat bantu untuk membuat suatu spesifikasi program yang merupakan struktur yang berisi tentang diagram-diagram, dimana di dalam program tersebut berisi input yang diproses dan menghasilkan output. Diagram HIPO memiliki tujuan utama yaitu untuk menjelaskan fungsi apa saja yang terdapat pada struktur sebuah program. Tujuannya untuk menjelaskan alur kerja sebuah program dari mulai input sampai output secara detail. Diagram HIPO aplikasi penghitung skor kelemahan umum software bisa diihat pada Gambar 3.5.



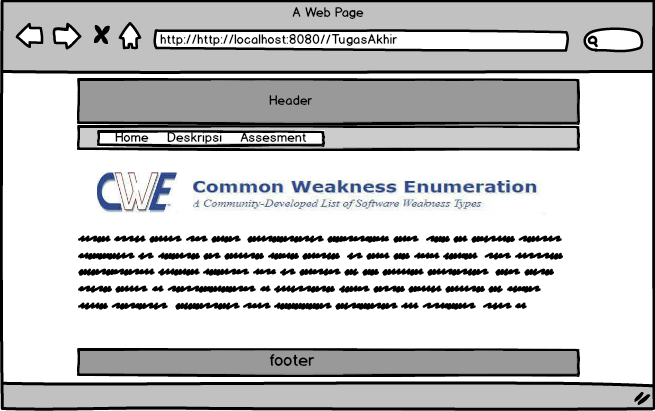
Gambar 3.5 Diagram HIPO

### Perancangan Antarmuka

Perancangan antarmuka sistem berguna untuk memberikan gambaran terkait sistem yang akan dibuat. Berikut rancangan antarmuka dari sistem ini.

1. Halaman Home

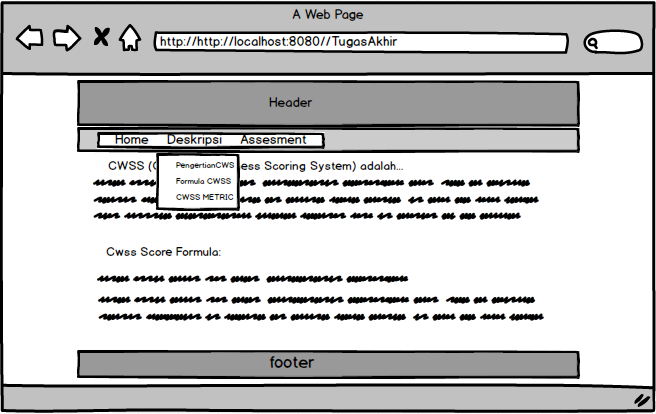
Halaman home ini menampilkan berisi informasi tentang gambaran umum CWE (Common Weakness Enumeration) selaku penyedia metode soring yang dipakai dalam penelitian. Rancangan antarmuka halaman home bisa dilihat pada Gambar 3.6.



Gambar 3.6 Perancangan Mockup Halaman Home

1. Halaman Deskripsi

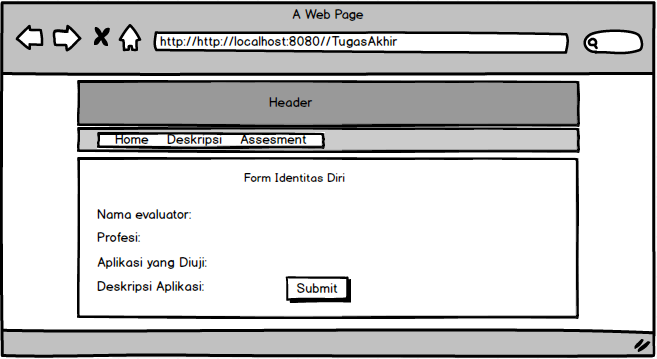
Halaman ini berisi informasi detail tentang metode scoring yang dipakai dalam penelitian seperti pengertian CWSS (Common Weakness Scoring System), rumus CWSS, bobot penilaian dll. Rancangan antarmuka halaman deskripsi bisa dilihat pada Gambar 3.7.



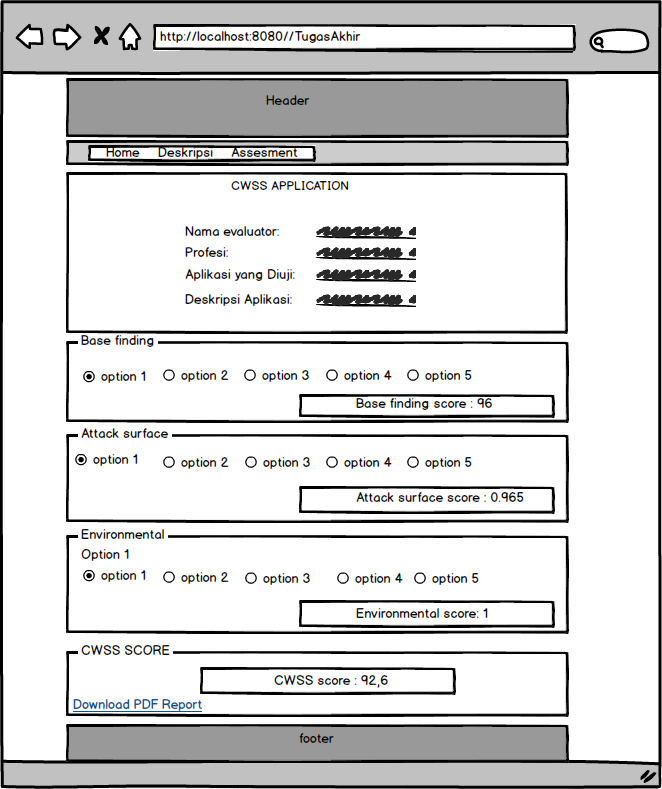
Gambar 3.7 Perancangan Mockup Halaman Deskripsi

1. Halaman Assesment

Halaman ini berisi form penilaian yang harus diisi oleh pengguna untuk mendapat skor kelemahan umum software. Pada halaman ini pengguna bisa melihat laporan hasil skor yang dapat diunduh dalam bentuk PDF. Rancangan antarmuka halaman Assesment bisa dilihat pada Gambar 3.8 dan Gambar 3.9.



