«SKRIPSI/TUGAS AKHIR»

«JUDUL BAHASA INDONESIA»

«Nama Lengkap»

NPM: «10 digit NPM UNPAR»

PROGRAM STUDI «MATEMATIKA/FISIKA/TEKNIK INFORMATIKA»
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
«tahun»

«FINAL PROJECT/UNDERGRADUATE THESIS»

«JUDUL BAHASA INGGRIS»

«Nama Lengkap»

NPM: «10 digit NPM UNPAR»

DEPARTMENT OF «MATHEMATICS/PHYSICS/INFORMATICS»
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
«tahun»

LEMBAR PENGESAHAN

«JUDUL BAHASA INDONESIA»

«Nama Lengkap»

NPM: «10 digit NPM UNPAR»

Bandung, «tanggal» «bulan» «tahun»

Menyetujui,

Pembimbing Utama	Pembimbing Pendamping
«pembimbing utama/1»	«pembimbing pendamping/2»

Ketua Tim Penguji	Anggota Tim Penguji
«penguji 1»	«penguji 2»

Mengetahui,

Ketua Program Studi

«Ketua Program Studi»

PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa «skripsi/tugas akhir» dengan judul:

«JUDUL BAHASA INDONESIA»

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.

Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuhkan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,
Tanggal «tanggal» «bulan» «tahun»

Meterai
Rp. 6000

«Nama Lengkap»
NPM: «10 digit NPM UNPAR»

ABSTRAK

«Tuliskan abstrak anda di sini, dalam bahasa Indonesia»
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Kata-kata kunci: «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Indonesia»

ABSTRACT

«Tuliskan abstrak anda di sini, dalam bahasa Inggris»
Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Keywords: «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Inggris»

«kepada siapa anda mempersembahkan skripsi ini. . . ?»

KATA PENGANTAR

«Tuliskan kata pengantar dari anda di sini . . . »
Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Bandung, «bulan» «tahun»

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1

PENDAHULUAN

1.1 Latar Belakang

SharIF Judge [1] adalah sebuah aplikasi gratis dan *open source* untuk menilai code berbahasa C , C++, Java dan Python. SharIF Judge adalah pencabangan dari Sharif Judge yang telah dibuat oleh Mohammed Javad Naderi. Versi dari pencabangan ini memuat fitur baru yang diperlukan oleh jurusan teknik informatika UNPAR. Aplikasi ini dibuat menggunakan PHP (*CodeIgnitor framework*) dan bagian backendnya dibuat dengan BASH.

Web Content Accessibility Guidelines (*WCAG*) 2.1 [2] memuat rekomendasi untuk membuat konten web lebih mudah diakses. Pedoman-pedoman ini akan membuat konten lebih mudah diakses untuk orang disabilitas termasuk akomodasi untuk kebutaan dan penglihatan rendah, ketulian dan gangguan pendengaran, gerakan terbatas, fotosensitif, atau kombinasinya, dan beberapa akomomodasi untuk kesulitan belajar dan keterbatasan kognitif; tetapi tidak akan memenuhi setiap kebutuhan pengguna dengan disabilitas. Di dalam *WCAG* 2.1 ada 78 kriteria sukses. Kriteria sukses adalah pedoman untuk membuat konten lebih mudah diakses. Ada 3 tingkat kepatuhan yaitu A (terkecil), AA, AAA (terbesar). Tingkat kepatuhan A adalah tingkat kepatuhan terkecil yang diperoleh jika seluruh kriteria sukses tingkat A terpenuhi atau versi alternatifnya tersedia. Tingkat kepatuhan AA adalah tingkat kepatuhan yang diperoleh jika seluruh kriteria sukses tingkat A dan AA terpenuhi atau versi alternatif tingkat AA tersedia. Tingkat kepatuhan AAA adalah tingkat kepatuhan yang diperoleh jika seluruh kriteria sukses tingkat A, AA, dan AAA terpenuhi atau veri alternatif tingkat AAA tersedia.

Pada skripsi ini, akan dilakukan analisis tingkat kepatuhan dan rekomendasi perbaikan aplikasi SharIF Judge berdasarkan *Web Content Accessibility Guideline* 2.1. Selain itu, aplikasi SharIF Judge juga akan diuji dengan beberapa kondisi keterbatasan seperti keterbatasan visual, keterbatasan gerak, keterbatasan pendengaran. Dengan perbaikan ini diharapkan aplikasi SharIF Judge dapat diakses oleh banyak kalangan.

1.2 Rumusan Masalah

- Bagaimana tingkat kepatuhan SharIF Judge terhadap *WCAG* 2.1 ?
- Rekomendasi perbaikan apa saja yang perlu dilakukan terhadap SharIF Judge untuk menaikkan tingkat kepatuhannya ?

1.3 Tujuan

- Mengetahui tingkat kepatuhan SharIF Judge terhadap WCAG 2.1.
- Membuat rekomendasi perbaikan yang perlu dilakukan terhadap SharIF Judge untuk menaikkan tingkat kepatuhannya.

1.4 Batasan Masalah

Batasan masalah pada skripsi ini adalah sebagai berikut :

1. content

1.5 Metodologi

Metodologi yang dilakukan pada skripsi ini adalah sebagai berikut :

1. Studi literatur mengenai *WCAG* 2.1 dan SharIF Judge
2. Mengukur tingkat kepatuhan SharIF Judge terhadap *WCAG* 2.1
3. Memberikan rekomendasi perbaikan pada setiap kriteria kesuksesan.
4. Mengimplementasikan rekomendasi perbaikan.
5. Menguji hasil perbaikan.

1.6 Sistematika Pembahasan

Setiap bab dalam skripsi ini memiliki sistematika penulisan ke dalam poin-poin sebagai berikut :

1. Bab 1: Pendahuluan, akan membahas gambaran umum dari skripsi ini. Bab ini berisi latar belakang, rumusan masalah, tujuan, batasan masalah, metode penelitian, dan sistematika pembahasan.
2. Bab 2: Landasan Teori, akan membahas dasar teori yang menjadi acuan dalam pembuatan skripsi ini. Dasar teori yang digunakan yaitu *WCAG* 2.1 dan SharIF Judge.
3. Bab 3: Analisis, akan membahas hasil analisis mengenai tingkat kepatuhan situs web SharIF Judge terhadap *WCAG* 2.1.
4. Bab 4: Perancangan, akan membahas mengenai perubahan-perubahan yang dapat dilakukan untuk meningkatkan kepatuhan situs web SharIF Judge terhadap *WCAG* 2.1.
5. Bab 5: Implementasi dan Pengujian, akan membahas hasil implementasi dan pengujian yang telah dilakukan pada situs web SharIF Judge.
6. Bab 6: Kesimpulan dan saran, akan berisi kesimpulan dari hasil penelitian yang telah dilakukan dan saran yang dapat diberikan untuk penelitian berikutnya.

BAB 2

LANDASAN TEORI

2.1 WCAG 2.1

Web Content Accessibility Guidelines (*WCAG*) 2.1 memuat rekomendasi untuk membuat konten web lebih mudah diakses. Pedoman-pedoman ini akan membuat konten lebih mudah diakses untuk orang disabilitas termasuk akomodasi untuk kebutaan dan penglihatan rendah, ketulian dan gangguan pendengaran, gerakan terbatas, fotosensitif, atau kombinasinya, dan beberapa akomomodasi untuk kesulitan belajar dan keterbatasan kognitif; tetapi tidak akan memenuhi setiap kebutuhan pengguna dengan disabilitas. *WCAG* dikembangkan oleh *World Wide Web Consortium* melalui kerja sama dengan individu dan organisasi di seluruh dunia dengan tujuan memberikan standar bersama untuk aksesibilitas konten web yang memenuhi kebutuhan individu, organisasi, dan pemerintah internasional. *WCAG* 2.1 merupakan pembaruan dari *WCAG* 2.0 yang dibuat pada 11 Desember 2008. Ada 78 kriteria sukses dalam *WCAG* 2.1. Kriteria sukses adalah pedoman untuk membuat konten lebih mudah diakses. Kriteria Sukses *WCAG* 2.1 ditulis sebagai pernyataan yang dapat diuji yang tidak teknologi spesifik. Pedoman ini mencakup aksesibilitas konten web di desktop, laptop, tablet, dan perangkat bergerak. Dengan mengikuti pedoman ini juga akan sering membuat konten web lebih bermanfaat bagi pengguna secara umum.

Ada beberapa kondisi yang harus dipenuhi untuk sebuah Kriteria Sukses yaitu :

1. Semua Kriteria Sukses harus menjadi masalah akses penting bagi orang disabilitas yang mengatasi masalah di luar masalah kegunaan yang dihadapi oleh semua pengguna. Dengan kata lain, masalah akses harus menyebabkan masalah yang lebih besar bagi orang disabilitas daripada orang yang tidak disabilitas agar dianggap sebagai masalah aksesibilitas.

2. Semua Kriteria Sukses harus dapat diuji. Hal ini penting karena jika tidak, maka tidak mungkin untuk menentukan apakah suatu halaman memenuhi Kriteria Sukses. Kriteria Sukses dapat diuji dengan kombinasi evaluasi mesin dan manusia selama pengujian dapat menentukan apakah sebuah Kriteria Sukses terpenuhi dengan tingkat kepercayaan yang tinggi.

Kriteria Sukses memiliki tiga tingkat kesesuaian yaitu tingkat A (terkecil), AA, AAA (terbesar). Ada beberapa faktor yang menentukan tingkat tersebut. Faktor tersebut termasuk :

1. Apakah Kriteria Sukses esensial (dalam kata lain, jika Kriteria Sukses tidak terpenuhi maka teknologi bantuan juga tidak dapat membuat konten dapat diakses).

2. Apakah mungkin untuk memenuhi Kriteria Sukses untuk semua situs web dan jenis konten yang akan diterapkan Kriteria Sukses.

3. Apakah Kriteria Sukses membutuhkan keterampilan yang dapat dicapai secara wajar oleh pembuat konten (Pengetahuan dan keterampilan untuk memenuhi Kriteria Sukses dapat diperoleh dalam pelatihan seminggu atau kurang).

4. Apakah Kriteria Sukses dapat memaksakan batasan tampilan dan fungsi dari halaman web (batasan dari fungsi, presentasi, kebebasan berekspresi, desain atau estetika)

5. Apakah tidak ada solusi jika Kriteria Sukses tidak terpenuhi

Berikut adalah uraian kriteria sukses *WCAG* 2.1 :

2.1.1 *Perceivable*

Informasi dan antarmuka pengguna harus ditampilkan kepada pengguna dengan cara yang dapat dipahami.

