

SKRIPSI

**KUSTOMISASI SHARIF JUDGE UNTUK KEBUTUHAN
PROGRAM STUDI TEKNIK INFORMATIKA**



Stillmen Vallian

NPM: 2014730083

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2018**

UNDERGRADUATE THESIS

**CUSTOMIZATION OF SHARIF JUDGE FOR NEEDS OF
INFORMATICS DEPARTMENT**



Stillmen Vallian

NPM: 2014730083

**DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
2018**

ABSTRAK

Sharif Judge adalah *online judge* gratis untuk bahasa pemrograman C, C++, Java dan Python. *Sharif Judge* diciptakan oleh Mohammad Javad Naderi pada tahun 2014. *Sharif Judge* dibangun menggunakan *framework CodeIgniter* dan bersifat *open source*. Pada perangkat lunak *Sharif Judge*, terdapat banyak fitur yang dapat digunakan. Fitur-fitur tersebut seperti deteksi kecurangan (kode yang mirip), mengunduh nilai peserta dalam bentuk *excel*, mengunduh kode jawaban peserta dalam bentuk *zip*, menampilkan nilai seluruh peserta dan lain sebagainya. *Sharif Judge* digunakan oleh banyak kalangan, salah satunya adalah Program Studi Teknik Informatika Universitas Katolik Parahyangan.

Perangkat lunak *Sharif Judge* digunakan oleh Program Studi Teknik Informatika Universitas Katolik Parahyangan pada mata kuliah seperti Algoritma dan Struktur Data serta Desain dan Analisis Algoritma. Perangkat lunak ini sangat membantu para dosen dan mahasiswa. Sistem penilaian otomatis merupakan salah satu fitur yang sering digunakan oleh para dosen. Dengan memanfaatkan fitur di atas, dosen dapat dengan mudah memberikan nilai tugas, kuis dan ujian.

Sharif Judge terakhir dicommit di *GitHub* pada bulan Juli 2015 dan meninggalkan beberapa *bug*. Selain meninggalkan beberapa *bug*, Program Studi Teknik Informatika memiliki kebutuhan-kebutuhan yang lebih spesifik. Hal ini menyebabkan *Sharif Judge* kurang memenuhi kebutuhan Program Studi Teknik Informatika. Pengembangan perangkat lunak *Sharif Judge* perlu dilakukan untuk mengatasi masalah-masalah di atas.

Kustomisasi *Sharif Judge* memiliki tujuan agar *Sharif Judge* dapat kembali memenuhi kebutuhan Program Studi Teknik Informatika. Dari kebutuhan-kebutuhan tersebut, dirancang fitur-fitur untuk diimplementasi ke dalam perangkat lunak *Sharif Judge*. Dengan mengimplementasikan fitur-fitur baru, diharapkan perangkat lunak *Sharif Judge* dapat kembali memenuhi kebutuhan Program Studi Teknik Informatika.

Kata-kata kunci: Kustomisasi, pengembangan, kebutuhan, fitur, *Sharif Judge*, Program Studi Teknik Informatika

ABSTRACT

Sharif Judge is a free online judge for C, C++, Java and Python programming courses. *Sharif Judge* was created by Mohammad Javad Naderi in 2014. Sharif Judge was built by CodeIgniter Framework and has an open source characteristic. At Sharif Judge software, there are many features that can be used. Those features are like cheat detection (resemble codes), downloading participant's score in excel format, downloading participant's answer code in zip format, display the score of all those participants and so many else. Sharif Judge is used by many circles, one of them is Department of Informatics Parahyangan Catholic University.

Sharif Judge software is used by Department of Informatics Parahyangan Catholic University in many lessons like Algorithms and Data Structures also Design and Analysis of Algorithm. This software is very helpful for the lecturers and students. Automatic scoring system is one of the feature that often used by lecturers. By using that feature, lecturers can easily give the score of task, quiz and examination.

Sharif Judge was lastly committed in GitHub at July 2015 and leaving some bugs. Beside leaving some bugs, Department of Informatics have the requirements that are more specific. In this case Sharif Judge doesn't quite fulfill needs of Informatics Department. The development of Sharif Judge software needs to be done in order to solve all those problems.

Customization of Sharif Judge has an objective to make Sharif Judge can fulfill needs of Informatics Department. From all those needs, many features is designed to be implemented into Sharif Judge software. By implementing new features, it is expected that Sharif Judge software can re-fulfill needs of Informatics Department.

Keywords: Customization, development, needs, features, Sharif Judge, Informatics Department

DAFTAR ISI

DAFTAR ISI	ix
DAFTAR GAMBAR	xiii
DAFTAR TABEL	xv
1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Tujuan	2
1.4 Batasan Masalah	2
1.5 Metodologi	2
1.6 Sistematika Pembahasan	3
2 LANDASAN TEORI	5
2.1 <i>CodeIgniter</i>	5
2.1.1 <i>Flow Chart Aplikasi</i>	6
2.1.2 <i>Model-View-Controller</i>	6
2.2 <i>Sharif Judge</i>	11
2.2.1 Instalasi	12
2.2.2 <i>Clean URLs</i>	13
2.2.3 <i>Users</i>	14
2.2.4 Menambah <i>Assignment</i>	15
2.2.5 Struktur Pengujian	21
2.2.6 Deteksi Kecurangan	24
2.2.7 Database	25
2.3 <i>Twig</i>	26
3 ANALISIS	29
3.1 Analisis Kebutuhan Perangkat Lunak <i>Sharif Judge</i>	29
3.1.1 <i>Security with PHP</i>	31
3.1.2 <i>Securing Assignment</i>	32
3.1.3 <i>New Function</i>	32
3.1.4 <i>Solve Problem Indicator</i>	32
3.1.5 <i>Some Problem & Sugestion</i>	32
3.1.6 <i>Queue failed to process if submission take too long to complete?</i>	33
3.1.7 <i>Compilation Error on All Language</i>	33
3.1.8 Membatasi Soal (Deskripsi & PDF) Hanya Bisa Diakses saat <i>Assignment</i> "Open" dan Setelah Waktu Mulai	33
3.1.9 Menguji Kemiripan Kode Antar Mahasiswa	34
3.1.10 Satu Akun Hanya Dapat Login Satu Waktu	34
3.1.11 Membatasi Soal (Deskripsi & PDF) Hanya Bisa Diakses saat <i>Assignment</i> "Open" dan Setelah Waktu Mulai	35

3.1.12	Mengumpulkan <i>File</i> dengan Format TXT	35
3.1.13	Membatasi Soal (Deskripsi & PDF) Hanya Bisa Diakses saat <i>Assignment</i> "Open" dan Setelah Waktu Mulai	35
3.1.14	Mengumpulkan <i>File</i> dengan Format TXT	35
3.1.15	Perlu Ditambah Petunjuk Penamaan <i>File Input</i> dan <i>Output</i>	35
3.1.16	Membatasi Soal (Deskripsi & PDF) Hanya Bisa Diakses saat <i>Assignment</i> "Open" dan Setelah Waktu Mulai	35
3.1.17	Pendaftaran Peserta Disertai dengan <i>Display Name</i>	36
3.1.18	Nama Pengguna <i>Sharif Judge</i> Seharusnya Tidak Bisa Diubah	36
3.1.19	<i>Sharif Judge</i> Diharapkan Memiliki Fungsi <i>Assignment</i> Dapat Dikumpulkan Tanpa Adanya Batasan Waktu	36
3.1.20	<i>Sharif Judge</i> Diharapkan Memiliki <i>Scoreboard Global</i> untuk Semua <i>Assignment</i>	37
3.1.21	Membatasi Soal (Deskripsi & PDF) Hanya Bisa Diakses saat <i>Assignment</i> "Open" dan Setelah Waktu Mulai	37
3.1.22	Mengumpulkan <i>File</i> dengan Format TXT	37
3.1.23	UI Masih Merepotkan	37
3.1.24	UI Ada yang Tidak Berguna	38
3.1.25	<i>Sharif Judge</i> Diharapkan Memiliki Fungsi <i>Reminder</i>	38
3.1.26	Membatasi Soal (Deskripsi & PDF) Hanya Bisa Diakses saat <i>Assignment</i> "Open" dan Setelah Waktu Mulai	38
3.1.27	Integrasi <i>Login</i> ke <i>Server RADIUS</i>	38
3.1.28	Mengumpulkan <i>File</i> dengan Format TXT	39
3.1.29	Mengumpulkan <i>File</i> JAR (Java Multi Kelas)	39
3.1.30	Branding Teknik Informatika	39
3.2	Analisis Sistem Kini	40
3.2.1	<i>Model</i>	40
3.2.2	<i>View</i>	43
3.2.3	<i>Controller</i>	54
4	PERANCANGAN	59
4.1	Mengganti Method <i>shell_exec("rm ...")</i> Menjadi <i>unlink()</i>	59
4.2	Menambahkan Method Rekoneksi ke <i>Database</i>	60
4.3	Membatasi Pengaksesan Soal (deskripsi & PDF)	60
4.4	Mensupport <i>File</i> dengan Ekstensi TXT	61
4.5	Menambahkan Halaman <i>Logs</i>	64
4.6	Menambahkan Parameter " <i>Display Name</i> " pada Pendaftaran Peserta <i>Sharif Judge</i>	69
4.7	Menambahkan Fitur " <i>Lock Student's Display Name</i> "	73
4.8	Menambahkan Fitur " <i>Archived Assignment</i> "	77
4.9	Menambahkan Halaman <i>Hall of Fame</i>	81
4.10	Integrasi <i>Login</i> ke <i>Server RADIUS</i>	85
4.11	Branding Teknik Informatika	86
5	IMPLEMENTASI DAN PENGUJIAN	89
5.1	Lingkungan untuk Implementasi dan Pengujian	89
5.2	Implementasi	90
5.3	Pengujian Fungsional	95
5.3.1	Mengunduh Soal (deskripsi & PDF) yang Telah Dibatasi	95
5.3.2	Membuat <i>Assignment</i> yang Menerima <i>File</i> dengan Ekstensi TXT	96
5.3.3	Mengakses Halaman <i>24-hour Logs</i>	98
5.3.4	Mendaftarkan Peserta Menggunakan Tambahan Parameter " <i>Display Name</i> "	99
5.3.5	<i>Disable Display Name</i> Peserta Menggunakan Fitur " <i>Lock Student's Display Name</i> "	101

5.3.6	Menambahkan <i>Assignment</i> yang Mengaktifkan Fitur " <i>Archived Assignment</i> "	102
5.3.7	Mengakses Halaman <i>Hall of Fame</i>	103
5.4	Pengujian Ekseprimental	104
5.4.1	<i>Remove the Assignments Folder</i>	105
5.4.2	<i>Add User Email are not formatted correctly</i>	106
5.4.3	<i>Cannot Create Assignments</i>	107
5.4.4	<i>Link ke New Home of Sharif-Judge</i>	108
5.4.5	Hapus Perbedaan antara <i>Admin</i> dan <i>Student</i> pada <i>PDF Download</i>	113
5.4.6	<i>Download Excel</i> Tidak Berfungsi pada Halaman <i>Submission</i>	114
5.5	Masukan dari Pengguna <i>GitHub</i>	120
6	KESIMPULAN DAN SARAN	123
6.1	Kesimpulan	123
6.2	Saran	124
	DAFTAR REFERENSI	125
A	KODE PROGRAM HALAMAN <i>24-hour Logs</i>	127
B	KODE PROGRAM HALAMAN <i>Hall of Fame</i>	129
C	KODE PROGRAM <i>shj_functions.js</i>	133
D	KODE PROGRAM <i>secrets.php</i>	139

DAFTAR GAMBAR

2.1	<i>Flow Chart</i> Aplikasi	6
2.2	Tampilan Halaman <i>Assignments</i>	15
3.1	Logo dan Ikon	39
3.2	<i>Banner Sharif Judge</i>	40
3.3	Halaman <i>Dashboard</i>	44
3.4	Halaman <i>Notifications</i>	44
3.5	Halaman <i>Assignments</i>	45
3.6	Halaman <i>Problems</i>	46
3.7	Halaman <i>Submit</i>	46
3.8	Halaman <i>Profile</i>	47
3.9	Halaman <i>Scoreboard</i>	48
3.10	Halaman <i>All Submission</i>	48
3.11	Halaman <i>Final Submission</i>	49
3.12	Halaman <i>Settings</i>	49
3.13	Halaman <i>User</i>	50
3.14	Halaman <i>Add User</i>	50
3.15	Halaman <i>Add Notification</i>	51
3.16	Halaman <i>Add Assignment</i>	52
3.17	Halaman <i>Rejudge</i>	52
3.18	Halaman <i>Login</i>	53
3.19	Halaman <i>Register</i>	53
3.20	Halaman <i>Lost</i>	54
4.1	Diagram <i>Flowchart</i> Menambah <i>Assignment</i> TXT	64
4.2	Rancangan tampilan halaman <i>logs</i>	66
4.3	Diagram <i>Flowchart</i> Mengakses Halaman Logs	69
4.4	Diagram <i>Flowchart</i> Mengaktifkan Fitur " <i>Lock Student's Display Name</i> "	77
4.5	Diagram <i>Flowchart</i> Mengaktifkan Fitur " <i>Archived Assignment</i> "	81
4.6	Rancangan tampilan halaman <i>Hall of Fame</i>	83
4.7	Rancangan tampilan <i>details Hall of Fame</i> peserta tertentu	83
4.8	Diagram <i>Flowchart</i> Mengakses Halaman <i>Hall of Fame</i> dan Melihat Detail <i>Hall of Fame</i>	85
4.9	Logo dan Ikon	86
4.10	<i>Banner Sharif Judge</i>	87
5.1	Tampilan Halaman Login	92
5.2	Tampilan Halaman Hall of Fame	93
5.3	Tampilan Detail dari Hall of Fame	93
5.4	Tampilan Halaman <i>24-hour Logs</i>	94
5.5	Tampilan <i>Side Menu SharIF Judge</i>	94
5.6	Ikon <i>SharIF Judge</i> pada <i>Title Bar</i>	95
5.7	Logo <i>SharIF Judge</i> pada <i>Top Bar</i>	95

5.8	Empat Buah <i>Assignment</i> yang Dibuat	95
5.9	Pesan <i>Error "You are not registered for submitting."</i>	96
5.10	Pesan <i>Error "Selected assignment has not started."</i>	96
5.11	Pesan <i>Error "Selected assignment has finished."</i>	96
5.12	Pesan <i>Error "Selected assignment has been closed."</i>	96
5.13	Pembuatan <i>Assignment Upload</i> TXT	97
5.14	<i>Submit File</i> TXT	97
5.15	Halaman <i>All Submission</i> setelah Mengumpulkan <i>File</i> TXT	97
5.16	<i>File</i> TXT Hasil Unduh	98
5.17	<i>File</i> TXT Utama	98
5.18	Halaman <i>24-hour Logs</i> yang Tampil	98
5.19	Halaman <i>24-hour Logs</i> Mencatat Aktivitas <i>Login</i> Pengguna	99
5.20	Halaman <i>Add User</i>	100
5.21	Pengguna Berhasil Didaftarkan	100
5.22	<i>Display Name</i> yang Tampil Sesuai dengan Parameter yang Dimasukan	101
5.23	Mengaktifkan Fitur <i>Lock Student's Display Name</i>	101
5.24	<i>Text Field Display Name</i> Menjadi <i>Disable</i>	102
5.25	Mengaktifkan Fitur <i>Archived Assignment</i>	102
5.26	<i>Finish Time</i> dengan Nilai "2038-01-18 00:00:00"	103
5.27	<i>Archived Assignment</i> Tidak Muncul pada Kalendar	103
5.28	Halaman <i>Hall of Fame</i>	104
5.29	<i>Details</i> Nilai yang Diperoleh	104
5.30	Perubahan yang Dilakukan pada Halaman <i>Settings</i>	105
5.31	Format <i>Email</i> Tidak Beraturan	107
5.32	Isi <i>Excel judge_all_submissions.xlsx</i>	120
5.33	Isi <i>Excel sharifjudge_users.xlsx</i>	120

DAFTAR TABEL

2.1	<i>User Roles Table</i>	14
2.2	<i>Permission Table</i>	14
2.3	<i>Problem 1 (Penjumlahan)</i>	18
2.4	<i>Problem 2 (Max)</i>	19
3.1	Tabel Analisis Kebutuhan <i>Sharif Judge</i>	30
4.1	Perancangan Tabel <i>shj_logins</i>	64
4.2	Perincian fungsi <i>insert_to_logs</i>	65
4.3	Perincian fungsi <i>get_all_logs</i>	66
4.4	Perincian fungsi <i>consturct__</i>	67
4.5	Perincian fungsi <i>index</i>	67
4.6	Perincian fungsi <i>get_all_final_submission</i>	82
4.7	Perincian fungsi <i>get_all_user_assignments</i>	82
4.8	Perincian fungsi <i>consturct__</i>	84
4.9	Perincian fungsi <i>index</i>	84
4.10	Perincian fungsi <i>hof_details</i>	84
5.1	Perangkat Keras Lingkungan Pertama	89
5.2	Perangkat Lunak Lingkungan Pertama	89
5.3	Perangkat Keras Lingkungan Kedua	90
5.4	Perangkat Lunak Lingkungan Kedua	90
5.5	Struktur Tabel <i>shj_logins</i>	90
5.6	Struktur Tabel <i>shj_settings</i>	91
5.7	Struktur Tabel <i>shj_assignments</i>	92

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Online judge merupakan sebuah sistem *online* yang dirancang untuk mengevaluasi kode algoritma yang dikumpulkan oleh pengguna. Sistem dapat mengcompile dan mengeksekusi kode serta mengujinya ke dalam lingkungan homogen menggunakan data yang telah disediakan. Kode yang dikumpulkan dapat dijalankan dengan batasan-batasan seperti *time limit*, *memory limit*, keamanan dan sebagainya. Penggunaan *online judge* sering ditemukan pada kompetisi pemrograman dan mata kuliah pemrograman [1].

Sharif Judge adalah *online judge* gratis untuk bahasa pemrograman C, C++, Java dan Python. Perangkat lunak ini diciptakan oleh Mohammad Javad Naderi pada tahun 2014 dan bersifat *open source*. Antarmuka *Sharif Judge* ditulis menggunakan bahasa pemrograman PHP (*framework CodeIgniter*) dan *backend* menggunakan *BASH* [2].

Fungsi utama *Sharif Judge* adalah menilai jawaban yang telah dikumpulkan oleh para peserta secara otomatis. Cara kerja penilaian otomatis pada *Sharif Judge* dimulai dari dosen membuat *assignment*. Untuk membuat *assignment* dibutuhkan data-data seperti nama *assignment*, waktu mulai, waktu berhenti, waktu tambahan, daftar peserta, deskripsi soal dan kunci jawaban. Para peserta dapat mengunduh deskripsi soal lalu mengerjakan *assignment* tersebut. Peserta yang telah selesai mengerjakan *assignment* dapat mengumpulkan jawaban dalam bentuk kode program. *Sharif Judge* akan menjalankan kode program, membandingkan dengan kunci jawaban. Setelah menyesuaikan hasil keluaran dari kode program dengan kunci jawaban, *Sharif Judge* akan menilai jawaban peserta.

Sharif Judge digunakan oleh Program Studi Teknik Informatika Universitas Katolik Parahyangan pada mata kuliah seperti Algoritma dan Struktur Data serta Desain dan Analisis Algoritma. Perangkat lunak *Sharif Judge* sangat membantu dosen dan mahasiswa dalam bidang akademik. Sistem penilaian otomatis merupakan salah satu fitur yang sering digunakan oleh para dosen. Dengan memanfaatkan fitur di atas, dosen dapat dengan mudah memberikan nilai tugas, kuis dan ujian. Mahasiswa juga dapat melihat nilai secara langsung setelah jawaban dikumpulkan. Para mahasiswa dapat memperbaiki jawaban yang telah dikumpulkan jika *assignment* yang dikerjakan tidak melewati batas waktu pengumpulan.

Pada prakteknya, perangkat lunak *Sharif Judge* versi terkini masih butuh pengembangan. Pengembangan tersebut dibutuhkan karena Program Studi Teknik Informatika memiliki kebutuhan yang lebih spesifik. Kebutuhan spesifik tersebut seperti *login* yang terintegrasi dengan server RADIUS Teknik Informatika, membatasi pengaksesan deskripsi soal pada *assignment* dan kebutuhan spesifik

1 lainnya. *Sharif Judge* terakhir dicommit di *GitHub* pada bulan Juli 2015 dan masih memiliki
2 beberapa *bug* yang belum diperbaiki. Hal-hal di atas menyebabkan *Sharif Judge* kurang memenuhi
3 kebutuhan Program Studi Teknik Informatika.

4 Pada skripsi ini, akan dikembangkan *Sharif Judge* agar sesuai dengan kebutuhan yang di-
5 sebutkan di atas. Dari kebutuhan yang disebutkan di atas, akan dirancang fitur-fitur untuk
6 diimplementasikan pada *Sharif Judge*. Dengan pengimplementasian fitur yang baru, diharapkan
7 kebutuhan mahasiswa dan dosen dapat terpenuhi.

8 1.2 Rumusan Masalah

- 9 1. Fitur-fitur apa saja yang dibutuhkan oleh Teknik Informatika?
- 10 2. Bagaimana mengembangkan *Sharif Judge* sehingga memenuhi kebutuhan Teknik Informatika?

11 1.3 Tujuan

- 12 1. Menganalisa dan mengetahui fitur-fitur yang dibutuhkan Teknik Informatika.
- 13 2. Mengimplementasi kebutuhan Program Studi Teknik Informatika pada *Sharif Judge*.

14 1.4 Batasan Masalah

15 Batasan masalah yang dibuat terkait dengan pengerjaan skripsi ini adalah sebagai berikut :

- 16 1. Perangkat lunak diuji pada mata kuliah AIF102 (Algoritma & Struktur Data).

17 1.5 Metodologi

18 Metodologi yang dilakukan dalam pengerjaan skripsi ini adalah sebagai berikut :

- 19 1. Studi literatur mengenai:
 - 20 • *CodeIgniter* sebagai *framework* untuk mengembangkan perangkat lunak.
 - 21 • Dokumentasi *Sharif Judge* sebagai panduan untuk mengembangkan perangkat lunak.
- 22 2. Menganalisis kebutuhan-kebutuhan dari para dosen pengguna *Sharif Judge* dan daftar *issue*
23 pada repositori *Sharif Judge* pada *Github*.
- 24 3. Menentukan dan merancang fitur yang akan diimplementasikan.
- 25 4. Mengimplementasikan fitur terhadap perangkat lunak.
- 26 5. Menguji perangkat lunak ke mata kuliah selama satu semester.
- 27 6. Membuat dokumentasi perangkat lunak.

1.6 Sistematika Pembahasan

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

1. Bab 1 : Pendahuluan

Bab 1 membahas mengenai gambaran umum penelitian ini. Berisi tentang latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi dan sistematika pembahasan.

2. Bab 2 : Landasan Teori

Bab 2 membahas mengenai teori-teori yang mendukung berjalannya penelitian ini serta tentang *CodeIgniter* dan dokumentasi *Sharif Judge*.

3. Bab 3 : Analisis

Bab 3 membahas mengenai analisis fitur-fitur yang akan diimplementasi pada *Sharif Judge*.

4. Bab 4 : Perancangan

Bab 4 membahas mengenai perancangan yang dilakukan sebelum masuk ke tahap implementasi.

5. Bab 5 : Implementasi dan Pengujian

Bab 5 membahas mengenai implementasi dan pengujian yang telah dilakukan.

6. Bab 6 : Kesimpulan dan Saran

Bab 6 membahas hasil kesimpulan dari keseluruhan penelitian ini dan saran-saran yang dapat diberikan untuk penelitian berikutnya.

BAB 2

LANDASAN TEORI

Bab ini membahas tentang landasan teori yang digunakan dalam skripsi ini. Landasan teori yang digunakan, diambil dari dua sumber, yaitu "*CodeIgniter Documentation*" karya *British Columbia Institute of Technology* [3] dan "*Sharif Judge Documentation*" karya Mohammad Javad Naderi [4].

2.1 *CodeIgniter*

CodeIgniter merupakan sebuah *framework* bagi pengguna yang ingin membangun aplikasi web menggunakan PHP. Tujuan utamanya adalah memungkinkan para pengguna untuk mengembangkan proyek-proyek menjadi lebih cepat dibandingkan menulis kode dari awal. *Framework* ini memiliki banyak *library* untuk fungsi-fungsi yang biasa diperlukan, serta antarmuka dan struktur logis yang sederhana untuk mengakses *library* ini. *CodeIgniter* membuat para pengguna lebih fokus pada proyek dengan cara meminimalkan jumlah kode yang dibutuhkan [3].

Beberapa keunggulan dari *CodeIgniter* yaitu:

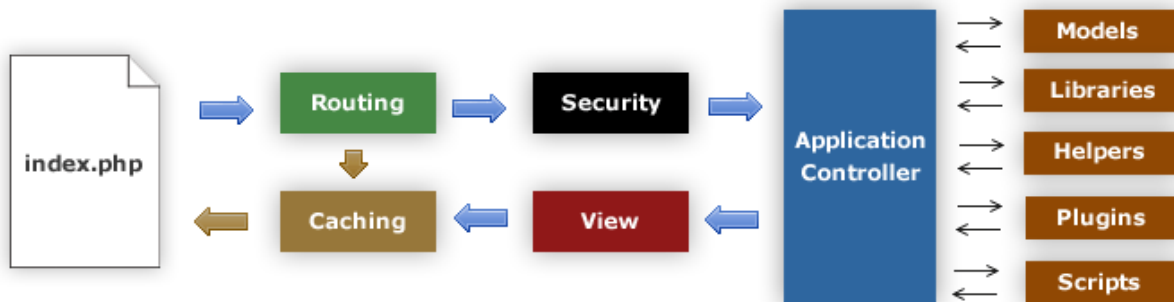
- *Framework* yang ringan
Inti dari sistem *CodeIgniter* membutuhkan *library* yang kecil. Hal ini sangat berbeda dengan *framework* lain yang membutuhkan *resource* lebih. *Library* tambahan dimuat secara dinamis atau sesuai dengan permintaan sehingga sistem dapat berjalan cepat.
- Menggunakan konsep M-V-C
CodeIgniter menggunakan pendekatan *Model-View-Controller* yang memungkinkan pemisahan antara logika dan presentasi. Pendekatan *Model-View-Controller* dibahas pada sub bab 2.1.2.
- Menghasilkan *Clean URLs*
CodeIgniter menghasilkan *Clean URLs* dan *search-engine friendly*. *Clean URLs* bertujuan untuk mempermudah pengguna dalam membaca *URL*. Contoh perbandingan *URL* biasa dengan *Clean URLs*: *URL* biasa: `\\example.com\\index.php?page=news`, *Clean URLs*: `\\example.com\\news`.
- Memiliki banyak fitur bantuan
CodeIgniter dilengkapi dengan *library* yang umumnya diperlukan untuk mengembangkan *web*, seperti mengakses *database*, mengirim *email*, memvalidasi data *form*, menjaga *session*, memanipulasi gambar dan masih banyak lagi.

- Mudah diperluas

CodeIgniter dapat diperluas dengan menambahkan *library* pengguna dan kelas bantuan (*helper*).

2.1.1 *Flow Chart* Aplikasi

Gambar 2.1 menunjukkan bagaimana data mengalir ke seluruh sistem [3]:



Gambar 2.1: *Flow Chart* Aplikasi

1. File *index.php* berfungsi sebagai *front controller* dan menginisialisasi *resource* utama yang dibutuhkan untuk menjalankan *CodeIgniter*.
2. *Router* memeriksa *HTTP request* untuk menentukan apa yang harus dilakukan.
3. Jika terdapat *file cache*, maka akan langsung dikirimkan ke *browser*. File cache merupakan data yang pernah diakses oleh pengguna. Data tersebut dapat berupa teks, gambar atau *file*.
4. *HTTP request* dan data pengguna yang dikirim akan terlebih dahulu disaring untuk alasan keamanan. *Application controller* akan dimuat setelah proses penyaringan selesai.
5. *Controller* akan memuat kelas *Model*, *library* utama dan kelas bantuan.
6. *View* akan memuat tampilan akhir dan dikirim ke web *browser*. Jika caching diaktifkan, maka tampilan dimasukkan ke dalam cache terlebih dahulu sehingga pada permintaan selanjutnya tampilan tersebut dapat diakses lebih cepat.

2.1.2 *Model-View-Controller*

CodeIgniter merupakan *framework* yang menggunakan pola pengembangan *Model-View-Controller*. *MVC* adalah pendekatan perangkat lunak yang memisahkan logika aplikasi dari presentasi. Hal tersebut memungkinkan halaman web pengguna memiliki *scripting* yang sedikit karena presentasi terpisah dari *scripting* PHP [3].

2.1.2.1 Model

Model merepresentasikan bagian struktur data pengguna. Biasanya kelas *Model* berisi fungsi-fungsi yang membantu pengguna untuk mengambil, menyimpan, dan memperbarui informasi pada *database*. Berikut ini adalah beberapa hal penting yang terdapat pada *Model*:

- Susunan dari *Model*

Kelas *Model* berada di direktori `application/models/`. *Model* dapat disimpan ke dalam sub direktori. Bentuk dasar kode pada kelas *model* digambarkan seperti berikut ini:

```
class Model_name extends CI_Model {  
    public function __construct()  
    {  
        parent::__construct();  
        //constructor code  
    }  
}
```

Model_name adalah nama kelas dari kelas *model* yang pengguna buat. Penamaan kelas *Model* harus dimulai dengan huruf kapital. Selain itu, kelas *model* harus merupakan turunan (*class* *CI_Model* atau *MY_Model*).

- Menghubungkan Sebuah *Model*

Pada dasarnya *model* akan dimuat dan dipanggil dari *method* atau fungsi yang ada pada *controller*. Untuk menghubungkan *model*, pengguna harus menggunakan method berikut:

```
$this->load->model('model_name');
```

Jika *model* yang pengguna buat terletak di dalam sebuah sub-direktori, maka pengguna harus menyertakan alamat relatif (*relative path*) dari *model* yang dibuat. Sebagai contoh, jika *model* yang pengguna buat berlokasi di `application/models/blog/Queries.php` pengguna dapat menghubungkannya dengan cara:

```
$this->load->model('blog/queries');
```

Pengguna dapat mengakses *method* yang terdapat pada *model* menggunakan sebuah objek dengan nama yang sama dengan nama kelas yang pengguna buat sebelumnya:

```
$this->load->model('model_name');
```

```
$this->model_name->method();
```

Jika pengguna ingin menggunakan objek yang berbeda untuk sebuah *model*, maka pengguna dapat menggunakan penamaan (*alias*) di parameter kedua:

```
$this->load->model('model_name', 'foobar');
```

```
$this->foobar->method();
```

Berikut merupakan contoh sebuah *controller* yang terhubung dengan sebuah *model* dan menampilkan data hasil olahan *model* ke *view*:

```
class Blog_controller extends CI_Controller {

    public function blog()
    {

        $this->load->model('blog');

        $data['query'] = $this->blog->data_sepuluh
            _artikel_terakhir();

        $this->load->view('blog', $data);

    }

}
```

- *Auto-loading Model*

Auto-loading (menghubungkan secara otomatis) model tertentu secara global dapat pengguna lakukan dengan menggunakan pengaturan yang ada pada berkas `application/config/autoload.php`. Kode yang ditambahkan untuk menghubungkan *model* secara otomatis selama sistem berjalan adalah `$autoload['model'] = array('model_name');`

- Koneksi ke *Database*

Ketika sebuah *model* dipanggil, *model* tidak akan terhubung ke *database* secara otomatis. Beberapa opsi yang dapat digunakan untuk menghubungkan *model* ke *database*:

- Pengguna dapat menghubungkan dengan menggunakan metode standar *database* antara *class Controller* atau *class Model*. Metode standar database dibagi menjadi dua cara yaitu *auto connect* dan *manual connect*. Fitur auto connect akan memuat kelas database pada setiap halaman. Manual connect dapat digunakan ketika pengguna menginginkan halaman tertentu yang dapat terhubung dengan database.
- Pengguna dapat mengatur sebuah *model* melakukan *auto-connect* dengan menambahkan nilai `TRUE` (boolean) di parameter ketiga atau mengatur konektivitas sebagaimana yang telah didefinisikan di dalam file `application/config/database.php`

```
$this->load->model('model_name', '', TRUE);
```

- Pengguna dapat mengatur koneksi secara manual dengan menambahkan *item-item* berupa *array* pada *parameter* ketiga seperti contoh berikut:

```
$config['hostname'] = 'localhost';
$config['username'] = 'username';
$config['password'] = 'katasandi';
$config['database'] = 'database_name';
```



```

1      $config['dbdriver'] = 'mysqli';
2      $config['dbprefix'] = '';
3      $config['pconnect'] = FALSE;
4      $config['db_debug'] = TRUE;
5
6      $this->load->model('model_name', '', $config);
7

```

8 2.1.2.2 *View*

9 *View* merupakan informasi yang akan ditampilkan kepada pengguna. Umumnya *View* merupakan
10 sebuah halaman web, namun pada *CodeIgniter*, *View* dapat berupa bagian-bagian halaman seperti
11 *header* atau *footer*. *View* tidak pernah dipanggil secara langsung, melainkan harus melalui *controller*
12 karena dalam *framework MVC*, *controller* berfungsi sebagai pengatur. Untuk memuat tampilan
13 tertentu, pengguna dapat menggunakan *method* berikut:

```

14
15 $this->load->view('name');
16

```

17 *CodeIgniter* dapat menangani beberapa panggilan *method \$this->load->view()* dari dalam
18 *controller*. Jika lebih dari satu panggilan terjadi, maka panggilan tersebut ditambahkan bersama.
19 Contohnya pengguna ingin memiliki *header view*, *menu view*, *content view* dan *footer view*. Kode
20 program yang digunakan seperti berikut:

```

21
22 class Page{
23     public function INDEX(){
24         $data['page_title'] = 'Your title';
25         $load->view('header');
26         $load->view('menu');
27         $load->view('content', $data);
28         $load->view('footer');
29     }
30 }
31

```

32 2.1.2.3 *Controller*

33 *Controller* berfungsi sebagai perantara antara *Model*, *View* dan *resource* lain yang dibutuhkan
34 untuk memproses *HTTP request* dan menghasilkan halaman web. *Controller* merupakan sebuah
35 kelas yang dinamakan demikian agar dapat dikaitkan dengan URI. Sebagai contoh URI **example.**
36 **com/index.php/blog/**. Pada contoh tersebut *segment* terakhir dari URI adalah *blog*, sehingga
37 *CodeIgniter* akan mencari kelas *controller* bernama *Blog.php* dan memuatnya ke dalam *browser*.
38 Nama *controller* harus diawali dengan huruf kapital. Selain itu *controller* juga harus merupakan
39 turunan kelas "*CI_Controller*". Contoh yang benar :

```

40
41 <?php
42 class Blog extends CI_Controller {
43

```

```
}

```

Contoh yang salah :

```
<?php
class blog extends CI_Controller {

}

```

Berikut beberapa hal penting yang terdapat pada *Controller*:

- *Method*

Untuk menjalankan suatu *method*, pengguna perlu menuliskannya pada segmen kedua URI.