Gambar 3.8 Perancangan Mockup Halaman Assesment 1



Gambar 3.9 Perancangan Mockup Halaman Assesment 2

### Perancangan Pengujian

Sebelum penulis dapat membuat kesimpulan, tentunya penelitian ini harus melalui tahap pengujian untuk memastikan apakah kebutuhan sistem dan semua fungsi pada sistem telah berkerja dengan baik dan benar. Pengujian pada tahapan ini dilakukan dengan beberapa jenis pengujian seperti pengujian skor hasil perhitungan pada sistem dan pengujian fungsionalitas pada sistem yang dibangun. Adapun penjelasan detail terkait pengujian tersebut akan dijelaskan pada poin-poin berikut:

#### Rancangan Pengujian Skor Hasil Perhitungan Pada Sistem

Pengujian skor ini dilakukan menggunakan 2 skenario yang telah disediakan oleh CWSS (*Common Weakness Scoring System*) dengan menggunakan metode *blackbox*. Tujuan dari rancang pengujian ini adalah untuk mendapatkan informasi tentang fungsi-fungsi dalam aplikasi apakah telah sesuai dengan apa yang direncanakan dan berjalan dengan baik atau tidak, mulai dari input hingga output. Maka dari itu penulis menguji ini berdasarkan tabel rancangan *blackbox testing* seperti pada Tabel 3.1 dan Tabel 3.2 berikut.

Tabel 3.1 Rancangan Skenario 1 Pengujian Skor

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric** | **Faktor** | **Skenario pilihan** | **Hasil yang diharapkan** |
| Base finding | Technical impact | High | 96.0 |
| Acquired privilege | Administrator |
| Acquired privilege layer | Application |
| Internal control effectiveness | None |
| Finding confidence | Proven True |
| Attack surface | Required privilege | Guest | 0.965 |
| Required privilege layer | Application |
| Acces vector | Internet |
| Authentication strength | None |
| Level of interaction | Typical/limited |
| Deployment scope | All |
| Environmental | Business impact | Critical | 1.0 |
| Likelihood of discovery | High |
| Likelihood of exploit | High |
| External control effectiveness | None |
| Prevalence | Not applicable |
| **CWSS Score** | | | 92.6 |

Tabel 3.2 Rancangan Skenario 2 Pengujian Skor

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric** | **Faktor** | **Skenario pilihan** | **Hasil yang diharapkan** |
| Base finding | Technical impact | Medium | 84.0 |
| Acquired privilege | Administrator |
| Acquired privilege layer | Application |
| Internal control effectiveness | None |
| Finding confidence | Proven True |
| Attack surface | Required privilege | Regular user | 0.935 |
| Required privilege layer | Application |
| Acces vector | Internet |
| Authentication strength | Weak |
| Level of interaction | Automated |
| Deployment scope | Not applicable |
| Environmental | Business impact | Low | 0.65 |
| Likelihood of discovery | Not applicable |
| Likelihood of exploit | Not applicable |
| External control effectiveness | None |
| Prevalence | Not applicable |
| **CWSS Score** | | | 51.1 |

#### Perancangan Pengujian Implementasi Sistem

Rancangan pengujian sistem dilakukan saat sistem telah selesai dibuat sesuai rancangan. Pengujian dilakukan dengan memberikan kuesioner kepada responden dan mengisi kuesioner yang telah disediakan. Pengujian dilakukan untuk memastikan apakah semua fungsi pada sistem telah berkerja dengan baik dan benar Berikut desain kuesioner yang akan digunakan dalam pengujian. Rancangan kuesioner pengujian bisa dilihat pada Tabel 3.3.

Tabel 3.3 Rancangan Kuesioner Pengujian

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Keterangan** | **STS** | **TS** | **N** | **S** | **SS** |
| 1. | Sistem memiliki tampilan yang menarik |  |  |  |  |  |
| 2. | Sistem memiliki warna elemen antarmuka yang nyaman dilihat pengguna |  |  |  |  |  |
| 3. | Informasi yang disajikan dalam Aplikasi web penghitung skor kelemahan umum software menggunakan skor CWSS (Common Weakness Scoring System) mudah dipahami |  |  |  |  |  |
| 4. | Sistem mudah digunakan |  |  |  |  |  |
| 5. | Sistem dapat membantu para *software developer, researcher, end user,* praktisi keamanan mengetahui kelemahan pada *software*nya |  |  |  |  |  |
| 6. | Sistem dapat membantu para *software developer, researcher, end user,* praktisi keamanan mengidentifikasi faktor-faktor yang dapat menyebabkan kelemahan pada software |  |  |  |  |  |
| 7. | Sistem mampu menampilkan skor perhitungan dengan benar |  |  |  |  |  |
| 8. | Sistem sudah berjalan dengan baik |  |  |  |  |  |
|  | **Total** |  |  |  |  |  |

Keterangan:

1. STS : Sangat Tidak Setuju (1 Poin)
2. TS : Tidak Setuju (2 Poin)
3. N : Netral (3 Poin)
4. S : Setuju (4 Poin)
5. SS : Sangat Setuju (5 Poin)

Untuk mengetahui penilaian responden terhadap pengujian sistem, perlu dilakukan perhitungan skor pengujian. Terkait dengan formula yang digunakan untuk melakukan perhitungan skor pengujian dapat dilihat pada

Skor total merupakan skor yang diperoleh dari penjumlahan untuk setiap jumlah responden yang menjawab kategori tertentu dikali dengan bobot per kategori. Formula untuk memperoleh skor total dapat dilihat pada persamaan

Adapun terkait dengan skor maks merupakan skor tertinggi yang dapat diperoleh dari setiap subpenilaian. Formula untuk memperoleh Skor maksimal dapat dilihat pada

Setelah memperoleh skor pengujian, dapat diketahui kategori hasil penilaian kuesioner berdasarkan rentang skor pengujian. Kategori hasil penilaian kuesioner dapat dilihat pada Tabel 3.4.