Kriteria Sukses 1.1.1 Non-text Content

(Level A)

Semua konten bukan teks yang ditampilkan ke pengguna memiliki teks alternatif yang tujuannya sama. Kecuali pada kondisi sebagai berikut :

- Kontrol, dan masukan : Jika konten bukan teks adalah kontrol atau masukan user maka konten tersebut harus memiliki nama yang menjelaskan tujuannya.
- Media berbasis waktu : Jika konten bukan teks adalah media berbasis waktu, maka setidaknya disediakan alternatif berupa teks untuk identifikasi deskriptif dari konten bukan teks.
- Tes : Jika konten bukan teks adalah tes atau latihan yang tidak valid jika ditampilkan dalam teks, maka setidaknya disediakan teks alternatif untuk identifikasi deskriptif dari konten bukan teks.
- Indra : Jika konten bukan teks digunakan untuk menciptakan pengalaman indra tertentu, maka setidaknya disediakan teks alternatif untuk identifikasi deskriptif dari konten bukan teks.
- *CAPTCHA* : Jika tujuan dai konten bukan teks digunakan untuk memastikan apakah konten tersebut diakses oleh manusia dan bukannya komputer, maka disediakan teks alternatif untuk mengidentifikasi dan menjelaskan tujuan dari konten bukan teks, dan disediakan bentuk alternatif dari *CAPTCHA* menggunakan mode keluaran untuk berbagai jenis persepsi indra untuk mengakomodasi berbagai disabilitas.
- Dekorasi, pemformatan, tak kentara : Jika konten bukan teks adalah dekorasi saja, digunakan untuk pemformatan, atau tidak ditampilkan kepada pengguna, maka konten tersebut diterapkan dengan cara yang dapat diacuhkan oleh teknologi bantuan.

Kriteria Sukses 1.2.1 Audio-only dan Video-only (Prerecorded)

(Level A)

Untuk rekaman audio saja dan rekaman video saja, berikut ini benar, kecuali ketika audio atau video adalah media alternatif untuk teks dan diberi label dengan jelas :

- Rekaman audio saja : Tersedia alternatif untuk media berbasis waktu yang isinya mewakili informasi yang sama dengan konten rekaman audio saja.
- Rekaman video saja : Tersedia alternatif untuk media berbasis waktu atau trek audio yang isinya mewakili informasi yang sama dengan konten rekaman video saja.

Kriteria Sukses 1.2.2 Captions (Prerecorded)

(Level A)

Caption disediakan untuk semua konten rekaman audio di media berbasis waktu kecuali medianya adalah media alternatif untuk teks dan diberi label dengan jelas.

Kriteria Sukses 1.2.3 Audio Descriptive atau Media Alternative (Prerecorded)

(Level A)

Tersedianya alternatif untuk media berbasis waktu atau deskripsi audio dari konten rekaman video untuk media yang disingkronkan, kecuali medianya adalah media alternatif untuk teks dan diberi label dengan jelas.

Kriteria Sukses 1.2.4 Captions (Live)

(Level AA)

Keterangan tersedia untuk semua konten audio yang disiarkan langsung di media yang disingkronkan.

Kriteria Sukses 1.2.5 Audio Description (Prerecorded)

(Level AA)

Deskripsi audio disediakan untuk semua konten rekaman video di media yang disingkronkan.

Kriteria Sukses 1.2.6 Sign Language (Prerecorded)

(Level AAA)

Intepretasi bahasa isyarat disediakan untuk semua konten rekaman audio di media yang disingkronkan.

Kriteria Sukses 1.2.7 Extended Audio Description (Prerecorded)

(Level AAA)

Ketika keheningan di audio tidak memadai untuk menyampaikan maksud video tersebut, deskripsi audio tambahan disediakan untuk semua konten rekaman video pada media yang disingkronkan.

Kriteria Sukses 1.2.8 Media Alternative (Prerecorded)

(Level AAA)

Tersedia alternatif untuk media berbasis waktu untuk semua rekaman media yang disingkronkan dan untuk semua rekaman media video saja.

Kriteria Sukses 1.2.9 Audio-only (Live)

(Level AAA)

Tersedia alternatif untuk media berbasis waktu yang menampilkan informasi yang setara untuk konten siaran langsung audio saja.

Kriteria Sukses 1.3.1 Info dan Relationships

(Level A)

Informasi , struktur, dan hubungan yang ditampilkan melalui presentasi dapat ditentukan secara pemrograman atau tersedia dalam teks.

Kriteria Sukses 1.3.2 Meaningful Sequence

(Level A)

Ketika urutan konten disajikan memengaruhi maknanya, urutan bacaan yang benar dapat ditentukan secara pemrograman.

Kriteria Sukses 1.3.3 Sensory Characteristics

(Level A)

Petunjuk yang diberikan untuk memahami dan mengoperasikan konten tidak hanya bergantung pada komponen karakteristik sensorik seperti bentuk, warna, ukuran, lokasi visual, orientasi, atau suara.

Kriteria Sukses 1.3.4 Orientation

(Level AA)

Tampilan dan pengoperasian konten tidak bergantung pada satu orientasi tampilan , seperti *portrait* atau *landscape*, kecuali jika orientasi tampilan tertentu esensial.

Kriteria Sukses 1.3.5 Identify Input Purpose

(Level AA)
Tujuan untuk setiap bidang masukan yang digunakan untuk mendapatkan informasi pengguna dapat ditentukan secara pemrograman ketika :

- Bidang masukan menyajikan tujuan yang diidentifikasi di bagian tujuan masukan untuk komponen atarmuka pengguna.
- Konten diimplementasikan menggunakan teknologi dengan dukungan untuk mengidentifikasi makna yang diharapkan masukan data formulir.

Kriteria Sukses 1.3.6 Identify Purpose

(Level AAA)

Dalam konten yang diimplementasi dengan bahasa *markup*, tujuan dari komponen antarmuka, ikon, dan bidang dapat ditentukan secara pemrograman.

Kriteria Sukses 1.4.1 Use of Color

(Level A)

Warna tidak hanya digunakan sebagai satu-satunya cara visual untuk menyampaikan informasi, menunjukkan aksi, menampilkan respon, atau membedakan elemen visual.

Kriteria Sukses 1.4.2 Audio Control

(Level A)

Jika ada audio yang diputar secara otomatis di halaman web yang berdurasi lebih dari 3 detik, maka setidaknya ada mekanisme untuk menjeda atau menghentikan audio, atau ada mekanisme untuk mengontrol volume audio secara independen dari tingkat volume sistem secara keseluruhan.

Kriteria Sukses 1.4.3 Contrast (Minimum)

(Level AA)

Presentasi visual dari teks, gambar teks, memiliki rasio kontras setidaknya 4.5:1, kecuali jika :

- Teks besar : Teks berukuran besar dan gambar teks berukuran besar memiliki rasio kontras setidaknya 3:1.
- Insidental : Teks atau gambar teks yang merupakan bagian dari komponen antarmuka yang tidak aktif, atau hanya dekorasi saja, atau yang tidak tampak, atau bagian dari gambar yang memiliki konten visual yang signifikan, tidak memiliki syarat kontras.
- Logo : Teks yang merupakan bagian dari logo atau nama merek tidak memiliki syarat kontras.

Kriteria Sukses 1.4.4 Resize text

(Level AA)

Teks dapat diubah ukurannya tanpa teknologi bantuan hingga 200 persen tanpa kehilangan konten atau fungsionalitasnya kecuali untuk keterangan dan gambar teks.

Kriteria Sukses 1.4.5 Images of Text

(Level AA)

Jika suatu teknologi yang digunakan dapat mencapai presentasi visual, maka teks digunakan untuk menyampaikan informasi daripada gambar teks kecuali dalam kondisi berikut :

1. Customizable : Gambar teks dapat di sesuaikan dengan kebutuhan pengguna.
2. Esensial : Tampilan teks esensial untuk menyampaikan informasi.

Kriteria Sukses 1.4.6 Contrast (Enhanced)

(Level AAA)

Presentasi visual dari teks, gambar teks, memiliki rasio kontras setidaknya 7:1, kecuali jika :

1. Teks besar : Teks berukuran besar dan gambar teks berukuran besar memiliki rasio kontras setidaknya 4.5:1.
2. Insidental : Teks atau gambar teks yang merupakan bagian dari komponen antarmuka yang tidak aktif, atau hanya dekorasi saja, atau yang tidak tampak, atau bagian dari gambar yang memiliki konten visual yang signifikan, tidak memiliki syarat kontras.
3. Logo : Teks yang merupakan bagian dari logo atau nama merek tidak memiliki syarat kontras.

Kriteria Sukses 1.4.7 Low atau No Background Audio

(Level AAA)

Untuk konten rekaman audio saja yang berisi pidato di latar depan, bukan audio *CAPTCHA* atau audio logo, dan bukan suara musik atau rap, setidaknya salah satu kondisi berikut ini benar :

- Tidak ada latar belakang : Audio tidak berisi suara di latar belakangnya.
- Mematikan suara : Suara pada latar belakang dapat dimatikan
- 20 Desibel : Suara pada latar belakang setidaknya 20 desibel lebih kecil daripada pidato pada latar depan, dengan pengecualian untuk suara yang berdurasi satu atau dua detik.

Kriteria Sukses 1.4.8 Visual Presentation

(Level AAA)

Untuk presentasi visual dari blok teks, sebuah mekanisme harus ada untuk mencapai kondisi berikut :

- Warna pada latar belakang dan latar depan dapat dipilih oleh pengguna.
- Lebar tidak lebih dari 80 karakter.
- Teks tidak diratakan (sejajar dengan margin kiri dan kanan)
- Jeda baris dalam paragraf setidaknya satu setengah spasi, dan jeda paragraf setidaknya 1.5 lebih besar dari jeda baris.
- Teks dapat diubah ukurannya tanpa teknologi bantuan hingga 200 persen dengan cara pengguna tidak perlu gulir secara horizontal untuk membaca teks dalam mode *full-screen.*

Kriteria Sukses 1.4.9 Images of Text (No Exception)

(Level AAA)

Gambar teks hanya digunakan untuk dekorasi murni atau dimana presentasi teks tertentu esensial untuk informasi yang disampaikan.

Kriteria Sukses 1.4.10 Reflow

(Level AA)

Konten dapat ditampilkan tanpa kehilangan informasi atau fungsionalitasnya, dan tanpa memerlukan gulir di 2 dimensi untuk :

- Gulir vertikal untuk konten yang lebarnya setara dengan 320 piksel *css.*
- Gulir horizontal untuk konten yang tingginya setara dengan 256 piksel *css.*

Kecuali bagian konten yang memerlukan tata letak dua dimensi untuk keperluan dan tujuannya.

Kriteria Sukses 1.4.11 Non-text Contrast

(Level AA)

Presentasi visual berikut harus memiliki rasio kontras setidaknya 3:1 terhadap warna yang bedekatan :

- Komponen antarmuka : Informasi visual diperlukan untuk mengidentifikasi komponen antarmuka dan statusnya, kecuali antarmuka yang tidak aktif atau tampilan komponen ditentukan oleh agen pengguna dan tidak diubah oleh pembuat web.
- Objek grafik : Bagian dari grafik diperlukan untuk menjelaskan kontennya, kecuali tampilan grafik esensial untuk informasi yang disampaikan.

Kriteria Sukses 1.4.12 Text Spacing

(Level AA)

Dalam konten yang diimplementasikan menggunakan bahasa *markup* yang mendukung properti gaya teks berikut, tidak ada kehilangan konten atau fungsionalitas ketika mengatur setelan semua hal berikut dan tidak mengganti properti tampilannya :

- Tinggi baris setidaknya 1.5 kali ukuran teks.
- Spasi antar paragraf setidaknya 2 kali ukuran teks.
- Spasi antar huruf setidaknya 0.12 kali ukuran teks
- Spasi antar kata setidaknya 0.16 kali ukuran teks.