Berikut beberapa *method* yang dibuat pada sebuah *controller*:

```
<?php
class Blog extends CI_Controller {

    public function index()
    {

        echo 'Hello World!';

    }

    public function comments()
    {

        echo 'Look at this!';

    }

    public function shoes($sandals, $id)
    {

        echo $sandals;
        echo $id;

    }

}

```

Jika pengguna memuat URL `example.com/index.php/blog/comments`, maka method *comments()* akan dijalankan pada *controller Blog.php*. Method *index()* akan dijalankan jika bagian kedua URI kosong. Jika URI mengandung lebih dari dua segment, maka segmen-segmen tersebut akan dimasukkan ke dalam *method* sebagai parameter. Contoh: pengguna memuat URL `example.com/index.php/blog/shoes/sandals/123`. Segmen "*sandals*" dan "*123*" akan dimasukkan sebagai *parameter* ke dalam *method shoes*.

- *Default Controller*

CodeIgniter dapat diperintahkan untuk menjalankan *default controller* jika tidak terdapat URI. Hal ini umumnya terjadi ketika terdapat permintaan menggunakan URL dasar *website*. Penentuan *default controller* terdapat pada file `application/config/routes.php`. Berikut contoh penentuan *default controller*:

```
$route['default_controller'] = 'blog';
```

Blog merupakan nama kelas *controller* yang ingin digunakan. Jika pengguna mengakses file *index.php* utama tanpa menentukan segmen URI, maka akan dijalankan *controller Blog*.

- *Private Method*

Method-method dengan tipe *private* tidak dapat diakses oleh publik. *Method* ini hanya dapat diakses oleh *method* lain dalam *controller* yang sama dan *method* ini juga tidak dapat diakses melalui URL. Contoh penulisan *private method*:

```
private function _utility()
{
    // kode program
}
```

Method di atas tidak dapat diakses dengan cara pemanggilan *method* yang umum seperti berikut:

```
example.com/index.php/blog/_utility/
```

- Mengorganisir *Controller* ke Dalam Sub Direktori

CodeIgniter mengizinkan pengguna untuk mengorganisir *controller* ke dalam sub direktori. Pengguna dapat membuat sub direktori di dalam direktori *application/controllers/* dan menyimpan kelas *controller* ke direktori tersebut. Ketika menggunakan fitur ini, pengguna harus menspesifikasikan folder tersebut ke dalam URI. Berikut contoh pemanggilan *controller* yang berada di dalam sub direktori: Sebuah *controller* berlokasi pada direktori *application/controllers/products/Shoes.php*. Untuk memanggil *controller* tersebut, URI pengguna yang tampil akan seperti *example.com/index.php/products/shoes/show/123*.

2.2 Sharif Judge

Sharif Judge adalah *online judge* gratis untuk bahasa pemrograman C, C++, Java dan Python. Perangkat lunak ini diciptakan oleh Mohammad Javad Naderi pada tahun 2014 dan bersifat *open source*. Antarmuka *Sharif Judge* ditulis menggunakan bahasa pemrograman PHP (*framework CodeIgniter*) dan *backend* menggunakan *BASH* [2]. *Sharif Judge* memiliki beberapa fitur seperti:

- Terdapat beberapa peran (*role*) *users* yaitu *admin*, *head instructor*, *instructor*, *student*
- *Sandboxing* (belum diterapkan untuk *python*). *Sandboxing* adalah sebuah mekanisme yang dapat menjalankan aplikasi dalam lingkungan virtual yang aman.
- Deteksi kecurangan (mendeteksi kode yang mirip) menggunakan *Moss*
- Pengaturan untuk menilai keterlambatan pengiriman
- Antrian pengiriman

- Mengunduh hasil dalam bentuk *file excel*
- Mengunduh kode yang telah dikirim dalam bentuk *file zip*
- Metode "*Output Comparison*" dan "*Tester Code*" untuk memeriksa kebenaran dari hasil keluaran.
- Menambahkan beberapa pengguna sekaligus
- Deskripsi *problem* (*PDF/Markdown/HTML*)
- Penilaian ulang (*rejudge*)
- Papan nilai
- Notifikasi

2.2.1 Instalasi

Untuk menjalankan *Sharif Judge*, dibutuhkan sebuah server *Linux* dengan persyaratan berikut [4]:

- *Webserver* menjalankan PHP versi 5.3 atau yang lebih baru
- Pengguna harus dapat menjalankan PHP dari *command line*. Pada *Ubuntu*, pengguna perlu menginstall paket *php5-cli*
- *Mysql database* (dengan ekstensi *mysqli* untuk PHP) atau *PostgreSQL database*
- PHP harus memiliki akses untuk menjalankan *shell commands* menggunakan fungsi *shell_exec*. Contohnya seperti *command* di bawah ini:

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

- Perkakas yang digunakan untuk melakukan proses kompilasi dan menjalankan kode yang dikumpulkan adalah (*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 dikirimkan. *Perl* merupakan salah satu bahasa pemrograman yang dapat digunakan untuk membangun perangkat lunak.

Jika persyaratan di atas telah terpenuhi, maka instalasi dapat dilakukan dengan cara sebagai berikut:

- Mengunduh versi terakhir dari *Sharif Judge* dan *unpack* hasil download di direktori *public html*
- Memindahkan folder *system* dan *application* keluar dari *public directory* dan masukan *path* lengkap di *file index.php*

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

- Membuat sebuah *Mysql* atau *PostgreSql* database 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
/*****

```

- 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 *assignments* di 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 *assignments* sehingga tidak dapat diakses publik.

2.2.2 Clean URLs

Secara default, `index.php` merupakan bagian dari seluruh URLs yang ada pada *Sharif Judge* [4]. Berikut contoh URLs yang dihasilkan oleh *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* adalah proses mengubah parameter yang terdapat pada URL. Berikut contoh URL yang telah melewati proses *URL rewriting*:

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

Untuk menggunakan *clean urls*, pengguna perlu melakukan beberapa perubahan yaitu:

- Mengubah nama file `.htaccess2` menjadi `.htaccess` yang berlokasi di direktori utama *Sharif Judge*. `.htaccess` merupakan sebuah konfigurasi file untuk 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 users* [4].

Tabel 2.1: *User Roles Table*

<i>User Role</i>	<i>User Level</i>
<i>Admin</i>	3
<i>Head Instructor</i>	2
<i>Instructor</i>	1
<i>Student</i>	0

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

Tabel 2.2: *Permission Table*

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

Pengguna dapat menambahkan *users* dengan mengklik pada bagian *Add Users* di halaman *Users*. Pengguna harus mengisi semua informasi yang ada pada *text area*. Baris dimulai dengan komentar *#*. Setiap baris lainnya mewakili pengguna dengan sintaks berikut:

```
USERNAME EMAIL PASSWORD ROLE
```

```
* Usernames dapat berisikan huruf kecil atau nomor dan harus terdiri
  antara 3 sampai 20 karakter.
```

```
* Passwords harus terdiri antara 6 sampai 30 karakter.
```

```
* Pengguna dapat menggunakan RANDOM[n] untuk menghasilkan password
  acak yang terdiri dari n-digit karakter.
```

```
* ROLE harus terdiri dari salah satu ini: 'admin', 'head_instructor',
  'instructor', 'student'
```

Berikut contoh penggunaan sintaks untuk menambahkan *users*:

```

1 # This is a comment!
2 # This is another comment!
3 instructor instructor@sharifjudge.ir 123456 head_instructor
4 instructor2 instructor2@sharifjudge.ir random[7] instructor
5 student1 st1@sharifjudge.ir random[6] student
6 student2 st2@sharifjudge.ir random[6] student
7 student3 st3@sharifjudge.ir random[6] student
8 student4 st4@sharifjudge.ir random[6] student
9 student5 st5@sharifjudge.ir random[6] student
10 student6 st6@sharifjudge.ir random[6] student
11 student7 st7@sharifjudge.ir random[6] student
12

```

2.2.4 Menambah *Assignment*

Pengguna dapat menambahkan *assignment* dengan cara mengklik *Add* di halaman *assignments* [4].

Pengguna akan melihat halaman seperti Gambar 2.2.

Gambar 2.2: 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*

Peserta tidak dapat mengumpulkan *assignment* sebelum "*Start Time*". Format yang digunakan untuk pengaturan *start time* adalah *MM/DD/YYYY HH:MM:SS*. Contoh: *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 ***, *-*, *+*, */*. Contoh *120* (2 jam) atau *48*60* (2 hari).

- *Participants*

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

- *Open*

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

- *Scoreboard*

Pengguna dapat mengaktifkan atau mematikan papan nilai dengan menggunakan pilihan ini.

- *Java Exceptions*

Pengguna dapat mengaktifkan dan mematikan *java exceptions* yang ditunjukkan kepada *students*. Perubahan pada pilihan ini tidak berdampak pada kode yang sebelumnya sudah dinilai. Nama *exception* akan muncul jika pada *file* *pathtester/java_exceptions_list* berisikan nama *exception* tersebut. Berikut hasil *exception* yang ditunjukkan 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
```



```

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

```

• *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 (bisa negatif). *Script* PHP pada bagian ini tidak mengandung *tags* `<?php` , `<? , ?>`. Berikut contoh *\$extra_time* 172800 (2 hari):

```

15 if ($delay<=0)
16 // no delay
17 $coefficient = 100;
18
19 elseif ($delay<=3600)
20 // delay less than 1 hour
21 $coefficient = ceil(100-((30*$delay)/3600));
22
23 elseif ($delay<=86400)
24 // delay more than 1 hour and less than 1 day
25 $coefficient = 70;
26
27 elseif (($delay-86400)<=3600)
28 // delay less than 1 hour in second day
29 $coefficient = ceil(70-((20*($delay-86400))/3600));
30
31 elseif (($delay-86400)<=86400)
32 // delay more than 1 hour in second day
33 $coefficient = 50;
34
35 elseif ($delay > $extra_time)
36 // too late
37 $coefficient = 0;
38

```

• *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 *kilobyte*, 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 default *Sharif Judge* menggunakan *diff*, namun pengguna dapat mengubah *command* pada bagian ini.

- *Diff Arguments*

Pengguna dapat mengatur argumen dari *Diff Command* disini. Untuk melihat daftar lengkap *diff* argumen, pengguna dapat melihat *man diff*. *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.

2.2.4.1 Contoh Assignment

Berikut contoh *assignment* untuk mencoba *Sharif Judge*. Menambah *assignment* dengan mengklik *Add* di halaman *Assignment*. *Assignment* dibagi menjadi 3 *problem*:

1. *Problem 1* (Penjumlahan)

Program pengguna akan menerima masukan bilangan *integer* *n*, kemudian menerima masukan lagi sebanyak *n* buah bilangan *integer* dan menampilkan hasil penjumlahan dari *n* nomor tersebut. Tabel 2.3 menunjukkan contoh *input* dan *output* untuk *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 masukan bilangan *integer* *n*, kemudian menerima masukan lagi sebanyak *n* buah bilangan *integer* dan menampilkan hasil penjumlahan dari dua nilai tertinggi. Tabel 2.4 menunjukkan contoh *input* dan *output* untuk *Problem 2*.

Tabel 2.4: *Problem 2 (Max)*

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

3. Problem 2 (*Upload!*)

Pengguna diharuskan mengunggah sebuah *file C* atau *zip*. *Problem* ini menggunakan pilihan "*Upload Only* sehingga tidak akan dinilai oleh *Sharif Judge*.

Pengguna dapat menemukan *file zip* pada *folder Assignments*. Berikut susunan pohon dari tugas 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
```

```

1 | | |-- output3.txt
2 | | |-- output4.txt
3 | | |-- output5.txt
4 | | |-- output6.txt
5 | | |-- output7.txt
6 | | |-- output8.txt
7 | | |-- output9.txt
8 | | --- output10.txt
9 | |-- desc.md
10 | --- Problem2.pdf
11 |-- p3
12 | --- desc.md
13 --- SampleAssignment.pdf
14

```

15 *Problem 1* menggunakan metode "*Tester*" untuk mengecek keluaran, sehingga memiliki *file tester.cpp*
 16 (*Tester Script*). *Problem 2* menggunakan metode *Output Comparison* untuk mengecek keluaran,
 17 sehingga memiliki dua *folder* (*in* dan *out*) yang berisikan *test case*. *Problem 3* merupakan *problem*
 18 yang menggunakan pilihan *Upload-Only*.

19 2.2.4.2 Aturan-aturan *Submission*

20 Untuk dapat mengumpulkan jawaban pada sebuah *assignment*, terdapat beberapa aturan yaitu:

- 21 1. Jawaban tidak dapat dikumpulkan jika waktu sekarang belum melewati '*Start time*' pada
 22 *assignment* yang dipilih.
- 23 2. Jawaban tidak dapat dikumpulkan jika waktu sekarang telah melewati '*Finish time*' pada
 24 *assignment* yang dipilih.
- 25 3. Jawaban tidak dapat dikumpulkan jika *assignment* yang dipilih berstatus *close*.
- 26 4. Jawaban tidak dapat dikumpulkan jika peserta tidak terdaftar pada *assignment* yang dipilih.

27 Jika seluruh aturan tersebut telah terpenuhi, maka peserta dapat mengumpulkan jawaban
 28 masing-masing. Berikut langkah-langkah untuk mengumpulkan jawaban pada sebuah *assignment*:

- 29 1. Memilih *assignment* yang ingin dikumpulkan pada halaman *Assignment*.
- 30 2. Pada halaman *Submit*, para peserta dapat memilih *problem* yang ingin dikumpulkan, bahasa
 31 pemrograman yang digunakan dan *file* jawaban.
- 32 3. Menekan tombol *Submit* untuk mengumpulkan jawaban.

33 Setelah menekan tombol *Submit*, pengguna akan diarahkan ke halaman *All Submission*. Pada
 34 tahap ini, jawaban para peserta telah berhasil dikumpulkan ke *Sharif Judge*.

2.2.5 Struktur Pengujian

Pengguna harus menyediakan sebuah *file zip* yang berisikan *test cases* ketika menambahkan *assignment*. *File zip* ini dapat berisi folder-folder untuk setiap *problem*. Pengguna harus memberikan nama pada folder sesuai aturan seperti *p1*, *p2*, *p3*, *dst*. *Assignment* yang menggunakan pilihan *Upload-Only* tidak membutuhkan *folder* [4].

2.2.5.1 Metode Pengecekan

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

- *Metode Input/Output Comparison*

Dengan metode ini, pengguna harus memasukan beberapa *file input* dan *output* pada *folder problem*. *Sharif Judge* akan memasukan 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*.

- *Metode Tester*

Dengan metode ini, pengguna harus menyediakan beberapa *file input* dan sebuah *file C++* (*tester.cpp*) dan beberapa *file output*. *Sharif Judge* akan memasukan 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 akan mengembalikan nilai 0, sebaliknya jika salah akan mengeluarkan nilai 1. Listing 2.1 menunjukan contoh kode 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]); /* Stream ini membaca
                               isi file input */
    ifstream test_out(argv[2]); /* Stream ini membaca
                                isi file output */
    ifstream user_out(argv[3]); /* Stream ini membaca
                                isi keluaran users */
```

```
1      /* Kode Pengguna */
2      /* Jika keluaran kode user benar, mengembalikan nilai 0,
3      sebaliknya mengembalikan 1 */
4      ...
5
6      ...
7
8  }
```

2.2.5.2 Contoh *File*

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

```
14 .
15 |-- p1
16 |   |-- in
17 |     |-- input1.txt
18 |     |-- input2.txt
19 |     |-- input3.txt
20 |     |-- input4.txt
21 |     |-- input5.txt
22 |     |-- input6.txt
23 |     |-- input7.txt
24 |     |-- input8.txt
25 |     |-- input9.txt
26 |     --- input10.txt
27 |   |-- out
28 |     --- output1.txt
29 |   --- tester.cpp
30 --- p2
31     |-- in
32     |   |-- input1.txt
33     |   |-- input2.txt
34     |   |-- input3.txt
35     |   |-- input4.txt
36     |   |-- input5.txt
37     |   |-- input6.txt
38     |   |-- input7.txt
39     |   |-- input8.txt
40     |   |-- input9.txt
41     |   --- input10.txt
42     --- out
```

```

1      |-- output1.txt
2      |-- output2.txt
3      |-- output3.txt
4      |-- output4.txt
5      |-- output5.txt
6      |-- output6.txt
7      |-- output7.txt
8      |-- output8.txt
9      |-- output9.txt
10     --- output10.txt
11

```

12 *Problem 1* menggunakan metode "*Tester*" untuk mengecek hasil keluaran, sehingga memiliki *file*
 13 *tester.cpp*. Listing 2.2 menunjukkan isi dari *file tester.cpp* untuk *problem 1*:

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

```

14
15  /*
16  * tester.cpp
17  */
18
19  #include <iostream>
20  #include <fstream>
21  #include <string>
22  using namespace std;
23  int main(int argc, char const *argv[])
24  {
25
26      ifstream test_in(argv[1]); /* Stream ini membaca
27      isi file input */
28      ifstream test_out(argv[2]); /* Stream ini membaca
29      isi file output */
30      ifstream user_out(argv[3]); /* Stream ini membaca
31      isi keluaran users */
32
33      /* Kode Pengguna */
34      /* Jika keluaran kode user benar, mengembalikan nilai 0,
35      sebaliknya mengembalikan 1 */
36      /* e.g.: Permasalahan: membaca n nomor dan keluarkan
37      hasil penjumlahannya: */
38
39      int sum, user_output;
40      user_out >> user_output;
41
42      if ( test_out.good() ) // if test's output file exists

```

```

1      {
2          test_out >> sum;
3      }
4      else
5      {
6          int n, a;
7          sum=0;
8          test_in >> n;
9          for (int i=0 ; i<n ; i++){
10             test_in >> a;
11             sum += a;
12         }
13     }
14
15     if (sum == user_output)
16         return 0;
17     else
18         return 1;
19
20 }
21

```

22 *Problem 2* menggunakan metode "*Input/Output Comparison*" untuk mengecek hasil keluaran,
 23 sehingga memiliki dua folder *in* dan *out* yang berisikan *test cases*. *Problem 3* menggunakan pilihan
 24 *Upload-Only*, sehingga tidak memiliki folder apapun.

25 2.2.6 Deteksi Kecurangan

26 *Sharif Judge* menggunakan *Moss* untuk mendeteksi kode yang mirip. *Moss (Measure Of Software*
 27 *Similarity)* merupakan sistem otomatis untuk menentukan kemiripan program. Pada saat ini, apli-
 28 kasi utama *Moss* telah digunakan untuk mendeteksi plagiarisme pada kelas *programming*. Pengguna
 29 dapat mengirimkan kode final (yang dipilih oleh *students* sebagai *Final Submission*) ke *server Moss*
 30 dengan satu klik [4].

31
 32 Sebelum menggunakan *Moss* ada beberapa hal yang harus diperhatikan yaitu:

- 33 • Pengguna harus mendapatkan *Moss user id* dan mengaturnya di *Sharif Judge*. Untuk
 34 mendapatkan *Moss user id*, pengguna harus terlebih dahulu mendaftar pada halaman <http://theory.stanford.edu/~aiken/moss/>. Pengguna akan mendapatkan sebuah *email* yang
 35 berisikan *script perl*. *Moss user id* berada pada *script* tersebut.

36
 37 Listing 2.3 menunjukan potongan *perl script* yang berisikan *user id*:

Listing 2.3: Potongan *perl script*

```

38 ...
39
40

```



```

1  $server = 'moss.stanford.edu';
2  $port = '7690';
3  $noreq = "Request not sent.";
4  $usage = "usage: moss [-x] [-l language] [-d]
5             [-b basefile1] ... [-b basefileN] [-m #]
6             [-c \"string\"] file1 file2 file3 ...";
7
8  #
9  # The userid is used to authenticate your queries to the server;
10     don't change it!
11  #
12  $userid=YOUR_MOSS_USER_ID;
13
14  #
15  # Process the command line options.  This is done in a non-standard
16  # way to allow multiple -b's.
17  #
18  $opt_l = "c";    # default language is c
19  $opt_m = 10;
20  $opt_d = 0;
21
22  ...
23  }
24

```

User id yang terdapat pada potongan *perl script* di atas, dapat digunakan pada *Sharif Judge* untuk mendeteksi kecurangan. Pengguna dapat menyimpan *user id* di *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* berakhir, 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*.

2.2.7 Database

Sebuah *database* dibutuhkan untuk menjalankan *Sharif Judge*. *Database* yang dapat digunakan yaitu *Mysql* dan *PostgreSQL*. Pada *database* tersebut terdapat beberapa tabel yang secara otomatis dibuat saat proses instalasi *Sharif Judge*. Beberapa tabel tersebut yaitu:

- *shj_assignments*
Tabel ini menyimpan seluruh informasi *assignment* yang ada pada *Sharif Judge*. Atribut yang terdapat pada tabel ini antara lain *id*, *name*, *problems*, *total_submits*, *open*, *scoreboard*,

javaexceptions, *description*, *start_time*, *finish_time*, *extra_time*, *late_rule*, *participants* dan *moss_update*.

- *shj_notifications*

Tabel ini menyimpan seluruh informasi notifikasi yang ada pada *Sharif Judge*. Atribut yang terdapat pada tabel ini antara lain *id*, *title*, *text* dan *time*.

- *shj_problems*

Tabel ini menyimpan seluruh informasi *problem* yang ada pada setiap *assignment*. Atribut yang terdapat pada tabel ini antara lain *assignment*, *id*, *name*, *score*, *is_upload_only*, *c_time_limit*, *python_time_limit*, *java_time_limit*, *memory_limit*, *allowed_languages*, *diff_cmd* dan *diff_arg*.

- *shj_queue*

Tabel ini menyimpan seluruh informasi antrian pada saat proses penilaian berlangsung. Atribut yang terdapat pada tabel ini antara lain *id*, *submit_id*, *username*, *assignment*, *problem* dan *type*.

- *shj_scoreboard*

Tabel ini menyimpan seluruh informasi *scoreboard* pada setiap *assignment*. Atribut yang terdapat pada tabel ini antara lain *assignment* dan *scoreboard*.

- *shj_session*

Tabel ini menyimpan seluruh informasi *session* setiap peserta *Sharif Judge*. Atribut yang terdapat pada tabel ini antara lain *session_id*, *ip_address*, *user_agent*, *last_activity* dan *user_data*.

- *shj_settings*

Tabel ini menyimpan seluruh informasi pengaturan yang ada pada *Sharif Judge*. Atribut yang terdapat pada tabel ini antara lain *shj_key* dan *shj_value*.

- *shj_submissions*

Tabel ini menyimpan seluruh informasi jawaban yang telah dikumpulkan ke *Sharif Judge*. Atribut yang terdapat pada tabel ini antara lain *submit_id*, *username*, *assignment*, *problem*, *is_final*, *time*, *status*, *pre_score*, *coefficient*, *file_name*, *main_file_name* dan *file_type*.

- *shj_users*

Tabel ini menyimpan seluruh informasi peserta yang terdaftar pada *Sharif Judge*. Atribut yang terdapat pada tabel ini antara lain *id*, *username*, *password*, *display_name*, *email*, *role*, *passchange_key*, *passchange_time*, *first_login_time*, *last_login_time*, *selected_assignment* dan *dashboard_widget_positions*.

2.3 Twig

Pada perangkat lunak *Sharif Judge*, kelas view menggunakan *framework* aplikasi yaitu *Twig*. *Twig* merupakan sebuah *template engine modern* untuk *PHP* [5]. Sebuah *template Twig* dapat

1 mengandung *variables* atau *expression* dan *tags*. *Variables* atau *expression* akan diubah pada saat
2 *template Twig* dievaluasi dan *tags* yang akan mengontrol logika dari *template* tersebut. Listing 2.4
3 menunjukkan kode template minimal yang menunjukkan beberapa hal mendasar.

Listing 2.4: Kode Template Minimal

```
4  
5 <!DOCTYPE html>  
6 <html>  
7     <head>  
8         <title>My Webpage</title>  
9     </head>  
10    <body>  
11        <ul id="navigation">  
12            {% for item in navigation %}  
13                <li><a href="{{ item.href }}">{{ item.caption }}</a></li>  
14            {% endfor %}  
15        </ul>  
16  
17        <h1>My Webpage</h1>  
18        {{ a_variable }}  
19    </body>  
20 </html>  
21
```

22 Ada dua jenis *delimiters* pada kode di atas, yaitu `{% ... %}` dan `{{ ... }}`. *Delimiters* `{%`
23 `... %}` digunakan untuk mengeksekusi sebuah *statements*. Pada kode di atas, *delimiters* `{% for`
24 `item in navigation %}` akan mengeksekusi *statements for-loops* atau pengulangan. *Delimiters* `{{`
25 `... }}` digunakan untuk menampilkan nilai. Pada kode di atas, *delimiters* `{{ item.href }}` akan
26 menampilkan nilai *item.href*, *delimiters* `{{ item.caption }}` akan menampilkan nilai *item.caption*
27 dan *delimiters* `{{ a_variable }}` akan menampilkan nilai *a_variable*.

BAB 3

ANALISIS

Bab ini membahas tentang analisis kebutuhan *Sharif Judge* yang diperlukan oleh Teknik Informatika dan solusi yang ditawarkan untuk memenuhi kebutuhan tersebut. Kebutuhan-kebutuhan tersebut didapat dari daftar *issue repository Sharif Judge* di *GitHub* dan dari para dosen pengguna *Sharif Judge*. Hasil dari analisis kebutuhan tersebut dicatat ke dalam *Google Sheets* dan didiskusikan dengan dosen pembimbing. Selain analisis kebutuhan, pada bab ini juga analisis sistem kini pada perangkat lunak *Sharif Judge*.

3.1 Analisis Kebutuhan Perangkat Lunak *Sharif Judge*

Analisis dilakukan dengan menganalisis setiap *issue* terbuka yang ada pada *repository* di *GitHub*. Pada setiap *issue* di *GitHub*, terdapat kode unik yang dimulai dengan tanda '#' dan diikuti dengan angka. Kode unik tersebut menunjukkan urutan *issue* yang dibuat oleh para pengguna *GitHub*. Dari analisa setiap *issue* yang ada, didapatkan beberapa pertanyaan dan usulan pengembangan. Beberapa *issue* yang memiliki usulan pengembangan akan dijadikan pertimbangan untuk mengembangkan *Sharif Judge*.

Analisis kebutuhan dari para dosen pengguna *Sharif Judge* dilakukan dalam bentuk wawancara secara langsung dan melalui *email*. Dosen-dosen yang telah diwawancarai antara lain:

1. Husnul Hakim
2. Claudio Franciscus
3. Vania Natali
4. Luciana Abednego
5. Joanna Helga

Tabel 3.1: Tabel Analisis Kebutuhan *Sharif Judge*

No	Deskripsi	Sumber	Issue Number / Nama Mata Kuliah	Pembuat Issue Nama Dosen	Status
1	<i>Security with PHP</i>	<i>GitHub</i>	#61	kathiedart	Akan diimplementasi
2	<i>Securing Assignment</i>	<i>GitHub</i>	#55	wojcik13	Tidak diimplementasi
3	<i>New Function</i>	<i>GitHub</i>	#53	wojcik13	Tidak diimplementasi
4	<i>Solved Problem Indicator</i>	<i>GitHub</i>	#46	atiabjobayer	Tidak diimplementasi
5	<i>Some Problem & Sugestion</i>	<i>GitHub</i>	#45	atiabjobayer	Tidak diimplementasi
6	<i>Queue failed to process if submission take too long to complete?</i>	<i>GitHub</i>	#32	truongan	Diimplemntasi
7	<i>Compilation Error on all language</i>	<i>Git</i> Hub	#34	Eririn07	Tidak diimplementasi
8	Membatasi soal (deskripsi dan PDF) hanya bisa diakses saat <i>assignment</i> "open" dan setelah waktu mulai	Dosen	AIF102	Husnul Hakim	Akan diimplementasi
9	Menguji kemiripan kode antar mahasiswa (Contek)	Dosen	AIF102	Husnul Hakim	Tidak diimplementasi
10	1 Akun hanya dapat <i>login</i> 1 waktu (Jika suatu akun sedang <i>login</i> , tidak ada lagi yang bisa <i>login</i> akun tersebut)	Dosen	AIF102	Husnul Hakim	Akan diimplementasi
11	Membatasi soal (deskripsi dan PDF) hanya bisa diakses saat <i>assignment</i> "open" dan setelah waktu mulai	Dosen	AIF102	Vania Natali	Akan diimplementasi
12	<i>Sharif Judge</i> tidak dapat menerima file dengan ekstensi *.txt untuk <i>Pre-Test</i>	Dosen	AIF102	Vania Natali	Akan diimplementasi
13	Membatasi soal (deskripsi dan PDF) hanya bisa diakses saat <i>assignment</i> "open" dan setelah waktu mulai	Dosen	AIF202	Luciana Abednego	Akan diimplementasi
14	<i>Sharif Judge</i> tidak dapat menerima file dengan ekstensi *.txt untuk <i>Pre-Test</i>	Dosen	AIF202	Luciana Abednego	Akan diimplementasi
15	Perlu ditambah petunjuk penamaan <i>file input & output</i> yg langsung bisa dilihat ketika hendak <i>upload</i>	Dosen	AIF202	Luciana Abednego	Tidak diimplementasi
16	Membatasi soal (deskripsi dan PDF) hanya bisa diakses saat <i>assignment</i> "open" dan setelah waktu mulai	Dosen	AIF202	Joanna Helga	Akan diimplementasi
17	<i>Register</i> peserta yg <i>mode batch</i> , <i>Sharif Judge</i> tidak minta nama orangnya (lebih baik ada nama orangnya)	Dosen	AIF202	Joanna Helga	Akan diimplementasi
18	Nama peserta seharusnya tidak bisa diganti (Bisa menjadi "mainan" dan tindak kecurangan karena dapat memberikan <i>hint</i>)	Dosen	AIF202	Joanna Helga	Akan diimplementasi
19	Ingin memiliki fungsi <i>Assignment</i> tidak memiliki batasan waktu (arsip soal tahun lalu dapat dikerjakan kapan saja)	Dosen	AIF202	Joanna Helga	Akan diimplementasi
20	Ingin memiliki <i>scoreboard global</i> untuk semua <i>assignment</i> .	Dosen	AIF202	Joanna Helga	Akan Diimplementasi
21	Membatasi soal (deskripsi dan PDF) hanya bisa diakses saat <i>assignment</i> "open" dan setelah waktu mulai	Dosen	AIF102 & AIF202	Claudio Fransiscus	Akan diimplementasi
22	<i>Sharif Judge</i> tidak dapat menerima file dengan ekstensi *.txt untuk <i>Pre-Test</i>	Dosen	AIF102 & AIF202	Claudio Fransiscus	Akan diimplementasi
23	UI masih merepotkan	Dosen	AIF102 & AIF202	Claudio Fransiscus	Tidak diimplementasi
24	UI ada yang tidak berguna (yang lebih banyak digunakan <i>assignment</i> , <i>submit</i> , <i>scoreboard</i> , dan hasil <i>submit</i>)	Dosen	AIF102 & AIF202	Claudio Fransiscus	Tidak diimplementasi

25	Ingin memiliki fungsi <i>reminder</i> . Banyak mahasiswa lupa mengerjakan tugas dan tidak bisa mengumpulkan. Fungsi <i>reminder</i> akan mengirimkan <i>reminder</i> ke <i>email</i> mahasiswa	Dosen	AIF102 & AIF202	Claudio Fransiscus	Tidak diimplementasi
26	Membatasi soal (deskripsi dan PDF) hanya bisa diakses saat <i>assignment</i> "open" dan setelah waktu mulai	Dosen	AIF4012	Pascal Alfadian	Akan diimplementasi
27	Integrasi login ke <i>RADIUS</i> (<i>password</i> sama dengan <i>login Windows</i>)	Dosen	AIF401	Pascal Alfadian	Akan diimplementasi
28	Mengumpulkan <i>file</i> TXT	Dosen	AIF401	Pascal Alfadian	Akan diimplementasi
29	Mengumpulkan <i>file</i> JAR (java multi kelas)	Dosen	AIF401	Pascal Alfadian	Akan diimplementasi
30	Branding Teknik Informatika	Dosen	AIF401	Pascal Alfadian	Akan diimplementasi

Tabel 3.1 menunjukkan daftar kebutuhan perangkat lunak *Sharif Judge*. Pada tabel di atas terdapat beberapa kolom, yaitu:

- Deskripsi

Jika sumber kebutuhan berasal dari *GitHub*, maka pada kolom deskripsi akan ditulis judul dari *issue* yang ada pada *repository*. Jika sumber kebutuhan berasal dari dosen, maka deskripsi kebutuhan langsung ditulis pada kolom tersebut.

- Sumber

Kolom sumber berisi sumber dari kebutuhan pengembangan *Sharif Judge* yaitu *GitHub* atau Dosen.

- *Issue Number* / Nama Mata Kuliah

Jika sumber kebutuhan berasal dari *GitHub*, maka pada kolom ini akan ditulis kode unik yang ada pada setiap *issue*. Jika sumber kebutuhan berasal dari dosen, maka pada kolom ini akan ditulis mata kuliah yang diajar oleh dosen tersebut. Keterangan kode mata kuliah AIF102 = Algoritma & Struktur Data, AIF202 = Desain & Analisis Algoritma dan AIF401 = Skripsi 1.

- Pembuat *Issue* / Nama Dosen

Jika sumber kebutuhan berasal dari *GitHub*, maka pada kolom ini akan ditulis *username* pembuat *issue* tersebut. Jika sumber kebutuhan berasal dari dosen, maka pada kolom ini akan ditulis nama dosen yang memberikan daftar kebutuhan.

- Status

Kolom Status berisikan status dari kebutuhan tersebut apakah akan diimplementasi atau tidak diimplementasi.

3.1.1 Security with PHP

Issue dengan kode unik #61 dibuat oleh pengguna *GitHub* dengan *username* danwdart pada tanggal 6 April 2017. Pada *issue* tersebut dikatakan bahwa seseorang pengguna *Sharif Judge* dapat mengubah parameter PHP *shell_exec(rm ...)* yang dikenal sebagai serangan command injection. *Command injection* adalah suatu jenis serangan yang bertujuan untuk mengeksekusi perintah secara sewenang-wenang pada perangkat lunak ¹. Untuk menghindari hal di atas, *username* danwdart menyarankan untuk mengganti perintah PHP *shell_exec(rm ...)* dengan method lain.

¹OWASP, "Command Injection" terakhir diubah 31 Mei 2018. https://www.owasp.org/index.php/Command_Injection

Pada skripsi ini, *issue* di atas akan diimplementasi. Solusi yang ditawarkan adalah mengganti penggunaan PHP *shell_exec(rm ...)* dengan *method* lain. Perintah *shell_exec("rm ...")* memiliki fungsi untuk menghapus sebuah file. Perintah tersebut dapat diganti menggunakan *method unlink()* yang memiliki fungsi sama yaitu menghapus sebuah file.

3.1.2 *Securing Assignment*

Issue dengan kode unik #55 dibuat oleh pengguna *GitHub* dengan *username wojcik13* pada tanggal 23 Oktober 2016. *Username wojcik13* menanyakan apakah ada pilihan pada *Sharif Judge* untuk mengamankan *assignment* dengan *password*.

Pada skripsi ini, *issue* di atas tidak diimplementasikan. *Issue* di atas merupakan sebuah pertanyaan fitur pada *Sharif Judge* untuk mengamankan *assignment* dengan menggunakan *password*. Oleh karena *issue* tersebut merupakan sebuah pertanyaan, maka pada skripsi ini *issue* di atas tidak diimplementasikan.

3.1.3 *New Function*

Issue dengan kode unik #53 dibuat oleh pengguna *GitHub* dengan *username wojcik13* pada tanggal 2 Oktober 2016. *Username wojcik13* menanyakan apakah ada fungsi pada *Sharif Judge* seperti menghentikan *scoreboard* dan fungsi mengatur sesi.