Tabel 3.4 Kategori Penilaian Kuisioner

|  |  |
| --- | --- |
| **Skor Pengujian** | **Kategori Penilaian** |
| 0%-19,99% | Sangat Kurang |
| 20%-39,99% | Kurang |
| 40%-59,99% | Cukup |
| 60%-79,99% | Baik |
| 80%-100% | Sangat Baik |

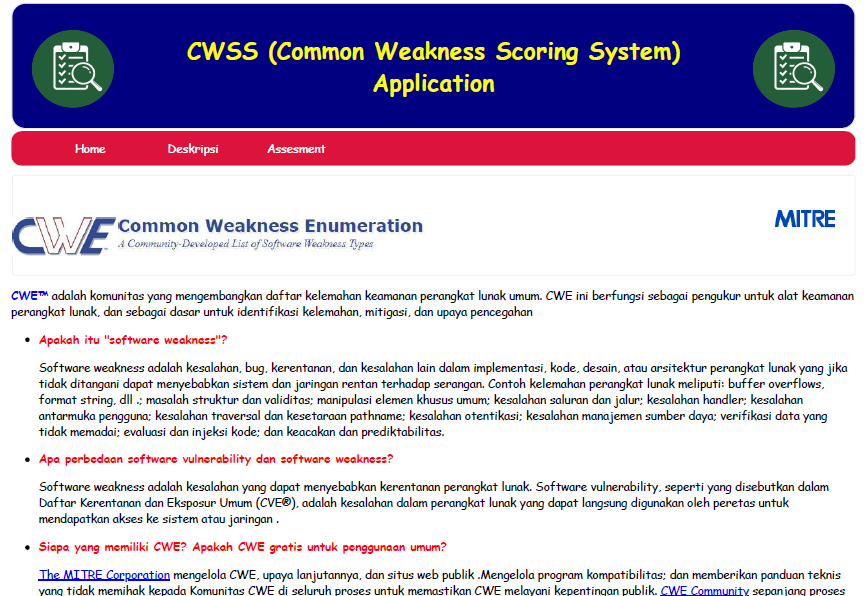
## BAB IV HASIL DAN PEMBAHASAN

### Implementasi Sistem

Tahapan implementasi merupakan tahapan lanjutan dari tahap perancangan bab sebelumnya. Pada bab hasil dan pembahasan menjelaskan cara kerja dan hasil dari sistem yang telah dibuat. Berikut ini adalah penjelasan dari implementasi aplikasi web penghitung kelemaham umum software menggunakan skor CWSS (Common Weakness Scoring System).

#### Halaman Home

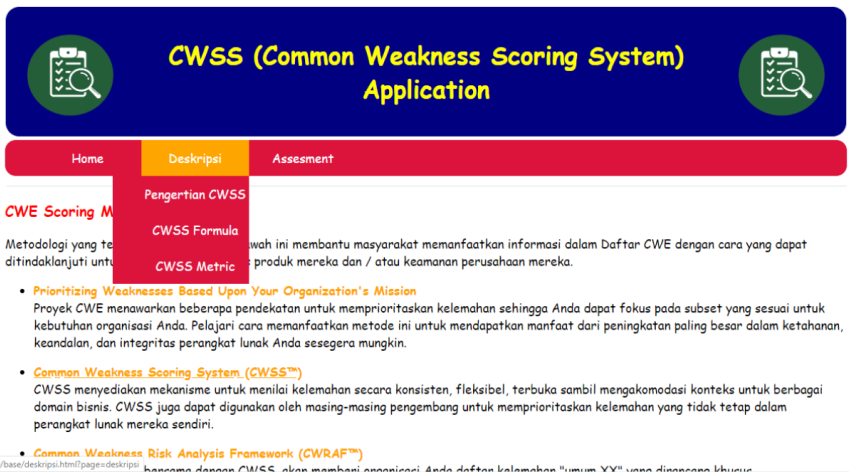
Halaman yang pertama kali muncul ketika aplikasi dijalankan adalah halaman home. Pada halaman ini berisi gambaran umum informasi tentang CWE (Common Weakness Enumeration) selaku komunitas penyedia metode scoring CWSS. Hasil halaman home bisa dilhat pada Gambar 4.1.



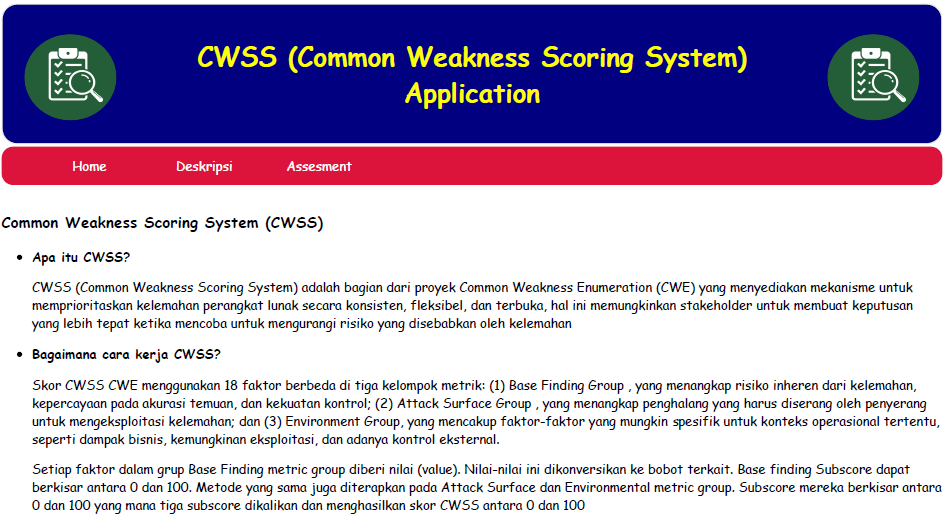
Gambar 4.1 Halaman Home

#### Halaman Deskripsi

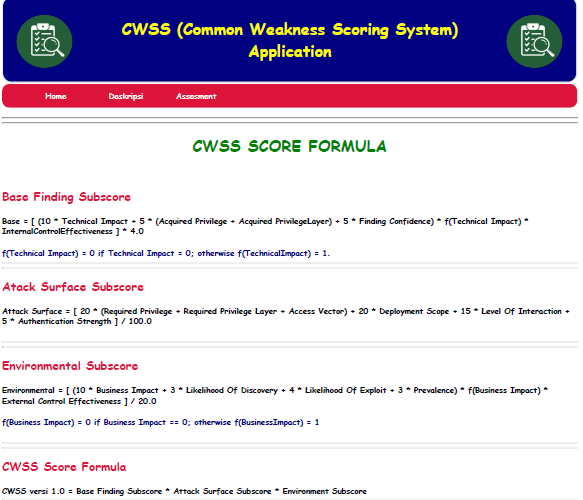
Halaman deskripsi ini memuat tentang informasi-infomasi detail metode scoring CWSS (Common Weakness Scoring System) seperti pengertian CWSS, cara kerja perhitungan CWSS, Grup matrik CWSS, formula yang dipakai dan bobot penilaian Hasil halaman deskripsi bisa dilihat pada Gambar 4.2, Gambar 4.3, Gambar 4.4, Gambar 4.5.