Kriteria Sukses 1.4.13 Content on Hover or Focus

(Level AA)

Ketika mendapatkan dan kemudian menghapus *pointer hover* atau fokus keyboard memicu konten tambahan untuk menjadi terlihat dan kemudian disembunyikan, berikut ini benar :

- Dapat disingkirkan : Tersedianya mekanisme untuk menyingkirkan konten tambahan tanpa menggerakkan *pointer hover* atau fokus keyboard, kecuali konten tambahan menunjukkan kesalahan input, atau tidak mengganggu atau menggantikan konten lain.
- *Hoverable* : Jika *pointer hover* dapat memicu konten tambahan, maka penunjuk dapat digerakkan di atas konten tambahan tanpa konten tambahan tersebut hilang.
- Persisten : Konten tambahan dapat terlihat sampai penunjuk atau fokus dihilangkan, pengguna menyingkirkannya, atau informasinya sudah tidak valid.

Kecuali : Presentasi visual dari konten tambahan dikontrol oleh agen pengguna dan tidak diubah oleh pembuat web.

2.1.2 *Operable*

Komponen antarmuka pengguna dan navigasi harus dapat dioperasikan.

Kriteria Sukses 2.1.1 Keyboard

(Level A)

Semua fungsionalitas konten dapat dioperasikan melalui antarmuka keyboard tanpa memerlukan waktu spesifik untuk penekanan tombolnya, kecuali jika fungsi yang mendasarinya membutuhkan input yang bergantung pada pergerakan pengguna dan bukan hanya pada titik akhir.

Kriteria Sukses 2.1.2 No Keyboard Trap

(Level A)

Jika fokus keyboard dapat dipindahkan ke komponen tertentu pada halaman menggunakan antarmuka keyboard, maka fokus dapat dipindahkan dari komponen itu hanya menggunakan antarmuka keyboard, dan jika memerlukan lebih dari sekadar penekanan tombol panah atau tombol *tab* atau metode keluar standar lainnya, pengguna diberikan informasi tentang metode tersebut untuk memindahkan fokus.

Kriteria Sukses 2.1.3 Keyboard (No Exception)

(Level AAA)
Semua fungsionalitas konten dapat dioperasikan melalui antarmuka keyboard tanpa memerlukan waktu spesifik untuk penekanan tombolnya.

Kriteria Sukses 2.1.4 Character Key Shortcuts

(Level A)

Jika pintasan keyboard diimplementasi dengan menggunakan huruf (termasuk huruf besar dan kecil), tanda baca, angka, atau karakter simbol, maka setidaknya salah satu dari yang berikut ini benar :

- Matikan : Tersedia mekanisme untuk mematikan pintasan.
- Dipetakan kembali : Tersedia mekanisme untuk memetakan kembali pintasan untuk menggunakan satu atau lebih karaktek keyboard yang tidak dapat dicetak.
- Hanya aktif saat fokus : Pintasan keyboard untuk komponen antarmuka pengguna hanya aktif saat komponen tersebut memiliki fokus.

Kriteria Sukses 2.2.1 Timing Adjustable

(Level A)

Untuk setiap batasan waktu yang ditentukan oleh konten, setidaknya salah satu dari yang berikut ini benar :

- Matikan : Pengguna dapat mematikan batas waktu sebelum mencapai batas tersebut; atau
- Sesuaikan : Pengguna dapat menyesuaikan batas waktu sebelum mencapai batas tersebut, dengan waktu tambahan yang setidaknya sepuluh kali lebih panjang dari setelan standar; atau
- Perpanjangan : Pengguna diperingati ketika batas waktu habis dan diberikan waktu setidaknya 20 detik untuk menambah batas waktu dengan perintah sederhana (misalnya, tekan tombol spasi), dan pengguna dapat menambah batas waktu setidaknya sepuluh kali lipat; atau

- Perkecualian waktu riil : Batas waktu diperlukan sebagai bagian dari kejadian waktu riil (misalnya, lelang), dan mustahil untuk menyediakan alternatif untuk batas waktu; atau
- Perkecualian esensial : Batas waktu esensial dan perpanjangan batas ini menyalahi inti dari kegiatan tersebut; atau
- Perkecualian 20 jam : Batas waktu yang diberikan lebih dari 20 jam.

Kriteria Sukses 2.2.2 Pause, Stop, Hide

(Level A)

Untuk informasi yang bergerak, berkedip, bergulir, atau diperbarui secara otomatis, semua hal berikut ini benar :

- Bergerak, berkedip, bergulir : Untuk informasi yang bergerak, berkedip, bergulir yang (1) mulainya otomatis, (2) berlangsung lebih dari lima detik, dan (3) ditampilkan paralel dengan konten lain, ada mekanisme bagi pengguna untuk memberi jeda, memberhentikan, atau menyembunyikan informasi tersebut; kecuali jika aktivitas bergerak, berkedip, atau bergulir tersebut merupakan bagian dari aktivitas yang esensial; dan
- Diperbarui otomatis : Untuk informasi mana pun yang diperbarui secara otomatis, yaitu yang (1) mulainya otomatis dan (2) ditampilkan paralel dengan konten lain, ada mekanisme bagi pengguna untuk memberi jeda, memberhentikan, atau menyembunyikan informasi tersebut; kecuali jika pembaruan otomatis tersebut merupakan bagian dari aktivitas yang esensial; dan

Kriteria Sukses 2.2.3 No Timing

(Level AAA)

Waktu bukanlah bagian esensial dari kejadian atau aktivitas yang disajikan oleh konten, kecuali untuk media yang disingkronisasi non-interaktif dan kejadian waktu riil.

Kriteria Sukses 2.2.4 Interruptions

(Level AAA)

Interupsi dapat ditunda oleh pengguna, kecuali interupsi yang melibatkan keadaan darurat.

Kriteria Sukses 2.2.5 Re-authenticating

(Level AAA)

Ketika sesi autentikasi berakhir, pengguna dapat melanjutkan aktivitas tanpa kehilangan data setelah autentikasi ulang.

Kriteria Sukses 2.2.6 Timeouts

(Level AAA)

Pengguna diperingatkan tentang waktu ketidakaktifan yang dapat menyebabkan kehilangan data, kecuali jika data tersebut disimpan lebih dari 20 jam ketika pengguna tidak melakukan tindakan apapun.

Kriteria Sukses 2.3.1 Three Flashes or Below Threshold

(Level A)

Halaman web tidak mengandung apapun yang berkelip lebih dari tiga kali dalam periode satu detik, atau kelipan berada dibawah batas umum kelipan dan kelipan merah

Kriteria Sukses 2.3.2 Three Flashes

(Level AAA)

Halaman web tidak mengandung apapun yang berkelip lebih dari tiga kali dalam periode satu detik

Kriteria Sukses 2.3.3 Animation from Interactions

(Level AAA)

Animasi gerak yang dipicu oleh interaksi dapat dinonaktifkan, kecuali jika animasi itu penting untuk fungsionalitas atau informasi yang sedang disampaikan.

Kriteria Sukses 2.4.1 Bypass Blocks

(Level A)

Tersedianya sebuah mekanisme untuk meloncati area konten yang diulang-ulang pada berbagai halaman web.

Kriteria Sukses 2.4.2 Page Titled

(Level A)

Halaman web memiliki judul yang menggambarkan topik atau tujuan.

Kriteria Sukses 2.4.3 Focus Order

(Level A)

Jika halaman web dapat dinavigasi secara berurutan dan urutan navigasi memengaruhi makna atau operasi, maka komponen yang dapat dapat menerima fokus akan menerima fokus dalam urutan yang menjaga makna dan pengoperasian.

Kriteria Sukses 2.4.4 Link Purpose (In Context)

(Level A)

Tujuan setiap tautan dapat ditentukan dari teks tautan saja atau dari kombinasi teks tautan beserta konteks tautan yang ditentukan secara pemrograman, kecuali jika tautan tersebut bersifat ambigu bagi pengguna secara umum.

Kriteria Sukses 2.4.5 Multiple Ways

(Level AA)

Ada berbagai cara untuk menemukan halaman web dalam satu set halaman web kecuali halaman web adalah hasil dari, atau langkah dalam suatu proses.

Kriteria Sukses 2.4.6 Headings and Labels

(Level AA)

Judul dan label menjelaskan topik atau tujuan.

Kriteria Sukses 2.4.7 Focus Visible

(Level AA)

Setiap antarmuka pegguna yang dapat dioperasikan dengan keyboard memiliki mode pengoperasian di mana indikator fokus keyboard terlihat jelas.

Kriteria Sukses 2.4.8 Location

(Level AAA)

Informasi mengenai lokasi pengguna dalam satu set halaman web tersedia.

Kriteria Sukses 2.4.9 Link Purpose (Link Only)

(Level AAA)

Tersedianya sebuah mekanisme untuk memungkinkan tujuan setiap tautan diidentifikasi dari teks tautan saja, kecuali jika tujuan tersebut ambigu bagi pengguna secara umum.

Kriteria Sukses 2.4.10 Section Headings

(Level AAA)

Judul bagian digunakan untuk mengatur konten.

Kriteria Sukses 2.5.1 Pointer Gestures

(Level A)

Semua fungsionalitas yang menggunakan gerakan *multipoint* atau berbasis jalur untuk operasi dapat dioperasikan dengan *pointer* tunggal tanpa gestur berbasis jalur, kecuali jika multipoint atau gestur berbasis jalur sangat penting.

Kriteria Sukses 2.5.2 Pointer Cancellation

(Level A)

Untuk fungsionalitas yang dapat dioperasikan menggunakan pointer tunggal, setidaknya salah satu dari yang berikut ini benar :

- Tidak Ada *Down-Event* : *Down-Event* dari *pointer* tidak dipakai untuk eksekusi bagian dari fungsi;
- Gagalkan atau Batalkan : Keberhasilan fungsi pada *up-event*, dan tersedia mekanisme untuk mengagalkan fungsi sebelum berhasil atau membatalkan fungsi setelah berhasil;
- *Up Reversal* :
- Esensial : Keberhasilan fungsi saat *down-event* esensial.

Kriteria Sukses 2.5.3 Label in Name

(Level A)

Untuk komponen antarmuka pengguna dengan label yang menyertakan teks atau gambar teks, nama tersebut berisi teks yang ditampilkan secara visual.

Kriteria Sukses 2.5.4 Motion Actuation

(Level A)

Fungsi yang dapat dioperasikan oleh gerakan perangkat atau gerakan pengguna juga dapat dioperasikan oleh komponen antarmuka pengguna dan merespons gerakan dapat dinonaktifkan untuk mencegah pergerakan tidak disengaja, kecuali ketika :

- Antarmuka yang Didukung : Gerakan digunakan untuk mengoperasikan fungsi melalui antarmuka yang mendukung aksesibilitas;
- Esensial : Gerakan sangat esensial untuk fungsi dan jika menonaktifkannya maka membuat aktivitas tersebut tidak valid.

Kriteria Sukses 2.5.5 Target Size

(Level AAA)

Ukuran target untuk *pointer* masukan setidaknya 44 *css* piksel kecuali jika :

- Setara : Target tersedia melalui tautan atau kontrol yang setara pada halaman yang sama berukuran setidaknya 44 *css* piksel;
- *Inline* : Target terdapat pada kalimat atau blok teks;
- Kontrol Agen Pengguna : Ukuran dari target ditentukan oleh agen pengguna dan tidak diubah oleh pembuat web;
- Esensial : Presentasi visual dari target sangat penting untuk informasi yang disampaikan.