Pada skripsi ini, *issue* di atas tidak diimplementasikan. *Issue* di atas merupakan sebuah pertanyaan fitur pada *Sharif Judge* untuk menghentikan *scoreboard* dan mengatur sesi. Oleh karena *issue* tersebut merupakan sebuah pertanyaan, maka pada skripsi ini *issue* di atas tidak diimplementasikan.

3.1.4 *Solve Problem Indicator*

Issue dengan kode unik #46 dibuat oleh pengguna *GitHub* dengan *username atiabjobayer* pada tanggal 16 April 2016. *Username atiabjobayer* mengatakan bahwa para pengguna *Sharif Judge* tidak dapat melihat masalah yang telah diselesaikan.

Pada skripsi ini, *issue* di atas tidak diimplementasikan. Pada *issue* di atas, *username atiabjobayer* kurang spesifik dalam menjelaskan kebutuhannya. Oleh karena *issue* tersebut kurang spesifik, maka pada skripsi ini *issue* di atas tidak diimplementasikan.

3.1.5 *Some Problem & Sugestion*

Issue dengan kode unik #45 dibuat oleh pengguna *GitHub* dengan *username atiabjobayer* pada tanggal 16 April 2016. *Username atiabjobayer* mengatakan bahwa ada beberapa persoalan yang ada pada perangkat lunak *Sharif Judge*. Beberapa masalah tersebut yaitu:

1. Beberapa kontes besar pemrograman diadakan dengan metode *ACM ICPC* namun *Sharif Judge* tidak mendukung sistem penilaian *ICPC*.
2. Pengguna dapat melihat deskripsi masalah sebelum kontes dimulai. Hal ini dapat membahayakan keamanan kontes pemrograman.

3. Tampilan yang digunakan pada *Sharif Judge* tidak responsif. Pengguna tidak dapat melihat pada *device* kecil/

4. Seharusnya ada Sistem Klarifikasi. Fitur ini harus ada pada *online judge*.

5. Pengguna tidak dapat mengumpulkan kode mereka dari *text-editor*.

Pada skripsi ini, *issue* di atas tidak diimplementasikan. *Issue* di atas memiliki cakupan persoalan yang terlalu luas. Oleh karena *issue* tersebut memiliki persoalan yang terlalu luas, maka pada skripsi ini *issue* di atas tidak diimplementasikan.

3.1.6 *Queue failed to process if submission take too long to complete?*

Issue dengan kode unik #32 dibuat oleh pengguna *GitHub* dengan *username truongan*. Pada *issue* tersebut dikatakan bahwa *assignment* yang memiliki masalah dengan *test case* besar akan menyebabkan *submission status* menjadi *pending*. *User truongan* memperkirakan hal di atas terjadi dikarenakan *database connection times out*. Untuk mengatasinya, *user truongan* menambahkan *method reconnect database* pada file *Queueprocess.php*. *User truongan* mengatakan bahwa perbaikan kode yang telah dilakukan dapat mengatasi persoalan di atas.

Pada skripsi ini, *issue* di atas akan diimplementasi. Solusi yang ditawarkan untuk mengatasi hal di atas adalah mengimplementasi kode *method reconnect database* yang telah dilakukan oleh *user truongan*. *Method reconnect database* telah disediakan oleh *framework CodeIgniter* yaitu

```
$this->db->reconnect();
```

Method reconnect database akan menghubungkan kembali *database* ketika terjadi kasus seperti di atas.

3.1.7 *Compilation Error on All Language*

Issue dengan kode unik #34 dibuat oleh pengguna *GitHub* dengan *username Eririn07*. *Username Eririn07* mencoba untuk menilai sebuah kode dan mendapatkan beberapa *error*. Kode *error* yang muncul ketika menilai kode dengan bahasa *Java* adalah *Java HotSpot(TM) 64-Bit Server VM warning: INFO: os::commit_memory(0x00007f9cfd000000, 2555904, 1) failed; error='Permission denied' (errno=13)* dan Kode *error File Not Found* muncul ketika menilai kode dengan bahasa *C*. *Username Eririn07* menanyakan bagaimana mengatasi masalah di atas.

Pada skripsi ini, *issue* di atas tidak diimplementasikan. *Issue* di atas merupakan sebuah pertanyaan untuk mengatasi kode *error* yang muncul. Oleh karena *issue* tersebut merupakan sebuah pertanyaan, maka pada skripsi ini *issue* di atas tidak diimplementasikan.

3.1.8 *Membatasi Soal (Deskripsi & PDF) Hanya Bisa Diakses saat Assignment "Open" dan Setelah Waktu Mulai*

Kebutuhan ini merupakan salah satu kebutuhan yang paling banyak disebut oleh para dosen pengguna *Sharif Judge*. Perangkat lunak *Sharif Judge* terkini masih belum dapat memenuhi kebutuhan di atas. Para peserta dapat mengunduh deskripsi soal & PDF sebelum waktu *assignment*

1 dimulai. Untuk menangani hal tersebut para dosen harus mengunggah *file* PDF tepat pada saat
2 *assignment* dimulai.

3 Pada skripsi ini, kebutuhan di atas akan diimplementasi. Solusi yang ditawarkan untuk memenuhi
4 kebutuhan di atas yaitu membatasi soal agar dapat diunduh pada saat *assignment* "open" dan
5 setelah waktu mulai. Jika ada peserta yang mencoba untuk mengunduh soal (deskripsi & PDF)
6 pada saat *assignment* belum dimulai, maka akan muncul pesan error "*Selected assignment*" has not
7 *started*. Deskripsi & PDF hanya dapat diunduh tepat setelah waktu *assignment* dimulai.

8 3.1.9 Menguji Kemiripan Kode Antar Mahasiswa

9 Perangkat lunak *Sharif Judge* terkini sudah memiliki fitur untuk menguji kemiripan kode antar
10 peserta dengan menggunakan *Moss* (*Measure Of Software Similarity*). *Moss* adalah sistem otomatis
11 untuk mendeteksi kemiripan program. Aplikasi *Moss* telah berkembang dari tahun 1994 hingga
12 sekarang. Algoritma yang digunakan pada aplikasi *Moss* sangat efektif dibandingkan algoritma
13 deteksi kecurangan lainnya [6].

14 Pada skripsi ini, kebutuhan di atas tidak diimplementasikan karena aplikasi *Moss* sedang tidak
15 dapat digunakan. Hal tersebut terjadi karena aplikasi *Moss* membutuhkan port 7690, namun port
16 tersebut diblok oleh UNPAR.

17 3.1.10 Satu Akun Hanya Dapat Login Satu Waktu

18 Para peserta *Sharif Judge* dapat *login* menggunakan akunnya di beberapa komputer. Peserta
19 yang mengetahui *user* dan *password* peserta lain dapat dengan mudah *login* ke *Sharif Judge*. Hal
20 tersebut sering dijadikan celah bagi beberapa peserta untuk melakukan tindak kecurangan. Peserta
21 yang sudah *login* menggunakan akun peserta lainnya, dapat melihat dan menyalin kode yang telah
22 dikumpulkan ke *Sharif Judge*. Tindak kecurangan ini sering terjadi pada saat kuis maupun ujian.
23 Bapak Husnul Hakim menginginkan setiap akun pada perangkat lunak *Sharif Judge* hanya dapat
24 *login* satu waktu. Jika sebuah akun telah *login* di satu komputer, maka akun tersebut tidak dapat
25 *login* di komputer lainnya. Diharapkan dengan adanya fitur tersebut dapat menekan jumlah tindak
26 kecurangan yang terjadi.

27 Pada skripsi ini, kebutuhan di atas tidak diimplementasikan. Jika fitur di atas diimplementasi,
28 maka dikhawatirkan akan merepotkan admin lab. Para admin harus membuka akun pengguna
29 setiap kali ada akun yang terkunci pada sebuah komputer. Sebagai gantinya, penulis menawarkan
30 sebuah solusi. Solusi yang ditawarkan untuk mengurangi tingkat kecurangan seperti di atas yaitu
31 membuat halaman baru yang berisikan *logs username* yang berhasil *login* ke *Sharif Judge*. Halaman
32 *logs* tersebut akan mencatat *username* yang *login* menggunakan *ip* berbeda dalam waktu dibawah
33 24 jam. Dengan adanya halaman *Logs* ini, para dosen dapat memantau *username* yang *login* pada
34 dua tempat berbeda. Penggunaan halaman *logs* dapat dilihat pada *user guide* atau dokumentasi
35 *SharIF Judge* di *GitHub*.

3.1.11 Membatasi Soal (Deskripsi & PDF) Hanya Bisa Diakses saat *Assignment* "Open" dan Setelah Waktu Mulai

Kebutuhan ini telah dibahas pada sub bab sebelumnya. Lihat sub bab 3.1.8 untuk status dari kebutuhan ini.

3.1.12 Mengumpulkan *File* dengan Format TXT

Pengumpulan *file* dengan format TXT dibutuhkan untuk *Pre-test*. Perangkat lunak *Sharif Judge* yang terkini hanya dapat menerima *file* C, C++, Java, Python 2, Python 3, Zip, dan PDF. Para peserta yang ingin mengumpulkan jawaban *Pre-test*, harus terlebih dahulu mengubah ekstensi *file* menjadi Java atau mengompres *file* ke dalam Zip.

Pada skripsi ini, kebutuhan di atas akan diimplementasi. Solusi yang ditawarkan untuk kebutuhan di atas yaitu menambahkan *file* dengan format TXT agar dapat dikumpul ke *Sharif Judge*. *Assignment* yang digunakan merupakan *assignment* yang bersifat "Upload Only". Dosen dapat menambahkan format TXT pada bagian "Allowed Language" sehingga para peserta dapat mengumpulkan jawaban *Pre-test* menggunakan *file* dengan ekstensi TXT. Penggunaan fitur ini dapat dilihat pada *user guide* atau dokumentasi *SharIF Judge* di *GitHub*.

3.1.13 Membatasi Soal (Deskripsi & PDF) Hanya Bisa Diakses saat *Assignment* "Open" dan Setelah Waktu Mulai

Kebutuhan ini telah dibahas pada sub bab sebelumnya. Lihat sub bab 3.1.8 untuk status dari kebutuhan ini.

3.1.14 Mengumpulkan *File* dengan Format TXT

Kebutuhan ini telah dibahas pada sub bab sebelumnya. Lihat sub bab 3.1.12 untuk status dari kebutuhan ini.

3.1.15 Perlu Ditambah Petunjuk Penamaan *File Input* dan *Output*

Dalam membuat sebuah *assignment* pada perangkat lunak *Sharif Judge* terdapat *file test case* yang harus disertakan. *File test case* yang disertakan memiliki beberapa ketentuan. Beberapa ketentuan tersebut seperti struktur direktori dan penamaan dalam *file test case*.

Pada skripsi ini, kebutuhan di atas tidak diimplementasikan. Pada halaman *add assignment* telah disediakan *link* menuju dokumentasi *Sharif Judge* di *GitHub* yang menjelaskan ketentuan dalam menyertakan *file test case*. Ketentuan tersebut harus terpenuhi agar sebuah *assignment* dapat berjalan dengan baik. Oleh sebab itu, kebutuhan ini tidak diimplementasikan.

3.1.16 Membatasi Soal (Deskripsi & PDF) Hanya Bisa Diakses saat *Assignment* "Open" dan Setelah Waktu Mulai

Kebutuhan ini telah dibahas pada sub bab sebelumnya. Lihat sub bab 3.1.8 untuk status dari kebutuhan ini.

3.1.17 Pendaftaran Peserta Disertai dengan *Display Name*

Pendaftaran peserta ke dalam *Sharif Judge* terkini tidak disertai dengan *Display Name*. Perangkat lunak *Sharif Judge* membutuhkan empat buah parameter yang dipisah menggunakan spasi untuk mendaftarkan peserta. Parameter tersebut antara lain, *username*, *email*, *password* dan *role*. Contoh penggunaannya seperti "i14085 i14085@unpar.ac.id random85 student" yang artinya peserta didaftarkan menggunakan *username* i14085, *email* i14085@unpar.ac.id, *password* random85 dan *role* sebagai *student*. Setiap peserta yang berhasil didaftarkan masih belum memiliki *Display Name*. Para peserta harus memasukan *Display Name* masing-masing secara manual.

Pada skripsi ini, kebutuhan di atas akan diimplementasi. Solusi yang ditawarkan untuk memenuhi kebutuhan di atas yaitu menambahkan parameter *Display Name* pada saat pendaftaran peserta. Pendaftaran peserta menggunakan lima buah paramater dan dipisah menggunakan tanda koma. Parameter tersebut antara lain, *username*, *email*, *display name*, *password*, dan *role*. Contoh penggunaan parameter di atas seperti "i14085,i14085@unpar.ac.id,Budi Simon,random85,student" yang artinya peserta didaftarkan menggunakan *username* i14085, *email* i14085@unpar.ac.id, *display name* Budi Simon, *password* random85 dan *role* sebagai *student*. Dengan pengimplementasian fitur ini, setiap peserta yang didaftarkan akan langsung memiliki *Display Name* masing-masing. Penggunaan fitur ini dapat dilihat pada *user guide* atau dokumentasi *SharIF Judge* di *GitHub*.

3.1.18 Nama Pengguna *Sharif Judge* Seharusnya Tidak Bisa Diubah

Display Name pada perangkat lunak *Sharif Judge* berfungsi sebagai nama peserta. Selain itu, nama peserta akan muncul pada halaman *Scoreboard* sebuah *assignment* yang dapat dilihat oleh seluruh peserta. *Sharif Judge* yang terkini mengizinkan para peserta untuk mengubah *Display Name* pada halaman *Profile*. Hal tersebut dapat dijadikan sebuah "mainan" dan tindakan kecurangan karena dapat memberikan *hint* untuk peserta lain. Oleh karena itu, Ibu Joanna Helga menginginkan nama peserta yang terdaftar pada *Sharif Judge* tidak dapat diubah.

Pada skripsi ini, kebutuhan di atas akan diimplementasi. Solusi yang ditawarkan untuk memenuhi kebutuhan di atas yaitu menambahkan sebuah fitur yang dapat mengunci *Display Name* peserta *Sharif Judge*. Fitur ini diletakan pada halaman *Settings* yang dapat diatur oleh *admin*. Jika *admin* mengaktifkan fitur tersebut, maka *input text Display Name* pada halaman *Profile* menjadi nonaktif sehingga para peserta tidak dapat mengubah *Display Name*. Sebaliknya jika *admin* menonaktifkan fitur tersebut, maka *input text Display Name* pada halaman *Profile* akan kembali aktif. Penggunaan fitur ini dapat dilihat pada *user guide* atau dokumentasi *SharIF Judge* di *GitHub*.

3.1.19 *Sharif Judge* Diharapkan Memiliki Fungsi *Assignment* Dapat Dikumpulkan Tanpa Adanya Batasan Waktu

Pada masa Pra UTS dan Pra UAS biasanya para dosen akan memberikan *assignment* sebagai bahan pembelajaran. Arsip-arsip soal ujian dan latihan tahun lalu akan dijadikan sebuah *assignment* yang dapat dikerjakan oleh para peserta. *Assignment* tersebut memiliki waktu pengumpulan yang cenderung lama. Ibu Joanna Helga menginginkan sebuah fitur yang *Sharif Judge* dapat mengatur *assignment* tertentu menjadi tidak memiliki batasan waktu dan dapat dikumpulkan kapan saja.

Pada skripsi ini, kebutuhan di atas akan diimplementasi. Solusi yang ditawarkan untuk memenuhi

kebutuhan di atas yaitu membuat sebuah fitur tambahan pada *assignment*. *Assignment* yang mengaktifkan fitur ini tidak akan muncul pada kalender yang terdapat di halaman *Dashboard*. Fitur tersebut akan membuat batas waktu pengumpulan menjadi tanggal 18 Januari 2038. Tanggal 18 Januari 2038 diambil karena merupakan batas dari waktu *UNIX*. Batas tersebut diperoleh karena setiap detik yang dilewati sejak tanggal 1 Januari 1970 disimpan menggunakan *integer 32-bit*. Tanggal 18 Januari 2038 waktu UTC+7 merupakan batas maksimum dari *integer 32-bit* tersebut. Masalah ini dikenal sebagai masalah "Year 2038 problem". Penggunaan fitur ini dapat dilihat pada *user guide* atau dokumentasi *SharIF Judge* di *GitHub*.

3.1.20 *Sharif Judge* Diharapkan Memiliki *Scoreboard Global* untuk Semua *Assignment*

Sharif Judge terkini memiliki halaman *Scoreboard* yang berfungsi menampilkan seluruh nilai akhir para peserta dari sebuah *assignment*. Pada halaman *Socreboard* juga menampilkan nilai dari setiap *problem* yang ada pada sebuah *assignment*. Nilai yang muncul pada halaman ini adalah nilai para peserta yang telah mengumpulkan jawabannya. Nilai yang muncul tersebut akan diurutkan mulai dari yang tertinggi hingga terendah. Ibu Joanna Helga menginginkan sebuah *Scoreboard global* untuk semua *assignment*. *Scoreboard global* tersebut menampilkan total skor beserta detail skor setiap *problem* yang telah dikerjakan para peserta diseluruh *assignment* yang ada.

Pada skripsi ini, kebutuhan di atas akan diimplementasi. Solusi yang ditawarkan untuk memenuhi kebutuhan di atas yaitu membuat halaman baru yang diberi nama *Hall of Fame*. Halaman *Hall of Fame* menampilkan berapa *problem* yang telah dikerjakan oleh para peserta diseluruh *assignment* yang ada. Nama peserta yang muncul pada halaman ini diurutkan sesuai dengan total skor dari seluruh *assignment* yang telah dikerjakan oleh para peserta. Penggunaan halaman *Hall of Fame* dapat dilihat pada *user guide* atau dokumentasi *SharIF Judge* di *GitHub*.

3.1.21 Membatasi Soal (Deskripsi & PDF) Hanya Bisa Diakses saat *Assignment* "Open" dan Setelah Waktu Mulai

Kebutuhan ini telah dibahas pada sub bab sebelumnya. Lihat sub bab 3.1.8 untuk status dari kebutuhan ini.

3.1.22 Mengumpulkan *File* dengan Format *TXT*

Kebutuhan ini telah dibahas pada sub bab sebelumnya. Lihat sub bab 3.1.12 untuk status dari kebutuhan ini.

3.1.23 UI Masih Merepotkan

Bapak Claudio Fransiscus mengeluhkan UI pada perangkat lunak *Sharif Judge* merepotkan. Contohnya seperti pada saat peserta ingin memilih *assignment*, para peserta harus masuk ke halaman *assignment* dan memilih *assignment* yang ingin dikerjakan. Contoh lainnya seperti skenario mengumpulkan jawaban, para peserta harus masuk ke halaman *submit*, memilih *problem* yang ingin dikumpulkan, memilih bahasa, memilih *file* yang akan dikumpulkan dan *submit*.

Pada skripsi ini, kebutuhan di atas tidak diimplementasikan. Bapak Claudio Fransiscus masih kurang spesifik dalam menjelaskan kebutuhannya. Oleh karena kebutuhan tersebut kurang spesifik, maka pada skripsi ini kebutuhan di atas tidak diimplementasikan.

3.1.24 UI Ada yang Tidak Berguna

Bapak Claudio Fransiscus mengeluhkan beberapa UI pada perangkat lunak *Sharif Judge* tidak berguna. Pada *side bar Sharif Judge* terdapat beberapa menu halaman. Menurut Bapak Claudio Fransiscus, beberapa *menu* tersebut tidak semuanya terpakai. *Menu* yang sering digunakan hanya *Assignments*, *Submit*, *Submission* dan *Scoreboard*.

Pada skripsi ini, kebutuhan di atas tidak diimplementasikan. Claudio Fransiscus masih kurang spesifik dalam menjelaskan kebutuhannya. Oleh karena kebutuhan tersebut kurang spesifik, maka pada skripsi ini kebutuhan di atas tidak diimplementasikan.

3.1.25 *Sharif Judge* Diharapkan Memiliki Fungsi *Reminder*

Setiap assignment pada perangkat lunak *Sharif Judge* memiliki batas pengumpulan. Jika *assignment* telah melewati batas pengumpulan maka para peserta tidak dapat mengumpulkan tugasnya. Banyak peserta sering kali lupa untuk mengerjakan *assignment* dan pada akhirnya melewati batas pengumpulan. Bapak Claudio Fransiscus menginginkan perangkat lunak *Sharif Judge* yang memiliki fitur *reminder*. Fitur *reminder* akan mengirimkan *email* ke setiap peserta yang berisikan peringatan bahwa ada *assignment* yang harus diselesaikan.

Pada skripsi ini, kebutuhan di atas tidak diimplementasikan. Kebutuhan di atas belum dapat diimplementasi karena masih belum ada sistem *scheduler*. Selain itu, bobot pekerjaan diperkirakan akan membutuhkan waktu lebih dari 1 semester.

3.1.26 Membatasi Soal (Deskripsi & PDF) Hanya Bisa Diakses saat *Assignment "Open"* dan Setelah Waktu Mulai

Kebutuhan ini telah dibahas pada sub bab sebelumnya. Lihat sub bab 3.1.8 untuk status dari kebutuhan ini.

3.1.27 Integrasi *Login* ke *Server RADIUS*

RADIUS (Remote Authentication Dial In User Service) merupakan protokol jaringan klien dan server. Klien mengirimkan informasi pengguna ke server *RADIUS* yang ditunjuk dan akan bertindak berdasarkan respons yang dikembalikan. Server *RADIUS* akan menerima permintaan koneksi pengguna, mengautentikasi pengguna dan kemudian mengembalikan informasi konfigurasi yang diperlukan agar klien dapat memberikan layanan kepada pengguna. Server *RADIUS* dapat bertindak sebagai klien *proxy* ke server *RADIUS* lain atau server autentikasi jenis lainnya². Lab FTIS UNPAR memiliki server *RADIUS* yang dapat memverifikasi ID mahasiswa terhadap kata sandinya. Server *RADIUS* juga berguna untuk autentikasi ID mahasiswa agar dapat menggunakan komputer di Lab FTIS UNPAR.

²Cisco, "How Does RADIUS Work?" How Does RADIUS Work? - Cisco. <https://www.cisco.com/c/en/us/support/docs/security-vpn/remote-authentication-dial-user-service-radius/12433-32.html> (diakses 22 Februari 2018)

Pada skripsi ini, kebutuhan di atas akan diimplementasi. Solusi yang ditawarkan untuk memenuhi kebutuhan di atas yaitu mengintegrasikan *login server RADIUS* pada perangkat lunak *Sharif Judge*. Dengan pengimplementasian integrasi *login RADIUS* pada *Sharif Judge*, para peserta dapat *login* ke *Sharif Judge* menggunakan akun yang terdapat pada *server RADIUS*.

3.1.28 Mengumpulkan *File* dengan Format TXT

Kebutuhan ini telah dibahas pada sub bab sebelumnya. Lihat sub bab 3.1.12 untuk status dari kebutuhan ini.

3.1.29 Mengumpulkan *File* JAR (Java Multi Kelas)

JAR (*Java ARchive*) adalah format *file* platform-independen yang menggabungkan banyak *file* menjadi satu. *File-file* seperti kelas, gambar dan suara dapat digabungkan dalam *file* JAR. Perangkat lunak *Sharif Judge* yang terkini tidak mendukung pengumpulan *file* dengan ekstensi JAR. *Sharif Judge* hanya menerima beberapa tipe file yaitu C, C++, Java, Python 2, Python 3, Zip, dan PDF.

Pada skripsi ini, kebutuhan di atas akan diimplementasi. Solusi yang ditawarkan untuk kebutuhan di atas yaitu menambahkan file dengan format JAR agar dapat dikumpul ke *Sharif Judge*. File dengan format JAR akan langsung dinilai oleh *Sharif Judge* sehingga para peserta dapat melihat hasil dari jawaban masing-masing.

3.1.30 Branding Teknik Informatika

Branding Teknik Informatika dilakukan dengan cara mengubah logo dan ikon *Sharif Judge* menjadi logo Teknik Informatika. Selain mengubah logo dan ikon, perubahan juga dilakukan terhadap nama perangkat lunak menjadi SharIF Judge dan link dokumentasi yang ada pada perangkat lunak *Sharif Judge*. Hal di atas dapat dilakukan karena *Sharif Judge* sendiri menggunakan lisensi GPL versi 3. GPL merupakan kepanjangan dari *General Public License* yang memberikan beberapa kebebasan pada setiap penggunaanya [7]. Kebebasan tersebut antara lain:

- Kebebasan untuk menggunakan perangkat lunak dengan tujuan apapun
- Kebebasan untuk mengubah perangkat lunak sesuai dengan kebutuhan
- Kebebasan untuk membagikan perangkat lunak kepada teman dan kerabat
- Kebebasan untuk membagikan perubahan yang telah dilakukan

Pada skripsi ini, kebutuhan di atas akan diimplementasi. Gambar 3.1 menunjukkan logo dan ikon yang akan digunakan. Gambar 3.2 menunjukkan *banner Sharif Judge* yang akan digunakan.



Gambar 3.1: Logo dan Ikon



Gambar 3.2: Banner Sharif Judge

3.2 Analisis Sistem Kini

Seperti yang telah dibahas pada Bab 2.2, Sharif Judge menggunakan framework Codeigniter. Framework ini menerapkan prinsip Model-View-Controller (M-V-C). Model-View-Controller merupakan metode untuk membuat sebuah aplikasi dengan memisahkan data (Model) dari tampilan (View) dan cara memprosesnya (Controller). Pada perangkat lunak Sharif Judge model, view dan controller terdapat pada folder Application serta dipisahkan ke dalam masing-masing folder.

3.2.1 Model

Direktori model perangkat lunak Sharif Judge terdapat pada Sharif-Judge\application\models. Di dalam folder models, terdapat beberapa kelas model yang berisikan fungsi-fungsi untuk membantu pengguna mengambil, menyimpan, dan memperbarui informasi pada database.

3.2.1.1 Assignment_Model.php

Pada file Assignment_Model.php terdapat beberapa fungsi yaitu:

- *add_assignment*: menambahkan *assignment* baru ke dalam *database* atau mengedit *assignment* yang ada
- *delete_assignment*: menghapus *assignment*
- *all_assignments*: menampilkan semua daftar *assignment* dan informasinya
- *new_assignment_id*: mengembalikan nilai int terkecil yang dapat digunakan sebagai id untuk *assignment* baru
- *all_problems*: mengembalikan *array* yang berisi semua masalah dalam *assignment* yang diberikan
- *problem_info*: mengembalikan baris *database* untuk masalah tertentu
- *assignment_info*: mengembalikan baris *database* untuk *assignment* tertentu
- *is_participant*: mengecek apakah pengguna terdaftar sebagai peserta pada *assignment* tertentu
- *increase_total_submits*: meningkatkan jumlah total *submit* sebanyak satu
- *set_moss_time*: mengedit "Moss Update Time" untuk *assignment* tertentu
- *get_moss_time*: mengembalikan "Moss Update Time" untuk *assignment* tertentu

- *save_problem_description*: menambahkan atau Memperbarui deskripsi masalah
- *_update_coefficients*: memperbarui koefisien yang terdapat pada *assignment* tertentu. Fungsi ini dipanggil oleh fungsi *add_assignment*

3.2.1.2 *Notifications_Model.php*

Pada file *Notifications_Model.php* terdapat beberapa fungsi yaitu:

- *get_all_notifications*: mengembalikan semua notifikasi
- *get_latest_notifications*: mengembalikan 10 notifikasi terkini
- *add_notification*: menambahkan notifikasi baru
- *update_notification*: mengedit notifikasi tertentu
- *delete_notification*: menghapus notifikasi tertentu
- *get_notification*: menampilkan notifikasi tertentu
- *have_new_notification*: mengecek apakah terdapat notifikasi setelah melewati waktu tertentu

3.2.1.3 *Queue_Model.php*

Pada file *Queue_Model.php* terdapat beberapa fungsi yaitu:

- *in_queue*: mengecek apakah sebuah submission telah berada dalam antrean
- *get_queue*: mengembalikan semua antrean *submission*
- *empty_queue*: mengosongkan antrean
- *add_to_queue*: memasukan *submission* ke dalam antrean
- *rejudge*: menambahkan *submission* ke dalam antrean untuk di-*rejudge*
- *rejudge_single*: menambahkan *submission* tunggal ke dalam antrean untuk di-*rejudge*
- *get_first_item*: mengembalikan *item* pertama dari antrean
- *remove_item*: menghapus *item* tertentu dari antrean
- *save_judge_result_in_db*: menyimpan hasil dari *judge* ke dalam *database*. Fungsi ini dipanggil oleh fungsi *Queueprocess* di *Controller*

3.2.1.4 *Scoreboard_Model.php*

Pada file *Scoreboard_Model.php* terdapat beberapa fungsi yaitu:

- *_generate_scoreboard*: menghasilkan *scoreboard* dari *Final Submissions* untuk *assignment* tertentu. Fungsi ini dipanggil oleh *update_scoreboard*

- *update_scoreboards*: mengupdate *cache scoreboard* dari semua *assignment*. Fungsi ini dipanggil setiap kali pengguna menghapus atau semua *assignment* pengguna dihapus.
- *update_scoreboard*: mengupdate *cache scoreboard* dari semua *assignment* dan menyimpan kode *html scoreboard* ke dalam *database*. Fungsi ini dipanggil setelah *judging/rejudging* sebuah *submission* dan ketika beberapa *settings* diubah pada *assignment* tertentu. *Setting* tersebut terdiri dari *Extra Time*, *Start Time*, *Finish Time*, *Coefficient's Rule* dan *Enable/Disable Scoreboard*.
- *get_scoreboard*: mengembalikan *cache scoreboard* untuk *assignment* tertentu sebagai teks *html*

3.2.1.5 *Settings_Model.php*

Pada file *Settings_Model.php* terdapat beberapa fungsi yaitu:

- *get_setting*: mengembalikan pengaturan tertentu
- *set_setting*: mengupdate pengaturan tertentu
- *get_all_settings*: menampilkan semua pengaturan
- *set_setting*: mengupdate seluruh pengaturan

3.2.1.6 *Submit_Model.php*

Pada file *Submit_Model.php* terdapat beberapa fungsi yaitu:

- *get_submission*: mengembalikan baris tabel sebuah *submission* tertentu
- *get_final_submissions*: mengembalikan baris tabel *final submission* para peserta dari sebuah *submission* tertentu
- *get_all_submissions*: mengembalikan baris tabel seluruh *submission*
- *count_final_submissions*: mengembalikan jumlah total *final submission* dari peserta tertentu
- *count_all_submissions*: mengembalikan jumlah total *submission* dari peserta tertentu
- *set_final_submission*: menentukan *submission* tertentu agar menjadi *final submission*
- *add_upload_only*: menambahkan hasil dari *submit "Upload only"* ke *database*

3.2.1.7 *User.php*

Pada file *User.php* terdapat beberapa fungsi yaitu:

- *select_assignment*: menetapkan *assignment* yang dipilih untuk username tertentu
- *save_widget_positions*: mengupdate posisi dari *dashboard widget* ke *database*
- *get_widget_positions*: mengembalikan posisi *dashboard widget* dari *database*

3.2.1.8 *User_Model.php*

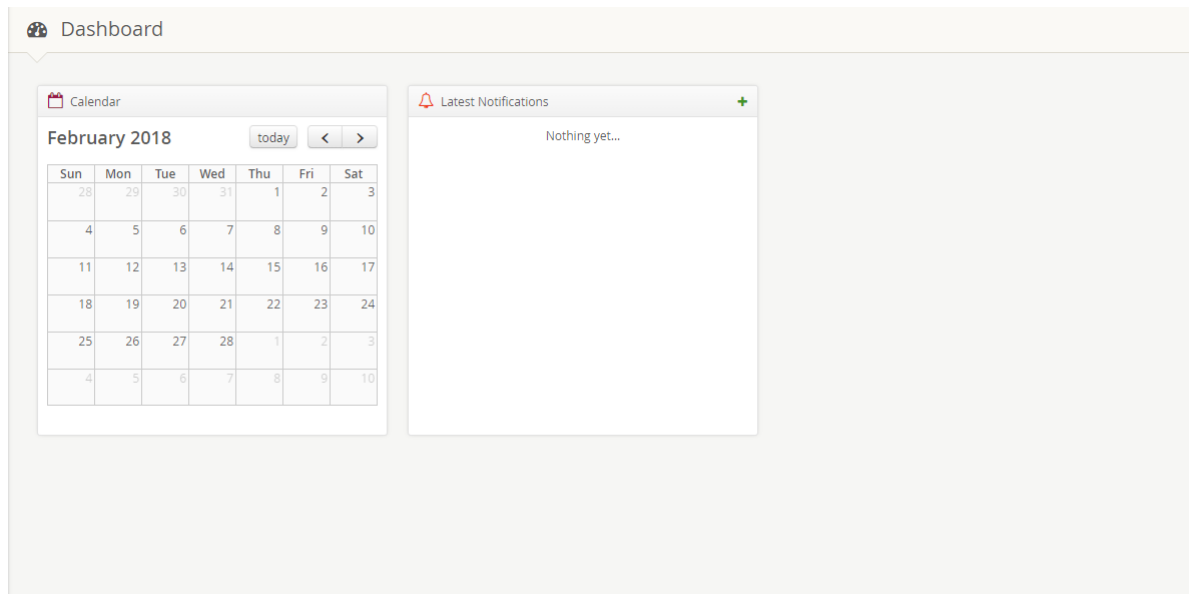
Pada file *User_Model.php* terdapat beberapa fungsi yaitu:

- *have_user*: mengecek apakah pengguna dengan *username* yang diinput terdapat di *database*
- *user_id_to_username*: mengubah *user id* menjadi *username*
- *username_to_user_id*: mengubah *username* menjadi *user id*
- *have_email*: mengecek apakah pengguna dengan *email* yang diberikan terdapat di *database*
- *add_user*: menambahkan pengguna tunggal
- *add_users*: menambahkan beberapa pengguna
- *delete_user*: menghapus pengguna tertentu
- *delete_submissions*: menghapus semua *submission* pengguna tertentu
- *validate_user*: mengecek apakah *user* dan *password* yang diinput cocok untuk keperluan *login*
- *selected_assignment*: mengembalikan *assignment* yang dipilih
- *get_names*: mengembalikan nama dari *username* tertentu
- *update_profile*: memperbarui *user profile* (*Name*, *Email*, *Password*, *Role*)
- *send_password_reset_mail*: Menghasilkan *password reset key* dan mengirim *email* berisi link untuk mereset *password*
- *passchange_is_valid*: mengecek apakah *password reset key* sesuai
- *reset_password*: mereset *password* untuk *password reset key* yang diberikan
- *get_all_users*: menampilkan seluruh *user* yang ada pada *database*
- *get_user*: mengembalikan baris tabel *user* dengan *user id* tertentu
- *update_login_time*: memperbarui *First Login Time* dan *Last Login Time* *username* tertentu

3.2.2 *View*

Direktori *view* perangkat lunak *Sharif Judge* terdapat pada *Sharif-Judge\application\views*. Di dalam folder *views*, terdapat beberapa folder yang memisahkan kelas *view* diantaranya adalah folder *error*, *pages* dan *templates*. Folder *error* berisikan tampilan ketika pengguna melakukan kesalahan seperti *404 Page Not Found* atau *Database Error*. Folder *template* berisikan tampilan dasar penyusun *Sharif Judge* seperti *Top Bar*, *Side Bar* dan *Header*. Folder *pages* berisikan tampilan utama *Sharif Judge*. Pada folder ini terdapat beberapa folder lain yaitu *admin* dan *authentication*. Folder *admin* berisikan tampilan yang hanya dapat dilihat oleh *admin* seperti tampilan *Settings*, *User*, *Install*, *Add Assignemnt*, *Add Notification* dan lain-lain. Folder *authentication* berisikan tampilan yang berhubungan dengan akses akun pengguna seperti tampilan *Login*, *Register* dan *Reset Password*.