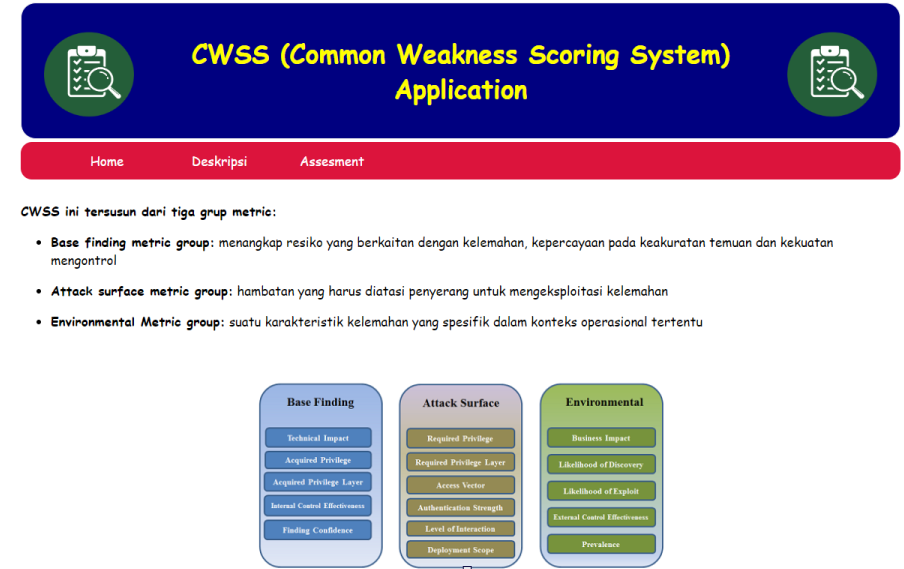
Gambar 4.2 Halaman Deskripsi



Gambar 4.3 Halaman Pengertian CWSS Submenu Deskripsi



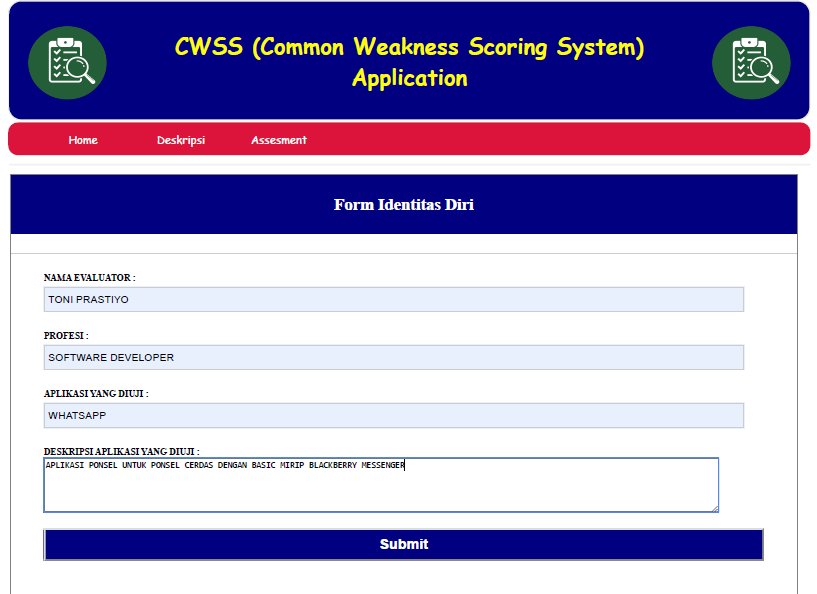
Gambar 4.4 Halaman CWSS Formula Submenu Deskripsi



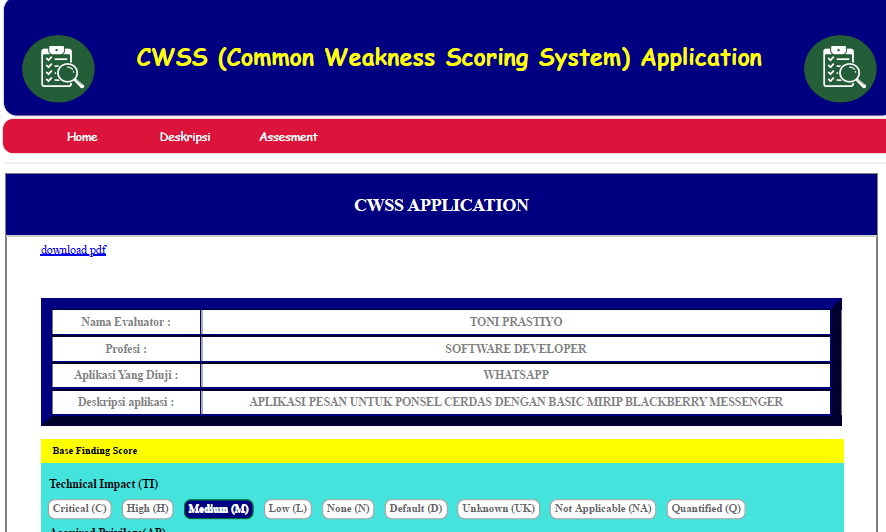
Gambar 4.5 Halaman CWSS Metric Grup Submenu Deskripsi

#### Halaman Assesment

Halaman assessment merupakan halaman yang digunakan untuk melakukan penilaian dimana pengguna akan memperoleh skor hasil perhitungan setelah mengisi beberapa opsi yang disediakan. Sebelum ke form penilain pengguna diminta untuk mengisi identitas diri terlebih dahulu seperti mengisi nama evaluator, profesi, aplikasi yang diuji dan deskripsi aplikasi yang diuji. Setelah mengisi identitas diri, pengguna akan melihat tampilan data identitasnya serta dapat melakukan penilaian. Pengisian identitas diri bisa dilihat pada Gambar 4.6 dan Gambar 4.7.



Gambar 4.6 Halaman Assesment Mengisi Identitas Diri



Gambar 4.7 Halaman Assesment

Pada Gambar 4.8 halaman assessment menunjukkan tampilan fieldset base finding score yang dibawahnya terdapat beberapa faktor yang harus dipilih pengguna dengan cara di klik dari factor technical impact sampai dengan finding confidence. Setelah pengguna mengisi semua faktor, hasil perhitungan base finding akan secara otomatis mengeluarkan skor.