Kriteria Sukses 2.5.6 Concurrent Input Mechanisms

(Level AAA)

Konten web tidak membatasi penggunaan modalitas masukan yang tersedia pada platform kecuali jika pembatasan itu penting, diperlukan untuk memastikan keamanan konten, atau diperlukan untuk menghormati pengaturan pengguna.

2.1.3 *Understandable*

Informasi dan pengoperasian antarmuka pengguna harus dapat dimengerti.

Kriteria Sukses 3.1.1 Language of Page

(Level A)

Standar bahasa manusia dari setiap halaman web dapat ditentukan secara pemrograman.

Kriteria Sukses 3.1.2 Language of Parts

(Level AA)

Bahasa manusia dari setiap bait atau frasa dalam konten dapat ditentukan secara pemrograman kecuali untuk nama diri, istilah teknis, kata-kata dari bahasa yang tidak ditentukan, dan kata-kata atau frasa yang telah menjadi bagian dari bahasa sehari-hari dari teks yang ada di sekelilingnya.

Kriteria Sukses 3.1.3 Unusual Words

(Level AAA)

Tersedianya mekanisme untuk mengidentifikasi definisi kata atau frasa tertentu yang digunakan dengan cara yang tidak biasa atau terbatas, termasuk idiom dan jargon.

Kriteria Sukses 3.1.4 Abbreviations

(Level AAA)

Tersedianya mekanisme untuk mengidentifikasi kepanjangan dari singkatan.

Kriteria Sukses 3.1.5 Reading Level

(Level AAA)

Ketika teks membutuhkan kemampuan membaca lebih maju daripada tingkat pendidikan menengah bawah setelah penghapusan nama diri dan jabatan, konten tambahan atau versi yang tidak memerlukan kemampuan membaca lebih maju daripada tingkat pendidikan menengah bawah harus tersedia.

Kriteria Sukses 3.1.6 Pronunciation

(Level AAA)

Tersedianya mekanisme untuk mengidentifikasi pengucapan kata tertentu ketika makna kata tersebut, dalam konteksnya, bersifat ambigu tanpa mengetahui pengucapannya.

Kriteria Sukses 3.2.1 On Focus

(Level A)

Tidak ada perubahan konteks jika komponen antarmuka pengguna menerima fokus.

Kriteria Sukses 3.2.2 On Input

(Level A)

Mengubah pengaturan komponen antarmuka pengguna mana pun tidak otomatis menyebabkan perubahan konteks kecuali jika pengguna telah diperingati tentang hal ini sebelum menggunakan komponen.

Kriteria Sukses 3.2.3 Consistent Navigation

(Level AA)

Mekanisme navigasi yang diulang pada beberapa halaman web dalam satu set halaman web muncul dalam urutan relatif yang sama setiap kali diulang, kecuali jika perubahan dilakukan oleh pengguna.

Kriteria Sukses 3.2.4 Consistent Identification

(Level AA)

Komponen yang memiliki fungsi yang sama dalam satu set halaman web diidentifikasi secara konsisten.

Kriteria Sukses 3.2.5 Change on Request

(Level AAA)

Perubahan konteks hanya terjadi oleh permintaan pengguna atau ada mekanisme tersedia untuk menonaktifkan perubahan tersebut.

Kriteria Sukses 3.3.1 Error Identification

(Level A)

Jika kesalahan masukan terdeteksi secara otomatis, *item* yang salah diidentifikasi dan kesalahan tersebut dijelaskan kepada pengguna dalam teks.

Kriteria Sukses 3.3.2 Labels or Instructions

(Level A)

Label atau instruksi diberikan saat konten membutuhkan masukan pengguna.

Kriteria Sukses 3.3.3 Error Suggestion

(Level AA)

Jika kesalahan masukan terdeteksi secara otomatis dan saran untuk mengoreksi diketahui, maka saran tersebut diberikan kepada pengguna, kecuali jika itu akan membahayakan keamanan atau tujuan konten.

Kriteria Sukses 3.3.4 Error Prevention (Legal, Financial, Data)

(Level AA)

Untuk halaman web yang menyebabkan komitmen legal atau transaksi finansial bagi pengguna, dan memodifikasi atau menghapus data yang dapat dikontrol pengguna dalam sistem penyimpanan data, atau yang mengirimkan tanggapan tes pengguna, setidaknya salah satu dari yang berikut ini benar :

- Bisa dibatalkan : Data yang dikirim bisa dibatalkan;
- Diperiksa : Data yang dimasukkan oleh pengguna dapat diperiksa untuk eror masukan dan pengguna dipersilakan untuk mengoreksinya;
- Dikonfirmasi : Tersedianya mekanisme untuk meninjau, mengonfirmasi, dan mengoreksi informasi sebelum informasi tersebut dikirim.

Kriteria Sukses 3.3.5 Help

(Level AAA)

Tersedianya bantuan pada konteks yang sedang berjalan.

Kriteria Sukses 3.3.6 Error Prevention (All)

(Level AAA)

Untuk halaman web yang mengharuskan pengguna untuk mengirimkan informasi, setidaknya salah satu dari yang berikut ini benar :

- Bisa dibatalkan : Data yang dikirim bisa dibatalkan;
- Diperiksa : Data yang dimasukkan oleh pengguna dapat diperiksa untuk eror masukan dan pengguna dipersilakan untuk mengoreksinya;
- Dikonfirmasi : Tersedianya mekanisme untuk meninjau, mengonfirmasi, dan mengoreksi informasi sebelum informasi tersebut dikirim.

2.1.4 *Robust*

Konten harus cukup handal sehingga dapat ditafsirkan oleh berbagai agen pengguna, termasuk teknologi alat bantu.

Kriteria Sukses 4.1.1 Parsing

(Level A)

Dalam konten yang diimplementasikan menggunakan bahasa *markup*, elemen memiliki tag awal dan akhir yang lengkap, elemen disusun berlapis sesuai dengan spesifikasinya, elemen tidak mengandung atribut duplikat, dan setiap ID unik, kecuali jika spesifikasi mengizinkan fitur ini.

Kriteria Sukses 4.1.2 Name, Role, Value

(Level A)

Untuk semua komponen antarmuka pengguna (termasuk tetapi tidak terbatas pada: elemen formulir, tautan, dan komponen yang dihasilkan oleh skrip), nama dan peran dapat ditentukan secara pemrograman; keadaan, properti, dan nilai-nilai yang dapat diatur oleh pengguna dapat diatur secara terprogram; dan pemberitahuan perubahan pada *item-item* ini tersedia untuk agen pengguna, termasuk teknologi bantu.

Kriteria Sukses 4.1.3 Status Messages

(Level AA)

Dalam konten yang diimplementasikan menggunakan bahasa *markup*, pesan status dapat ditentukan secara terprogram melalui peran atau sifat sedemikian rupa sehingga dapat disajikan kepada pengguna dengan teknologi bantuan tanpa menerima fokus.

2.2 SharIF Judge

SharIF Judge adalah sebuah aplikasi gratis dan *open source* untuk menilai code berbahasa *C , C++, Java* dan *Python*. *SharIF Judge* adalah pencabangan dari *Sharif Judge* yang telah dibuat oleh Mohammed Javad Naderi. Versi dari pencabangan ini memuat fitur baru yang diperlukan oleh jurusan teknik informatika UNPAR. Aplikasi ini dibuat menggunakan PHP (*CodeIgnitor framework*) dan bagian backendnya dibuat dengan *BASH*.

2.2.1 Instalasi

Berikut adalah persyaratan dan langkah-langkah instalasi yang perlu dilakukan untuk menjalankan *SharIF Judge* :

Persyaratan

Untuk menjalankan *SharIF Judge*, diperlukan sebuah *Linux server* dengan persyaratan berikut :

- *Webserver* menjalankan PHP versi 5.3 atau lebih baru
- Pengguna dapat menjalankan PHP melalui *command line*. Pada *Ubuntu*, pengguna perlu menginstal paket *php5-cli*
- *MySql database* (dengan ekstensi untuk PHP) atau *PostgreSql database*
- PHP harus diberikan akses untuk menjalankan perintah menggunakan fungsi *shell_exec*. Contoh seperti perintah berikut :

```
echo shell_exec("php -v");
```

- Perkakas yang digunakan untuk melakukan proses kompilasi dan menjalankan kode yang dikumpulkan (*gcc, g++, javac, java, python2, python3 commands*)
- *Perl* lebih baik diinstall untuk alasan ketepatan waktu, batas memori dan memaksimalkan batas ukuran pada *output* kode yang dikirim.

Instalasi

- Mengunduh versi terakhir dari *SharIF Judge* dan *unpack* hasil unduhan di direktori *public html*
- Memindahkan folder *system* dan *application* ke luar *public directory*, dan masukkan *path* lengkap di *file index.php*

```
$system_path = '/home/mohammad/secret/system';
$application_folder = '/home/mohammad/secret/application';
```

- Membuat sebuah database *Mysql* atau *PostgreSql* untuk Sharif Judge. Jangan menginstall paket koneksi database untuk *C/C++, Java*, atau *Python*
- Mengatur koneksi *database* di *file application/config/database.php*. Pengguna dapat menggunakan awalan untuk nama tabel.

```
/*  Enter database connection settings here:  */
'dbdriver' => 'postgre',    // database driver (mysqli, postgre)
'hostname' => 'localhost',  // database host
'username' => `,           // database username
'password' => `,           // database password
'database' => `,           // database name
'dbprefix' => 'shj_',       // table prefix
/*********************************************/
```

- Mengatur *RADIUS server* dan *mail server* pada *file application/config/secrets.example.php* dan simpan dengan nama *secrets.php*
- Membuat direktori *application/cache/Twig* agar dapat ditulis oleh PHP
- Membuka halaman utama *SharIF Judge* pada *web browser* dan ikuti proses instalasi berikutnya
- *Log in* menggunakan akun *admin*
- Memindahkan folder *tester* dan *assigments* ke luar public directory lalu simpan path lengkap di halaman *Settings*. Dua folder tersebut harus dapat ditulis oleh PHP. File-file yang diunggah akan disimpan di folder *assigments* sehingga tidak dapat diakses publik.

2.2.2 *Clean URLs*

Secara standar, *index.php* merupakan bagian dari seluruh *URLs* yang ada pada *SharIF Judge*. Berikut contoh *URLs SharIF Judge* :

- `http://example.mjnaderi.ir/index.php/dashboard`
- `http://example.mjnaderi.ir/index.php/users/add`

Pengguna dapat menghilangkan *index.php* di atas dan memiliki *URLs* yang baik jika sistem pengguna mendukung *URL rewriting*. *URL rewriting* merupakan proses mengubah parameter yang terdapat pada *URL*. Berikut adalah contoh *URL* yang telah melewati proses *URL rewriting* :

- `http://example.mjnaderi.ir/dashboard`
- `http://example.mjnaderi.ir/users/add`

Cara memakai *Clean URLs*

Pengguna dapat melakukan langkah berikut untuk menggunakan *clean URL* :

- Mengubah nama *file .htaccess2* menjadi *.htaccess* yang ada di direktori utama *SharIF Judge*. *.htaccess* merupakan sebuah file konfigurasi yang digunakan pada *web server*
- Mengubah *$config['index_page'] = 'index.php'*; menjadi *$config['index_page'] =* ''; pada file *application/config/config.php*

2.2.3 Users

Pada *SharIF Judge*, *users* dibagi menjadi 4 *role*. Keempat *role* tersebut adalah *Admins, Head Instructor, Instructor*, dan *Students*. Tabel 2.1 menunjukkan *level* yang ada untuk setiap *role*.