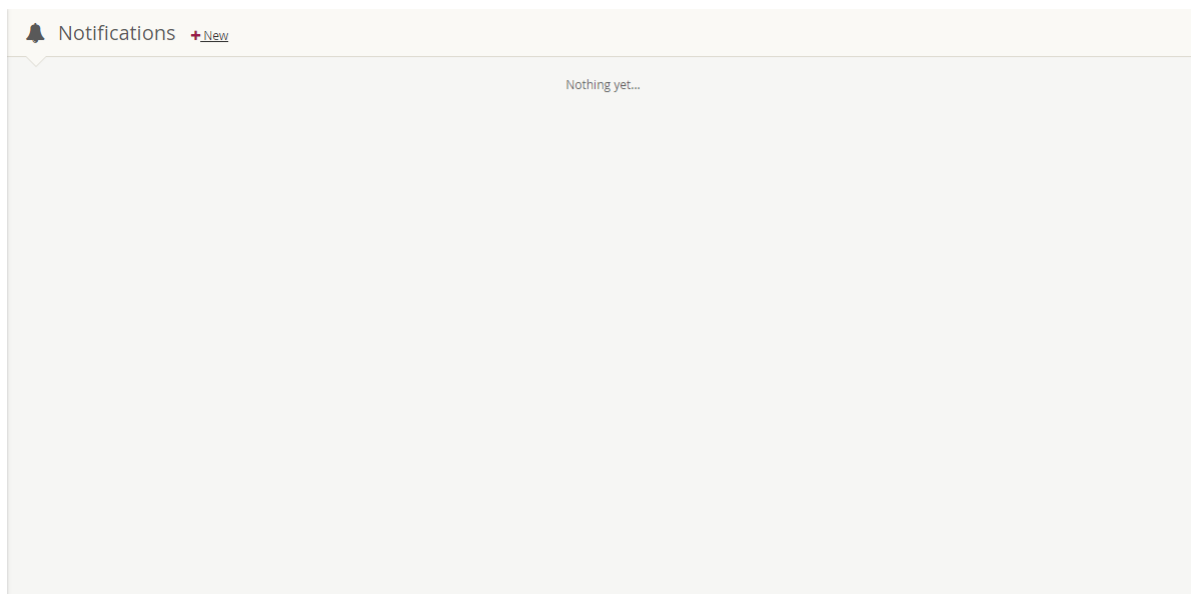
3.2.2.1 *dashboard.twig*



Gambar 3.3: Halaman *Dashboard*

Gambar 3.3 menunjukkan halaman *dashboard*. *Dashboard* merupakan tampilan utama setelah pengguna berhasil login pada perangkat lunak *Sharif Judge*. Pada tampilan dashboard, terdapat kalender dan tabel notifikasi. Kalender akan menampilkan keterangan *assignment* yang ada pada *Sharif Judge* dan tabel notifikasi akan berisikan 10 pengumuman terbaru yang telah dibuat oleh admin.

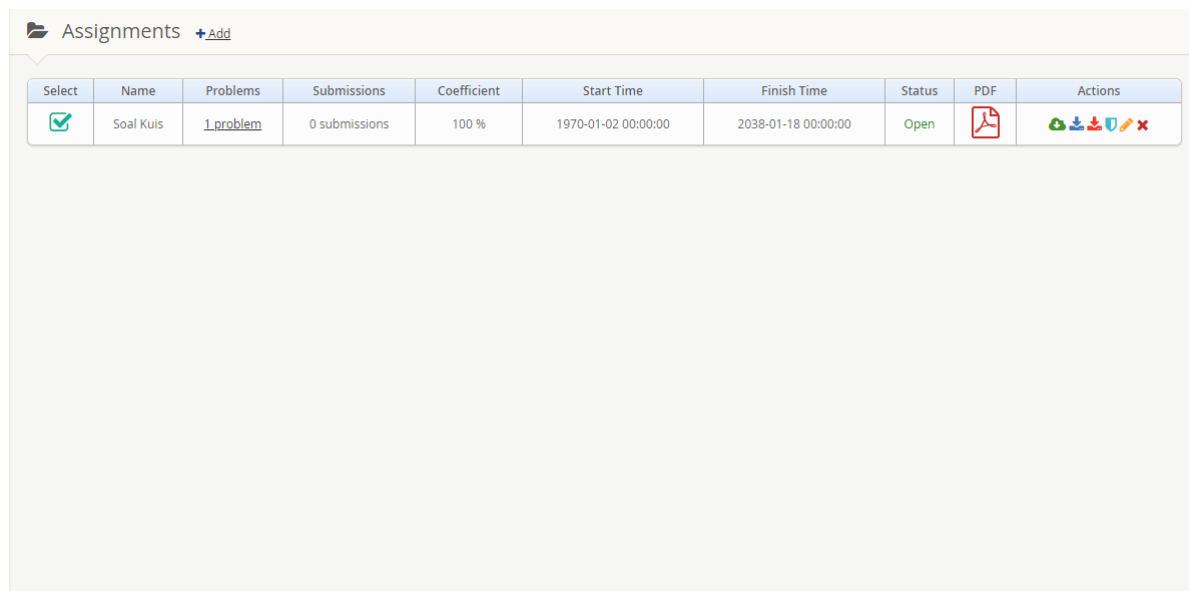
3.2.2.2 *notifications.twig*



Gambar 3.4: Halaman *Notifications*

- 1 [Gambar 3.4](#) menunjukkan halaman notifikasi. Halaman notifikasi berisikan seluruh notifikasi yang
 2 telah dibuat oleh admin. Jika admin yang memasuki halaman tersebut, maka terdapat pilihan
 3 "New". Fungsi "New" tersebut memungkinkan admin dapat menambahkan pengumuman baru untuk
 4 para pengguna *Sharif Judge*. Pengumuman tersebut akan muncul pada tabel notifikasi di *dashboard*.

5 3.2.2.3 *assignments.twig*



Select	Name	Problems	Submissions	Coefficient	Start Time	Finish Time	Status	PDF	Actions
<input checked="" type="checkbox"/>	Soal Kuis	1 problem	0 submissions	100 %	1970-01-02 00:00:00	2038-01-18 00:00:00	Open		

Gambar 3.5: Halaman *Assignments*

- 6 [Gambar 3.5](#) menunjukkan halaman *assignments*. Halaman *assignments* berisikan seluruh *assignment*
 7 yang ada pada *Sharif Judge*. Pada halaman ini, pengguna dapat memilih *assignment* mana yang
 8 akan dikerjakan. Jika admin yang memasuki halaman tersebut, maka terdapat beberapa pilihan
 9 tambahan. Beberapa pilihan tambahan seperti fungsi "Add" akan mengarahkan admin ke halaman
 10 baru untuk menambah *assignment* dan beberapa fungsi lain untuk mengedit, menghapus atau
 11 mengunduh *assignment* yang sudah ada.

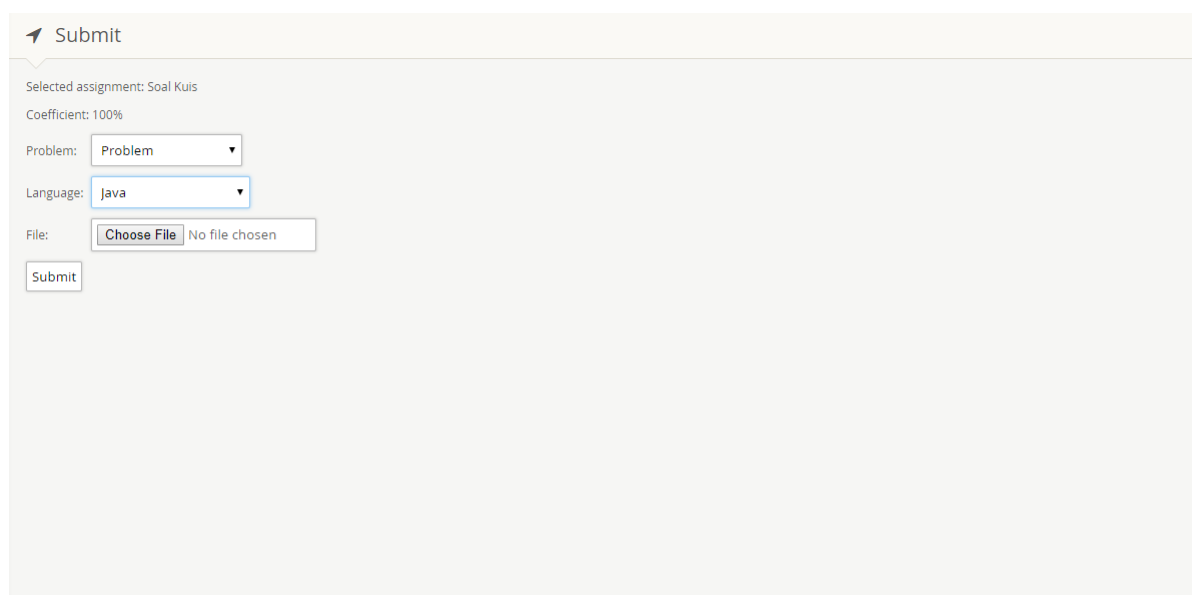
1 3.2.2.4 *problems.twig*



Gambar 3.6: Halaman *Problems*

2 Gambar 3.6 menunjukkan halaman *problems*. Halaman *problems* berisikan masalah-masalah yang ada
3 pada suatu *assignment*. Pada halaman ini, pengguna dapat melihat deskripsi pada masing-masing
4 masalah. Selain melihat deskripsi masalah, para pengguna juga dapat mengumpulkan jawaban dari
5 setiap masalah tersebut.

6 3.2.2.5 *submit.twig*

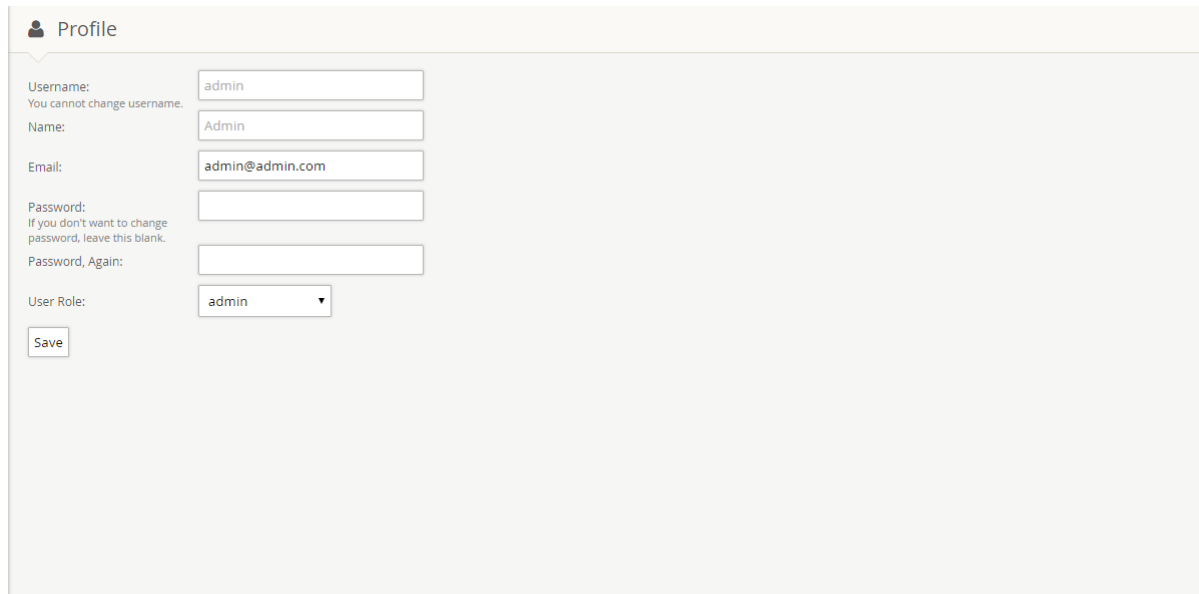


Gambar 3.7: Halaman *Submit*

7 Gambar 3.7 menunjukkan halaman *submit*. Halaman *submit* berfungsi untuk mengumpulkan jawaban
8 dari *assignment* yang telah dipilih pengguna sebelumnya. Pengguna dapat memilih masalah telah

- 1 diselsaikan lalu memilih bahasa yang digunakan dan memilih *file* jawaban yang akan dikumpulkan.
- 2 Setelah mengumpulkan jawaban, pengguna akan diarahkan ke halaman *All Submission* untuk
- 3 melihat hasil dari jawaban yang telah dikumpulkan sebelumnya.

4 3.2.2.6 *profile.twig*



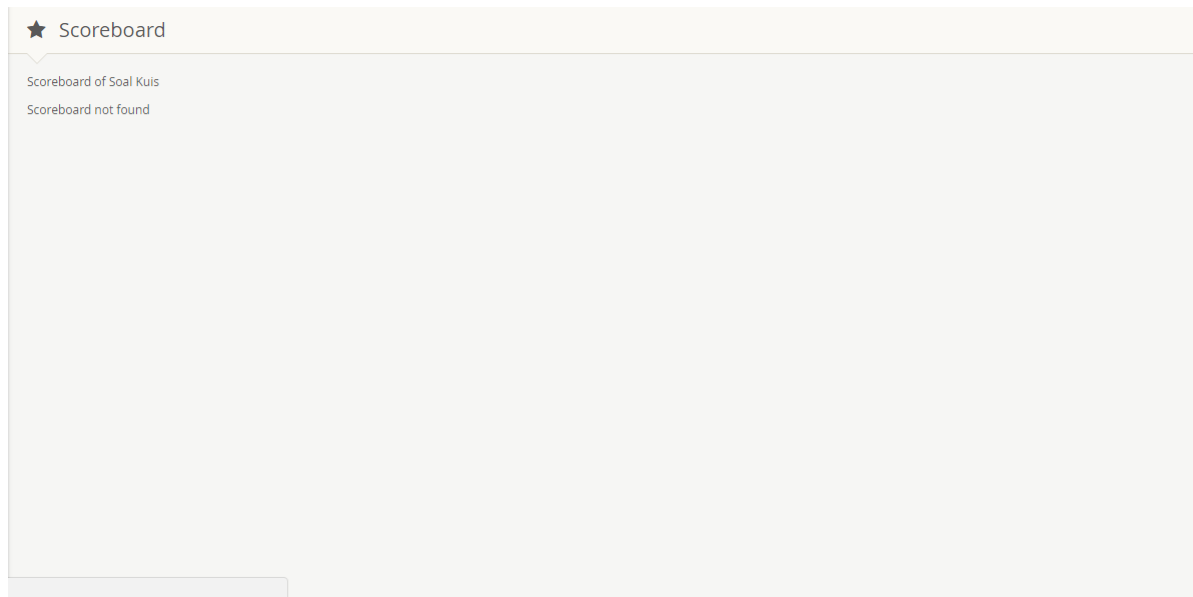
The screenshot shows a web interface titled "Profile" with a user icon. It contains several input fields for updating user information:

- Username:** A text box containing "admin" with a note below it: "You cannot change username."
- Name:** A text box containing "Admin".
- Email:** A text box containing "admin@admin.com".
- Password:** A text box with a note: "If you don't want to change password, leave this blank."
- Password, Again:** A text box for re-entering the password.
- User Role:** A dropdown menu currently showing "admin".
- Save:** A button at the bottom left of the form.

Gambar 3.8: Halaman *Profile*

- 5 [Gambar 3.8](#) menunjukkan halaman *profile*. Halaman *profile* berisikan keterangan akun pengguna
- 6 *Sharif Judge*. Pada halaman ini, pengguna dapat mengubah nama, *email* dan *password*. Jika
- 7 admin yang memasuki halaman ini, maka terdapat tambahan pilihan yaitu *user role*. Admin dapat
- 8 mengubah user role menjadi *head_instructor*, *instructor* dan *student*.

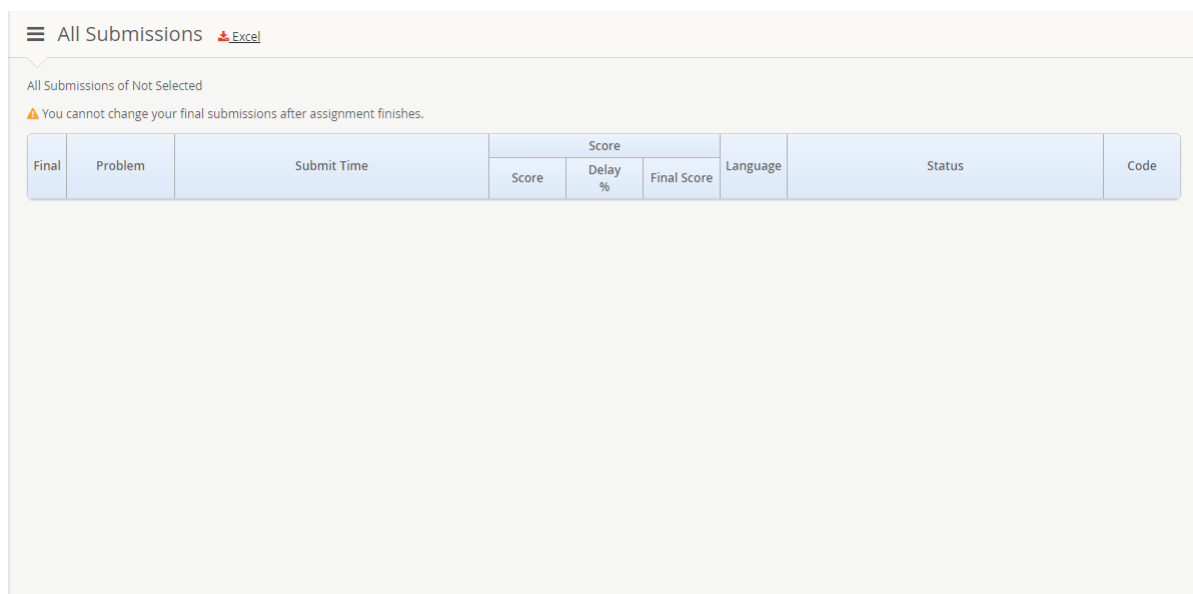
1 3.2.2.7 *scoreboard.twig*



Gambar 3.9: Halaman *Scoreboard*

2 Gambar 3.9 menunjukkan halaman *scoreboard*. Halaman *scoreboard* berisikan nilai seluruh pengguna
 3 *Sharif Judge* pada *assignment* tertentu. Nama pengguna yang muncul akan terurut secara menurun
 4 berdasarkan nilai *assignment* pengguna. Admin dapat menonaktifkan *scoreboard* dengan cara
 5 menghilangkan *checkbox* "Scoreboard" pada halaman *Add Assignment*.

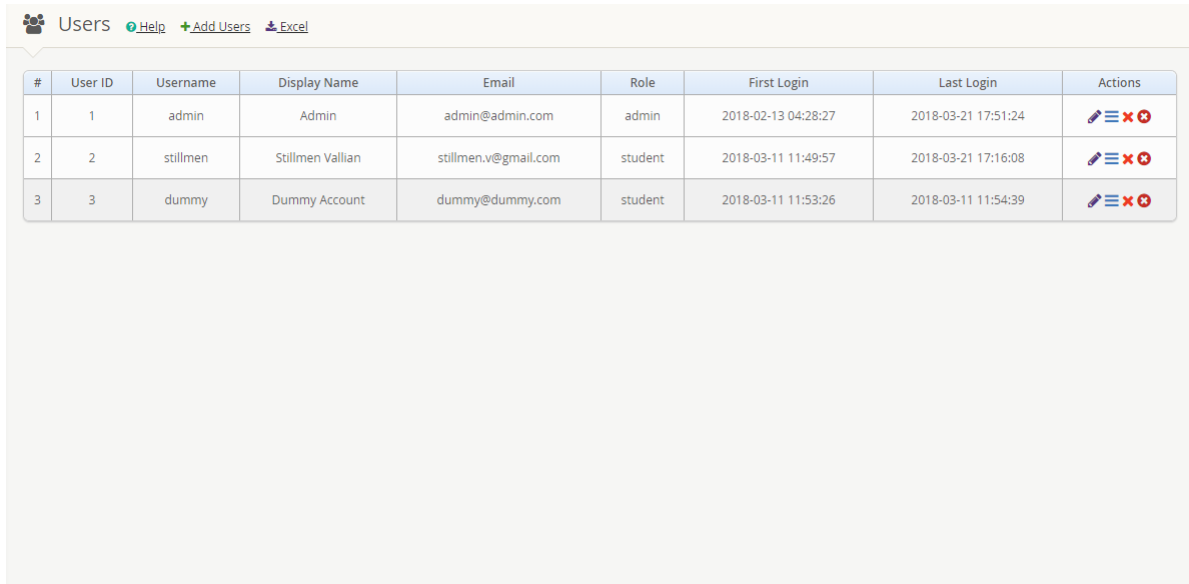
6 3.2.2.8 *submission.twig*












Gambar 3.10: Halaman *All Submission*

1 *tester, email, sandboxing, shield* dan lain-lain.

2 3.2.2.10 *user.twig*

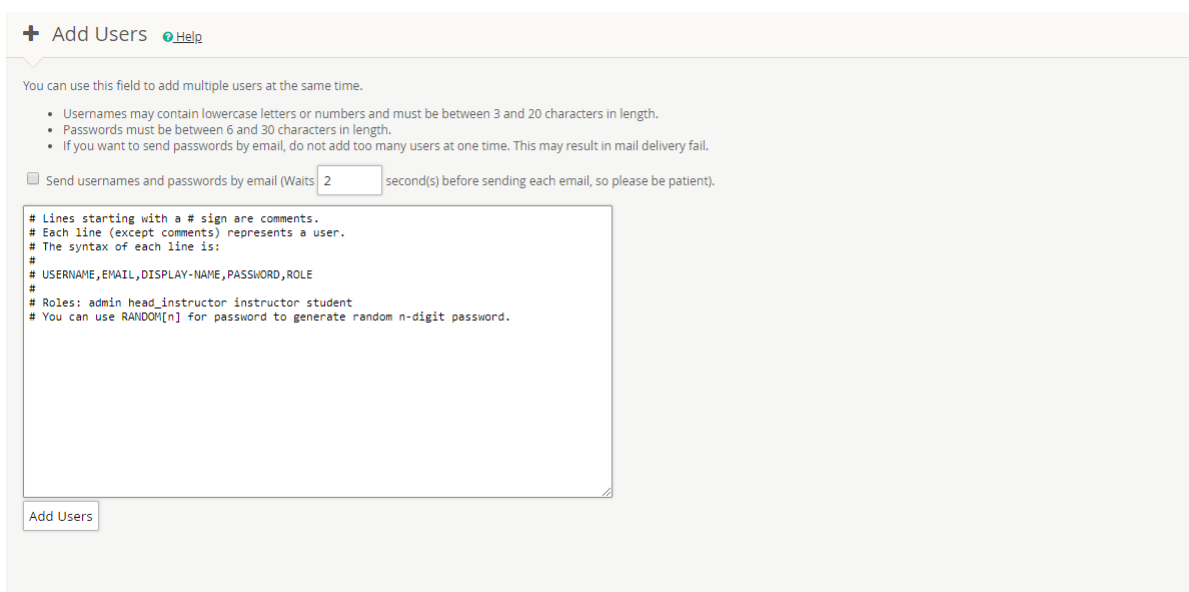


#	User ID	Username	Display Name	Email	Role	First Login	Last Login	Actions
1	1	admin	Admin	admin@admin.com	admin	2018-02-13 04:28:27	2018-03-21 17:51:24	  
2	2	stillmen	Stillmen Vallian	stillmen.v@gmail.com	student	2018-03-11 11:49:57	2018-03-21 17:16:08	  
3	3	dummy	Dummy Account	dummy@dummy.com	student	2018-03-11 11:53:26	2018-03-11 11:54:39	  

Gambar 3.13: Halaman *User*

3 Gambar 3.13 menunjukkan halaman *user*. Halaman *user* berisikan list pengguna yang terdaftar
 4 pada *Sharif Judge*. Pada halaman ini, admin dapat melakukan beberapa aksi seperti menambah
 5 pengguna, menghapus pengguna, melihat hasil jawaban pengguna, menghapus jawaban pengguna
 6 dan mengubah data diri pengguna.

7 3.2.2.11 *add_user.twig*



+ Add Users [Help](#)

You can use this field to add multiple users at the same time.

- 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.
- If you want to send passwords by email, do not add too many users at one time. This may result in mail delivery fail.

☐ Send usernames and passwords by email (Waits second(s) before sending each email, so please be patient).

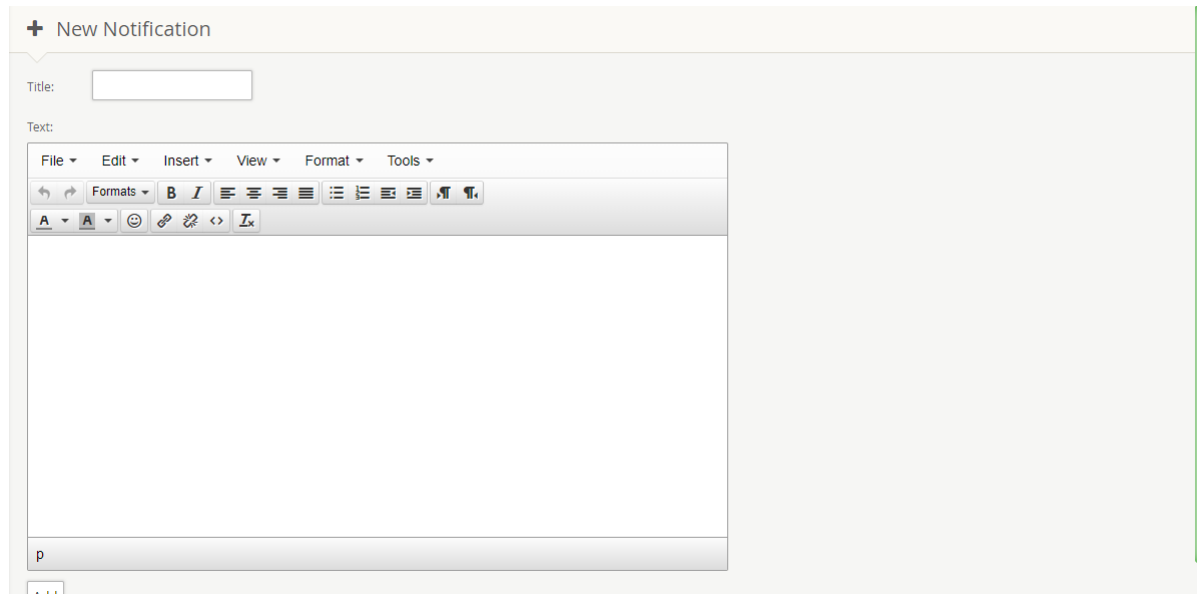
```
# Lines starting with a # sign are comments.
# Each line (except comments) represents a user.
# The syntax of each line is:
#
# USERNAME,EMAIL,DISPLAY-NAME,PASSWORD,ROLE
#
# Roles: admin head_instructor instructor student
# You can use RANDOM[n] for password to generate random n-digit password.
```

Add Users

Gambar 3.14: Halaman *Add User*

- 1 [Gambar 3.14](#) menunjukan halaman *add user*. Halaman *add user* berfungsi untuk menambah peserta
2 pada *Sharif Judge*. Pada halaman ini, admin dapat memasukan informasi pengguna yang ingin
3 didaftarkan pada *Sharif Judge*. Informasi tersebut berupa *username*, *email*, *password* dan *role*.

4 3.2.2.12 *add_notification.twig*



Gambar 3.15: Halaman *Add Notification*

- 5 [Gambar 3.15](#) menunjukan halaman *add notification*. Halaman *add notification* berfungsi untuk
6 menambah pengumuman pada *Sharif Judge*. Pada halaman ini, terdapat beberapa *form* seperti
7 judul pengumuman dan isi dari pengumuman tersebut.

1 3.2.2.13 *add_assignment.twig*

Add Assignment [Help](#)

Assignment Name:

Start Time:

Finish Time:

Extra Time (minutes):
Extra time for late submissions.

Participants:
Enter username of participants here (comma separated). Only these users are able to submit. You can use keyword "ALL".

Tests and Descriptions (zip file): No file chosen
[Use this structure](#)

PDF File: No file chosen
PDF File of Assignment

☐ Open Open or close this assignment

☐ Scoreboard Check this to enable scoreboard

☐ Java Exceptions Check this to show Java exceptions to users

☐ Archived Assignment Check this to make an archived assignment

Coefficient rule (?)
PHP script without <?php ?> tags

```
/*
 * Put coefficient (from 100) in variable $coefficient.
 * You can use variables $extra_time and $delay.
 * $extra_time is the total extra time given to users
 * (in seconds) and $delay is number of seconds passed
 * from finish time (can be negative).
 * In this example, $extra_time is 172800 (2 days):
 */

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;
```

Problems [+](#)

	Name	Score	Time Limit (ms)			Memory Limit (KB)	Allowed Languages (?)	Diff Command (?)	Diff Argument (?)	Upload Only (?)
			C/C++	Python	Java					
1	Problem	100	500	1500	2000	50000	C,C++,Python 2,Python 3,J	diff	-bB	<input type="checkbox"/>

Gambar 3.16: Halaman *Add Assignment*

2 **Gambar 3.16** menunjukkan halaman *add assignment*. Halaman *add assignment* berfungsi untuk
 3 menambah *assignment* pada *Sharif Judge*. Pada halaman ini, *admin* dan *head instructor* dapat
 4 mengatur beberapa pengaturan *assignment* tersebut. Beberapa pengaturan seperti nama, waktu
 5 mulai, waktu akhir, *scoreboard* dan lain-lain. *Admin* juga dapat mengunggah deskripsi serta *file*
 6 PDF untuk *assignment* tersebut.

7 3.2.2.14 *rejudge.twig*

Rejudge

Selected Assignment: Soal Kuis

By clicking on rejudge, all submissions of selected problem will change to **PENDING** state. Then Sharif Judge rejudges them one by one.

If you want to rejudge a single submission, you can click on rejudge button in [All Submissions](#) or [Final Submissions](#) page.

Rejudge Problem 1 (Problem)

Rejudge Problem 2 (Problem)

Rejudge Problem 3 (Problem)

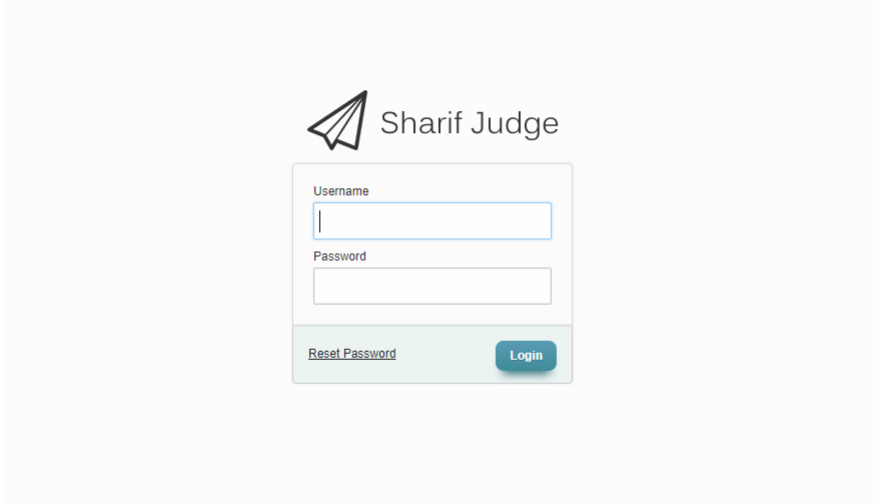
Rejudge Problem 4 (Problem)

Rejudge Problem 5 (Problem)

Gambar 3.17: Halaman *Rejudge*

- 1 [Gambar 3.17](#) menunjukan halaman *rejudge*. Halaman *rejudge* berfungsi untuk menilai ulang hasil
- 2 pekerjaan seluruh peserta *Sharif Judge*. Pada halaman ini, *admin* dan *head instructor* dapat menilai
- 3 ulang setiap masalah pada *assignment* yang dipilih.

4 3.2.2.15 *login.twig*



Gambar 3.18: Halaman *login*

- 5 [Gambar 3.18](#) menunjukan halaman *login*. Halaman *login* merupakan halaman pertama yang tampil
- 6 ketika pengguna membuka *Sharif Judge*. Untuk dapat menggunakan *Sharif Judge*, para pengguna
- 7 harus memasukan kombinasi *username* dan *password* yang tepat. Jika pengguna berhasil *login*,
- 8 maka akan diarahkan ke halaman *dashboard*. Jika pengguna gagal *login*, maka akan muncul pesan
- 9 kesalahan "*Incorrect username or password*".

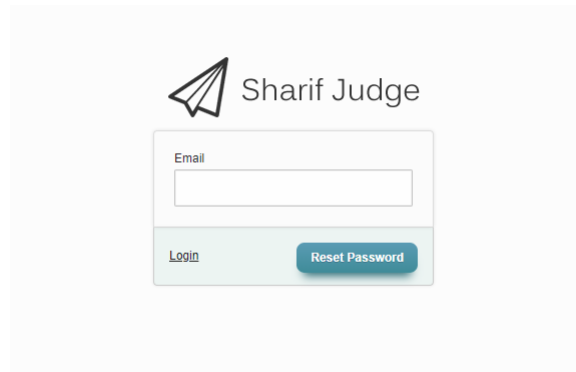
10 3.2.2.16 *register.twig*



Gambar 3.19: Halaman *Register*

1 **Gambar 3.19** menunjukkan halaman *register*. Halaman *register* berfungsi untuk mendaftar sebagai
 2 peserta *Sharif Judge*. Umumnya halaman ini tidak tersedia karena pengguna *Sharif Judge* telah
 3 ditentukan sebelumnya oleh *admin* dan *head instructor*. Halaman ini akan muncul jika *admin*
 4 mengaktifkan fitur *Open Public Registration* pada halaman *Settings*.

5 3.2.2.17 *lost.twig*



Gambar 3.20: Halaman *Lost*

6 **Gambar 3.20** menunjukkan halaman *lost*. Halaman *lost* berfungsi untuk para peserta yang lupa
 7 kombinasi *username* dan *password*. Peserta harus memasukan *email* yang didaftarkan pada *Sharif*
 8 *Judge*. *Sharif Judge* akan mengirimkan *link* untuk mereset *password* ke *email* yang telah dimasukan
 9 sebelumnya.

10 3.2.3 *Controller*

11 Direktori *controller* perangkat lunak *Sharif Judge* terdapat pada *Sharif-Judge\application*
 12 *controllers*. Di dalam *folder controllers*, terdapat beberapa kelas *controller* yang berisikan fungsi-
 13 fungsi sebagai perantara *model*, *view*, dan *resource* lainnya yang dibutuhkan untuk memproses
 14 *HTTP request*.