Gambar 4.8 Halaman Assesment Base Finding Score

Perhitungan otomatis pada base finding score halaman assessment ini menggunakan javascript. Untuk *source code* nya bisa dilihat pada Gambar 4.9.

|  |  |
| --- | --- |
|  | // fungsi mendapatkan nilai pada base finding score  function gettechnicalimpact()  {  var technical=0;  var theForm = document.forms["base finding"];  var ti = theForm.elements["ti"];  for(var i = 0; i < ti.length; i++)  {  if(ti[i].checked )  {  technical = technical\_impact[ti[i].value];  break;  }  }  return technical;  }  function getacquiredprivilege()  {  var acquired =0;  var theForm = document.forms["base finding"];  var ap = theForm.elements["ap"];  for(var i = 0; i < ap.length; i++)  {  if(ap[i].checked)  {  acquired = acquired\_privilege[ap[i].value];  break;  }  }  return acquired;  }  function getacquiredprivilegelayer()  {  var acquiredpl=0;  var theForm = document.forms["base finding"];  var al = theForm.elements["al"];  for(var i = 0; i < al.length; i++)  {  if(al[i].checked)  {  acquiredpl= acquired\_privilegeLayer[al[i].value];  break;  }  }  return acquiredpl;  }  function getinternalcontrol()  {  var internal=0;  var theForm = document.forms["base finding"];  var ic = theForm.elements["ic"];  for(var i = 0; i < ic.length; i++)  {  if(ic[i].checked)  {  internal = internalControl\_effectiveness[ic[i].value];  break;  }  }  return internal;  }  function getfindingconfidence()  {  var finding=0;  var theForm = document.forms["base finding"];  var fc = theForm.elements["fc"];  for(var i = 0; i < fc.length; i++)  {  if(fc[i].checked)  {  finding = finding\_confidence[fc[i].value];  break;  }  }  return finding;  }  // fungsi mendapatkan hasil base finding score  function calculateTotal()  {  var hasil = gettechnicalimpact();  var ftimp ;  if(hasil <= 0){  ftimp =0;  } else {  ftimp =1;  }  var basefinding =[(10 \* gettechnicalimpact() + 5 \* (getacquiredprivilege()+ getacquiredprivilegelayer()) + 5 \*getfindingconfidence()) \* ftimp\* getinternalcontrol() ] \* 4    //display the result  var divobj = document.getElementById('totalPrice');  divobj.style.display='block';  divobj.innerHTML = "Base Finding Score: "+basefinding;  } |

Gambar 4.9 *Source code* perhitungan base finding score

Pada Gambar 4.10 halaman assessment menunjukkan tampilan fieldset attack surface score yang dibawahnya terdapat beberapa faktor yang harus dipilih pengguna dengan cara di klik dari faktor required privilege sampai dengan deployment scope. Setelah pengguna mengisi semua faktor, hasil perhitungan attack surface akan secara otomatis mengeluarkan skor.



Gambar 4.10 Halaman Assesment Attack Surface Score

Perhitungan otomatis pada attack surface score halaman assessment ini menggunakan javascript. Untuk *source code* nya bisa dilihat pada Gambar 4.11.

|  |  |
| --- | --- |
|  | // fungsi attack surface  function getrequiredprivilege()  {  var required=0;  var theForm = document.forms["base finding"];  var rp = theForm.elements["rp"];  for(var i = 0; i < rp.length; i++)  {  if(rp[i].checked)  {  required = required\_privilege[rp[i].value];  break;  }  }  return required;  }  function getrequiredprivilegelayer()  {  var requiredpl=0;  var theForm = document.forms["base finding"];  var rl = theForm.elements["rl"];  for(var i = 0; i < rl.length; i++)  {  if(rl[i].checked)  {  requiredpl = required\_privilegeLayer[rl[i].value];  break;  }  }  return requiredpl;  }  function getaccesvector()  {  var acces=0;  var theForm = document.forms["base finding"];  var av = theForm.elements["av"];  for(var i = 0; i < av.length; i++)  {  if(av[i].checked)  {  acces = acces\_vector[av[i].value];  break;  }  }  return acces;  }  function getauthenticationstrength()  {  var authentication=0;  var theForm = document.forms["base finding"];  var as = theForm.elements["as"];  for(var i = 0; i < as.length; i++)  {  if(as[i].checked)  {  authentication =authentication\_strength[as[i].value];  break;  }  }  return authentication;  }  function getlevelinteraction()  {  var level=0;  var theForm = document.forms["base finding"];  var ini = theForm.elements["ini"];  for(var i = 0; i < ini.length; i++)  {  if(ini[i].checked)  {  level =level\_interaction[ini[i].value];  break;  }  }  return level;  }  function getdeploymentscope()  {  var deployment=0;  var theForm = document.forms["base finding"];  var sco = theForm.elements["sco"];  for(var i = 0; i < sco.length; i++)  {  if(sco[i].checked)  {  deployment =deployment\_scope[sco[i].value];  break;  }  }  return deployment;  }  function calculateTotal2()  {  var attack = [ 20 \* (getrequiredprivilege() + getrequiredprivilegelayer() + getaccesvector()) + 20\* getdeploymentscope() + 15\* getlevelinteraction() + 5\*getauthenticationstrength()] /100.0  //display the result  var divobj = document.getElementById('totalprices');  divobj.style.display='block';  divobj.innerHTML = "Attack surface score: "+attack;  } |

Gambar 4.11 *Source code* Perhitungan Attack Surface Score

Pada Gambar 4.12 halaman assessment menunjukkan tampilan fieldset environmental score yang business impact sampai dengan prevalence. Setelah pengguna mengisi semua faktor, hasil perhitungan environmental akan secara otomatis mengeluarkan skor.



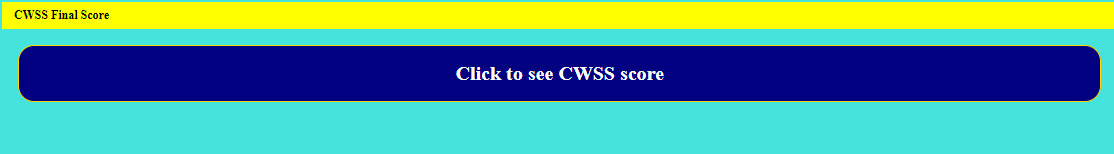
Gambar 4.12 Halaman Assesment Environmental Score

Perhitungan otomatis pada environmental score halaman assessment ini menggunakan javascript. Untuk *source code* nya bisa dilihat pada Gambar 4.13.