Tabel 2.1: *User Roles Table*

User Role	***User Level***
Admin	3
Head Instructor	2
Instructor	1
Student	0

Setiap *users* dapat melakukan aksi yang berbeda-beda. Aksi yang dapat dilakukan para *users* akan disesuaikan pada tingkat masing-masing. Tabel 2.2 menunjukkan aksi yang dapat dilakukan pada setiap *role*.

Tabel 2.2: *Permission Table*

Aksi	*Admin*	*Head Instructor*	*Instructor*	*Student*
Mengubah *Settings*	✓	×	×	×
Menambah/Menghapus *users*	✓	×	×	×
Mengubah Peran *users*	✓	×	×	×
Menambah/Menghapus/Mengubah *Assignment*	✓	✓	×	×
Mengunduh *Test*	✓	✓	×	×
Menambah/Menghapus/Mengubah Notifikasi	✓	✓	×	×
Rejudge	✓	✓	×	×
Melihat/*Pause*/Melanjutkan/*Submission Queue*	✓	✓	×	×
Mendeteksi Kode yang Mirip	✓	✓	×	×
Melihat Semua Kode	✓	✓	✓	×
Mengunduh Kode Final	✓	✓	✓	×
Memilih *Assignment*	✓	✓	✓	✓
Submit	✓	✓	✓	✓

Pengguna dapat menambahkan *users* dengan masuk ke bagian *Add Users* di halaman *Users*. Pengguna harus mengisi semua informasi yang ada pada *text area*. Baris yang diawali dengan # adalah komentar. Setiap baris lainnya mewakili pengguna dengan sintaksis berikut:

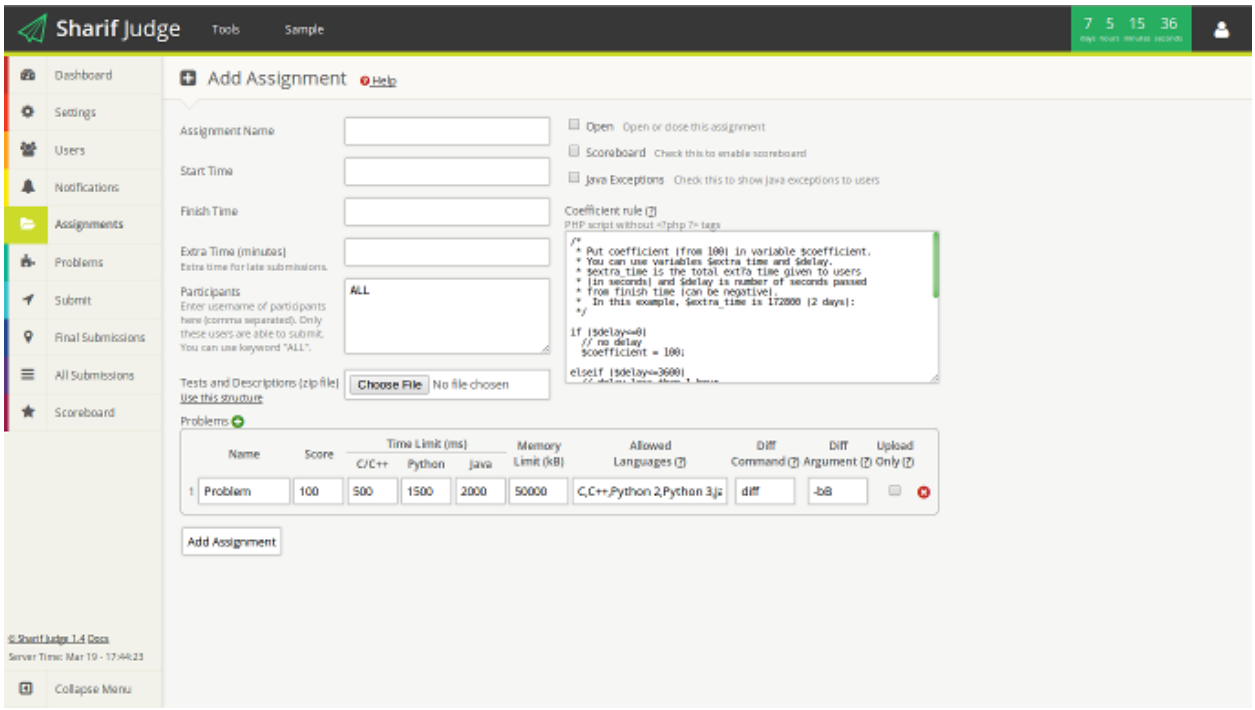
```
USERNAME,EMAIL,DISPLAY-NAME,PASSWORD,ROLE

* Usernames may contain lowercase letters or numbers and must be between 3 and 20
    characters in length.
* Passwords must be between 6 and 30 characters in length.
* You can use RANDOM[n] for password to generate random n-digit password.
* ROLE must be one of these: ‘admin‘, ‘head_instructor‘, ‘instructor‘, ‘student‘
```

Berikut adalah contohnya:

```
# This is a comment!
# This is another comment!
instructor instructor@sharifjudge.ir 123456 head_instructor
instructor2 instructor2@sharifjudge.ir random[7] instructor
student1 st1@sharifjudge.ir random[6] student
student2 st2@sharifjudge.ir random[6] student
student3 st3@sharifjudge.ir random[6] student
student4 st4@sharifjudge.ir random[6] student
student5 st5@sharifjudge.ir random[6] student
student6 st6@sharifjudge.ir random[6] student
student7 st7@sharifjudge.ir random[6] student
```

2.2.4 Menambah Assignment

Pengguna dapat menambahkan *assignment* dengan cara masuk ke bagian *Add* di halaman *assigments*. Gambar 2.1 menunjukkan contoh tampilan halaman untuk menambah *assignment*.



Gambar 2.1: Tampilan Halaman *Assignments*

Berikut ini adalah pengaturan yang terdapat pada halaman *Add Assignments*:

- *Assignment Name*
  Nama *assignment* yang akan ditampilkan dalam daftar *assignment*.
- *Start Time*
  User tidak dapat mengumpulkan assignment sebelum *"Start Time"*. Format yang digunakan untuk pengaturan start time adalah *MM/DD/YYYY HH:MM:SS*. Contohnya: 08/31/2013 12:00:00
- *Finish Time, Extra Time*
  Peserta tidak dapat mengumpulkan *assignment* setelah *Finish Time + Extra Time*. *Assignment*

yang telat akan dikalikan dengan koefisien tertentu. Pengguna harus menulis *script* PHP untuk menghitung koefisien pada bidang *"Coefficient Rule"*. Format yang digunakan untuk pengaturan *finish time* adalah *MM/DD/YYYY HH:MM:SS*. Contoh: *08/31/2013 23:59:59*. "*Extra Time*" akan terhitung dalam satuan menit. Pengguna juga dapat menggunakan operator aritmatika seperti *, -, +, /. Contohnya *120* (2 jam) atau *48*60* (2 hari).

- *Participants*
  Pengguna dapat menggunakan kata kunci *ALL* pada kolom *Participants* untuk mengijinkan seluruh peserta agar dapat mengumpulkan *assignment*. Untuk membatasi peserta tertentu, pengguna dapat memasukan *username* peserta pada kolom *Participants*. Setiap *username* dapat dipisahkan menggunakan tanda koma. Contoh: *admin, instructor1, instructor2, student1, student2, student3, student4*.

- *Test*
  Pengguna dapat mengunggah *test cases* dalam bentuk *zip file* dengan struktur yang sudah ditentukan.

- *Open*
  Pengguna dapat membuka atau menutup *assignment* untuk *user students* menggunakan pilihan ini. Jika pengguna menutup *assignment*, *non-student users* masih dapat mengumpulkan *assignment*.

- *Scoreboard*
  Pengguna dapat mengaktifkan atau menonaktifkan papan nilai dengan menggunakan pilihan ini.

- *Java Exceptions*
  Pengguna dapat mengaktifkan dan menonaktifkan *java exception* yang ditunjukan kepada *students*. Perubahan pada pilihan ini tidak memengaruhi kode yang sebelumnya sudah dinilai. Nama *exception* akan muncul jika pada *file* pathtester/java_exceptions_list berisikan nama *exception* tersebut. Berikut adalah hasil *exception* yang ditunjukan jika pengguna mengaktifkan pengaturan *Java Exceptions*:

```
Test 1
ACCEPT
Test 2
Runtime Error (java.lang.ArrayIndexOutOfBoundsException)
Test 3
Runtime Error (java.lang.ArrayIndexOutOfBoundsException)
Test 4
ACCEPT
Test 5
ACCEPT
Test 6
ACCEPT
Test 7
ACCEPT
Test 8
Runtime Error (java.lang.ArrayIndexOutOfBoundsException)
Test 9
Runtime Error (java.lang.StackOverflowError)
Test 10
Runtime Error (java.lang.ArrayIndexOutOfBoundsException)
```

- *Archieved Assignment*
  Jika fitur ini diaktifkan maka *assignment* akan dibuat dengan *Finish Time* 2038-01-18 00:00:00 ($UTC+7$) dalam kata lain pengguna memiliki waktu yang tidak terbatas untuk mengumpulkan *assignment* tersebut.

- *Coefficient Rule*
  Pengguna dapat menulis *script* PHP pada bagian ini. Pengguna harus memasukan koefisien (dari 100) pada variabel *$coefficient*. Pengguna dapat menggunakan variabel *$extra_time* dan *$delay*. *$extra_time* merupakan total waktu ekstra yang diberikan kepada *users* dalam satuan detik dan *$delay* merupakan jumlah detik berlalu dari waktu selesai (dapat negatif). *Script* PHP pada bagian ini tidak boleh mengandung *tags <?php , <? , ?>*. Berikut contoh *$extra_time* 172800 (2 hari):

```
if ($delay<=0)
// no delay
$coefficient = 100;

elseif ($delay<=3600)
// delay less than 1 hour
$coefficient = ceil(100-((30*$delay)/3600));

elseif ($delay<=86400)
// delay more than 1 hour and less than 1 day
$coefficient = 70;

elseif (($delay-86400)<=3600)
// delay less than 1 hour in second day
$coefficient = ceil(70-((20*($delay-86400))/3600));

elseif (($delay-86400)<=86400)
// delay more than 1 hour in second day
$coefficient = 50;

elseif ($delay > $extra_time)
// too late
$coefficient = 0;
```

- *Time Limit*
  Pengguna dapat mengatur batas waktu untuk menjalankan kode dalam satuan milisekon. Program yang ditulis menggunakan *Python* dan *Java* biasanya lebih lambat dari *C/C++*. Oleh karena itu *Python* dan *Java* membutuhkan waktu yang lebih.

- *Memory Limit*
  Pengguna dapat mengatur batas memori dalam satuan *kilobytes*, namun penggunaan *Memory Limit* tidak terlalu akurat.

- *Allowed Languages*
  Pengguna dapat mengatur bahasa untuk setiap *problem* (dipisahkan menggunakan koma). Bahasa yang tersedia seperti *C, C++, Java, Python 2, Python 3, zip, PDF*. Pengguna dapat

menggunakan *zip* atau *PDF* jika mengaktifkan pilihan *Upload Only*. Contoh: *C, C++ , zip* atau *Python 2,Python 3* atau *Java ,C*.