15 3.2.3.1 *Assignments.php*

16 Pada file *Assignments.php* terdapat beberapa fungsi yaitu:

- 17 • *index*: mempersiapkan data yang dibutuhkan untuk halaman *assignments.twig*. Data yang
 18 dipersiapkan, diambil menggunakan fungsi-fungsi yang terdapat pada *Assignment_model.php*
- 19 • *select*: memilih *assignment* yang ada pada *Sharif Judge*
- 20 • *pdf*: mengunduh file pdf atau deskripsi masalah pada *assignment* tertentu
- 21 • *downloadtestsdesc*: mengompres dan mengunduh test data dan deskripsi *assignment* tertentu
- 22 • *download_submissions*: mengompres dan mengunduh jawaban akhir para peserta pada
 23 *assignment* tertentu
- 24 • *delete*: menghapus sebuah *assignment*

- *add*: mengambil input dari user untuk menambah atau mengubah *assignment*
- *__add*: menambah atau mengubah *assignment*

3.2.3.2 *Dashboard.php*

Pada file *Dashboard.php* terdapat beberapa fungsi yaitu:

- *index*: mempersiapkan data yang dibutuhkan untuk halaman *dashboard.twig*. Data yang dipersiapkan, diambil menggunakan beberapa fungsi yang terdapat pada *Assignment_model.php*, *Settings_model.php* dan *Notification_model.php*
- *widget_positions*: menyimpan posisi *widget* pada *Dashboard* pengguna

3.2.3.3 *Install.php*

Pada file *Install.php* terdapat beberapa fungsi yaitu:

- *index*: fungsi ini membuat table-table yang dibutuhkan oleh *Sharif Judge* ke *database* yang telah ditentukan

3.2.3.4 *Login.php*

Pada file *Login.php* terdapat beberapa fungsi yaitu:

- *__registration_code*: memeriksa apakah kode pendaftaran yang dimasukkan sudah benar atau tidak
- *index*: memvalidasi kombinasi antara *username* dan *password* yang telah dimasukan pengguna
- *register*: mempersiapkan *form* registrasi dan memproses tampilan pada *register.twig*
- *logout*: keluar dari *Sharif Judge* dan mengalihkan pengguna ke halaman *login*
- *lost*: mempersiapkan *form* lupa *password* dan memproses tampilan pada *lost.twig*

3.2.3.5 *Notification.php*

Pada file *Notification.php* terdapat beberapa fungsi yaitu:

- *index*: mempersiapkan data yang dibutuhkan untuk halaman *notification.twig*. Data yang dipersiapkan, diambil menggunakan beberapa fungsi yang terdapat pada *Assignment_model.php* dan *Notification_model.php*
- *add*: menambah pengumuman pada *Sharif Judge*
- *edit*: mengubah pengumuman yang ada pada *Sharif Judge*
- *delete*: menghapus pengumuman yang ada pada *Sharif Judge*

3.2.3.6 *Problems.php*

Pada file *Problems.php* terdapat beberapa fungsi yaitu:

- *index*: menampilkan deskripsi *problem* yang diberikan
- *edit*: mengubah deskripsi *problem* yang ada pada *Sharif Judge*

3.2.3.7 *Profile.php*

Pada file *Profile.php* terdapat beberapa fungsi yaitu:

- *index*: mempersiapkan *form profile* yang berfungsi untuk mengubah informasi dari pengguna *Sharif Judge*
- *__password_check*: mengecek apakah *password* yang dibuat oleh pengguna *Sharif Judge* memenuhi syarat. Syarat *password* tersebut yaitu minimal terdiri dari 6 karakter.
- *__password_again_check*: mengecek apakah '*password again*' yang dimasukan sama dengan *password* yang telah dimasukan sebelumnya
- *__email_check*: mengecek apakah *email* yang dimasukan pengguna telah digunakan pengguna lain
- *__role_check*: memvalidasi *user role*

3.2.3.8 *Queueprocess.php*

Pada file *Queueprocess.php* terdapat beberapa fungsi yaitu:

- *run*: fungsi utama untuk memproses antrian (*queue*) dimana fungsi ini menjudge antrian satu demi satu

3.2.3.9 *Rejudge.php*

Pada file *Rejudge.php* terdapat beberapa fungsi yaitu:

- *index*: mempersiapkan data yang dibutuhkan untuk halaman *rejudge.twig*. Data yang dipersiapkan, diambil menggunakan beberapa fungsi yang terdapat pada *Assignment_model.php*
- *rejudge_single*: menilai ulang jawaban peserta pada satu masalah tertentu

3.2.3.10 *Scoreboard.php*

Pada file *Scoreboard.php* terdapat beberapa fungsi yaitu:

- *index*: mempersiapkan data yang dibutuhkan untuk halaman *scoreboard.twig*. Data yang dipersiapkan, diambil menggunakan beberapa fungsi yang terdapat pada *Assignment_model.php* dan *Scoreboard_model.php*

3.2.3.11 *Server_time.php*

Pada file *Server_time.php* terdapat beberapa fungsi yaitu:

- *index*: menampilkan waktu server yang berfungsi untuk sinkronisasi waktu server

3.2.3.12 *Settings.php*

Pada file *Settings.php* terdapat beberapa fungsi yaitu:

- *index*: mempersiapkan data yang dibutuhkan untuk halaman *settings.twig*. Data yang dipersiapkan, diambil menggunakan beberapa fungsi yang terdapat *settings_model.php*
- *update*: menyimpan pengaturan yang telah diubah pada halaman *settings.twig*

3.2.3.13 *Submission.php*

Pada file *Submission.php* terdapat beberapa fungsi yaitu:

- *__download_excel*: menggunakan library *PHPExcel* untuk menghasilkan file excel dari *submission* pada *assignment* tertentu
- *final_excel*: mengunduh file excel dari *submission* yang telah ditandai sebagai jawaban akhir
- *all_excel*: mengunduh file excel dari seluruh *submission*
- *the_final*: mempersiapkan dan menampilkan data yang dibutuhkan untuk halaman *submission.twig* bagian *Final Submissions*
- *all*: mempersiapkan dan menampilkan data yang dibutuhkan untuk halaman *submission.twig* bagian *All Submissions*
- *select*: memilih *submission* tertentu untuk dijadikan jawaban akhir untuk masalah tertentu
- *download_file*: mengunduh file jawaban yang telah dikumpulkan sebelumnya

3.2.3.14 *Submit.php*

Pada file *Submit.php* terdapat beberapa fungsi yaitu:

- *__language_to_type*: mengubah bahasa pemrograman menjadi ekstensi bahasa pemrograman tersebut. Contoh: "c++" akan diubah menjadi ".cpp"
- *__match*: memastikan ekstensi file jawaban yang dikumpul sesuai dengan bahasa pemrograman yang dipilih
- *__check_language*: memastikan bahasa pemrograman yang digunakan dapat ditangani oleh *Sharif Judge*
- *index*: menyiapkan data-data jawaban dan mengumpulkannya ke *Sharif Judge*
- *__upload*: menyimpan kode jawaban yang dikumpulkan dan menambahkannya ke antrian untuk dinilai

1 **3.2.3.15 *User.php***

2 Pada *file User.php* terdapat beberapa fungsi yaitu:

- 3 • *index*: menyiapkan data yang dibutuhkan untuk *halaman users.twig*. Data yang dipersiapkan, diambil menggunakan beberapa fungsi yang terdapat *assignment_model.php* dan
4 *user_model.php*
5
- 6 • *add: controller* untuk menambahkan pengguna baru
- 7 • *delete: controller*: untuk menghapus pengguna yang ada
- 8 • *delete_submissions: controller* untuk menghapus *submission* pengguna tertentu
- 9 • *list_excel*: mengunduh *file excel* dari *list user*

BAB 4

PERANCANGAN

Bab ini membahas tentang perancangan setiap fitur yang diimplementasi pada perangkat lunak *Sharif Judge*.

4.1 Mengganti Method *shell_exec("rm ...")* Menjadi *unlink()*

Method *shell_exec("rm ...")* yang memiliki fungsi untuk menghapus sebuah *file* terdapat pada kelas *controller Assignment.php*. Method *shell_exec("rm ...")* diubah menggunakan method *unlink()*. Listing 4.1 menunjukan perubahan kode program menggunakan *syntax diff* ¹.

Listing 4.1: Perubahan kode program pada *Assignment.php*

```
@@ -433,8 +433,7 @@ class Assignments extends CI_Controller
// Upload Tests (zip file)
-   shell_exec('rm -f ' . $assignments_root . '/*.zip');
+   unlink($assignments_root . '/*.zip');
$config = array(
    'upload_path' => $assignments_root,
    'allowed_types' => 'zip',

@@ -482,7 +481,7 @@ class Assignments extends CI_Controller
else
{
    foreach($old_pdf_files as $old_name)
    -   shell_exec("rm -f $old_name");
    +   unlink($old_name);
    $this->messages[] = array(
        'type' => 'success',
        'text' => 'PDF file uploaded successfully.'
```

¹GNU Operating System, "Comparing and Merging Files" terakhir diubah 6 Mei 2017. <https://www.gnu.org/software/diffutils/manual/diffutils.html#Detailed-Unified>

4.2 Menambahkan Method Rekoneksi ke *Database*

Method koneksi ke *database* ditambahkan pada kelas *controller Queueprocess.php*. Method yang digunakan yaitu *\$this->db->reconnect()*. Hal tersebut dilakukan untuk menghindari *connection times out* akibat pengujian yang memakan waktu lama. Listing 4.2 menunjukkan perubahan kode program yang dilakukan di *Queueprocess.php*

Listing 4.2: Perubahan kode program pada *Queueprocess.php*

```

@@ -131,6 +131,9 @@ class Queueprocess extends CI_Controller
$submission['status'] = $output;
}
+ //reconnect to database incase we have run test for a long time.
+ $this->db->reconnect();
+
// Save the result
$this->queue_model->save_judge_result_in_db($submission, $type);

```

4.3 Membatasi Pengaksesan Soal (deskripsi & PDF)

Fungsi untuk mengunduh soal (deskripsi & PDF) terdapat pada *controller Assignment.php*. Selain membatasi soal (deskripsi & PDF) hanya dapat diunduh saat *assignment "open"* dan setelah waktu mulai, pada fungsi ini juga ditambahkan fitur lain. Fitur lain tersebut yaitu membatasi soal hanya dapat diunduh oleh peserta yang terdaftar sebagai "*participant*" dan soal tidak dapat diunduh setelah melewati batas waktu pengumpulan. Rancangan algoritma kode yang digunakan yaitu

- Membuat atribut tambahan untuk menyimpan informasi waktu selesai, waktu mulai dan waktu tambahan sebuah assignment.
- Jika atribut "*open*" pada *assignment* tidak memiliki nilai, maka munculkan pesan *error "Selected assignment has been closed."*
- Jika pengguna tidak terdaftar sebagai "*participant*" dalam *assignment* yang dipilih, maka munculkan pesan *error "You are not registered for submitting."*
- Jika waktu sekarang telah melewati batas waktu selesai + waktu tambahan, maka munculkan pesan *error "Selected assignment has finished."*
- Jika waktu sekarang belum melewati waktu mulai, maka munculkan pesan *error "Selected assignment has not started."*

Listing 4.3 menunjukkan hasil pengimplementasian rancangan algoritma di atas ke dalam kode program *Assignments.php*

Listing 4.3: Perubahan kode program pada *Assignments.php*

```

@@ -99,6 +99,10 @@ class Assignments extends CI_Controller

```

```

1  */
2  public function pdf($assignment_id, $problem_id = NULL)
3  {
4      +   $finishtime = strtotime($this->assignment_model->assignment_info(
5          $assignment_id)['finish_time']);
6      +   $starttime = strtotime($this->assignment_model->assignment_info(
7          $assignment_id)['start_time']);
8      +   $extratime = $this->assignment_model->assignment_info($assignment_id)['
9          extra_time'];
10
11     // Find pdf file
12     if ($problem_id === NULL)
13     $pattern = rtrim($this->settings_model->get_setting('assignments_root'),'/' )."/
14         assignment_{$assignment_id}/*.pdf";
15
16     @@ -107,6 +111,14 @@ class Assignments extends CI_Controller
17     $pdf_files = glob($pattern);
18     if ( ! $pdf_files )
19     show_error("File not found");
20     +   elseif ( !$this->assignment_model->assignment_info($assignment_id)['open'] &&
21         $this->user->level == 0 )
22     +       show_error('Selected assignment has been closed. ');
23     +   elseif ( ! $this->assignment_model->is_participant($this->assignment_model->
24         assignment_info($assignment_id)['participants'],$this->user->username) )
25     +       show_error('You are not registered for submitting. ');
26     +   elseif ( shj_now() > $finishtime + $extratime && $this->user->level == 0 )
27     +       show_error('Selected assignment has finished. ');
28     +   elseif ( shj_now() < $starttime && $this->user->level == 0 )
29     +       show_error('Selected assignment has not started. ');
30
31     // Download the file to browser
32     $this->load->helper('download')->helper('file');
33

```

34 4.4 Mensupport *File* dengan Ekstensi TXT

35 Untuk dapat mensupport *file* dengan ekstensi TXT pada perangkat lunak *Sharif Judge*, diperlukan
36 perubahan kode pada beberapa kelas. Beberapa kelas tersebut antara lain *controller Submit.php*,
37 *model Assignment_model.php*, *view submissions.twig* dan kelas bantuan *shj_helper.php* yang terdapat
38 pada direktori *Sharif-Judge\application\helper*.

39 Perubahan di *Submit.php* untuk menerima *file* yang dikumpul menggunakan ekstensi TXT. Listing
40 4.4 menunjukkan hasil perubahan kode program yang dilakukan di *Submit.php*

Listing 4.4: Perubahan kode program pada *Submit.php*

```

1  @@ -58,6 +58,7 @@ class Submit extends CI_Controller
2
3      case 'java': return 'java';
4      case 'zip': return 'zip';
5      case 'pdf': return 'pdf';
6  +      case 'txt': return 'txt';
7      default: return FALSE;
8  }
9  }
10
11 @@ -76,6 +77,7 @@ class Submit extends CI_Controller
12
13      case 'java': return ($extension==='java'?TRUE:FALSE);
14      case 'zip': return ($extension==='zip'?TRUE:FALSE);
15      case 'pdf': return ($extension==='pdf'?TRUE:FALSE);
16  +      case 'txt': return ($extension==='txt'?TRUE:FALSE);
17  }
18
19 @@ -87,7 +89,7 @@ class Submit extends CI_Controller
20 {
21     if ($str=='0')
22         return FALSE;
23 -     if (in_array( strtolower($str),array('c', 'c++', 'python 2', 'python 3', '
24 java', 'zip', 'pdf')))
25 +     if (in_array( strtolower($str),array('c', 'c++', 'python 2', 'python 3', '
26 java', 'zip', 'pdf', 'txt')))
27         return TRUE;
28 return FALSE;
29 }
30

```

31
32 Perubahan kode di *Assignment_model.php* untuk membuat assignement dapat dikumpulkan dengan
33 menggunakan file TXT. Listing 4.5 menunjukan hasil perubahan kode program yang dilakukan di
34 *Assignment_model.php*

Listing 4.5: Perubahan kode program pada *Assignments_model.php*

```

35
36 @@ -98,7 +98,7 @@ class Assignment_model extends CI_Model
37
38     elseif ($item2 === 'pdf')
39         $item = 'PDF';
40     $item2 = strtolower($item);
41 -     if ( ! in_array($item2, array('c','c++','python 2','python 3','java','zip
42 ', 'pdf')))
43 +     if ( ! in_array($item2, array('c','c++','python 2','python 3','java','zip

```

```

1      ', 'pdf', 'txt'))))
2          continue;
3          // If the problem is not Upload-Only, its language should be one of {C,C
4      ++,Python 2, Python 3,Java}
5          if ( ! in_array($i, $uo) && ! in_array($item2, array('c','c++','python
6      2','python 3','java')) )
7

```

Perubahan kode di *shj_helper.php* untuk mengubah jenis kode ke ekstensi dan bahasa. Listing 4.6 menunjukkan hasil perubahan kode program yang dilakukan di *shj_helper.php*

Listing 4.6: Perubahan kode program pada *shj_helper.php*

```

11 @@ -81,6 +81,7 @@ if ( ! function_exists('filetype_to_extension'))
12
13     case 'java': return 'java';
14     case 'zip': return 'zip';
15     case 'pdf': return 'pdf';
16 +   case 'txt': return 'txt';
17     default: return FALSE;
18 }
19 }
20
21 @@ -104,6 +105,7 @@ if ( ! function_exists('filetype_to_language'))
22
23     case 'java': return 'Java';
24     case 'zip': return 'Zip';
25     case 'pdf': return 'PDF';
26 +   case 'txt': return 'TXT';
27     default: return FALSE;
28 }
29 }

```

Perubahan kode di halaman *submissions.twig* untuk mengunduh file dengan ekstensi TXT. Listing 4.7 menunjukkan hasil perubahan kode program yang dilakukan di halaman *submissions.twig*

Listing 4.7: Perubahan kode program pada halaman *submission.twig*

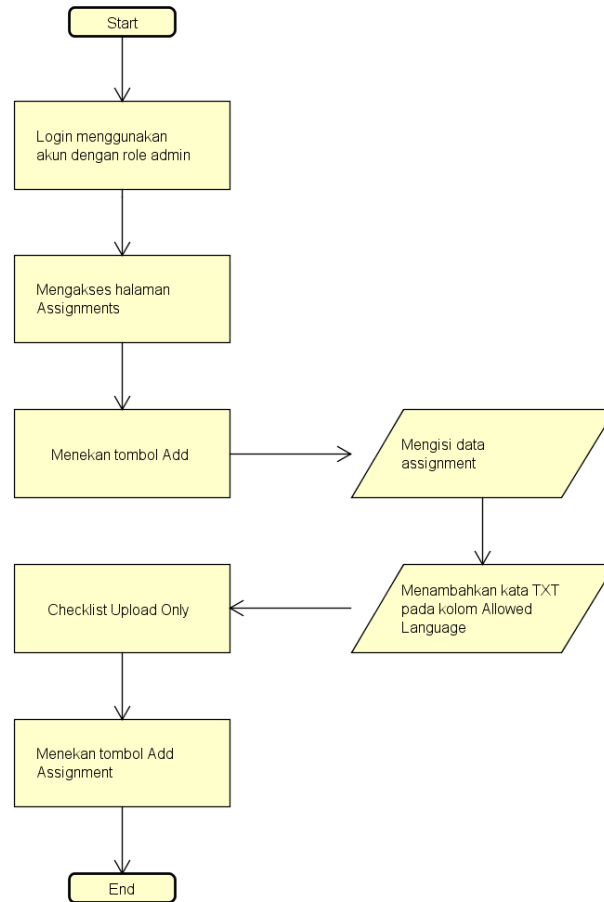
```

33 @@ -158,7 +158,7 @@
34     {% endif %}
35 </td>
36 <td>
37 -     {% if submission.file_type == 'zip' or submission.file_type == 'pdf' %}
38 +     {% if submission.file_type == 'zip' or submission.file_type == 'pdf' or
39     submission.file_type == 'txt' %}
40         <div class="btn shj-orange" data-type="download">Download</div>
41     {% else %}
42

```

1
2

```
<div class="btn shj-orange" data-type="code" >Code</div>
```



Gambar 4.1: Diagram *Flowchart* Menambah *Assignment* TXT

3 Gambar 4.1 merupakan diagram *flowchart* yang menunjukkan langkah-langkah untuk membuat
 4 *assignment* agar dapat menerima *file* dengan ekstensi TXT.

5 4.5 Menambahkan Halaman *Logs*

6 Agar halaman *Logs* dapat berjalan dengan baik, perlu ditambahkan tabel baru pada *database Sharif*
 7 *Judge*. Tabel baru tersebut bernama *shj_logins*.

Tabel 4.1: Perancangan Tabel *shj_logins*

Atribut	Tipe Data	Ukuran	Default
<i>login_id</i> (<i>primary key</i>)	int	11	None
<i>username</i>	varchar	20	None
<i>ip_address</i>	varchar	15	None
<i>timestamp</i>	timestamp	11	current_timestamp
<i>last_24h_login_id</i>	int	11	null

8 Tabel 4.1 menunjukkan perancangan tabel *shj_logins*. Tabel tersebut memiliki beberapa atribut,
 9 yaitu:

1. *login_id*: sebagai penanda yang membedakan setiap *login* peserta satu dengan yang lain. Memiliki *length default* int dari *phpMyAdmin* yaitu 11. Atribut *login_id* merupakan *primary key* karena id harus unik agar setiap *login* peserta dapat dibedakan. Atribut ini juga bersifat *auto increment*.
2. *username*: *username* peserta yang berhasil *login* pada *Sharif Judge*. Memiliki *length varchar* 20 karena *length username* pada tabel *shj_users* adalah 20.
3. *ip_address*: *ip address* peserta yang berhasil *login* pada *Sharif Judge*. Memiliki *length varchar* 15 karena *length* maksimal dari *ip address protocol version 4 (IPv4)* adalah 15. Contoh: 202.100.123.255
4. *timestamp*: waktu peserta saat berhasil *login* pada *Sharif Judge*. Menggunakan tipe data *timestamp* yang mencatat waktu *login* dengan format YYYY-MM-DD HH:MM:SS. Contoh: 2018-04-06 18:15:43
5. *last_24h_login_id*: id *login* peserta yang berhasil *login* pada *Sharif Judge* namun menggunakan *ip address* berbeda dalam waktu 24 jam terakhir.

Selain tabel diatas, halaman logs juga ditambahkan kelas *model*, *view* dan *controller*.

1. Model

Model untuk halaman *logs* bernama *Logs_model.php*. Pada *model Logs_model.php* terdapat beberapa fungsi. Tabel 4.2 menunjukkan perincian fungsi *insert_to_logs* yang terdapat pada *model Logs_model.php*

Tabel 4.2: Perincian fungsi *insert_to_logs*

Nama Method	<i>insert_to_logs</i>
Parameter Input	<i>\$username</i> dan <i>\$ip_address</i>
Parameter Output	-
Tabel yang berhubungan	<i>shj_logins</i>
Deskripsi	Proses untuk memasukan <i>logs</i> pengguna <i>Sharif Judge</i>
Algoritma	<ul style="list-style-type: none"> • Mengecek dan menghapus <i>logs</i> pada tabel <i>shj_logins</i> yang <i>timestampnya</i> lebih dari 24 jam. • Mengecek entri <i>login</i> terakhir untuk <i>\$username</i> yang menggunakan <i>IP address</i> tidak sama dengan <i>\$ip_address</i> • Jika tidak memiliki hasil, maka tambahkan entri baru menggunakan <i>\$username</i> dan <i>\$ip_address</i> tersebut. • Jika memiliki hasil, maka tambahkan entri baru menggunakan <i>\$username</i> dan <i>\$ip_address</i> serta. <i>last_24h_login_id</i> diisi dengan <i>login_id</i> sebelumnya

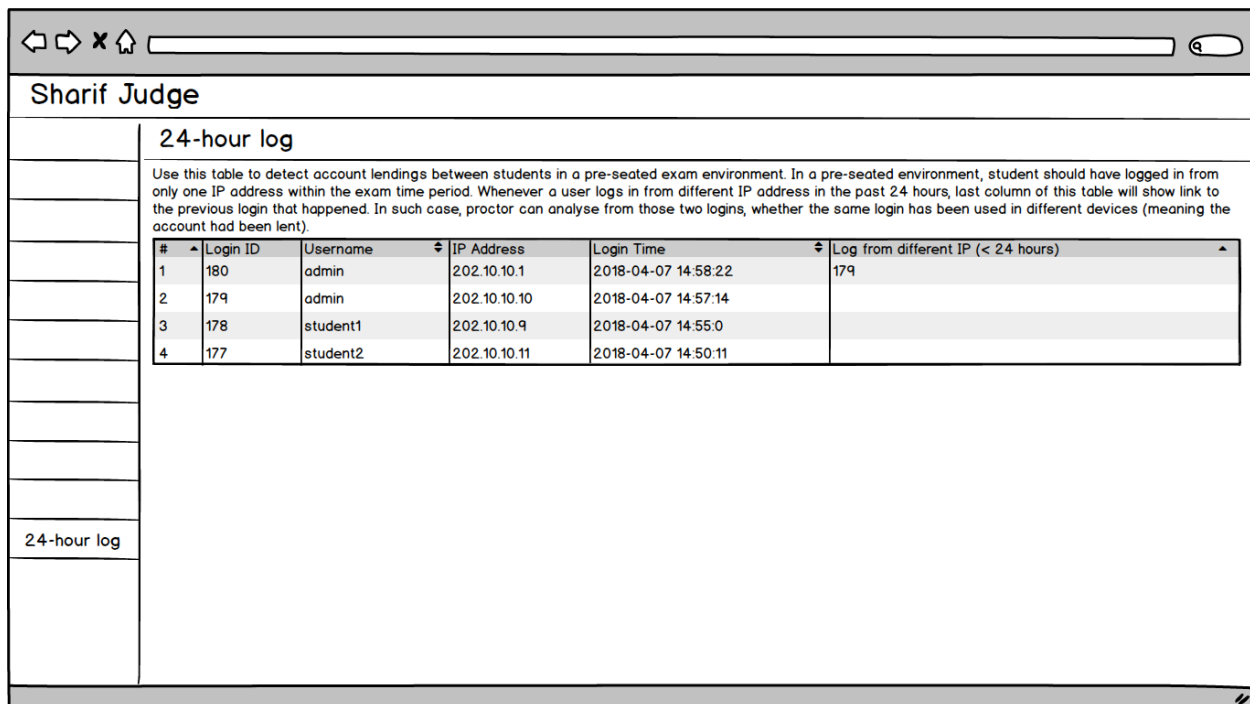
Tabel 4.3 menunjukkan perincian fungsi *get_all_logs* yang terdapat pada *model Logs_model.php*

Tabel 4.3: Perincian fungsi *get_all_logs*

Nama <i>Method</i>	<i>get_all_logs</i>
Parameter <i>Input</i>	-
Parameter <i>Output</i>	semua entri logs dari tabel <i>shj_logins</i>
Tabel yang berhubungan	<i>shj_logins</i>
Deskripsi	Proses untuk mengembalikan entri <i>logs</i> yang terdapat pada tabel <i>shj_logins</i>
Algoritma	<ul style="list-style-type: none"> Mengembalikan seluruh entri logs yang terdapat pada tabel <i>shj_logins</i> dalam bentuk <i>array</i>.

2. View

View untuk halaman *logs* bernama *logs.twig*. Menu halaman *logs* terletak di paling bawah menu lainnya dan bernama '24-hour log'. Gambar 4.2 menunjukkan rancangan tampilan halaman *logs*.



#	Login ID	Username	IP Address	Login Time	Log from different IP (< 24 hours)
1	180	admin	202.10.10.1	2018-04-07 14:58:22	179
2	179	admin	202.10.10.10	2018-04-07 14:57:14	
3	178	student1	202.10.10.9	2018-04-07 14:55:0	
4	177	student2	202.10.10.11	2018-04-07 14:50:11	

Gambar 4.2: Rancangan tampilan halaman *logs*

3. Controller

Controller untuk halaman *logs* bernama *Logs.php*. Pada *controller Logs.php* terdapat beberapa fungsi. Tabel 4.4 menunjukkan perincian fungsi *consturct___* yang terdapat pada *controller Logs.php*

Tabel 4.4: Perincian fungsi *consturct__*

Nama <i>Method</i>	<i>consturct__</i>
Parameter <i>Input</i>	-
Parameter <i>Output</i>	-
Tabel yang berhubungan	-
Deskripsi	membatasi pengguna yang dapat mengakses halaman <i>logs</i>
Algoritma	<ul style="list-style-type: none"> • Mengecek <i>session</i> pengguna yang akan mengakses halaman <i>logs</i>. • Jika <i>session</i> tidak berstatus '<i>logged_in</i>', maka pengguna akan dialihkan ke halaman <i>login</i>. • Mengecek <i>role</i> pengguna yang akan mengakses halaman <i>logs</i>. • Jika <i>role</i> pengguna bukan <i>admin</i>, maka pengguna akan dialihkan ke halaman '<i>404 Not Found</i>'.

1 Tabel 4.5 menunjukkan perincian fungsi *index* yang terdapat pada *controller Logs.php*

Tabel 4.5: Perincian fungsi *index*

Nama <i>Method</i>	<i>index</i>
Parameter <i>Input</i>	-
Parameter <i>Output</i>	-
Tabel yang berhubungan	<i>shj_logins</i>
Deskripsi	Proses untuk memuat seluruh entri <i>logs</i> pada halaman <i>logs.twig</i>
Algoritma	<ul style="list-style-type: none"> • Memuat data <i>logs</i> menggunakan fungsi <i>get_all_logs</i> dari <i>model Logs_model.php</i>. • Memproses data untuk tampilan <i>logs.twig</i>.

2 Selain menambahkan kelas *model*, *view* dan *controller*, terdapat perubahan kode pada kelas
3 *controller Login.php* dan *config Install.php*. Perubahan kode di *controller Login.php* untuk menyimpan
4 data login pengguna ke dalam database. Listing 4.8 menunjukan hasil perubahan kode program
5 yang dilakukan di *controller Login.php*

Listing 4.8: Perubahan kode program pada *Login.php*

```

6
7 @@ -53,6 +53,8 @@ class Login extends CI_Controller
8 $username = $this->input->post('username');
9 $password = $this->input->post('password');
10 if($this->user_model->validate_user($username, $password)){
11 +   $ip_adress = $this->input->ip_address();
12 +
13 // setting the session and redirecting to dashboard:
14 $login_data = array(
15     'username' => $username,
```

```

1
2 @@ -60,6 +62,7 @@ class Login extends CI_Controller
3 );
4     $this->session->set_userdata($login_data);
5     $this->user_model->update_login_time($username);
6 +     $this->logs_model->insert_to_logs($username,$ip_address);
7     redirect('/');
8 }
9 else
10

```

11

12 Perubahan kode di *config Install.php* untuk membuat tabel *shj_logins* pada *database* saat pengin-

13 stalan *SharIF Judge*. Listing 4.9 menunjukkan hasil perubahan kode program yang dilakukan di

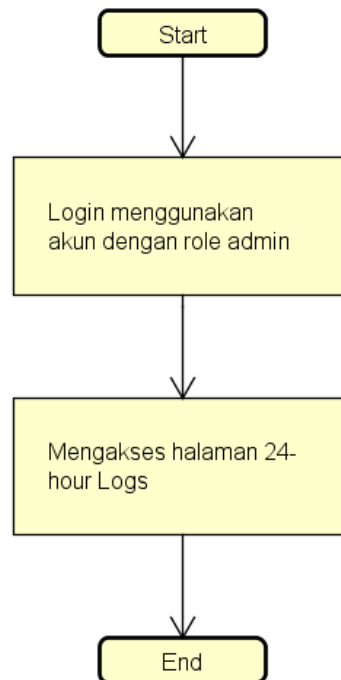
14 *config Install.php*

Listing 4.9: Perubahan kode program pada *Install.php*

```

15
16 @@ -249,7 +249,18 @@ class Install extends CI_Controller
17 if ( ! $this->dbforge->create_table('users', TRUE))
18 show_error("Error creating database table ".$this->db->dbprefix('users'));
19
20 -
21 + // create table 'logins'
22 + $fields = array(
23 +     'login_id'           => array('type' => 'INT', 'constraint' => 11, '
24     unsigned' => TRUE, 'auto_increment' => TRUE),
25 +     'username'           => array('type' => 'VARCHAR', 'constraint'
26     => 20),
27 +     'ip_address'         => array('type' => 'VARCHAR', 'constraint' => 15),
28 +     'timestamp'          => array('type' => 'TIMESTAMP'),
29 +     'last_24h_login_id'  => array('type' => 'INT', 'constraint' => 11, 'null'
30     => TRUE),
31 + );
32 + $this->dbforge->add_field($fields);
33 + $this->dbforge->add_key('login_id', TRUE); // PRIMARY KEY
34 + if ( ! $this->dbforge->create_table('logins', TRUE))
35 +     show_error("Error creating database table ".$this->db->dbprefix('logins'))
36     ;
37
38 // add admin user
39 $this->user_model->add_user(
40

```

Gambar 4.3: Diagram *Flowchart* Mengakses Halaman Logs

1 Gambar 4.3 merupakan diagram *flowchart* yang menunjukkan langkah-langkah untuk mengakses
2 halaman *Logs*.

3 4.6 Menambahkan Parameter "*Display Name*" pada Pendaftaran 4 Peserta *Sharif Judge*

5 Untuk dapat menambahkan parameter "*Display Name*" pada pendaftaran peserta *Sharif Judge*,
6 diperlukan beberapa perubahan kode program. Berikut rancangan algoritma yang dilakukan

- 7 1. Menambahkan parameter "*Display Name*" pada fungsi *add_user* yang terdapat di *model*
8 *User_model.php*.
- 9 2. Mengubah pemisah (*separator*) antar parameter pada fungsi *add_user*. Pemisah antar
10 parameter yang awalnya menggunakan spasi diubah menggunakan tanda koma.
- 11 3. Menambahkan keterangan parameter "*Display Name*" pada halaman *add_user.twig* dan
12 *add_user_result.twig*.
- 13 4. Menambahkan *Display Name* untuk *admin* pada proses *install Sharif Judge*.
- 14 5. Menambahkan *text field Display Name* pada halaman *Register (Open Public Registration)*.

15 Dari rancangan algoritma yang diterapkan, terdapat perubahan kode pada beberapa kelas. Beberapa
16 kelas tersebut antara lain *controller Install.php*, *controller Login.php*, *model User_model.php*, *view*
17 *add_user.twig*, *view add_user_result.twig* dan *view register.twig*.