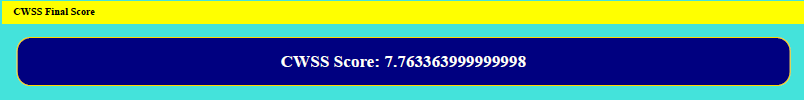
|  |  |
| --- | --- |
|  | // fungsi get environmental  function getbusinessimpact()  {  var business=0;  var theForm = document.forms["base finding"];  var bim = theForm.elements["bim"];  for(var i = 0; i < bim.length; i++)  {  if(bim[i].checked)  {  business =business\_impact[bim[i].value];  break;  }  }  return business;  }  function getlikelihooddiscovery()  {  var likelihood=0;  var theForm = document.forms["base finding"];  var di = theForm.elements["di"];  for(var i = 0; i < di.length; i++)  {  if(di[i].checked)  {  likelihood =likelihoodOf\_discovery[di[i].value];  break;  }  }  return likelihood;  }  function getlikelihoodexploit()  {  var lexploit=0;  var theForm = document.forms["base finding"];  var ex = theForm.elements["ex"];  for(var i = 0; i < ex.length; i++)  {    if(ex[i].checked)  {  lexploit =likelihoodOf\_exploit[ex[i].value];  break;  }  }  return lexploit;  }  function getexternalcontrol()  {  var external=0;  var theForm = document.forms["base finding"];  var ec = theForm.elements["ec"];  for(var i = 0; i < ec.length; i++)  {  if(ec[i].checked)  {  external =externalControl\_effectiveness[ec[i].value];  break;  }  }  return external;  }  function getprevalence()  {  var prevalen=0;  var theForm = document.forms["base finding"];  var p = theForm.elements["p"];  for(var i = 0; i < p.length; i++)  {  if(p[i].checked)  {  prevalen =prevalence[p[i].value];  break;  }  }  return prevalen;  }  function calculateTotal3(){  var hasil2 = getbusinessimpact();    var fbimp ;  if(hasil2 <=0){  fbimp =0;  } else {  fbimp =1;  }    var business = [ (10\*getbusinessimpact() + 3 \* getlikelihooddiscovery() + 4 \* getlikelihoodexploit() + 3\* getprevalence() ) \* fbimp \* getexternalcontrol()] / 20  //display the result  var divobj = document.getElementById('totalPriced');  divobj.style.display='block';  divobj.innerHTML = "Environmental score: "+business;  } |

Gambar 4.13 *Source code* Perhitungan Environmental Score

Setelah mengisi beberapa faktor dari fieldset base finding score, attack surface score, environmental score terdapat fieldset CWSS score, untuk melihat skor CWSS pengguna melakukan klik pada label yang bertuliskan “Click to see CWSS score” bisa dilihat pada Gambar 4.14 serta hasilnya bisa langsung terlihat pada Gambar 4.15.



Gambar 4.14 Halaman Assesment *click* CWSS Score



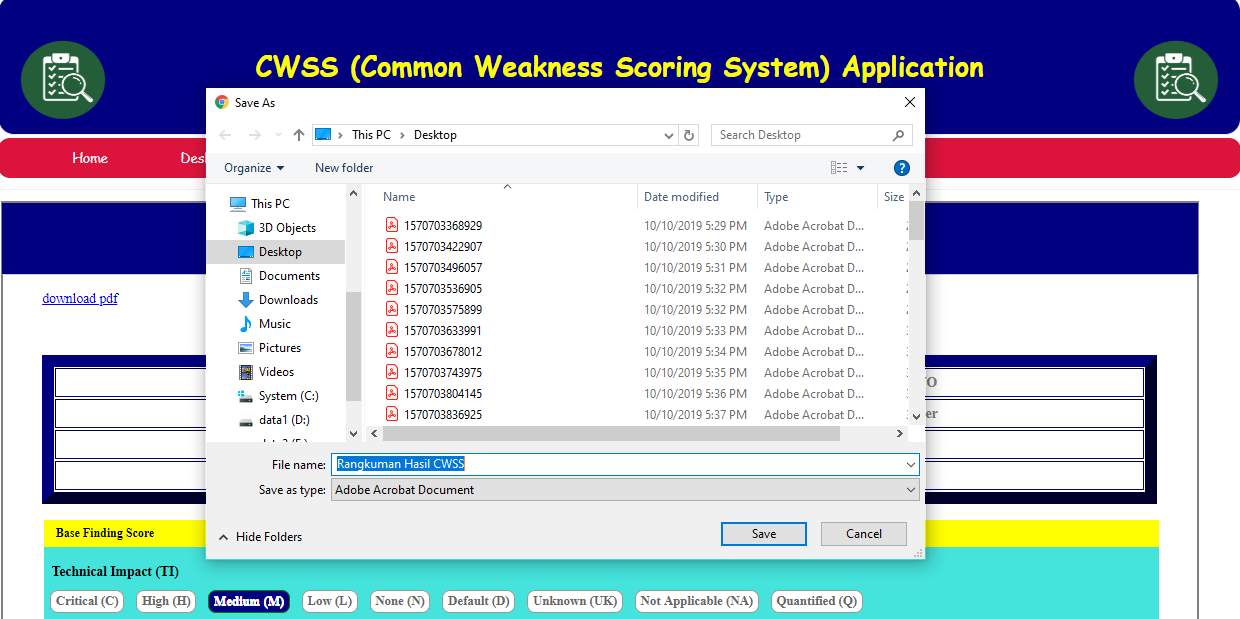
Gambar 4.15 Halaman Assesment CWSS Score

Perhitungan otomatis pada CWSS score halaman assessment ini menggunakan javascript. Untuk *source code* nya bisa dilihat pada Gambar 4.16.