- *Diff Command*
  Command ini digunakan untuk membandingkan keluaran dengan keluaran yang benar. Secara standar *SharIF Judge* menggunakan *diff*, namun pengguna dapat mengubah *command* pada bagian ini. Bidang ini tidak boleh mengandung spasi.

- *Diff Arguments*
  Pengguna dapat mengatur argumen dari *Diff Command* disini. Pengguna dapat melihat *man diff* untuk melihat daftar lengkap *diff* argumen. *SharIF Judge* menambahkan dua pilihan baru yaitu *ignore* dan *identical*. *Ignore* akan menghiraukan semua baris baru dan spasi. *Identical* tidak akan menghiraukan apapun namun keluaran dari file yang dikumpulkan harus identik dengan keluaran *test case* agar dapat diterima.

- *Upload Only*
  Jika pengguna mengatur *problem* sebagai *Upload-Only*, maka *SharIF Judge* tidak akan menilai *assignment* pada *problem* tersebut. Pengguna dapat menggunakan *zip* dan *PDF* pada *allowed languages* jika mengaktifkan pilihan ini.

2.2.5 *Sample Assignment*

Berikut adalah contoh *assignment* untuk testing pada *SharIF Judge*. Untuk menambah *assignment* pengguna dapat menekan *Add* pada halaman *Assignments*.

1. *Problem* 1 (Penjumlahan)
   Program pengguna akan menerima bilangan integer n, kemudian menerima masukkan sebanyak n buah bilangan integer dan akan menampilkan hasil penjumlahan n buah bilangan integer tersebut. Tabel 2.3 menunjukkan contoh hasil masukkan dan keluaran *problem 1*.

Tabel 2.3: *Problem* 1 (Penjumlahan)

Sample Input	Sample Output
5 54 78 0 4 9	145

2. *Problem* 2 (*Max*)
   Program pengguna akan menerima bilangan integer n, kemudian menerima masukkan sebanyak n buah bilangan integer dan akan menampilkan hasil penjumlahan dari dua nilai tertinggi. Tabel 2.4 menunjukkan contoh hasil masukkan dan keluaran *problem 2*.

Tabel 2.4: *Problem* 2 (*Max*)

Sample Input	Sample Output
7 162 173 159 164 181 158 175	356

3. *Problem* 3 *Upload*
   Program pengguna akan menerima sebuah *file C* atau *zip*. *Problem* ini menggunakan pilihan *Upload Only* sehingga tidak dinilai oleh *SharIF Judge*.

Pengguna dapat menemukan *file zip* pada folder *Assignment*. Berikut susunan pohon dari tiga *problem* di atas:

```
.
|-- p1
|   |-- in
|   |   |-- input1.txt
|   |   |-- input2.txt
|   |   |-- input3.txt
|   |   |-- input4.txt
|   |   |-- input5.txt
|   |   |-- input6.txt
|   |   |-- input7.txt
|   |   |-- input8.txt
|   |   |-- input9.txt
|   |   --- input10.txt
|   |-- out
|   |   --- output1.txt
|   |-- tester.cpp
|   --- desc.md
|-- p2
|   |-- in
|   |   |-- input1.txt
|   |   |-- input2.txt
|   |   |-- input3.txt
|   |   |-- input4.txt
|   |   |-- input5.txt
|   |   |-- input6.txt
|   |   |-- input7.txt
|   |   |-- input8.txt
|   |   |-- input9.txt
|   |   --- input10.txt
|   |-- out
|   |   |-- output1.txt
|   |   |-- output2.txt
|   |   |-- output3.txt
|   |   |-- output4.txt
|   |   |-- output5.txt
|   |   |-- output6.txt
|   |   |-- output7.txt
|   |   |-- output8.txt
|   |   |-- output9.txt
|   |   --- output10.txt
|   |-- desc.md
|   --- Problem2.pdf
|-- p3
|   --- desc.md
--- SampleAssignment.pdf
```

Problem 1 menggunakan metode *"Tester"* untuk memeriksa keluaran, sehingga memiliki *file tester.cpp (Tester Script). Problem* 2 menggunakan metode *Output Comparison* untuk memeriksa keluaran, sehingga memiliki dua folder (*in* dan *out*) yang berisi *test case. Problem* 3 menggunakan pilihan *Upload-Only.*

2.2.6 *Test Structure*

Saat menambah *assignments*, pengguna harus menyediakan *file zip* yang berisi *test cases*. *File zip* ini berisi folder-folder untuk setiap *problem*. Pengguna harus memberi nama folder sesuai aturan seperti *p1, p2, p3*, dst. *Assignment* yang menggunakan pilihan *Upload-Only* tidak membutuhkan *folder*.

Metode Pengecekan

SharIF Judge memiliki dua metode pengecekan untuk setiap *problem* yaitu metode *"Input/Output Comparison"* dan metode *"Tester"*.

1. Metode *Input/Output Comparison*
   Dalam metode ini pengguna harus memasukkan beberapa *file input* dan beberapa *output* ke dalam folder *problem*. *SharIF Judge* akan memasukkan nilai dari *file input* ke kode *users* dan membandingkan hasil keluaran dari kode *users* dengan *file output*. *Input files* harus berada dalam folder *"in"* dengan nama *input1.txt, input2.txt,* dst. *Output files* harus berada dalam folder *"out"* dengan nama *output1.txt, output2.txt,* dst.

2. Metode *Tester*
   Dalam metode ini pengguna harus memasukkan beberapa *file input*, sebuah *file C++ (tester.cpp)*, dan beberapa *file output*. *SharIF Judge* akan memasukkan nilai dari *file input* ke kode *users* dan mengambil keluaran dari kode *users*. *tester.cpp* akan mengambil nilai dari *file input, file output,* dan keluaran *users*. Jika keluaran dari kode *users* benar maka akan mengembalikan nilai 0, sedangkan jika salah akan mengembalikan nilai 1.

   Pengguna dapat menggunakan kode pada listing 2.1 sebagai templat untuk menulis *tester.cpp*:

Listing 2.1: Contoh kode *tester.cpp*

```
/*
* tester.cpp
*/

#include <iostream>
#include <fstream>
#include <string>
using namespace std;
int main(int argc, char const *argv[])
{

        ifstream test_in(argv[1]);    /* This stream reads from test's input
    file   */
        ifstream test_out(argv[2]);   /* This stream reads from test's
   output file  */
        ifstream user_out(argv[3]);   /* This stream reads from user's
   output file  */

        /* Your code here */
        /* If user's output is correct, return 0, otherwise return 1       */


        ...

}
```

Contoh *File*

Pengguna dapat menemukan contoh *file* penguji dalam folder *Assignments*. Berikut contoh susunan pohon dari *file* tersebut:

```
.
|-- p1
|   |-- in
|   |   |-- input1.txt
|   |   |-- input2.txt
|   |   |-- input3.txt
|   |   |-- input4.txt
|   |   |-- input5.txt
|   |   |-- input6.txt
|   |   |-- input7.txt
|   |   |-- input8.txt
|   |   |-- input9.txt
|   |   --- input10.txt
|   |-- out
|   |   --- output1.txt
|   --- tester.cpp
--- p2
|-- in
|   |-- input1.txt
|   |-- input2.txt
|   |-- input3.txt
|   |-- input4.txt
|   |-- input5.txt
|   |-- input6.txt
|   |-- input7.txt
|   |-- input8.txt
|   |-- input9.txt
|   --- input10.txt
--- out
|-- output1.txt
|-- output2.txt
|-- output3.txt
|-- output4.txt
|-- output5.txt
|-- output6.txt
|-- output7.txt
|-- output8.txt
|-- output9.txt
--- output10.txt
```

Problem 1 menggunakan metode *"Tester"* untuk mengecek hasil keluarannya sehingga memiliki *file tester.cpp*. Listing 2.2 menunjukkan isi dari *file tester.cpp* untuk *Problem* 1:

Listing 2.2: Isi *File tester.cpp* untuk *Problem* 1

```
/*
```

```
* tester.cpp
*/

#include <iostream>
#include <fstream>
#include <string>
using namespace std;
int main(int argc, char const *argv[])
{

        ifstream test_in(argv[1]);    /* This stream reads from test's input file
       */
        ifstream test_out(argv[2]);   /* This stream reads from test's output
    file  */
        ifstream user_out(argv[3]);   /* This stream reads from user's output
    file  */

        /* Your code here */
        /* If user's output is correct, return 0, otherwise return 1       */
        /* e.g.: Here the problem is: read n numbers and print their sum:  */

        int sum, user_output;
        user_out >> user_output;

        if ( test_out.good() ) // if test's output file exists
        {
                test_out >> sum;
        }
        else
        {
                int n, a;
                sum=0;
                test_in >> n;
                for (int i=0 ; i<n ; i++){
                        test_in >> a;
                        sum += a;
                }
        }

        if (sum == user_output)
                return 0;
        else
                return 1;

}
```

Problem 2 menggunakan metode *"Input/Output Comparison"* untuk mengecek hasil keluarannya sehingga memiliki dua folder *in* dan *out* yang berisi *test cases*. *Problem* 3 menggunakan pilihan *Upload-Only* sehingga tidak memiliki folder apapun.

2.2.7 Deteksi Kecurangan

SharIF Judge menggunakan *Moss* untuk mendeteksi kode yang mirip. *Moss (Measure Of Software Similarity)* merupakan sistem otomatis untuk menentukan kemiripan program. Pada saat ini, aplikasi utama *Moss* telah digunakan untuk mendeteksi plagiarisme pada kelas *programming*. Pengguna dapat mengirimkan kode final (yang dipilih oleh *students* sebagai *Final Submission*) ke server *Moss* dengan satu klik.

Sebelum menggunakan *Moss* ada beberapa hal yang perlu diperhatikan yaitu:

- Pengguna harus mendapatkan *Moss user id* dan mengaturnya di *SharIF Judge*. Untuk mendapatkan *Moss user id*, pengguna harus terlebih dahulu mendaftar pada halaman `http://theory.stanford.edu/~aiken/moss/`. Pengguna akan menerima sebuah *email* yang berisi *script perl*. Dalam *script* tersebut terdapat *Moss user id*.

  Listing 2.3 menunjukkan potongan *script perl* yang mengandung *user id*:

  Listing 2.3: Potongan *script perl*

```
...

$server = 'moss.stanford.edu';
$port = '7690';
$noreq = "Request not sent.";
$usage = "usage: moss [-x] [-l language] [-d]
[-b basefile1] ... [-b basefilen] [-m #]
[-c \"string\"] file1 file2 file3 ...";

#
# The userid is used to authenticate your queries to the server;
don't change it!
#
$userid=YOUR_MOSS_USER_ID;

#
# Process the command line options.  This is done in a non-standard
# way to allow multiple -b's.
#
$opt_l = "c";   # default language is c
$opt_m = 10;
$opt_d = 0;

...
}
```

  User id yang terdapat pada potongan *script perl* di atas, dapat digunakan pada *SharIF Judge* pada halaman *Moss*. *SharIF Judge* akan menggunakan *user id* tersebut di *Moss perl script*. *Perl* harus terinstal pada server agar dapat menggunakan *Moss*.