18 Perubahan kode di *Install.php* untuk menambahkan *Display Name admin*. Listing 4.10 menunjukkan
19 hasil perubahan kode program yang dilakukan di *Install.php*

Listing 4.10: Perubahan kode program pada *Install.php*

```

1  @@ -255,6 +255,7 @@ class Install extends CI_Controller
2
3      $this->user_model->add_user(
4          $this->input->post('username'),
5          $this->input->post('email'),
6  +      'Admin',
7          $this->input->post('password'),
8          'admin'
9  );
10

```

11
12 Perubahan kode di halaman *controller Login.php* untuk menerima *input* dari *text field Display Name*
13 halaman *register.twig*. Listing 4.11 menunjukkan hasil perubahan kode program yang dilakukan di
14 *Login.php*

Listing 4.11: Perubahan kode program pada *Login.php*

```

15
16  @@ -102,6 +102,7 @@ class Login extends CI_Controller
17
18      $this->user_model->add_user(
19          $this->input->post('username'),
20          $this->input->post('email'),
21  +      $this->input->post('displayname'),
22          $this->input->post('password'),
23          'student'
24  );
25

```

26 Perubahan kode di *User_model.php* untuk menambahkan parameter "*Display Name*" dan mengubah
27 pemisah antar parameter menggunakan tanda koma. Listing 4.12 menunjukkan hasil perubahan
28 kode program yang dilakukan di *User_model.php*

Listing 4.12: Perubahan kode program pada *User_model.php*

```

29
30  @@ -118,7 +118,7 @@ class User_model extends CI_Model
31
32  * @param $role
33  * @return bool|string
34  */
35  - public function add_user($username, $email, $password, $role)
36  + public function add_user($username, $email, $display_name, $password, $role)
37  {
38      if ( ! $this->form_validation->alpha_numeric($username) )
39      return 'Username may only contain alpha-numeric characters.';
40
41  @@ -137,6 +137,7 @@ class User_model extends CI_Model
42
43      $user=array(
44          'username' => $username,

```

```

1      'email' => $email,
2 +      'display_name' => $display_name,
3      'password' => $this->password_hash->HashPassword($password),
4      'role' => $role
5  );
6
7 @@ -173,26 +174,26 @@ class User_model extends CI_Model
8 if (strlen($line) == 0 OR $line[0] == '#')
9 continue; //ignore comments and empty lines
10
11 -      $parts = preg_split('/\s+/', $line);
12 -      if (count($parts) != 4)
13 -          continue; //ignore lines that not contain 4 parts
14 +      $parts = preg_split('/,+/', $line);
15 +      if (count($parts) != 5)
16 +          continue; //ignore lines that not contain 5 parts
17
18 -      if (strtolower(substr($parts[2], 0, 6)) == 'random')
19 +      if (strtolower(substr($parts[3], 0, 6)) == 'random')
20 {
21 // generate random password
22 -      $len = trim(substr($parts[2], 6), '[]');
23 +      $len = trim(substr($parts[3], 6), '[]');
24          if (is_numeric($len)){
25              $this->load->helper('string');
26 -              $parts[2] = shj_random_password($len);
27 +              $parts[3] = shj_random_password($len);
28          }
29 }
30
31 -      $result = $this->add_user($parts[0], $parts[1], $parts[2], $parts[3]);
32 +      $result = $this->add_user($parts[0], $parts[1], $parts[2], $parts[3], $parts
33 [4]);
34
35          if ($result === TRUE)
36 -              array_push($users_ok, array($parts[0], $parts[1], $parts[2], $parts[3]));
37 +              array_push($users_ok, array($parts[0], $parts[1], $parts[2], $parts[3],
38 $parts[4]));
39          else
40 -              array_push($users_error, array($parts[0], $parts[1], $parts[2], $parts[3],
41 $result));
42 +              array_push($users_error, array($parts[0], $parts[1], $parts[2], $parts[3],

```

```

1      $parts[4], $result));
2
3  } // end of loop
4

```

5

6 Perubahan kode di halaman *add_user.twig* dan *add_user_result.twig* untuk menambahkan ketere-
7 rangian parameter "*Display Name*". Listing 4.13 menunjukkan hasil perubahan kode program yang
8 dilakukan di halaman *add_user.twig*. Listing 4.14 menunjukkan hasil perubahan kode program yang
9 dilakukan di halaman *add_user_result.twig*

Listing 4.13: Perubahan kode program pada halaman *add_user.twig*

```

10
11 @@ -64,7 +64,7 @@
12 # Each line (except comments) represents a user.
13 # The syntax of each line is:
14 #
15 - # USERNAME EMAIL PASSWORD ROLE
16 + # USERNAME,EMAIL,DISPLAY-NAME,PASSWORD,ROLE
17 #
18 # Roles: admin head_instructor instructor student
19 # You can use RANDOM[n] for password to generate random n-digit password.
20

```

21

22

Listing 4.14: Perubahan kode program pada halaman *add_user_result.twig*

```

23
24 @@ -7,7 +7,7 @@
25 <p class="shj_ok">These users added successfully:</p>
26 <ol>
27 {% for item in ok %}
28 - <li>Username: {{ item[0] }} Email: {{ item[1] }} Password: <code>{{ item[2]
29   }}</code> Role: {{ item[3] }}</li>
30 + <li>Username: {{ item[0] }} Email: {{ item[1] }} Display Name: {{ item[2] }}
31   Password: <code>{{ item[3] }}</code> Role: {{ item[4] }} </li>
32 {% endfor %}
33 </ol>
34 {% endif %}
35 @@ -15,7 +15,7 @@
36 <p class="shj_error">Error adding these users:</p>
37 <ol>
38 {% for item in error %}
39 - <li>Username: {{ item[0] }} Email: {{ item[1] }} Password: <code>{{ item[2]
40   }}</code> Role: {{ item[3] }} ({{ item[4] }})</li>
41 + <li>Username: {{ item[0] }} Email: {{ item[1] }} Display Name: {{ item[2] }}
42   Password: <code>{{ item[3] }}</code> Role: {{ item[4] }} ({{ item[5] }})</li>

```



```
1 {% endfor %}
2
3 </ol>
```

Perubahan kode di halaman *register.twig* untuk menambahkan *text field Display Name*. Listing 4.15 menunjukkan hasil perubahan kode program yang dilakukan di halaman *register.twig*

Listing 4.15: Perubahan kode program pada halaman *register.twig*

```
7
8 @@ -32,6 +32,11 @@
9 <input id="form_email" type="email" autocomplete="off" name="email" required="
10     required" class="sharif_input" value="{ { set_value('email') } }"/>
11 { { form_error('email', '<div class="shj_error">', '</div>') } }
12 </p>
13 +     <p>
14 +         label for="form_displayname">Display Name</label><br/>
15 +         <input id="form_displayname" type="text" name="displayname" required="
16             required" pattern="[A-Za-z\s]+" title="The Display Name field must be contain
17             only alphabetical letters" class="sharif_input" value="{ { set_value('
18             displayname') } }"/>
19 +         { { form_error('form_displayname', '<div class="shj_error">', '</div>')
20             } }
21 +     </p>
22 <p>
23 <label for="form_password">Password</label><br/>
24 <input id="form_password" type="password" name="password" required="required"
25     pattern=".{6,200}" title="The Password field must be at least 6 characters in
26     length" class="sharif_input"/>
27
```

4.7 Menambahkan Fitur "*Lock Student's Display Name*"

Fitur "*Lock Student's Display Name*" membutuhkan sebuah "*key*" pada *database*. "*Key*" tersebut berfungsi untuk menyimpan sebuah nilai. Nilai yang disimpan akan menentukan apakah para peserta dapat mengubah *Display Name* atau tidak. "*Key*" disimpan pada tabel *shj_settings* pada kolom *shj_key* dengan nama *lock_student_display_name*. *lock_student_display_name* memiliki nilai *default shj_value* = 0. Jika nilai dari *lock_student_display_name* = 1, maka para peserta tidak dapat mengubah *Display Name*, sebaliknya jika bernilai 0, maka para peserta dapat mengubah *Display Name*.

Rancangan algoritma yang digunakan untuk menambahkan fitur "*Lock Student's Display Name*" yaitu

1. Menambahkan *shj_key* dengan nama *lock_student_display_name* yang memiliki nilai *shj_value* = 0 pada tabel *shj_settings*.

2. Menambahkan *check box* pada halaman *settings.twig* untuk mengaktifkan atau menonaktifkan fitur "*Lock Student's Display Name*".
3. Jika fitur "*Lock Student's Display Name*" diaktifkan, maka *text field Display Name* pada halaman *profile.twig* akan dinonaktifkan (*disabled*).
4. Jika fitur "*Lock Student's Display Name*" dinonaktifkan, maka *text field Display Name* pada halaman *profile.twig* akan kembali aktif.
5. Menambahkan fungsi untuk mengecek kembali nilai dari *lock_student_display_name* pada saat peserta menyimpan perubahan yang dilakukan di halaman *profile.twig*. Hal tersebut dilakukan untuk menangani para peserta yang "memaksa" agar dapat mengubah *Display Name* dengan cara *inspect element* lalu menghapus kode "*disabled*" pada *text field Display Name*.

Dari rancangan algoritma yang diterapkan, terdapat perubahan kode pada beberapa kelas. Beberapa kelas tersebut antara lain *controller Profile.php*, *controller Settings.php*, *model User_model.php*, *view settings.twig*, *view profile.twig* dan *config Install.php*.

Perubahan kode di *Profile.php* untuk mengirimkan nilai '*lock_student_display_name*' ke halaman *profile.twig*. Listing 4.16 menunjukkan hasil perubahan kode program yang dilakukan di *Profile.php*

Listing 4.16: Perubahan kode program pada *Profile.php*

```

@@ -64,6 +64,7 @@ class Profile extends CI_Controller
    'display_name' => $user->display_name,
    'role' => $user->role,
    'form_status' => $this->form_status,
+   'lock_student_display_name' => $this->settings_model->get_setting(
    lock_student_display_name),
    );

$this->twig->display('pages/profile.twig', $data);

```

Perubahan kode di *Settings.php* untuk menyimpan nilai dari *check box* fitur "*Lock Student's Display Name*" dari halaman *settings.twig* ke *database*. Listing 4.17 menunjukkan hasil perubahan kode program yang dilakukan di *Settings.php*

Listing 4.17: Perubahan kode program pada *Settings.php*

```

@@ -113,9 +113,10 @@ class Settings extends CI_Controller
    'results_per_page_all' => $this->input->post('rpp_all'),
    'results_per_page_final' => $this->input->post('rpp_final'),
    'week_start' => $this->input->post('week_start'),
+   'lock_student_display_name' => $this->input->post('
    lock_student_display_name')===NULL?0:1,
    )
);

```

1
2 Perubahan kode di *User_model.php* untuk mengecek apakah fitur "Lock Student's Display Name"
3 aktif atau tidak aktif. Listing 4.18 menunjukkan hasil perubahan kode program yang dilakukan di
4 *User_model.php*

Listing 4.18: Perubahan kode program pada *User_model.php*

```

5 @@ -404,9 +404,15 @@ class User_model extends CI_Model
6 return FALSE;
7 $the_user = $query->row();
8 $username = $the_user->username;
9 +
10 +
11 + $display_name = $this->input->post('display_name');
12 + $locked = $this->settings_model->get_setting(lock_student_display_name);
13 + if ($locked == 1) {
14 +     $display_name = $the_user->display_name;
15 + }
16
17 $user=array(
18 -     'display_name' => $this->input->post('display_name'),
19 +     'display_name' => $display_name,
20     'email' => $this->input->post('email')
21 );
22

```

23
24 Perubahan kode di *settings.twig* untuk menambahkan check box fitur "Lock Student's Display Name".
25 Listing 4.19 menunjukkan hasil perubahan kode program yang dilakukan di halaman *settings.twig*

Listing 4.19: Perubahan kode program pada halaman *settings.twig*

```

26 @@ -114,6 +114,11 @@ $(document).ready(function(){
27 <label for="form_log">Log</label><br>
28 <span class="form_comment">Enable Log</span>
29 </p>
30 + <p class="input_p">
31 +     <input id="form_lock_student_display_name" type="checkbox" name="
32 lock_student_display_name" value="1" {{ lock_student_display_name ? 'checked'
33 }}/>
34 +
35 +     <label for="form_lock_student_display_name">Lock Student's
36 Display Name</label><br>
37 +     <span class="form_comment">Student's can't change their display name
38 </span>
39 + </p>
40 <p class="input_p">
41     <label for="form_late_rule">Default Coefficient Rule</label>

```

```

1      <span class="form_comment clear">PHP script without <?php ?> tags</
2      span><br>
3

```

Perubahan kode di halaman *profile.twig* untuk menentukan apakah *text field Display Name* akan diaktifkan atau tidak. Listing 4.20 menunjukkan hasil perubahan kode program yang dilakukan di halaman *profile.twig*

Listing 4.20: Perubahan kode program pada halaman *profile.twig*

```

8
9 @@ -28,7 +28,13 @@
10 </p>
11 <p class="input_p clear">
12     <label for="form_name" class="short2">Name:</label>
13 -     <input id="form_name" type="text" name="display_name" class="sharif_input
14         medium" value="{{ display_name }}" />
15 +
16 + {% if lock_student_display_name == 1 %}
17 + <input id="form_name" type="text" name="display_name" class="sharif_input
18         medium" value="{{ display_name }}" disabled/>
19 + {% else %}
20 + <input id="form_name" type="text" name="display_name" class="sharif_input
21         medium" value="{{ display_name }}" />
22 + {% endif %}
23 +
24     {{ form_error('display_name', '<div class="shj_error">', '</div>')}}
25 </p>
26 <p class="input_p clear">
27

```

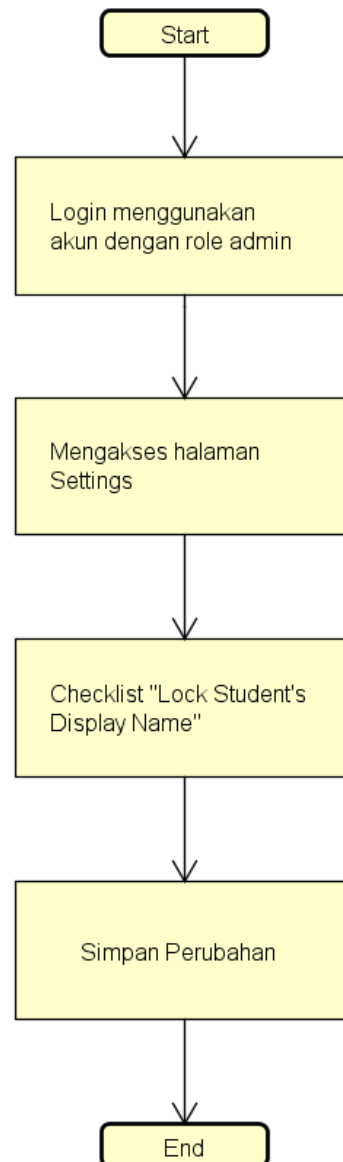
Perubahan kode di *config Install.php* untuk menambahkan atribut *lock_student_display_name* pada tabel *shj_settings* di *database* saat penginstalan *SharIF Judge*. Listing 4.21 menunjukkan perubahan kode di *config Install.php*

Listing 4.21: Perubahan kode program pada *Install.php*

```

32
33 @@ -220,6 +221,7 @@ class Install extends CI_Controller
34 array('shj_key' => 'results_per_page_all', 'shj_value' => '40'),
35 array('shj_key' => 'results_per_page_final', 'shj_value' => '80'),
36 array('shj_key' => 'week_start', 'shj_value' => '0'),
37 + array('shj_key' => 'lock_student_display_name', 'shj_value'
38     => '1'),
39 ));
40 if ( ! $result)
41 show_error("Error adding data to table ".$this->db->dbprefix('settings'));
42

```



Gambar 4.4: Diagram *Flowchart* Mengaktifkan Fitur "*Lock Student's Display Name*"

1 Gambar 4.4 merupakan diagram *flowchart* yang menunjukkan langkah-langkah untuk mengak-
2 tifkan fitur "*Lock Student's Display Name*".

3 4.8 Menambahkan Fitur "*Archived Assignment*"

4 Fitur "*Archived Assignment*" membutuhkan sebuah atribut baru pada *database*. Atribut tersebut
5 berfungsi untuk menyimpan sebuah nilai. Nilai yang disimpan akan menentukan apakah *assignment*
6 tersebut bersifat *Archived Assignment* atau tidak. Atribut baru tersebut ditambahkan pada
7 tabel *shj_assignments* dengan nama *archived_assignment* yang menggunakan tipe data *tinyint*.
8 *archived_assignment* memiliki nilai *default* = 0. Jika nilai dari *archived_assignment* = 1, maka
9 *assignment* tersebut merupakan sebuah *archived_assignment*, sebaliknya jika bernilai 0, maka
10 *assignment* tersebut merupakan *assignment* biasa.

11 Rancangan algoritma yang digunakan untuk menambahkan fitur "*Archived Assignment*" yaitu

1. Menambahkan atribut baru dengan nama *archived_assignment* yang menggunakan tipe data *tinyint*. Atribut baru tersebut ditambahkan pada tabel *shj_assignments*
2. Menambahkan *check box* pada halaman *assignments.twig* untuk mengaktifkan atau menonaktifkan fitur "Archived Assignment".
3. Jika fitur "Archived Assignment" diaktifkan, maka secara otomatis *text field Start time* bernilai 1970-01-02 00:00:00, *text field Finish Time* bernilai 2038-01-18 00:00:00 dan *text field Extra Time* bernilai 0.
4. *Assignment* yang bersifat *Archived Assignment* tidak muncul pada kalender halaman *dashboard*.

Dari rancangan algoritma yang diterapkan, terdapat perubahan kode program pada beberapa kelas. Beberapa kelas tersebut antara lain *model Assignment_model.php*, *view add_assignment.twig*, *view dashboard.twig* dan *config Install.php*.

Perubahan kode di *Assignment_model.php* untuk menyimpan nilai dari *check box* fitur "Archived Assignment" dari halaman *add_assignment.twig* ke *database*. Listing 4.22 menunjukkan hasil perubahan kode program yang dilakukan di *Assignment_model.php*

Listing 4.22: Perubahan kode program pada *Assignments_model.php*

```

@@ -38,6 +38,9 @@ class Assignment_model extends CI_Model
    {
        $extra_time *= $extra_item;
    }
+
+ $archived_assignment = $archived_assignment = $this->input->post('
    archived_assignment')!=NULL ? 1 : 0;
+
$assignment = array(
    'id' => $id,
    'name' => $this->input->post('assignment_name'),
+
+
@@ -51,7 +54,8 @@ class Assignment_model extends CI_Model
    'finish_time' => date('Y-m-d H:i:s', strtotime($this->input->post
    ('finish_time'))),
    'extra_time' => $extra_time*60,
    'late_rule' => $this->input->post('late_rule'),
-    'participants' => $this->input->post('participants')
+    'participants' => $this->input->post('participants'),
+    'archived_assignment' => $archived_assignment
    );
    if($edit)
    {

```

Perubahan kode di halaman *add_assignment.twig* untuk menambahkan *check box* fitur "Archived

1 *Assignment*" dan mengisi nilai *text field Start time* menjadi 1970-01-02 00:00:00, *text field Finish*
 2 *Time* menjadi 2038-01-18 00:00:00 serta *text field Extra Time* menjadi 0. Listing 4.23 munujukan
 3 hasil perubahan kode program yang dilakukan di halaman *add_assignment.twig*

Listing 4.23: Perubahan kode program pada halaman *add_assignment.twig*

```

4
5 @@ -39,6 +39,18 @@
6         shj.num_of_problems++;
7         $('#nop').attr('value', shj.num_of_problems);
8     });
9 +     $("#form_a_archived_assignment").click(function(){
10 +         if ($("#form_a_archived_assignment").is(':checked')) {
11 +             $("#start_time").val('1970-01-02 00:00:00');
12 +             $("#finish_time").val('2038-01-18 00:00:00');
13 +             $("#form_extra_time").val('0');
14 +         }
15 +         else{
16 +             $("#start_time").val('');
17 +             $("#finish_time").val('');
18 +             $("#form_extra_time").val('');
19 +         }
20 +     });
21     $(document).on('click', '.delete_problem', function(){
22         if (shj.num_of_problems==1) return;
23         var row = $(this).parents('tr');
24
25 @@ -153,6 +165,12 @@
26         <span class="form_comment space-left">Check this to show Java
27         exceptions to users</span>
28         {{ form_error('javaexceptions', '<div class="shj_error">', '</div
29         >') }}
30         </p>
31 + <p class="input_p">
32 +         <input id="form_a_archived_assignment" type="checkbox" name="
33         archived_assignment" value="1" {{ edit ? (edit_assignment.archived_assignment
34         ? 'checked') : set_checkbox('archived_assignment', '1')|raw }} />
35 +         <label for="form_a_archived_assignment" class="default">Archived
36         Assignment</label>
37 +         <span class="form_comment space-left">Check this to make an archived
38         assignment</span>
39 +         {{ form_error('archived_assignment', '<div class="shj_error">', '</
40         div>') }}
41 + </p>
42         <p class="input_p">

```

```

1      <label for="form_late_rule">Coefficient rule (<a target="_blank"
2      href="https://github.com/mjnaderi/Sharif-Judge/blob/docs/v1.4/add_assignment.
3      md#coefficient-rule">?</a></label><br>
4      <span class="form_comment medium clear" style="display: block;">
5      PHP script without <?php ?> tags</span>
6

```

Perubahan kode di halaman *dashboard.twig* untuk mengatur *assignment* yang bersifat *Archived Assignment* agar tidak muncul pada kalender. Listing 4.24 menunjukkan hasil perubahan kode program yang dilakukan di halaman *dashboard.twig*

Listing 4.24: Perubahan kode program pada halaman *dashboard.twig*

```

11 @@ -27,8 +27,13 @@ $(document).ready(function () {
12     events: [
13         {% set colors = ['#812C8C', '#FF750D', '#2C578C', '#013440', '#A6222C
14         ', '#42758C', '#02A300', '#BA6900'] %}
15         {% for assignment in all_assignments %}
16         -         {id:{{ assignment.id }},title:'{{ assignment.name|e('js') }}',
17         start:'{{ assignment.start_time }}', end:' {{ assignment.finish_time }}',
18         -         allDay:false,color:'{{ colors[(loop.index0)%
19         colors|length] }}'}
20         +         {% if assignment.archived_assignment == '0' %}
21         +         {id:{{ assignment.id }},title:'{{ assignment.name|e('js') }}',
22         start:'{{ assignment.start_time }}', end:' {{ assignment.finish_time }}',
23         +         allDay:false,color:'{{ colors[(loop.index0)%colors|length] }}'}
24         +         {% endif %}
25         +         {% if assignment.archived_assignment == '1' %}
26         +         {}
27         +         {% endif %}
28         +         {% if not loop.last %},{% endif %}
29         {% endfor %}
30     ]
31 }
32

```

Perubahan kode di *config Install.php* untuk menambahkan atribut *archived_assignment* pada tabel *shj_assignments* di *database* saat penginstalan *SharIF Judge*. Listing 4.25 menunjukkan hasil perubahan kode di *config Install.php*

Listing 4.25: Perubahan kode program pada *Install.php*

```

37 @@ -105,6 +105,7 @@ class Install extends CI_Controller
38     'late_rule'      => array('type' => 'TEXT'),
39     'participants'  => array('type' => 'TEXT', 'default' => ''),
40     'moss_update'   => array('type' => 'VARCHAR', 'constraint' => 30, '
41     default' => 'Never'),
42

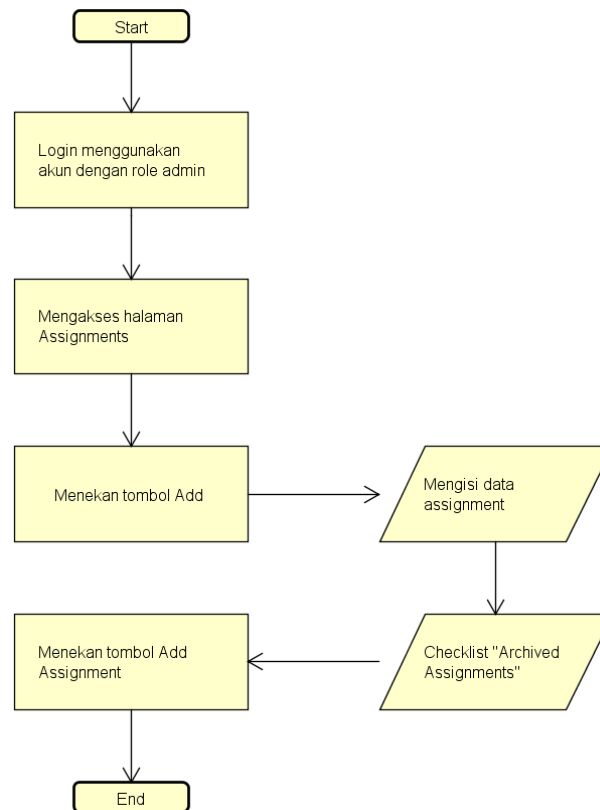
```



```

1 +      'archived_assignment'    => array('type' => 'TINYINT', 'constraint' => 1),
2
3 );
4 $this->dbforge->add_field($fields);
5 $this->dbforge->add_key('id', TRUE); // PRIMARY KEY
6

```



Gambar 4.5: Diagram *Flowchart* Mengaktifkan Fitur "*Archived Assignment*"

Gambar 4.5 merupakan diagram *flowchart* yang menunjukkan langkah-langkah untuk mengaktifkan fitur "*Archived Assignment*".

4.9 Menambahkan Halaman *Hall of Fame*

Halaman *Hall of Fame* tidak membutuhkan atribut atau tabel baru pada *database* namun perlu ditambahkan *model*, *view* dan *controller*.

1. *Model*

Model untuk halaman *Hall of Fame* bernama *Hof_model.php*. Pada *model Hof_model.php* terdapat beberapa fungsi. Tabel 4.6 menunjukkan perincian fungsi *get_all_final_submission* yang terdapat pada *Hof_model.php*

Tabel 4.6: Perincian fungsi *get_all_final_submission*

Nama <i>Method</i>	<i>get_all_final_submission</i>
Parameter <i>Input</i>	-
Parameter <i>Output</i>	semua entri nilai submissions yang telah dijumlahkan
Tabel yang berhubungan	<i>shj_submissions</i>
Deskripsi	Proses untuk mengembalikan semua entri nilai submission yang telah dijumlahkan pada tabel <i>shj_submissions</i>
Algoritma	<ul style="list-style-type: none"> • Menghitung dan menjumlahkan nilai seluruh problem pada setiap assignment peserta tertentu dengan menggunakan rumus perhitungan nilai dari <i>Controller Submissions.php</i>. Rumus yang digunakan adalah $SCORE = pre_score * coefficient / 10000$. • Menjumlahkan seluruh nilai assignment setiap peserta yang tidak bersifat "<i>Upload Only</i>". Nilai yang telah dijumlahkan akan disimpan sebagai total skor. • Mengurutkan total skor dari yang paling besar. • Mengembalikan seluruh entri yang telah dijumlahkan dalam bentuk <i>array</i>.

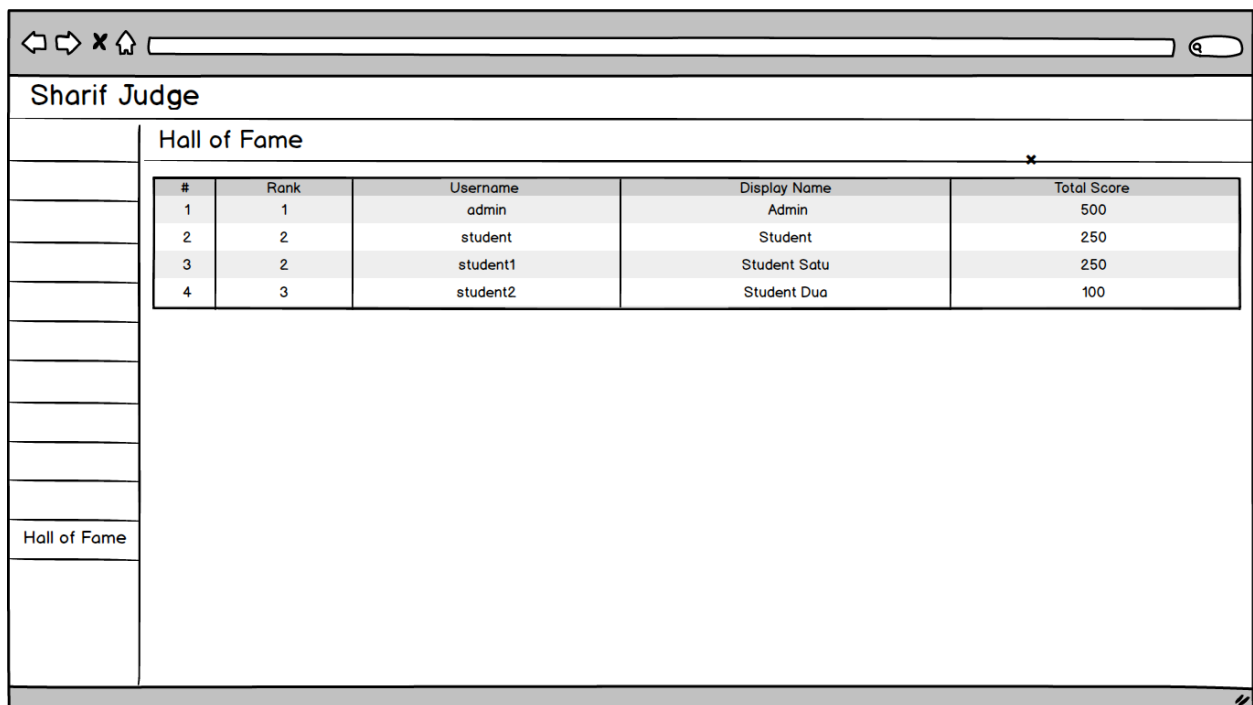
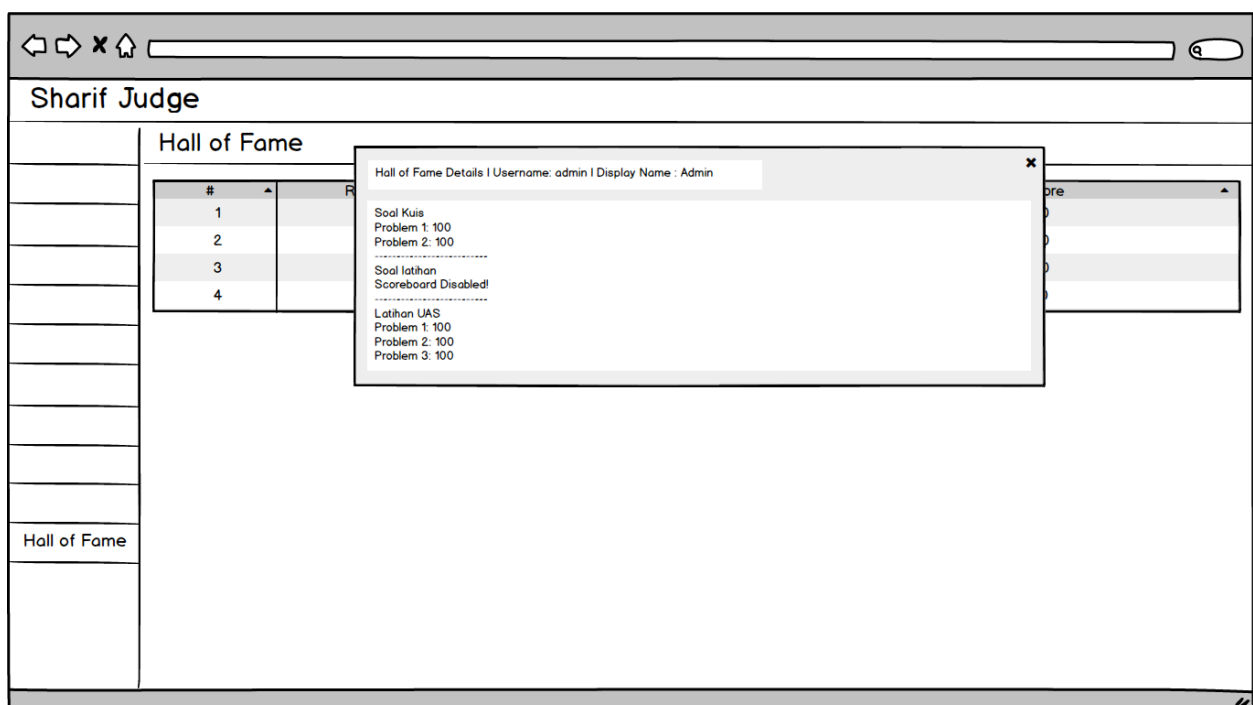
Tabel 4.7 menunjukkan perincian fungsi *get_all_user_assignments* yang terdapat pada *Hof_model.php*

Tabel 4.7: Perincian fungsi *get_all_user_assignments*

Nama <i>Method</i>	<i>get_all_user_assignments</i>
Parameter <i>Input</i>	<i>\$username</i>
Parameter <i>Output</i>	mengembalikan seluruh <i>details</i> dari <i>assignment</i> pengguna tertentu
Tabel yang berhubungan	<i>shj_submissions</i>
Deskripsi	Proses untuk mengembalikan <i>details assignment</i> pengguna tertentu. <i>Details</i> berisikan nama <i>assignment</i> , nama <i>problem</i> dan skor
Algoritma	<ul style="list-style-type: none"> • Menyimpan nama <i>assignment</i>, nama <i>problem</i> dan skor setiap <i>problem</i> dari sebuah <i>assignment</i> pengguna tertentu. • Mengembalikan <i>details</i> di atas dalam bentuk <i>array</i>.

2. View

View untuk halaman *Hall of Fame* bernama *halloffame.twig*. Menu halaman *Hall of Fame* terletak di bawah menu *Scoreboard*. Pada halaman ini juga berlaku sistem ranking sehingga nama para peserta yang muncul diurutkan berdasarkan total skor. Jika total skor yang dimiliki peserta memiliki nilai yang sama dengan peserta lainnya, maka peserta tersebut memiliki ranking yang sama dengan peserta lainnya. Gambar 4.6 menunjukkan rancangan tampilan halaman *Hall of Fame*. Gambar 4.7 menunjukkan rancangan tampilan *detail* dari *Hall of Fame*.

Gambar 4.6: Rancangan tampilan halaman *Hall of Fame*Gambar 4.7: Rancangan tampilan *details Hall of Fame* peserta tertentu

3. Controller

Controller untuk halaman *Hall of Fame* bernama *Halloffame.php*. Pada controller *Halloffame.php* terdapat beberapa fungsi. Tabel 4.8 menunjukkan perincian fungsi *consturct__* yang terdapat pada *Halloffame.php*

Tabel 4.8: Perincian fungsi *consturct__*

Nama <i>Method</i>	<i>consturct__</i>
Parameter <i>Input</i>	-
Parameter <i>Output</i>	-
Tabel yang berhubungan	-
Deskripsi	membatasi pengguna yang dapat mengakses halaman <i>Hall of Fame</i>
Algoritma	<ul style="list-style-type: none"> • mengecek <i>session</i> pengguna yang akan mengakses halaman <i>Hall of Fame</i>. • Jika <i>session</i> tidak berstatus '<i>logged_in</i>', maka pengguna akan dialihkan ke halaman <i>login</i>. • Memuat <i>model Hof_model.php</i>.

1 Tabel 4.9 menunjukkan perincian fungsi *index* yang terdapat pada *Halloffame.php*

Tabel 4.9: Perincian fungsi *index*

Nama <i>Method</i>	<i>index</i>
Parameter <i>Input</i>	-
Parameter <i>Output</i>	-
Tabel yang berhubungan	<i>shj_submissions</i>
Deskripsi	Proses untuk memuat seluruh entri <i>submissions</i> yang telah dijumlahkan pada halaman <i>halloffame.twig</i>
Algoritma	<ul style="list-style-type: none"> • Memuat data <i>Hall of Fame</i> menggunakan fungsi <i>get_all_final_submission</i> dari <i>model Hof_model.php</i>. • Memproses data untuk tampilan <i>halloffame.twig</i>.

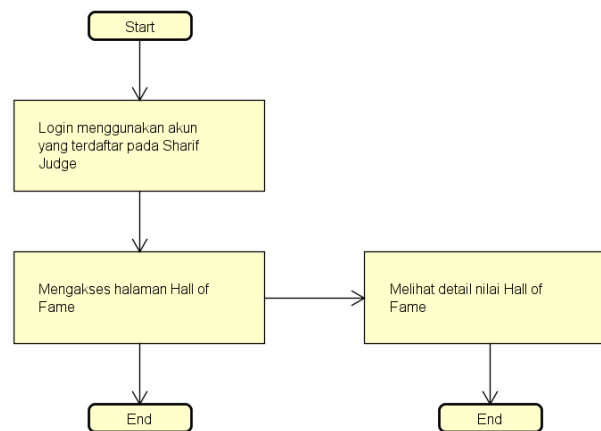
2 Tabel 4.10 menunjukkan perincian fungsi *hof_details* yang terdapat pada *Halloffame.php*

Tabel 4.10: Perincian fungsi *hof_details*

Nama <i>Method</i>	<i>hof_details</i>
Parameter <i>Input</i>	-
Parameter <i>Output</i>	-
Tabel yang berhubungan	<i>shj_submissions</i>
Deskripsi	Proses untuk memuat details <i>submissions</i> pada halaman <i>halloffame.twig</i>
Algoritma	<ul style="list-style-type: none"> • Memuat details <i>Hall of Fame</i> peserta tertentu menggunakan fungsi <i>get_all_user_assignments</i> dari <i>model Hof_model.php</i>. • Memproses data untuk tampilan details <i>Hall of Fame</i> dari peserta tertentu pada halaman <i>halloffame.twig</i>.