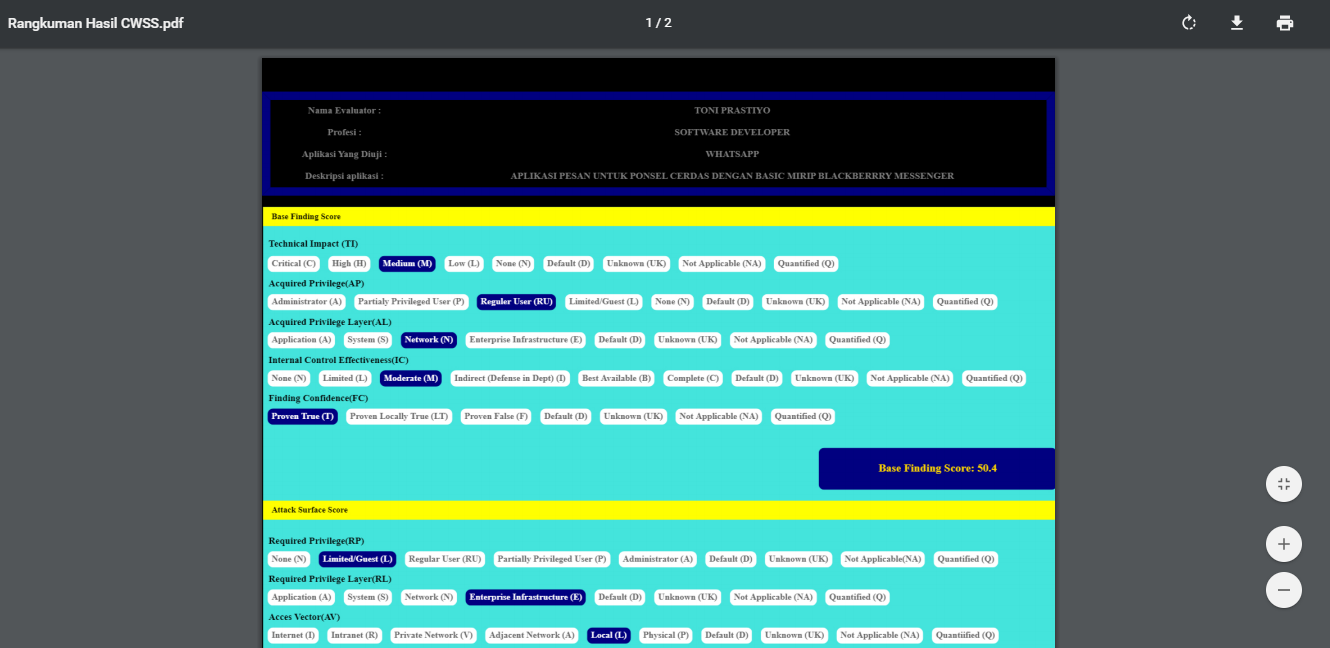
|  |  |
| --- | --- |
|  | function calculateTotal4()  {  var hasil = gettechnicalimpact();    var ftimp ;  if(hasil <= 0){  ftimp =0;  } else {  ftimp =1;  }    var hasil2 = getbusinessimpact();    var fbimp ;  if(hasil2 <= 0){  fbimp =0;  } else {  fbimp =1;  }  var cwss =[(10 \* gettechnicalimpact() + 5 \* (getacquiredprivilege()+ getacquiredprivilegelayer()) + 5 \*getfindingconfidence()) \*. ftimp \* getinternalcontrol() ] \* 4.0 \* [ 20 \* (getrequiredprivilege() + getrequiredprivilegelayer() + getaccesvector()) + 20\* getdeploymentscope() + 15\* getlevelinteraction() + 5\*getauthenticationstrength()] /100  \* [ (10\*getbusinessimpact() + 3 \* getlikelihooddiscovery() + 4 \* getlikelihoodexploit() + 3\* getprevalence() ) \* fbimp \* getexternalcontrol()] / 20  //display the result  var divobj = document.getElementById('totalPricede');  divobj.style.display='block';  divobj.innerHTML = "CWSS Score: "+cwss;  } |

Gambar 4.16 *Source code* Perhitungan CWSS Score

Setelah mengisi penilaian pada halaman assessment pengguna dapat melihat kembali hasil rangkuman yang telah diisi dengan cara klik tombol “download pdf” yang bisa dilihat pada Gambar 4.17. Rangkuman hasil penilaian ini memiliki format pdf dan bisa dilihat pada Gambar 4.18.



Gambar 4.17 Download Rangkuman Hasil Penilaian Format PDF



Gambar 4.18 Hasil Rangkuman Penilaian dengan Format PDF

### Pengujian

Pengujian merupakan tahap terakhir dalam tahapan penelitian seperti yang sudah dijelaskan pada bab sebelumnya. Pada tahap ini, terdapat dua jenis pengujian yang telah dilakukan yaitu:

#### Pengujian Skor Hasil Perhitungan Pada Sistem

Pengujian skor hasil perhitungan pada sistem dilakukan dengan menggunakan pengujian *blackbox* dengan menggunakan skenario yang telah disediakan CWSS. Tujuan dari pengujian ini untuk mengetahui apakah fungsi masukan dan keluaran sudah sesuai dengan yang diharapkan. Aplikasi diuji dengan menggunakan pengujian *blackbox* dengan rancangan seperti pada Tabel 3.1 dan Tabel 3.2 serta didapatkan hasil seperti pada Tabel 4.1 dan Tabel 4.2 berikut.

Tabel 4.1 Hasil Pengujian Skor Skenario 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metric** | **Faktor** | **Skenario pilihan** | **Hasil yang diharapkan** | **Hasil pengujian** |
| Base finding | Technical impact | High | 96.0 | Sesuai dengan hasil yang diharapkan |
| Acquired privilege | Administrator |
| Acquired privilege layer | Application |
| Internal control effectiveness | None |
| Finding confidence | Proven True |
| Attack surface | Required privilege | Guest | 0.965 | Sesuai dengan hasil yang diharapkan |
| Required privilege layer | Application |
| Acces vector | Internet |
| Authentication strength | None |
| Level of interaction | Typical/limited |
| Deployment scope | All |
| Environmental | Business impact | Critical | 1.0 | Sesuai dengan hasil yang diharapkan |
| Likelihood of discovery | High |
| Likelihood of exploit | High |
| External control effectiveness | None |
| Prevalence | Not applicable |
| **CWSS Score** | | | 92.6 | Sesuai dengan hasil yang diharapkan |

Tabel 4.2 Hasil Pengujian Skor Skenario 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metric** | **Faktor** | **Skenario pilihan** | **Hasil yang diharapkan** | **Hasil pengujian** |
| Base finding | Technical impact | Medium | 84.0 | Sesuai dengan hasil yang diharapkan |
| Acquired privilege | Administrator |
| Acquired privilege layer | Application |
| Internal control effectiveness | None |
| Finding confidence | Proven True |
| Attack surface | Required privilege | Regular user | 0.935 | Sesuai dengan hasil yang diharapkan |
| Required privilege layer | Application |
| Acces vector | Internet |
| Authentication strength | Weak |
| Level of interaction | Automated |
| Deployment scope | Not applicable |
| Environmental | Business impact | Low | 0.65 | Sesuai dengan hasil yang diharapkan |
| Likelihood of discovery | Not applicable |
| Likelihood of exploit | Not applicable |
| External control effectiveness | None |
| Prevalence | Not applicable |
| **CWSS Score** | | | 51.1 | Sesuai dengan hasil yang diharapkan |