- Pengguna dianjurkan untuk mendeteksi kode yang mirip setelah waktu *assignment* selesai, karena para peserta masih dapat mengubah *Final Submissions* masing-masing sebelum waktu habis. Dengan cara tersebut, *SharIF Judge* dapat mengirimkan *Final submissions* para peserta ke *server Moss*.

2.2.8 Keamanan

Pengguna dapat mengatur keamanan *SharIF Judge* dengan beberapa langkah. Berikut adalah langkah-langkah yang dapat dilakukan.

Memakai *Sandbox*

Pengguna harus memakai *Sandbox* untuk *C/C++* dan mengaktifkan *Java Security Manager (Java Policy)* untuk *Java*. Penjelasan lebih lanjut tentang *Sandbox* ada pada point 2.2.9.

Memakai *Shield*

Shield bukan merupakan pertahanan yang asli, tetapi lebih baik ada daripada tidak sama sekali. Pengguna dapat mengaktifkan *Shield* untuk *C, C++, Python*. Penjelasan lebih lanjut tentang *Shield* ada pada point **??**.

Jalankan sebagai *Non-Privileged User*

Pengguna diharapkan menjalankan kode yang sudah dikumpulkan sebagai *non-privileged user* - *user* yang tidak memiliki akses jaringan, tidak dapat menulis *file* apapun, dan tidak dapat membuat banyak proses.

Pengguna harus menjalankan *PHP* pada server sebagai *user www-data*. Membuat *user* baru dengan nama *restricted_user* dan passwordnya. Jalankan *sudo visudo* dan tambahkan baris berikut pada akhir file *sudoers* :

```
www-data ALL=(restricted_user) NOPASSWD: ALL
```

Pada file *tester/runcode.sh* ganti

```
if $TIMEOUT_EXISTS; then
        timeout -s9 $((TIMELIMITINT*2)) $CMD <$IN >out 2>err
else
        $CMD <$IN >out 2>err
fi
```

menjadi

```
if $TIMEOUT_EXISTS; then
        sudo -u restricted_user timeout -s9 $((TIMELIMITINT*2)) $CMD <$IN >out 2>
    err
else
        sudo -u restricted_user $CMD <$IN >out 2>err
fi
```

dan *uncomment* baris berikut:

```
sudo -u restricted_user pkill -9 -u restricted_user
```

- Menonaktifkan jaringan pada *restricted_user*
  restricted_user seharusnya tidak dapat mengakses jaringan. Pengguna dapat menonaktifkan jaringan untuk sebuah *user* pada *Linux* dengan menggunakan *iptables*. Setelah dinonaktifkan, uji dengan *ping* sebagai *restricted_user*.

- Menolak izin menulis
  Pastikan tidak ada *file* atau *directory* yang dapat ditulis oleh *restricted_user*.

- Membatasi jumlah proses
  Pengguna dapat membatasi jumlah proses dengan cara menambahkan baris berikut pada file */etc/security/limits.conf*:

```
restricted_user     soft    nproc   3
restricted_user     hard    nproc   5
```

Memakai Dua *Server*

Pengguna dapat memakai sebuah *server* untuk antarmuka web dan menangani permintaan web, dan memakai *server* lainnya untuk menjalankan kode yang dikumpulkan. Hal ini mengurangi risiko menjalankan kode yang dikumpulkan. Untuk memakai dua *server* pengguna harus mengganti *source code ShariF Judge*.

2.2.9 *Sandbox*

ShariF Judge menjalankan banyak *arbitary codes* yang *user* kumpulkan sehingga memerlukan perkakas untuk kode *sandbox* yang sudah dikumpulkan. Pengguna dapat meningkatkan keamanan dengan cara mengaktifkan *shield* dan *sandbox* secara bersamaan.

C/C++ Sandboxing

ShariF Judge menggunakan *EasySandbox* untuk *sandboxing* kode *C/C++*. *EasySandbox* membatasi jalannya kode mengguna *seccomp*, mekanisme *sandboxing* pada *Linux kernel*.

Secara standar, *EasySandbox* dinonaktifkan pada *ShariF Judge*. Pengguna dapat mengaktifkannya pada halaman *Settings*. Pengguna harus *"Build EasySandbox"* sebelum mengaktifkannya.

Build EasySandbox *EasySandbox files* terletak pada *tester/easysandbox*. Untuk membangun *EasySandbox* jalankan :

```
$ cd tester/easysandbox
$ chmod +x runalltests.sh
$ chmod +x runtest.sh
$ make runtests
```

Jika pesan yang ditampilkan adalah *"All tests passed!"*, maka *EasySandbox* telah berhasil dibangun dan dapat dipakai pada sistem. Pengguna dapat mengaktifkan *EasySandbox* pada halaman *Settings*.

Java Sandboxing

Java Sandbox dapat diaktifkan secara standar menggunakan *Java Security Manager*. Pengguna dapat mengaktifkan/menonaktifkan hal ini pada halaman *Settings*.

2.2.10 *Shield*

Shield merupakan mekanisme sederhana yang melarang jalannya kode yang berbahaya. *Shield* bukan solusi *sandboxing*. *Shield* hanya menyediakan sebagian pertahanan terhadap serangan sepele. Pertahanan yang asli terhadap kode yang tidak terpercaya hanya didapatkan dengan mengaktifkan *Sandbox*.

Shield untuk *C/C++*

Dengan mengaktifkan *Shield* untuk *C/C++*, *SharIF Judge* hanya menambahkan *#define* pada bagian awal kode *C/C++* sebelum menjalankannya.

Pengguna dapat melarang menggunakan *goto* dengan cara menambahkan baris berikut pada bagian awal kode yang dikumpulkan :

```
#define goto YouCannotUseGoto
```

Dengan baris ini pada awal *files*, semua kode yang sudah dikumpulkan yang menggunakan *goto* akan mendapatkan *compilation error*.

Jika pengguna menggunakan *Shield*, semua kode yang berisi *#undef* akan mendapatkan *compilation error*.

Mengaktifkan *Shield* untuk *C* atau *C++*

Pengguna dapat mengaftikan dan menonaktifkan *Shield* pada halaman *Settings*.

Menambahkan aturan untuk *C/C++*

Daftar aturan *#define* terletak pada *file tester/shield/defc.h* (untuk *C*) dan *tester/shield/defcpp.h* (untuk *C++*). Pengguna dapat menambahkan aturan *#define* baru pada *file* ini. Konten dari *file* ini dapat diganti pada halaman *Settings*.

Berikut adalah sintaksis yang digunakan pada *files* tersebut :

```
/*

@file defc.h
There should be a newline at end of this file.
Put the message displayed to user after // in each line

e.g. If you want to disallow goto, add this line:
#define goto errorNo13    //Goto is not allowd

*/

#define system errorNo1      //"system" is not allowed
#define freopen errorNo2     //File operation is not allowed
#define fopen errorNo3       //File operation is not allowed
#define fprintf errorNo4     //File operation is not allowed
#define fscanf errorNo5      //File operation is not allowed
#define feof errorNo6        //File operation is not allowed
#define fclose errorNo7      //File operation is not allowed
#define ifstream errorNo8    //File operation is not allowed
#define ofstream errorNo9    //File operation is not allowed
#define fork errorNo10       //Fork is not allowed
#define clone errorNo11      //Clone is not allowed
#define sleep errorNo12      //Sleep is not allowed
```

Pada akhir *files defc.h* dan *defcpp.h* harus ada baris baru. Ada banyak aturan yang tidak dapat dipakai pada *g++*. Sebagai contoh pengguna tidak dapat memakai *#define fopen errorNo3* pada *C++* karena hasilnya *compile error*.

Shield untuk *Python*

Dengan mengaktifkan *Shield* untuk *Python*, *SharIF Judge* hanya menambahkan beberapa kode pada bagian awal kode *Python* yang telah dikumpulkan sebelum dijalankan untuk mencegah penggunaan fungsi yang berbahaya. Kode tersebut terletak pada *files tester/shield/shield_py2.py* dan *tester/shield/shield_py3.py*. Pengguna dapat mengaktifkan/menonaktifkan *Shield* untuk *Python* pada halaman *Settings*.

Berikut adalah cara untuk keluar dari *Shield* dan menggunakan fungsi yang telah di *blacklist* :

```
# @file shield_py3.py

import sys
sys.modules['os']=None

BLACKLIST = [
        #'__import__', # deny importing modules
        'eval', # eval is evil
        'open',
        'file',
        'exec',
        'execfile',
        'compile',
        'reload',
        #'input'
        ]
for func in BLACKLIST:
        if func in __builtins__.__dict__:
                del __builtins__.__dict__[func]
```

BAB 3

ANALISIS

3.1 Tingkat Kepatuhan *SharIF Judge*

Kesuksesan aplikasi SharIF Judge dalam mematuhi kriteria sukses *WCAG* 2.1 ditulis dalam tabel-tabel berikut.

Tabel 3.1: Tabel kepatuhan *SharIF Judge* terhadap prinsip *Perceivable*

Kriteria Sukses	Tingkat Kepatuhan	Hasil
1.1.1	A	
1.2.1	A	Sukses
1.2.2	A	Sukses
1.2.3	A	Sukses
1.2.4	AA	Sukses
1.2.5	AA	Sukses
1.2.6	AAA	Sukses
1.2.7	AAA	Sukses
1.2.8	AAA	Sukses
1.2.9	AAA	Sukses
1.3.1	A	
1.3.2	A	Sukses
1.3.3	A	
1.3.4	A	Sukses
1.3.5	AA	
1.3.6	AAA	
1.4.1	A	
1.4.2	A	Sukses
1.4.3	AA	
1.4.4	AA	Tidak Sukses
1.4.5	AA	
1.4.6	AAA	
1.4.7	AAA	Sukses
1.4.8	AAA	
1.4.9	AAA	
1.4.10	AA	
1.4.11	AA	
1.4.12	AA	
1.4.13	AA	

Tabel 3.2: Tabel kepatuhan *SharIF Judge* terhadap prinsip *Operable*

Kriteria Sukses	Tingkat Kepatuhan	Hasil
2.1.1	A	
2.1.2	A	
2.1.3	AAA	
2.1.4	A	
2.2.1	A	
2.2.2	A	
2.2.3	AAA	
2.2.4	AAA	
2.2.5	AAA	
2.2.6	AAA	
2.3.1	A	
2.3.2	AAA	
2.3.3	AAA	
2.4.1	A	
2.4.2	A	
2.4.3	A	
2.4.4	A	
2.4.5	AA	
2.4.6	AA	
2.4.7	AA	
2.4.8	AAA	
2.4.9	AAA	
2.4.10	AAA	
2.5.1	A	
2.5.2	A	
2.5.3	A	
2.5.4	A	
2.5.5	AAA	
2.5.6	AAA	

Tabel 3.3: Tabel kepatuhan *SharIF Judge* terhadap prinsip *Understandable*

Kriteria Sukses	Tingkat Kepatuhan	Hasil
3.1.1	A	
3.1.2	AA	
3.1.3	AAA	
3.1.4	AAA	
3.1.5	AAA	
3.1.6	AAA	
3.2.1	A	
3.2.2	A	
3.2.3	AA	
3.2.4	AA	
3.2.5	AAA	
3.3.1	A	
3.3.2	A	
3.3.3	AA	
3.3.4	AA	
3.3.5	AAA	
3.3.6	AAA	

Tabel 3.4: Tabel kepatuhan *SharIF Judge* terhadap prinsip *Robust*

Kriteria Sukses	Tingkat Kepatuhan	Hasil
4.1.1	A	
4.1.2	A	
4.1.3	AA	

3.1.1 *Perceivable*

Kriteria Sukses 1.1.1 Non-text Content

Kriteria Sukses 1.2.1 Audio-only dan Video-only (Prerecorded)

(Sukses)

Kriteria ini sukses dipatuhi karena pada aplikasi *SharIF Judge* tidak terdapat konten berbasis waktu.