3 Selain menambahkan kelas *model*, *view* dan *controller*, terdapat penambahan fungsi pada file
 4 *shj_functions.js* yang terletak di *Sharif-Judge\assets\js*. Penambahan fungsi tersebut berguna

- 1 untuk meminta *details* dari Hall of Fame peserta tertentu menggunakan fungsi *hof_details* pada
 2 *controller Halloffame.php* lalu menampilkannya. Hasil penamabahan dan perubahan kode program
 3 yang dilakukan di *shj_functions.js* dapat dilihat di [Lampiran C](#)



Gambar 4.8: Diagram *Flowchart* Mengakses Halaman *Hall of Fame* dan Melihat Detail *Hall of Fame*

- 4 Gambar 4.8 merupakan diagram *flowchart* yang menunjukkan langkah-langkah untuk mengakses
 5 halaman *Hall of Fame* dan melihat detail nilai *Hall of Fame*.

6 4.10 Integrasi *Login* ke *Server RADIUS*

- 7 *Sharif Judge* memerlukan *library* baru agar dapat mengintegrasikan *login* ke *server RADIUS*.
 8 *Library* yang digunakan adalah *Dapphp/Radius*. *Dapphp/Radius* adalah klien *RADIUS PHP* untuk
 9 mengautentikasi pengguna terhadap *server RADIUS*. Cara yang disarankan untuk menginstal
 10 *Dapphp/Radius* adalah menggunakan *Composer*. Jika *Composer* telah terinstall, maka jalankan
 11 perintah "*composer require dapphp/radius*" atau tambahkan "*dapphp/radius*" pada file *composer.json*
 12 bagian "*section*" [8].

- 13 Rancangan algoritma yang digunakan untuk menambahkan mengintegrasikan *login* ke *server*
 14 *RADIUS* yaitu

- 15 1. Menginstall *Composer* pada perangkat lunak *Sharif Judge*.
- 16 2. Menambahkan *library Dapphp/Radius* menggunakan *Composer*.
- 17 3. Membuat file *secrets.php* pada direktori *Sharif-Judge\application\config* untuk menyimp-
- 18 an konfigurasi alamat *server RADIUS*.
- 19 4. Mengintegrasikan *login* pengguna *Sharif Judge* ke *server RADIUS*.

- 20 Dari rancangan algoritma yang diterapkan, terdapat perubahan kode pada kelas *model User_model.php*.

21

- 22 Perubahan kode di model *User_model.php* untuk memanggil *library Dapphp/Radius* dan menginte-
- 23 grasikan *login* pengguna *Sharif Judge* ke *server RADIUS*. Listing 4.26 menunjukkan hasil perubahan
- 24 kode program yang dilakukan di *User_model.php*

Listing 4.26: Perubahan kode program pada *User_model.php*

```

1  @@ -5,6 +5,7 @@
2
3  * @author Mohammad Javad Naderi <mjnaderi@gmail.com>
4  */
5  defined('BASEPATH') OR exit('No direct script access allowed');
6  + use Dappphp\Radius\Radius;
7
8  class User_model extends CI_Model
9  {
10
11  @@ -330,6 +331,15 @@ class User_model extends CI_Model
12      return FALSE;
13      if ($this->password_hash->CheckPassword($password, $query->row()->
14  password))
15          return TRUE;
16  +
17  + $this->load->config('secrets');
18  + if($this->config->item('shj_authenticate') == 'radius') {
19  +     $client = new Radius();
20  +     $client->setServer($this->config->item('shj_radius')['server']) //
21  RADIUS server address
22  +     ->setSecret($this->config->item('shj_radius')['secret']);
23  +     if($client->accessRequest($username, $password))
24  +         return TRUE;
25  + }
26  return FALSE;
27  }
28

```

29 Kode program file secrets.php dapat dilihat di [Lampiran D](#).

30 4.11 Branding Teknik Informatika

31 Branding Teknik Informatika dilakukan dengan cara mengubah seluruh logo dan ikon *Sharif Judge*
 32 menggunakan logo dan ikon Teknik Informatika. Gambar 4.9 menunjukan logo dan ikon Teknik
 33 Informatika yang digunakan. Gambar 4.10 banner *Sharif Judge* yang digunakan.



Gambar 4.9: Logo dan Ikon



Gambar 4.10: *Banner Sharif Judge*

1 Agar dapat menggunakan logo dan ikon Teknik Informatika dibutuhkan penggantian beberapa
2 file. Beberapa *file* tersebut antara lain *banner.png*, *favicon.ico* dan *logo_small.png* yang terdapat
3 pada `Sharif-Judge/assets/images`. Penggantian beberapa *file* tersebut mencakup *banner* pada
4 halaman *login*, icon pada *title bar* dan *top bar*.

5 Langkah-langkah yang dilakukan untuk proses branding Teknik Informatika adalah

- 6 1. Menggunakan logo dan ikon Teknik Informatika pada *Sharif Judge*.
- 7 2. Mengubah nama *Sharif Judge* menjadi SharIF Judge.
- 8 3. Mengubah link dokumentasi *Github Sharif Judge* menjadi [https://github.com/ftisunpar/](https://github.com/ftisunpar/Sharif-Judge)
9 `Sharif-Judge`.

BAB 5

IMPLEMENTASI DAN PENGUJIAN

Bab ini membahas tentang implementasi dan pengujian perangkat lunak berdasarkan rancangan yang sudah dibuat. Ada dua jenis pengujian yang dilakukan, yaitu pengujian fungsional dan pengujian eksperimental. Bab ini juga membahas tentang lingkungan yang digunakan untuk pengujian perangkat lunak ini.

5.1 Lingkungan untuk Implementasi dan Pengujian

Terdapat dua lingkungan yang digunakan untuk melakukan implementasi dan pengujian. Lingkungan pertama digunakan untuk mengembangkan perangkat lunak *SharIF Judge*. Lingkungan kedua digunakan untuk menguji fitur-fitur hasil pengembangan. Berikut spesifikasi lingkungan perangkat keras dan perangkat lunak yang digunakan

1. Lingkungan Pertama

Tabel 5.1 menunjukan perangkat keras lingkungan pertama.

Tabel 5.1: Perangkat Keras Lingkungan Pertama

Parameter	Nilai
<i>Processor</i>	<i>Intel Core i5 4200u</i>
<i>Graphics Processing Unit (GPU)</i>	<i>Intel HD Graphics HD4000 dan Nvidia GeForce 840M</i>
<i>Random Access Memory (RAM)</i>	12.00GB DDR3
<i>Storage</i>	120GB SSD dan 1TB Harddisk

Tabel 5.2 menunjukan perangkat lunak lingkungan pertama.

Tabel 5.2: Perangkat Lunak Lingkungan Pertama

Parameter	Nilai
Sistem Operasi	Windows 10 10 Education 64-bit
Bahasa Pemrograman	PHP, JavaScript, CSS dan HTML
<i>Text Editor</i>	<i>Atom</i>
<i>Framework</i>	<i>CodeIgniter</i>
Perangkat Lunak pendukung	<i>XAMPP Control Panel v3.2.2</i> <i>Google Chrome Version 65.0.3325.181 (Official Build) (64-bit)</i> <i>Firefox Quantum 59.0.2 (64-bit)</i> <i>Microsoft Excel 365</i>

2. Lingkungan Kedua

Tabel 5.3 menunjukkan perangkat keras lingkungan kedua.

Tabel 5.3: Perangkat Keras Lingkungan Kedua

Parameter	Nilai
<i>Processor</i>	<i>Intel Xeon E5-2603v4</i>
<i>Graphics Processing Unit (GPU)</i>	<i>AMD RADEON R7 240</i>
<i>Random Access Memory (RAM)</i>	8.00GB DDR4
<i>Storage</i>	1TB Harddisk

Tabel 5.4 menunjukkan perangkat lunak lingkungan kedua.

Tabel 5.4: Perangkat Lunak Lingkungan Kedua

Parameter	Nilai
Sistem Operasi	<i>Ubuntu 16.04.1</i>
Perangkat Lunak pendukung	<i>Apache Server 2.4.18</i>
	<i>MySQL 5.7.22</i>
	PHP 7.1.16-1

5.2 Implementasi

Hasil implementasi dari rancangan perangkat lunak yang sudah dibuat, terdiri dari tiga bagian, yaitu

1. Kode Program

Perubahan dan penambahan kode program untuk mengimplementasi kebutuhan *Sharif Judge*, ditulis dalam bahasa pemrograman PHP. Seluruh perubahan kode program telah dijabarkan dalam setiap sub bab pada bab 4. Kode program untuk halaman *Logs* dapat dilihat di [Lampiran A](#). Kode program untuk halaman *Hall of Fame* dapat dilihat di [Lampiran B](#).

2. Basis Data

Terdapat penambahan tabel dalam mengimplementasi kebutuhan *Sharif Judge* pada sub bab 4.5. Tabel tersebut diberi nama *shj_logins*. Tabel 5.5 menunjukkan struktur tabel *shj_logins*

Tabel 5.5: Struktur Tabel *shj_logins*

Atribut	Tipe Data	Ukuran	Default
<i>login_id (primary key)</i>	int	11	None
<i>username</i>	varchar	20	None
<i>ip_address</i>	varchar	15	None
<i>timestamp</i>	timestamp	11	current_timestamp
<i>last_24h_login_id</i>	int	11	null

Terdapat penambahan *key* dan *value* dalam mengimplementasi kebutuhan *Sharif Judge* pada sub bab 4.7. Pada tabel *shj_settings* ditambahkan *shj_key* dengan nama *lock_student_display_name* dan *shj_value* dengan nilai 0. Tabel 5.6 menunjukkan struktur tabel *shj_settings*

Tabel 5.6: Struktur Tabel *shj_settings*

shj_key	shj_value
<i>timezone</i>	Asia/Jakarta
<i>tester_path</i>	pathC:/xampp/htdocs/tester
<i>assignments_root</i>	pathC:/xampp/htdocs/assignments
<i>file_size_limit</i>	50
<i>output_size_limit</i>	1024
<i>queue_is_working</i>	0
<i>default_late_rule</i>	/** Put coefficient (from 100) in variable \$co...
<i>enable_easysandbox</i>	1
<i>enable_c_shield</i>	1
<i>enable_cpp_shield</i>	1
<i>enable_py2_shield</i>	1
<i>enable_py3_shield</i>	1
<i>enable_java_policy</i>	1
<i>enable_log</i>	1
<i>submit_penalty</i>	300
<i>enable_registration</i>	1
<i>registration_code</i>	0
<i>mail_from</i>	shj@example.com
<i>mail_from_name</i>	Sharif Judge
<i>reset_password_mail</i>	<p> Someone requested a password reset for your S...
<i>add_user_mail</i>	<p> Hello! You are registered in SharIF Judge at ...
<i>moss_userid</i>	
<i>results_per_page_all</i>	40
<i>results_per_page_final</i>	80
<i>week_start</i>	0
<i>lock_student_display_name</i>	1

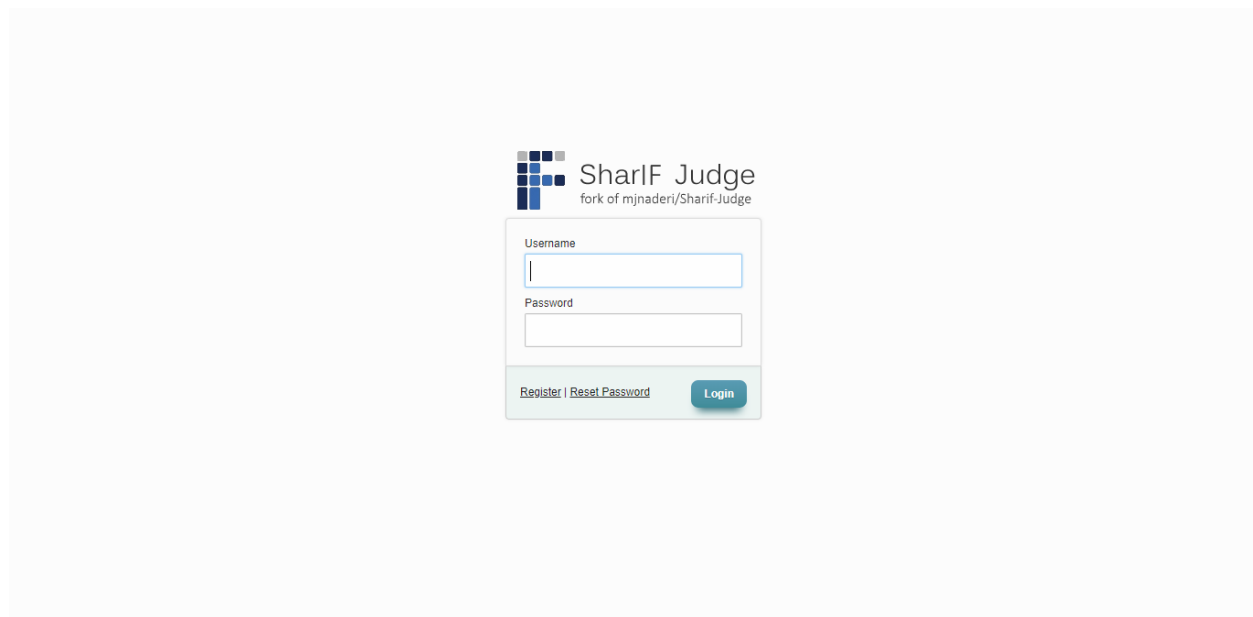
- 1 Terdapat penambahan atribut dalam mengimplementasi kebutuhan *Sharif Judge* pada [sub bab](#)
- 2 [4.8](#). Pada tabel *shj_assignments* ditambahkan atribut baru dengan nama *archived_assignment*.
- 3 Tabel [5.7](#) menunjukkan struktur tabel *shj_assignments*

Tabel 5.7: Struktur Tabel *shj_assignments*

Atribut	Tipe Data	Ukuran	Default
<i>id (primary key)</i>	int	11	None
<i>name</i>	varchar	50	-
<i>problems</i>	smallint	4	None
<i>total_submits</i>	int	11	None
<i>open</i>	tinyint	1	None
<i>scoreboard</i>	tinyint	1	None
<i>javaexceptions</i>	tinyint	1	None
<i>description</i>	text	-	None
<i>start_time</i>	datetime	1	None
<i>finish_time</i>	datetime	1	None
<i>extra_time</i>	int	11	None
<i>late_rule</i>	text	-	None
<i>participants</i>	text	-	None
<i>moss_update</i>	varchar	30	None
<i>archived_assignment</i>	tinyint	1	None

3. Tampilan

Tampilan untuk pengembangan *Sharif Judge* ini dirancang berdasarkan rancangan tampilan yang sudah dibuat. Gambar 5.1 menunjukkan tampilan halaman *login SharIF Judge*.

Gambar 5.1: Tampilan Halaman *Login*

Gambar 5.2 menunjukkan tampilan halaman *Hall of Fame*.

#	Rank	Username	Display Name	Total Score
1	1	i17010	William Oktavianus Kurniawan	4267
2	2	i17017	Cristopher	4160
3	3	i17054	William Xaverius Walah	3920
4	4	i17007	Ignatius Michael Liondy	3872
5	5	i17038	Henrico Leodra	3750
6	6	i17067	Harry Senjaya Darmawan	3527
7	7	i17047	Richard Morris Yonggi	3446
8	8	i17023	Kevin Draven Kenanga	3320
9	9	i17050	Cristine Artanty	3164
10	10	i17022	Kelvin Dravin Kenanga	3100
11	11	i17014	Sterenlie	3068
12	12	i17020	Yohan Kurnia Wijaya	2883

Gambar 5.2: Tampilan Halaman *Hall of Fame*

1

Gambar 5.3 menunjukkan tampilan *detail* dari *Hall of Fame*.

#	Rank	Username	Display Name	Total Score
1	1	i17010	William Oktavianus Kurniawan	4267
2	2	i17017	Cristopher	4160
3	3	i17054	William Xaverius Walah	3920
4	4	i17007	Ignatius Michael Liondy	3872
5	5	i17038	Henrico Leodra	3750
6	6	i17067	Harry Senjaya Darmawan	3527
7	7	i17047	Richard Morris Yonggi	3446
8	8	i17023	Kevin Draven Kenanga	3320
9	9	i17050	Cristine Artanty	3164
10	10	i17022	Kelvin Dravin Kenanga	3100
11	11	i17014	Sterenlie	3068
12	12	i17020	Yohan Kurnia Wijaya	2883

Gambar 5.3: Tampilan *Detail* dari *Hall of Fame*

2

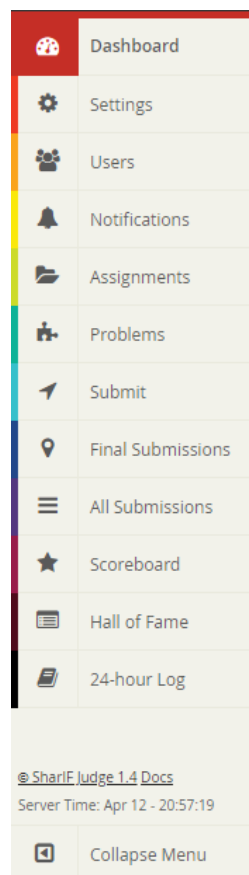
Gambar 5.4 menunjukkan tampilan halaman *24-hour Logs*.

Use this table to detect account lendings between students in a pre-seated exam environment. In a pre-seated environment, student should have logged in from only one IP address within the exam time period. Whenever a user logs in from different IP address in the past 24 hours, last column of this table will show link to the previous login that happened. In such case, proctor can analyse from those two logins, whether the same login has been used in different devices (meaning the account had been lent).

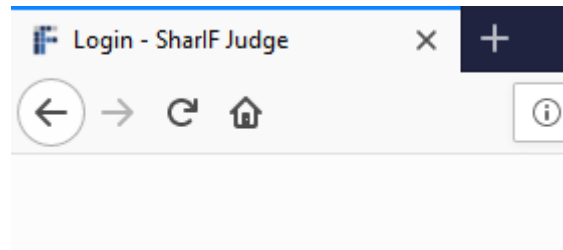
#	Login ID	Username	IP Address	Login Time	Log from different IP (< 24 hours)
1	620	stillmen	172.68.253.50	2018-05-04 00:13:25	
2	619	i17020	172.68.255.77	2018-05-04 00:07:09	547
3	618	husnulhakim	172.68.253.116	2018-05-03 23:23:11	520
4	617	i17067	162.158.146.151	2018-05-03 23:10:24	534
5	616	i17050	162.158.178.18	2018-05-03 22:44:37	606
6	615	i17008	172.68.255.29	2018-05-03 22:10:21	515
7	614	i17074	172.68.46.224	2018-05-03 22:04:24	524
8	613	natalia	172.68.46.224	2018-05-03 22:03:41	507
9	612	i17010	103.22.200.141	2018-05-03 19:49:56	521
10	611	i17050	162.158.178.18	2018-05-03 19:21:33	606
11	610	i17016	172.68.253.158	2018-05-03 19:04:24	531

Gambar 5.4: Tampilan Halaman *24-hour Logs*

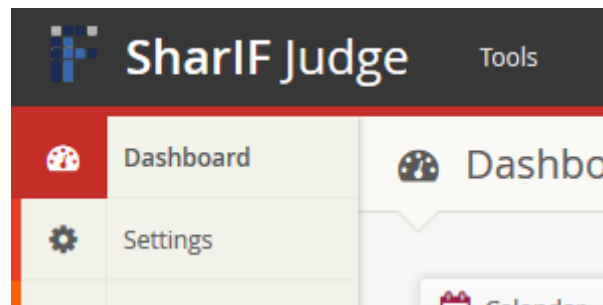
1 Gambar 5.5 menunjukkan tampilan *side menu SharIF Judge*.

Gambar 5.5: Tampilan *Side Menu SharIF Judge*

2 Gambar 5.6 menunjukkan tampilan Ikon *SharIF Judge* pada *Title Bar*.

Gambar 5.6: Ikon *SharIF Judge* pada *Title Bar*

Gambar 5.7 menunjukkan tampilan Logo *SharIF Judge* pada *Top Bar*.

Gambar 5.7: Logo *SharIF Judge* pada *Top Bar*

5.3 Pengujian Fungsional

Pengujian fungsional bertujuan untuk memastikan setiap fitur baru pada *SharIF Judge* dapat berfungsi dengan baik.

5.3.1 Mengunduh Soal (deskripsi & PDF) yang Telah Dibatasi

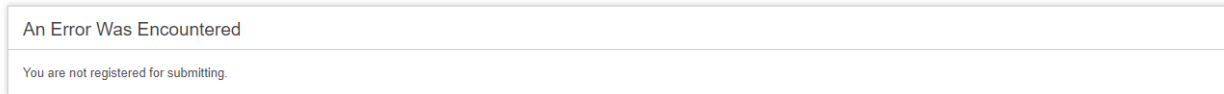
Hasil yang diharapkan dari pengujian fitur ini adalah soal dapat dibatasi sesuai dengan ketentuan yang telah dirancang pada sub bab 4.3. Pengujian dimulai dengan membuat empat buah *assignment*. Gambar 5.8 menunjukkan empat buah *assignment* yang dibuat.

Select	Name	Problems	Submissions	Coefficient	Start Time	Finish Time	Status	PDF	Actions
<input checked="" type="checkbox"/>	Participant Hanya Username Tertentu	1 problem	1 submission	100 %	2018-04-14 13:23:11	2018-04-16 00:00:00	Open		
<input type="checkbox"/>	Assignment Belum Dimulai	1 problem	3 submissions	100 %	2018-04-16 00:00:00	2018-04-21 00:00:00	Open		
<input type="checkbox"/>	Assignment Telah Selesai	1 problem	2 submissions	Finished	2018-04-09 00:00:00	2018-04-13 00:00:00	Open		
<input type="checkbox"/>	Closed Assignment	5 problems	4 submissions	Finished	2018-04-14 13:23:31	2018-04-14 13:23:36	Close		

Gambar 5.8: Empat Buah *Assignment* yang Dibuat

Assignment dengan nama "*Participant Hanya Username Tertentu*", merupakan *assignment* yang dikhususkan untuk peserta tertentu. *Assignment* dengan nama "*Assignment Belum Mulai*", merupakan *assignment* yang belum dimulai. *Assignment* dengan nama "*Assignment Telah Selesai*", merupakan *assignment* yang waktu pengumpulannya telah habis. *Assignment* dengan nama "*Closed Assignment*", merupakan *assignment* yang memiliki status *close*.

- 1 Jika peserta yang tidak terdaftar sebagai *participant*, mencoba untuk mengunduh *assignment*
2 dengan nama "*Participant Hanya Username Tertentu*", maka muncul pesan *error* "*You are not*
3 *registered for submitting.*" seperti [Gambar 5.9](#)



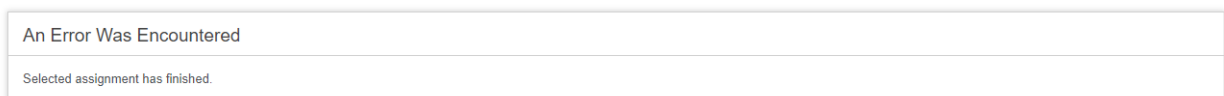
Gambar 5.9: Pesan *Error* "*You are not registered for submitting.*"

- 4 Jika peserta mencoba untuk mengunduh *assignment* dengan nama "*Assignment Belum Mulai*",
5 maka muncul pesan *error* "*Selected assignment has not started.*" seperti [Gambar 5.10](#)



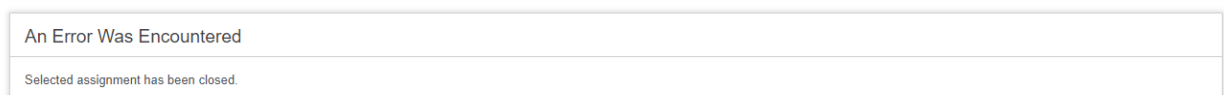
Gambar 5.10: Pesan *Error* "*Selected assignment has not started.*"

- 6 Jika peserta mencoba untuk mengunduh *assignment* dengan nama "*Assignment Telah Selesai*",
7 maka muncul pesan *error* "*Selected assignment has finished.*" seperti [Gambar 5.11](#)



Gambar 5.11: Pesan *Error* "*Selected assignment has finished.*"

- 8 Jika peserta mencoba untuk mengunduh *assignment* dengan nama "*Closed Assignment*", maka
9 muncul pesan *error* "*Selected assignment has been closed.*" seperti [Gambar 5.12](#)



Gambar 5.12: Pesan *Error* "*Selected assignment has been closed.*"

10 5.3.2 Membuat *Assignment* yang Menerima *File* dengan Ekstensi TXT

- 11 Hasil yang diharapkan dari pengujian fitur ini adalah *assignment* yang dibuat bisa menerima *file*
12 dengan ekstensi TXT dan pengguna bisa mengumpulkan file dengan ekstensi TXT ke *assignment*
13 tersebut. Pengujian dimulai dengan membuat *assignment* yang bersifat "*Upload Only*" dan menam-
14 bakan '*txt*' pada *text field Allowed Language*. [Gambar 5.13](#) menunjukkan pembuatan *assignment*
15 *upload* TXT

	Name	Score	Time Limit (ms)			Memory Limit (kB)	Allowed Languages (?)	Diff Command (?)	Diff Argument (?)	Upload Only (?)
			C/C++	Python	Java					
1	Testing Upload	100	500	1500	2000	50000	python 2, Python 3, Java, txt	diff	-bB	<input checked="" type="checkbox"/>

Gambar 5.13: Pembuatan *Assignment Upload* TXT

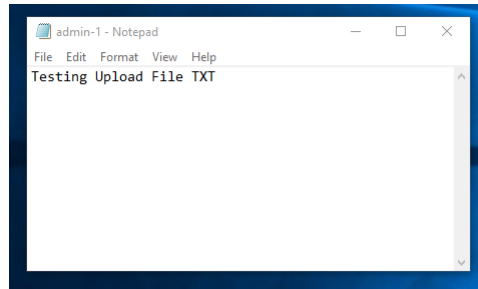
- 1 Setelah *assignment* berhasil dibuat, selanjutnya dicoba untuk mengumpulkan *file* dengan ekstensi
- 2 TXT pada halaman *Submit* seperti Gambar 5.14. Jika berhasil mengumpulkan *file* TXT, maka
- 3 pengguna langsung diarahkan ke halaman *All Submission* seperti Gambar 5.15

Gambar 5.14: *Submit File* TXT

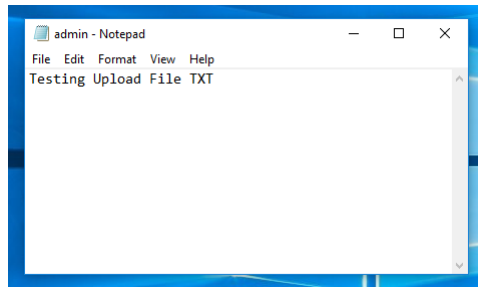
Final	ID	Username	Name	Problem	Submit Time	Score			Language	Status	Code	Log	Actions
						Score	Delay %	Final Score					
✓	1	admin	Admin	1	2018-04-14 14:07:12	0	No Delay 100%	0	TXT	Uploaded	Download	---	↺

Gambar 5.15: Halaman *All Submission* setelah Mengumpulkan *File* TXT

- 4 Pengujian dilanjutkan dengan mengunduh dan mencocokkan isi dari *file* TXT yang baru saja
- 5 dikumpulkan. Jika isi dari *file* TXT hasil unduh sama dengan *file* TXT utama, maka fitur ini dapat
- 6 berjalan dengan baik. Gambar 5.16 merupakan isi dari *file* TXT hasil unduh dan Gambar 5.17 isi
- 7 dari *file* TXT utama.



Gambar 5.16: File TXT Hasil Unduh



Gambar 5.17: File TXT Utama

5.3.3 Mengakses Halaman *24-hour Logs*

- Hasil yang diharapkan dari pengujian fitur ini adalah halaman *24-hour Logs* dapat menampilkan seluruh aktivitas *login* dari pengguna. Pengujian dimulai dari *login* menggunakan *username* dengan *role admin*. Setelah berhasil *login*, akses halaman *24-hour Logs* yang terletak di paling bawah *side menu*. Halaman *24-hour Logs* tampil seperti Gambar 5.18

🏠

Dashboard

⚙️

Settings

👤

Users

🔔

Notifications

📁

Assignments

🔖

Problems

🏹

Submit

📍

Final Submissions

☰

All Submissions

★

Scoreboard

🏆

Hall of Fame

📄

24-hour Log

Tools

09Stack(L)

0000

days hours minutes seconds

👤

#

Login ID

Username

IP Address

Login Time

Log from different IP (< 24 hours)

1

620

stillmen

172.68.253.50

2018-05-04 00:13:25

2

619

i17020

172.68.255.77

2018-05-04 00:07:09

[547](#)

3

618

husnulhakim

172.68.253.116

2018-05-03 23:23:11

[520](#)

4

617

i17067

162.158.146.151

2018-05-03 23:10:24

[534](#)

5

616

i17050

162.158.178.18

2018-05-03 22:44:37

[606](#)

6

615

i17008

172.68.255.29

2018-05-03 22:10:21

[515](#)

7

614

i17074

172.68.46.224

2018-05-03 22:04:24

[524](#)

8

613

natalia

172.68.46.224

2018-05-03 22:03:41

[507](#)

9

612

i17010

103.22.200.141

2018-05-03 19:49:56

[521](#)

10

611

i17050

162.158.178.18

2018-05-03 19:21:33

[606](#)

11

610

i17016

172.68.253.158

2018-05-03 19:04:24

[531](#)

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24-hour Log

📄

24

Gambar 5.18: Halaman *24-hour Logs* yang Tampil

- Pengujian dilanjutkan dengan mencoba *login* menggunakan *username* yang sama namun menggunakan *ip address* yang berbeda. Hasil yang diharapkan adalah halaman *24-hour Logs* dapat

- 1 mencatat dan menampilkan *Login ID* yang menggunakan *ip address* berbeda.

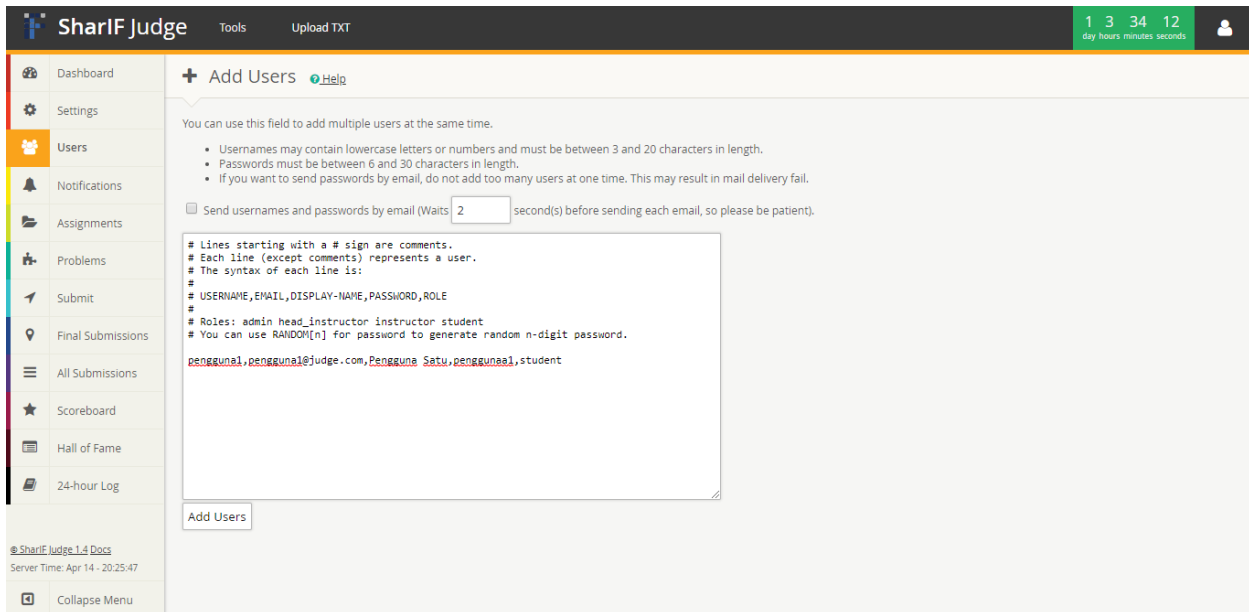
#	Login ID	Username	IP Address	Login Time	Log from different IP (< 24 hours)
1	621	stillmen	103.22.200.183	2018-05-04 00:13:54	620
2	620	stillmen	172.68.253.50	2018-05-04 00:13:25	
3	619	i17020	172.68.255.77	2018-05-04 00:07:09	547
4	618	husnulhakim	172.68.253.116	2018-05-03 23:23:11	520
5	617	i17067	162.158.146.151	2018-05-03 23:10:24	534
6	616	i17050	162.158.178.18	2018-05-03 22:44:37	606
7	615	i17008	172.68.255.29	2018-05-03 22:10:21	515
8	614	i17074	172.68.46.224	2018-05-03 22:04:24	524
9	613	natalia	172.68.46.224	2018-05-03 22:03:41	507
10	612	i17010	103.22.200.141	2018-05-03 19:49:56	521
11	611	i17050	162.158.178.18	2018-05-03 19:21:33	606

Gambar 5.19: Halaman *24-hour Logs* Mencatat Aktivitas *Login* Pengguna

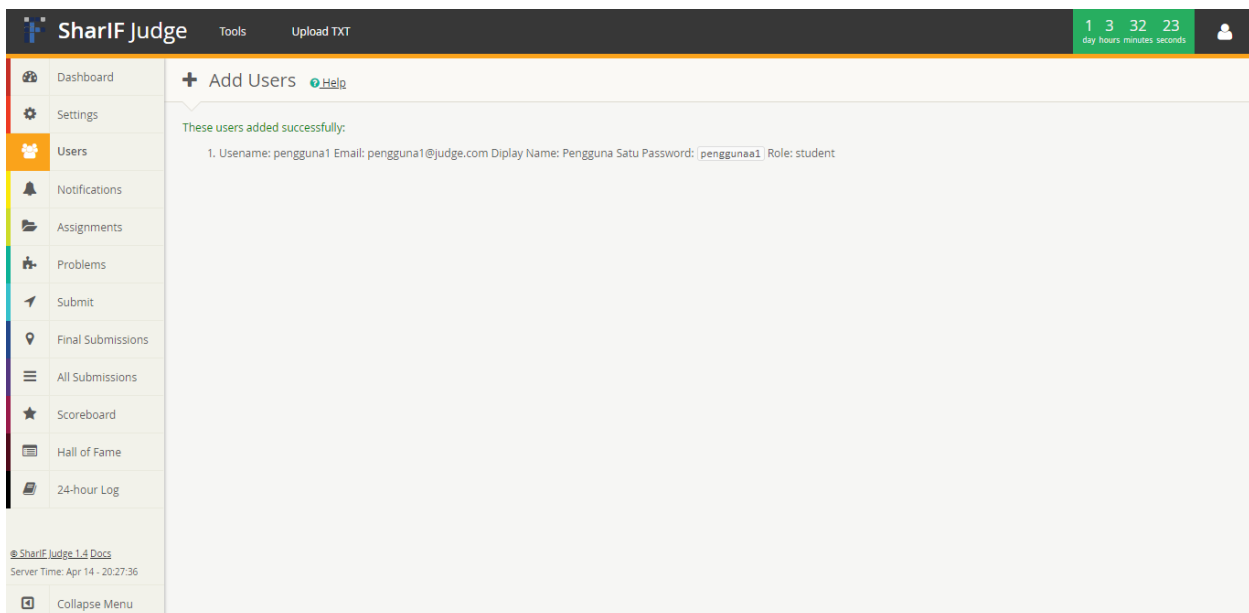
- 2 Terlihat pada [Gambar 5.19](#), *username* stillmen pertama kali *login* menggunakan *ip address*
- 3 172.68.253.50 (baris 2). Setelah mencoba untuk *login* menggunakan *ip address* yang berbeda,
- 4 halaman *24-hour Logs* dapat mencatat dan menampilkan *Login ID* dari *username* stillmen (baris 1).