#### Pengujian Implementasi Sistem

Pengujian dilakukan ketika tahapan implementasi dari perancangan sebelumnya telah selesai dibuat. Setelah dilakukan implementasi sistem, dilakukan pengujian kepuasan pengguna terhadap sistem yang sudah dibuat. Pengujian dilakukan oleh 5 orang yang terdiri dari Software Developer Codenesia dan Staff IT BSI UII. Pengujian dilakukan dengan cara mengisi kuesioner yang telah disediakan, sehingga didapatkan presentase kelayakan dari sistem yang telah dibuat. Penulis mendapatkan hasil kuesioner ini dengan metode penghitungan skala likert.

Metode skala likert adalah perhitungan yang mengacu pada pandangan seseorang, sikap dan pendapat berdasarkan kondisi yang sudah ditetapkan terlebih dahulu oleh penulis. Penulis menggunakan perhitungan skala likert berdasarkan 3 sebab yaitu skala likert yang paling banyak dugunakan dalam riset berupa survei, memudahkan responden untuk menjawab kuesioner terdapat pilihan sangat setuju sampai sangat tidak setuju, secara tampilan menggunakan skala likert lebih menarik dan mudah diisi oleh responden. Berikut hasil dari pengujian yang telah dilakukan dapat dilihat pada Tabel 4.3.

Tabel 4.3 Hasil Kuesioner Pengujian Implementasi Sistem

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Keterangan** | **STS** | **TS** | **N** | **S** | **SS** |
| 1. | Sistem memiliki tampilan yang menarik |  | 1 | 2 | 2 |  |
| 2. | Sistem memiliki warna elemen antarmuka yang nyaman dilihat pengguna |  | 2 | 1 | 1 | 1 |
| 3. | Informasi yang disajikan dalam Aplikasi web penghitung skor kelemahan umum software menggunakan skor CWSS (Common Weakness Scoring System) mudah dipahami |  | 1 | 2 | 2 |  |
| 4. | Sistem mudah digunakan |  |  |  | 3 | 2 |
| 5. | Sistem dapat membantu para *software developer, researcher, end user,* praktisi keamanan mengetahui kelemahan pada *software*nya |  | 1 | 2 | 2 |  |
| 6. | Sistem dapat membantu para *software developer, researcher, end user,* praktisi keamanan mengidentifikasi faktor-faktor yang dapat menyebabkan kelemahan pada software |  |  | 2 | 2 | 1 |
| 7. | Sistem mampu menampilkan skor perhitungan dengan benar |  |  |  | 4 | 1 |
| 8. | Sistem sudah berjalan dengan baik |  |  |  | 4 | 1 |
|  | **Total** |  | **5** | **9** | **20** | **6** |

Berdasarkan rumus perhitungan yang telah ditentukan sebelumnya, berikut perhitungan dari hasil pengujian yang telah dilakukan:

* 1. Responden yang menjawab sangat setuju (bobot 5) : 6 × 5 = 30
  2. Responden yang menjawab setuju (bobot 4) : 20 × 4 = 80
  3. Responden yang menjawab netral (bobot 3) : 9 × 3 = 27
  4. Responden yang menjawab tidak setuju (bobot 2) : 5 × 2 =10
  5. Responden yang menjawab sangat tidak setuju (bobot 1) : 0 × 1 = 0

Skor total : 30 + 80 + 27 + 10 + 0 = 147

Skor maks : 8 × 5 × 5 = 200

Skor pengujian : (147/200) × 100% = **73,5 %**

## BAB V KESIMPULAN DAN SARAN

### Kesimpulan

Format paragraf dengan style **isi paragraf** diterapkan sama di seluruh dokumen. Baris pertama berjarak 1 cm dari margin kiri rata kanan, diikuti baris seterusnya dari margin kiri.

### Saran

Format paragraf dengan style **isi paragraf** diterapkan sama di seluruh dokumen. Baris pertama berjarak 1 cm dari margin kiri rata kanan, diikuti baris seterusnya dari margin kiri.

.

## DAFTAR PUSTAKA

Andrade, H., & Du, Y. (2007). Student responses to criteria‐referenced self‐assessment. *Assessment & Evaluation in Higher Education*, *32*(2), 159–181.

Finkelstein, A. C. W., Kappel, G., & Retschitzegger, W. (2002). *Ubiquitous web application development-a framework for understanding*. na.

Flanagan, D. (2006). *JavaScript: the definitive guide*. “ O’Reilly Media, Inc.”

Martin, B., & Coley, S. C. (2014). Common Weakness Scoring System (CWSS). *Internet], Http://cwe. Mitre. Org/cwss*.

Martin, R. A. (2007). Common Weakness Enumeration, (May).

Mimin, H. (2007). Model dan teknik penilaian pada tingkat satuan pendidikan. *Jakarta: Gaung Persada Prees. Http/haryati. Blogspot. Com. id/2*, *7*(6).

Pawelana, A. F., Juliharta, I. G. P. K., & Suwirmayanti, N. L. G. P. (2015). Aplikasi Perhitungan Kerentanan Menggunakan Metode Common Vulnerability Scoring System Version 2. *JOSIKOM: Jurnal Online Sistem Komputer*, *1*(1).

Pressman, R. S. (2012). Rekayasa Perangkat Lunak (Buku Satu): Penerbit Andi. Yogyakarta.

Rosa, A. S., & Shalahuddin, M. (2011). Modul Pembelajaran Rekayasa Perangkat Lunak (Terstruktur dan Berorientasi Objek). *Bandung: Modula*.

Satzinger, J. W., Jackson, R. B., & Burd, S. D. (2011). *Systems analysis and design in a changing world*. Cengage learning.

Sunyoto, A., & Kom, M. (2007). *Ajax Membangun Web dengan Teknologi Asynchronouse JavaScript&XML*. Penerbit Andi.

Tam, A. (2005). CHARACTERISTICS OF WEB APPLICATIONS THAT AFFECT USABILITY : A REVIEW, 1–4.

## LAMPIRAN