Kriteria Sukses 1.2.2 Captions (Prerecorded)

(Sukses)

Kriteria ini sukses dipatuhi karena pada aplikasi *SharIF Judge* tidak terdapat konten berbasis waktu.

Kriteria Sukses 1.2.3 Audio Descriptive atau Media Alternative (Prerecorded)

(Sukses)

Kriteria ini sukses dipatuhi karena pada aplikasi *SharIF Judge* tidak terdapat konten berbasis waktu.

Kriteria Sukses 1.2.4 Captions (Live)

(Sukses)

Kriteria ini sukses dipatuhi karena pada aplikasi *SharIF Judge* tidak terdapat konten berbasis waktu.

Kriteria Sukses 1.2.5 Audio Description (Prerecorded)

(Sukses)

Kriteria ini sukses dipatuhi karena pada aplikasi *SharIF Judge* tidak terdapat konten berbasis waktu.

Kriteria Sukses 1.2.6 Sign Language (Prerecorded)

(Sukses)

Kriteria ini sukses dipatuhi karena pada aplikasi *SharIF Judge* tidak terdapat konten berbasis waktu.

Kriteria Sukses 1.2.7 Extended Audio Description (Prerecorded)

(Sukses)

Kriteria ini sukses dipatuhi karena pada aplikasi *SharIF Judge* tidak terdapat konten berbasis waktu.

Kriteria Sukses 1.2.8 Media Alternative (Prerecorded)

(Sukses)

Kriteria ini sukses dipatuhi karena pada aplikasi *SharIF Judge* tidak terdapat konten berbasis waktu.

Kriteria Sukses 1.2.9 Audio-only (Live)

(Sukses)

Kriteria ini sukses dipatuhi karena pada aplikasi *SharIF Judge* tidak terdapat konten berbasis waktu.

Kriteria Sukses 1.3.1 Info dan Relationships

Kriteria Sukses 1.3.2 Meaningful Sequence

(Sukses)

Kriteria ini sukses dipatuhi karena setiap halaman pada *SharIF Judge* memiliki urutan baca dan navigasi yang benar.

Kriteria Sukses 1.3.3 Sensory Characteristics

Kriteria Sukses 1.3.4 Orientation

(Sukses)

Kriteria ini sukses dipatuhi karena konten dari *SharIF Judge* dapat ditampilkan dalam orientasi *portrait* atau *landscape*.

Kriteria Sukses 1.3.5 Identify Input Purpose

Kriteria Sukses 1.3.6 Identify Purpose

Kriteria Sukses 1.4.1 Use of Color

Kriteria Sukses 1.4.2 Audio Control

(Sukses))

Kriteria ini sukses dipatuhi karena pada aplikasi *SharIF Judge* tidak terdapat konten berbasis waktu.

Kriteria Sukses 1.4.3 Contrast (Minimum)

Kriteria Sukses 1.4.4 Resize text

(Tidak Sukses)

Kriteria ini tidak sukses karena fungsionalitas *navigation bar* tidak berjalan pada pembesaran 125%.

Kriteria Sukses 1.4.5 Images of Text

Kriteria Sukses 1.4.6 Contrast (Enhanced)

Kriteria Sukses 1.4.7 Low atau No Background Audio

(Sukses)

Kriteria ini sukses dipatuhi karena pada aplikasi *SharIF Judge* tidak terdapat konten berbasis waktu.

Kriteria Sukses 1.4.8 Visual Presentation

Kriteria Sukses 1.4.9 Images of Text (No Exception)

Kriteria Sukses 1.4.10 Reflow

Kriteria Sukses 1.4.11 Non-text Contrast

Kriteria Sukses 1.4.12 Text Spacing

Kriteria Sukses 1.4.13 Content on Hover or Focus

DAFTAR REFERENSI

[1] 642120bb10213e035e09dd56805125d2e0eb67ac (2015) *SharIF-Judge*.

[2] Version 2.1 (2018) *Web Content Accessibility Guidelines*. W3C Web Accessibility Initiative (WAI). Massachusetts, USA.

LAMPIRAN A

KODE PROGRAM

Listing A.1: MyCode.c

```c

// This does not make algorithmic sense,
// but it shows off significant programming characters.

#include<stdio.h>

void myFunction( int input, float* output ) {
    switch ( array[i] ) {
        case 1: // This is silly code
            if ( a >= 0 || b <= 3 && c != x )
                *output += 0.005 + 20050;
            char = 'g';
            b = 2^n + ~right_size - leftSize * MAX_SIZE;
            c = (--aaa + &daa) / (bbb++ - ccc % 2 );
            strcpy(a,"hello $@?");
    }
    count = ~mask | 0x00FF00AA;
}

// Fonts for Displaying Program Code in LATEX
// Adrian P. Robson, nepsweb.co.uk
// 8 October 2012
// http://nepsweb.co.uk/docs/progfonts.pdf
```

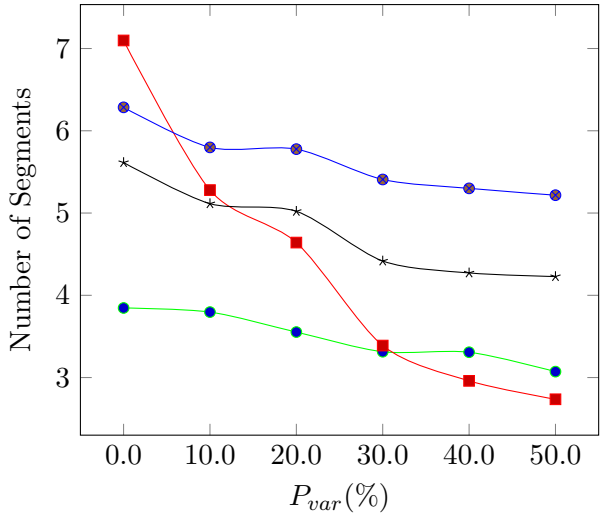
Listing A.2: MyCode.java

```java
import java.util.ArrayList;
import java.util.Collections;
import java.util.HashSet;

//class for set of vertices close to furthest edge
public class MyFurSet {
    protected int id;                                   //id of the set
    protected MyEdge FurthestEdge;                      //the furthest edge
    protected HashSet<MyVertex> set;                    //set of vertices close to furthest edge
    protected ArrayList<ArrayList<Integer>> ordered;    //list of all vertices in the set for each trajectory
    protected ArrayList<Integer> closeID;               //store the ID of all vertices
    protected ArrayList<Double> closeDist;              //store the distance of all vertices
    protected int totaltrj;                             //total trajectories in the set

    /*
     * Constructor
     * @param id : id of the set
     * @param totaltrj : total number of trajectories in the set
     * @param FurthestEdge : the furthest edge
     */
    public MyFurSet(int id,int totaltrj,MyEdge FurthestEdge) {
        this.id = id;
        this.totaltrj = totaltrj;
        this.FurthestEdge = FurthestEdge;
        set = new HashSet<MyVertex>();
        ordered = new ArrayList<ArrayList<Integer>>();
        for (int i=0;i<totaltrj;i++) ordered.add(new ArrayList<Integer>());
        closeID = new ArrayList<Integer>(totaltrj);
        closeDist = new ArrayList<Double>(totaltrj);
        for (int i = 0;i <totaltrj;i++) {
            closeID.add(-1);
            closeDist.add(Double.MAX_VALUE);
        }
    }

}
```

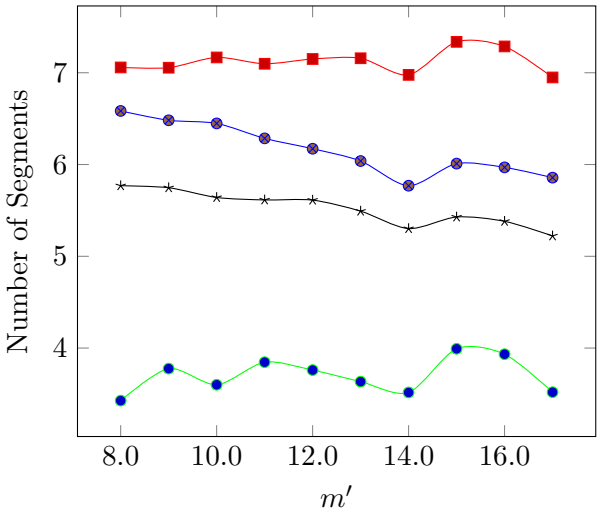
LAMPIRAN B

HASIL EKSPERIMEN

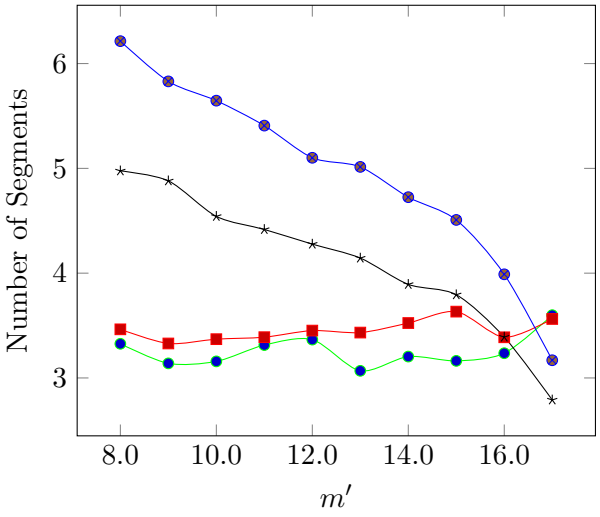
Hasil eksperimen berikut dibuat dengan menggunakan TIKZPICTURE (bukan hasil excel yg diubah ke file bitmap). Sangat berguna jika ingin menampilkan tabel (yang kuantitasnya sangat banyak) yang datanya dihasilkan dari program komputer.



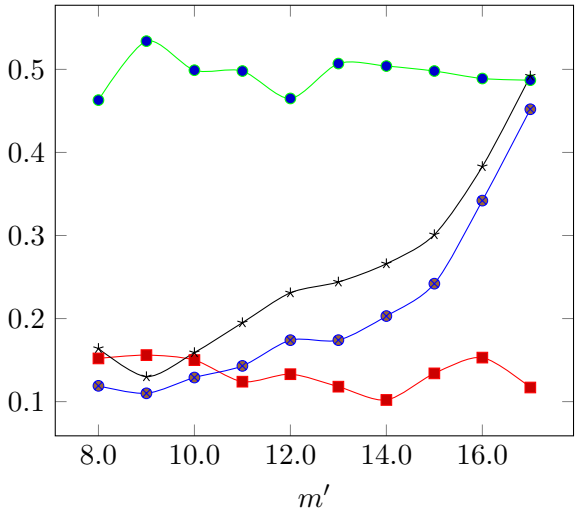
Gambar B.1: Hasil 1



Gambar B.2: Hasil 2



Gambar B.3: Hasil 3



Gambar B.4: Hasil 4