5.3.4 Mendaftarkan Peserta Menggunakan Tambahan Parameter "*Display Name*"

- 7 Hasil yang diharapkan dari pengujian fitur ini adalah peserta yang didaftarkan langsung memiliki
- 8 *Display Name*. Pengujian dimulai dengan menekan tombol *Add Users* pada halaman *Users*. Gu-
- 9 nakan parameter "pengguna1,pengguna1@judge.com,Pengguna Satu,pengguna1,student" untuk
- 10 menambah peserta baru seperti [Gambar 5.20](#)

Gambar 5.20: Halaman *Add User*

- 1 Jika pengguna baru berhasil ditambahkan, maka akan muncul pesan bahwa tersebut berhasil
- 2 didaftarkan seperti Gambar 5.21



Gambar 5.21: Pengguna Berhasil Didaftarkan

- 3 Pengujian dilanjutkan dengan *login* menggunakan *user* yang baru didaftarkan dan mengecek
- 4 *Display Name*. Gambar 5.22 menunjukkan bahwa *Display Name* yang muncul sesuai dengan parameter
- 5 *Display Name* yang telah dimasukan sebelumnya.

The screenshot shows the SharIF Judge web interface. On the left is a sidebar menu with options: Dashboard, Notifications, Assignments, Problems, Submit, Final Submissions, All Submissions, Scoreboard, and Hall of Fame. The main content area is titled 'Profile'. It contains a form with the following fields: Username (pennguna1), Name (Pengguna Satu), Email (pennguna1@judge.com), Password, and Password, Again. A 'Save' button is at the bottom of the form. At the top right, there is a timer showing 00:00:00 and a user icon. At the bottom left, it says '© SharIF Judge 1.4 Docs' and 'Server Time: Apr 14 - 20:35:08'.

Gambar 5.22: *Display Name* yang Tampil Sesuai dengan Parameter yang Dimasukan

5.3.5 *Disable Display Name* Peserta Menggunakan Fitur "*Lock Student's Display Name*"

Hasil yang diharapkan dari pengujian fitur ini adalah *text field Display Name* pada halaman *Profile* menjadi *disable*, sehingga para peserta tidak dapat mengubahnya. Pengujian dimulai dengan mengaktifkan fitur *Lock Student's Display Name* pada halaman *Settings* seperti Gambar 5.23.

The screenshot shows the SharIF Judge web interface with the 'Settings' page selected in the sidebar. The 'Registration' section has a checkbox 'Open Public Registration' which is checked. The 'Log' section has a checkbox 'Enable Log' which is checked. The 'Lock Student's Display Name' section has a checkbox 'Lock Student's Display Name' which is checked. Below this checkbox is a text box for 'Default Coefficient Rule' containing PHP code. The 'Email' section has three text boxes: 'Send Emails From' (no-reply+shj@labftis.net), 'Send Emails "From" Name' (Judge from FTIS Administrator), and 'Password Reset Email' (You can use {SITE_URL}, {RESET_LINK} and {VALID_TIME}). At the top right, there is a timer showing 13:04:15 and a user icon. At the bottom left, it says '© SharIF Judge 1.4 Docs' and 'Server Time: Apr 14 - 20:35:44'.

Gambar 5.23: Mengaktifkan Fitur *Lock Student's Display Name*

Pengujian dilanjutkan dengan mengecek *text field Display Name* pada halaman *Profile*. Gambar 5.24 menunjukkan bahwa *text field Display Name* telah disable.

The screenshot shows the SharIF Judge user interface. On the left is a sidebar with navigation links: Dashboard, Notifications, Assignments, Problems, Submit, Final Submissions, All Submissions, Scoreboard, and Hall of Fame. The main content area is titled 'Profile'. It contains a form with the following fields: Username (pengguna1, with a note 'You cannot change username.'), Name (Pengguna Satu, which is disabled), Email (pengguna1@judge.com), Password, and Password, Again. A 'Save' button is at the bottom of the form. At the top right, there is a timer showing 0 days, 0 hours, 0 minutes, and 0 seconds. At the bottom left, it says '© SharIF Judge 1.4 Docs' and 'Server Time: Apr 14 - 20:58:36'.

Gambar 5.24: *Text Field Display Name Menjadi Disable*

5.3.6 Menambahkan *Assignment* yang Mengaktifkan Fitur "*Archived Assignment*"

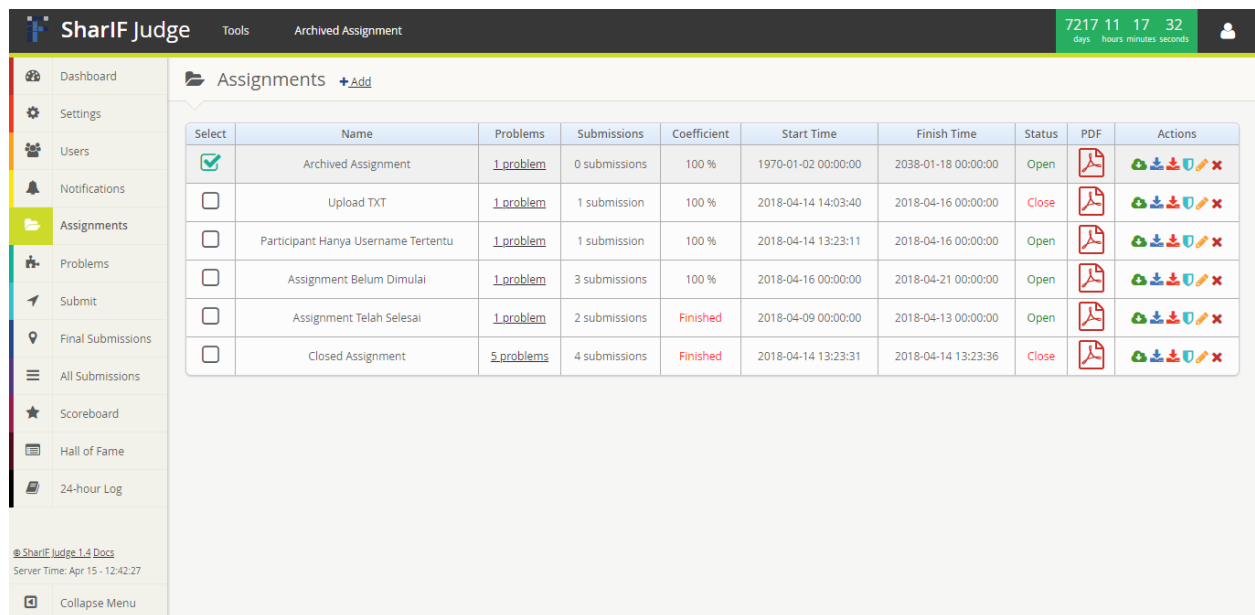
Hasil yang diharapkan dari pengujian fitur ini adalah *assignment* yang dibuat memiliki batas waktu pengumpulan sampai tanggal 18 Januari 2038. Pengujian dimulai dengan membuat *assignment* dan mengaktifkan fitur *Archived Assignment* pada halaman *Add Assignment* seperti Gambar 5.25.

The screenshot shows the 'Add Assignment' page in SharIF Judge. The 'Assignment Name' is 'Archived Assignment'. The 'Start Time' is '1970-01-02 00:00:00' and the 'Finish Time' is '2038-01-18 00:00:00'. The 'Extra Time (minutes)' is '0'. The 'Participants' field contains 'ALL'. The 'Tests and Descriptions (zip file)' and 'PDF File' fields both show 'Choose File' and 'No file chosen'. On the right, there are checkboxes for 'Open', 'Scoreboard', 'Java Exceptions', and 'Archived Assignment', all of which are checked. Below these is a 'Coefficient rule (2)' section with a PHP script. At the bottom, there is a 'Problems' table with one problem listed.

Name	Score	Time Limit (ms)			Memory Limit (kB)	Allowed Languages (2)	Diff Command (2)	Diff Argument (2)	Upload Only (2)
		C/C++	Python	Java					
1 Arc Assignmer	100	500	1500	2000	50000	C,C++,Python 2,Python 3,J	diff	-bB	<input type="checkbox"/>

Gambar 5.25: Mengaktifkan Fitur *Archived Assignment*

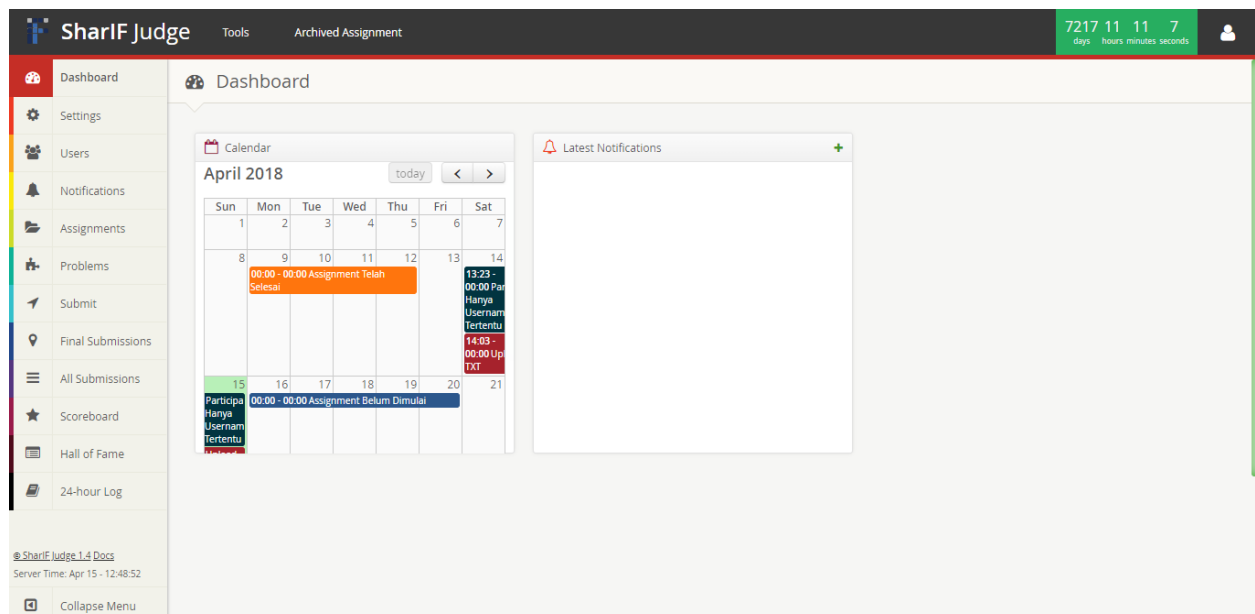
Gambar 5.26 menunjukkan bahwa *assignment* yang baru dibuat (baris 1) memiliki nilai *Finish Time* "2038-01-18 00:00:00" yang artinya *assignment* tersebut dapat dikumpulkan sampai tanggal 18 Januari 2038.



Select	Name	Problems	Submissions	Coefficient	Start Time	Finish Time	Status	PDF	Actions
<input checked="" type="checkbox"/>	Archived Assignment	1 problem	0 submissions	100 %	1970-01-02 00:00:00	2038-01-18 00:00:00	Open		
<input type="checkbox"/>	Upload TXT	1 problem	1 submission	100 %	2018-04-14 14:03:40	2018-04-16 00:00:00	Close		
<input type="checkbox"/>	Participant Hanya Username Tertentu	1 problem	1 submission	100 %	2018-04-14 13:23:11	2018-04-16 00:00:00	Open		
<input type="checkbox"/>	Assignment Belum Dimulai	1 problem	3 submissions	100 %	2018-04-16 00:00:00	2018-04-21 00:00:00	Open		
<input type="checkbox"/>	Assignment Telah Selesai	1 problem	2 submissions	Finished	2018-04-09 00:00:00	2018-04-13 00:00:00	Open		
<input type="checkbox"/>	Closed Assignment	5 problems	4 submissions	Finished	2018-04-14 13:23:31	2018-04-14 13:23:36	Close		

Gambar 5.26: *Finish Time* dengan Nilai "2038-01-18 00:00:00"

- 1 Pengujian dilanjutkan dengan mengecek kalender pada halaman *Dashboard*. *Assignment* yang
- 2 mengaktifkan fitur *Archived Assignment* tidak akan muncul pada kalender seperti Gambar 5.27



Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
						13:23 - 00:00 Par Hanya Usernam Tertentu
						14:03 - 00:00 Up TXT
15	16	17	18	19	20	21
Participa Hanya Usernam Tertentu						

Gambar 5.27: *Archived Assignment* Tidak Muncul pada Kalender

3 5.3.7 Mengakses Halaman *Hall of Fame*

- 4 Hasil yang diharapkan dari pengujian fitur ini adalah halaman *Hall of Fame* dapat menampilkan
- 5 total nilai semua pengguna dari setiap *assignment* yang telah dikumpulkan. Pengujian dimulai
- 6 dengan mengakses halaman *Hall of Fame* yang terletak di bawah *side menu Scoreboard*. Halaman
- 7 *Hall of Fame* tampil seperti Gambar 5.28

#	Rank	Username	Display Name	Total Score
1	1	i17010	William Oktavianus Kurniawan	4267
2	2	i17017	Cristopher	4160
3	3	i17054	William Xaverius Walah	3920
4	4	i17007	Ignatius Michael Liondy	3872
5	5	i17038	Henrico Leodra	3750
6	6	i17067	Harry Senjaya Darmawan	3527
7	7	i17047	Richard Morris Yonggi	3446
8	8	i17023	Kevin Draven Kenanga	3320
9	9	i17050	Cristine Artanty	3164
10	10	i17022	Kelvin Dravin Kenanga	3100
11	11	i17014	Sterenlie	3068
12	12	i17020	Yohan Kurnia Wijaya	2883

Gambar 5.28: Halaman *Hall of Fame*

- 1 Pengujian dilanjutkan dengan mengklik baris pada tabel untuk melihat *details* nilai pengguna.
- 2 Gambar 5.29 menunjukkan *details* nilai yang diperoleh pengguna dengan *username* *i17010*.

#	Rank	Username	Display Name	Total Score
1	1	i17010	William Oktavianus Kurniawan	4267
2	2	i17017	Cristopher	4160
3	3	i17054	William Xaverius Walah	3920
4	4	i17007	Ignatius Michael Liondy	3872
5	5	i17038	Henrico Leodra	3750
6	6	i17067	Harry Senjaya Darmawan	3527
7	7	i17047	Richard Morris Yonggi	3446
8	8	i17023	Kevin Draven Kenanga	3320
9	9	i17050	Cristine Artanty	3164
10	10	i17022	Kelvin Dravin Kenanga	3100
11	11	i17014	Sterenlie	3068
12	12	i17020	Yohan Kurnia Wijaya	2883

Hall of Fame Details Username: i17010 Display Name: William Oktavianus Kurniawan	
Latihan Judge	Penjumlahan : 100
Recursive (L)	Fibonacci : 100
Recursive (T)	Belajar Menulis : 100
Recursive (T2)	noX : 100
Triangle	Concatenation : 100
Target	Target : 100
Marionbat	Marionbat : 100
Basic Sorting (L)	04BasicSorting(L)-02 : 100
Basic Sorting (L)	04BasicSorting(L)-01 : 100
Advanced Sorting (L)	05AdvancedSorting(L)-L0502 : 100
Advanced Sorting (L)	05AdvancedSorting(L)-L0501 : 100
Advanced Sorting (T)	T0503xxxxx : 100
Advanced Sorting (T)	T0502xxxxx : 100
Advanced Sorting (T)	T0501xxxxx : 100
Searching (L)	Sum : 100
Searching (L)	Lempar : 17
Searching (L)	Alamat : 100
Searching (T)	

Gambar 5.29: *Details* Nilai yang Diperoleh

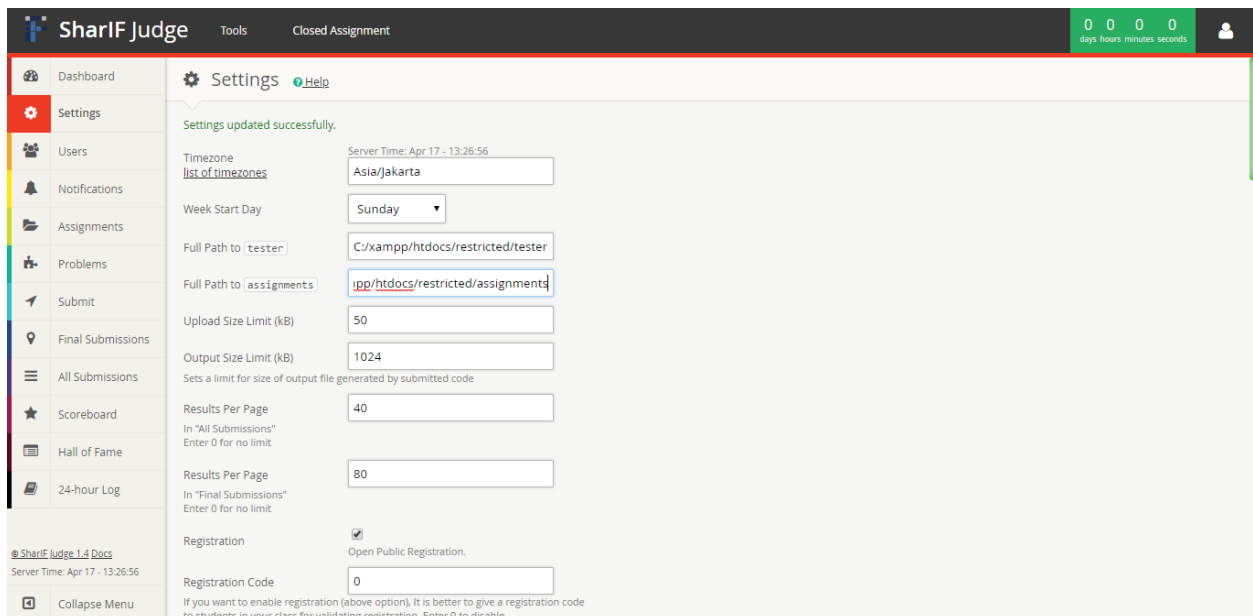
3 5.4 Pengujian Ekseprimental

- 4 Pengujian eksperimental dilakukan dengan cara menginstal perangkat lunak *SharIF Judge* pada
- 5 server lab FTIS UNPAR. Pengujian ini bertujuan untuk menguji coba secara langsung fitur-fitur baru
- 6 *SharIF Judge* pada mata kuliah pemrograman. *SharIF Judge* diuji untuk mata kuliah Algoritma
- 7 & Struktur Data (ASD) semester genap 2017/2018. *SharIF Judge* yang telah diinstall, dapat
- 8 diakses pada URL <http://asd-lat.ftis.unpar/> dan <http://asd-quiz.ftis.unpar/>. Selama

1 pengujian berlangsung, terdapat beberapa persoalan yang muncul. Persoalan yang muncul dicatat
 2 ke dalam *issue repository SharIF Judge* di *GitHub*. Semua persoalan dapat dilihat di <https://github.com/ayenz/Sharif-Judge/issues>.

4 5.4.1 Remove the Assignments Folder

5 *Issue* dengan kode unik #4 meminta agar *folder assignments* untuk dihapus. Hal ini bertujuan
 6 agar direktori *URL /assignments* dapat diakses. *Folder assignments* merupakan *folder default*
 7 untuk menyimpan *assignment*. *Folder default* penyimpanan *assignment* tersebut harus diubah
 8 terlebih dahulu. Untuk mengubah *folder default* penyimpanan, diperlukan perubahan nilai pada
 9 halaman *Settings* dan kode pada kelas *Controller Install.php*. Selain mengubah *folder default*
 10 penyimpanan *assignment*, perubahan juga dilakukan pada *folder default tester* dan *default timezone*
 11 yang bertujuan untuk merapihkan struktur direktori.



Gambar 5.30: Perubahan yang Dilakukan pada Halaman *Settings*

12 Gambar 5.30 menunjukkan perubahan pada halaman *Settings* dilakukan pada *text field "Timezone"*,
 13 *"Full Path to tester"* dan *"Full Path to assignments"*.

- 14 • Nilai yang digunakan untuk *text field "Full Path to assignments"* adalah `C:/xampp/htdocs/restricted/assignments`. Nilai tersebut memiliki arti bahwa *folder default* penyimpanan *assignment* diletakan pada direktori `C:/xampp/htdocs/restricted/assignments`.
- 17 • Nilai yang digunakan untuk *text field "Full Path to tester"* adalah `C:/xampp/htdocs/restricted/tester`. Nilai tersebut memiliki arti bahwa *folder default tester* diletakan pada direktori `C:/xampp/htdocs/restricted/tester`.
- 20 • Nilai yang digunakan untuk *text field "Timezone"* adalah `Asia/Jakarta`. Nilai tersebut memiliki arti bahwa *default timezone* yang digunakan pada *SharIF Judge* mengikuti *timezone* wilayah `Asia/Jakarta`.

1 Perubahan kode pada kelas *Controller Install.php* bertujuan agar penginstalan *SharIF Judge*
 2 langsung menggunakan *default* nilai seperti di atas. Listing 5.1 menunjukkan perubahan kode
 3 program yang dilakukan di *Install.php*

Listing 5.1: Perubahan kode program pada *Install.php*

```

4  @@ -196,9 +196,9 @@ class Install extends CI_Controller
5
6
7  // insert default settings to table 'settings'
8  $result = $this->db->insert_batch('settings', array(
9      -      array('shj_key' => 'timezone',          'shj_value' => 'Asia/Tehran
10         '),
11      -      array('shj_key' => 'tester_path',        'shj_value' => '/home/shj/
12         tester'),
13      -      array('shj_key' => 'assignments_root',    'shj_value' => '/home/shj/
14         assignments'),
15      +      array('shj_key' => 'timezone',          'shj_value' => 'Asia/Jakarta
16         '),
17      +      array('shj_key' => 'tester_path',        'shj_value' => dirname(
18         __FILE__, 3) . "/restricted/tester"),
19      +      array('shj_key' => 'assignments_root',    'shj_value' => dirname(
20         __FILE__, 3) . "/restricted/assignments"),
21

```

22 Setelah melakukan beberapa perubahan, folder assignment dapat dihapus.

23 5.4.2 *Add User Email are not formatted correctly*

24 *Issue* dengan kode unik #5 mengatakan bahwa pendaftaran peserta via *email* tidak berfungsi dengan
 25 baik. Pengguna mendapatkan *email* dengan format yang tidak beraturan. Contohnya *username*
 26 yang tertukar dengan *role* dan *password* tertukar dengan *display name* seperti [Gambar 5.31](#)

Sharif Judge Username and Password



Hello! You are registered in Sharif Judge at <http://asd-lat.ftis.unpar/> as

Your username: pascal

Your password: Pascal

You can log in at <http://asd-lat.ftis.unpar/>

Gambar 5.31: Format *Email* Tidak Beraturan

Hal di atas terjadi karena penambahan parameter "*Display Name*" pada fungsi `add_user` yang terdapat di `model User_model.php`. Persoalan ini dapat diatasi dengan memperbaiki urutan data yang dikirimkan ke email. Listing 5.2 menunjukkan perubahan kode program yang dilakukan di `User_model.php`

Listing 5.2: Perubahan kode program pada `User_model.php`

```

@@ -219,9 +219,9 @@ class User_model extends CI_Model
    $this->email->subject('Sharif Judge Username and Password');
    $text = $this->settings_model->get_setting('add_user_mail');
    $text = str_replace('{SITE_URL}', base_url(), $text);
-   $text = str_replace('{ROLE}', $user[3], $text);
+   $text = str_replace('{ROLE}', $user[4], $text);
    $text = str_replace('{USERNAME}', $user[0], $text);
-   $text = str_replace('{PASSWORD}', htmlspecialchars($user[2]), $text);
+   $text = str_replace('{PASSWORD}', htmlspecialchars($user[3]), $text);
    $text = str_replace('{LOGIN_URL}', base_url(), $text);
    $this->email->message($text);
    $this->email->send();

```

Setelah melakukan perubahan kode program, pendaftaran peserta via *email* dapat berfungsi dengan baik. *Email* yang diterima pengguna telah menggunakan format yang benar.

5.4.3 *Cannot Create Assignments*

Issue dengan kode unik #7 mengatakan bahwa Bapak Husnul selaku dosen Algoritma & Struktur Data, tidak bisa membuat *assignment*. *Error log* yang dihasilkan tidak menunjukkan pesan kesalahan

apapun. Berikut isi *error log* yang terdapat di direktori `/var/log/apache2/error.log`

```
[Wed Jan 31 06:25:02.516154 2018] [mpm_prefork:notice] [pid 1734] AH00163: Apache
/2.4.18 (Ubuntu) configured -- resuming normal operations
[Wed Jan 31 06:25:02.516182 2018] [core:notice] [pid 1734] AH00094: Command line:
'/usr/sbin/apache2'
```

Persoalan terjadi karena tidak ada atribut *archived_assignment* di tabel *shj_assignments*. Persoalan ini dapat diatasi dengan menambahkan atribut *archived_assignment* di tabel *shj_assignments*, sehingga struktur tabel *shj_assignments* menjadi seperti Tabel 5.5. *Assignment* berhasil dibuat setelah atribut *archived_assignment* ditambahkan.

5.4.4 Link ke New Home of Sharif-Judge

Issue dengan kode unik #8 mengatakan bahwa pada halaman *Add* atau *Edit Assignment*, terdapat *link* menuju ke dokumentasi *Sharif Judge* milik mjnaderi. *Link* tersebut diupdate menuju ke dokumentasi *SharIF Judge* yang baru yaitu <https://github.com/ftisunpar/Sharif-Judge/>. Persoalan ini dapat diselesaikan dengan mengubah *link* dokumentasi pada halaman *add_assignment.twig*. Listing 5.3 menunjukkan perubahan kode program yang dilakukan di halaman *user.twig*

Listing 5.3: Perubahan kode program pada halaman *user.twig*

```
@@ -12,7 +12,7 @@
{% block title_menu %}
-     <span class="title_menu_item"><a href="https://github.com/mjnaderi/Sharif-
Judge/blob/docs/v1.4/users.md" target="_blank"><i class="fa fa-question-circle
color6"></i> Help</a></span>
+     <span class="title_menu_item"><a href="https://github.com/ftisunpar/
Sharif-Judge/blob/docs/v1.4/users.md" target="_blank"><i class="fa fa-question
-circle color6"></i> Help</a></span>
<span class="title_menu_item"><a href="{{ site_url('users/add') }}"><i class="fa
fa-plus color11"></i> Add Users</a></span>
<span class="title_menu_item"><a href="{{ site_url('users/list_excel') }}"><i
class="fa fa-download color9"></i> Excel</a></span>
{% endblock %}
```

Listing 5.4 menunjukkan perubahan kode program yang dilakukan di halaman *settings.twig*

Listing 5.4: Perubahan kode program pada halaman *settings.twig*

```
@@ -24,7 +24,7 @@ $(document).ready(function(){
{% block title_menu %}
<span class="title_menu_item">
-     <a href="https://github.com/mjnaderi/Sharif-Judge/blob/docs/v1.4/settings.
md" target="_blank"><i class="fa fa-question-circle color6"></i> Help</a>
```

```

1  +      <a href="https://github.com/ftisunpar/Sharif-Judge/blob/docs/v1.4/
2      settings.md" target="_blank"><i class="fa fa-question-circle color6"></i> Help
3      </a>
4  </span>
5  {% endblock %}
6
7  @@ -151,7 +151,7 @@ $(document).ready(function(){
8
9  <h2 class="shj_form">
10  Sandboxing <span class="title_menu_item">
11  -      <a href="https://github.com/mjnaderi/Sharif-Judge/blob/docs/v1.4/
12      sandboxing.md" target="_blank"><i class="fa fa-question-circle color11"></i>
13      Help</a>
14  +      <a href="https://github.com/ftisunpar/Sharif-Judge/blob/docs/v1.4/
15      sandboxing.md" target="_blank"><i class="fa fa-question-circle color11"></i>
16      Help</a>
17  </span>
18  </h2>
19
20  @@ -160,7 +160,7 @@ $(document).ready(function(){
21  <label for="form_easysandbox">EasySandbox</label>
22  </p>
23  <p class="form_comment">Enable EasySandbox for C/C++.<br>
24  -      You must <a href="https://github.com/mjnaderi/Sharif-Judge/blob/docs/v1
25      .4/sandboxing.md#build-easysandbox" target="_blank">build EasySandbox</a>
26      before enabling it.<br>
27  +      You must <a href="https://github.com/ftisunpar/Sharif-Judge/blob/docs/v1
28      .4/sandboxing.md#build-easysandbox" target="_blank">build EasySandbox</a>
29      before enabling it.<br>
30  {% if not sandbox_built %}
31  <span style="color: red;">You have not built EasySandbox yet.</span>
32  {% endif %}
33
34  @@ -171,35 +171,35 @@ $(document).ready(function(){
35  <label for="form_java_policy">Java Policy</label>
36  </p>
37  <p class="form_comment">
38  -      Enable <a href="https://github.com/mjnaderi/Sharif-Judge/blob/docs/v1.4/
39      sandboxing.md#java-sandboxing" target="_blank">Java Sandboxing</a>
40  +      Enable <a href="https://github.com/ftisunpar/Sharif-Judge/blob/docs/v1.4/
41      sandboxing.md#java-sandboxing" target="_blank">Java Sandboxing</a>
42  </p>

```

```

1
2 <h2 class="shj_form">
3 Shield <span class="title_menu_item">
4 -     <a href="https://github.com/mjnaderi/Sharif-Judge/blob/docs/v1.4/shield.
5 md" target="_blank"><i class="fa fa-question-circle color11"></i> Help</a>
6 +     <a href="https://github.com/ftisunpar/Sharif-Judge/blob/docs/v1.4/shield.
7 md" target="_blank"><i class="fa fa-question-circle color11"></i> Help</a>
8 </span>
9 </h2>
10
11 <p class="input_p">
12 <input id="form_c_sh" type="checkbox" name="enable_c_shield" value="1" {{
13     enable_c_shield ? 'checked' }}/> <label for="form_c_sh">C Shield</label><br>
14 -     <span class="form_comment">Enable <a href="https://github.com/mjnaderi/
15 Sharif-Judge/blob/docs/v1.4/shield.md" target="_blank">Shield</a> for C</span>
16 +     <span class="form_comment">Enable <a href="https://github.com/ftisunpar/
17 Sharif-Judge/blob/docs/v1.4/shield.md" target="_blank">Shield</a> for C</span>
18 </p>
19 <p class="input_p">
20 <input id="form_cpp_sh" type="checkbox" name="enable_cpp_shield" value="1" {{
21     enable_cpp_shield ? 'checked' }}/>
22 <label for="form_cpp_sh">C++ Shield</label><br>
23 -     <span class="form_comment">Enable <a href="https://github.com/mjnaderi/
24 Sharif-Judge/blob/docs/v1.4/shield.md" target="_blank">Shield</a> for C++</
25 span>
26 +     <span class="form_comment">Enable <a href="https://github.com/ftisunpar/
27 Sharif-Judge/blob/docs/v1.4/shield.md" target="_blank">Shield</a> for C++</
28 span>
29 </p>
30 <p class="input_p">
31 <input id="form_py2_sh" type="checkbox" name="enable_py2_shield" value="1" {{
32     enable_py2_shield ? 'checked' }}/>
33 <label for="form_py2_sh">Python 2 Shield</label><br>
34 -     <span class="form_comment">Enable <a href="https://github.com/mjnaderi/
35 Sharif-Judge/blob/docs/v1.4/shield.md" target="_blank">Shield</a> for Python
36 2</span>
37 +     <span class="form_comment">Enable <a href="https://github.com/ftisunpar/
38 Sharif-Judge/blob/docs/v1.4/shield.md" target="_blank">Shield</a> for Python
39 2</span>
40 </p>
41 <p class="input_p">
42 <input id="form_py3_sh" type="checkbox" name="enable_py3_shield" value="1" {{

```

```

1   enable_py3_shield ? 'checked' }}/>
2   <label for="form_py3_sh">Python 3 Shield</label><br>
3   -   <span class="form_comment">Enable <a href="https://github.com/mjnaderi/
4       Sharif-Judge/blob/docs/v1.4/shield.md" target="_blank">Shield</a> for Python
5       3</span>
6   +   <span class="form_comment">Enable <a href="https://github.com/ftisunpar/
7       Sharif-Judge/blob/docs/v1.4/shield.md" target="_blank">Shield</a> for Python
8       3</span>
9   </p>
10  <p class="input_p">
11  <label for="form_def_c">Shield Rules (for C)</label>
12

```

Listing 5.5 menunjukkan perubahan kode program yang dilakukan di halaman *add_user.twig*

Listing 5.5: Perubahan kode program pada halaman *add_user.twig*

```

15  @@ -12,7 +12,7 @@
16
17
18  {% block title_menu %}
19  -   <span class="title_menu_item"><a href="https://github.com/mjnaderi/Sharif-
20      Judge/blob/docs/v1.4/users.md#add-users" target="_blank"><i class="fa fa-
21      question-circle color6"></i> Help</a></span>
22  +   <span class="title_menu_item"><a href="https://github.com/ftisunpar/
23      Sharif-Judge/blob/docs/v1.4/users.md#add-users" target="_blank"><i class="fa
24      fa-question-circle color6"></i> Help</a></span>
25  {% endblock %}
26

```

Listing 5.6 menunjukkan perubahan kode program yang dilakukan di halaman *add_assignment.twig*

Listing 5.6: Perubahan kode program pada halaman *add_assignment.twig*

```

29  @@ -78,7 +78,7 @@
30
31
32  {% block title_menu %}
33  <span class="title_menu_item">
34  -   <a href="https://github.com/mjnaderi/Sharif-Judge/blob/docs/v1.4/
35      add_assignment.md" target="_blank"><i class="fa fa-question-circle color1"></i>
36      > Help</a>
37  +   <a href="https://github.com/ftisunpar/Sharif-Judge/blob/docs/v1.4/
38      add_assignment.md" target="_blank"><i class="fa fa-question-circle color1"></i>
39      > Help</a>
40  </span>
41  {% endblock %}
42

```

```

1 @@ -132,7 +132,7 @@
2 <p class="input_p clear">
3 <label for="form_tests_desc">Tests and Descriptions (zip file)<br>
4 <span class="form_comment">
5 -     <a href="https://github.com/mjnaderi/Sharif-Judge/blob/docs/v1.4/
6     tests_structure.md" target="_blank">Use this structure</a>
7 +     <a href="https://github.com/ftisunpar/Sharif-Judge/blob/docs/v1.4/
8     tests_structure.md" target="_blank">Use this structure</a>
9 </span>
10 </label>
11 <input id="form_tests_desc" type="file" name="tests_desc" class="sharif_input
12     medium"/>
13
14 @@ -172,7 +172,7 @@
15 {{ form_error('archived_assignment', '<div class="shj_error">', '</div>')}}
16 </p>
17 <p class="input_p">
18 -     <label for="form_late_rule">Coefficient rule (<a target="_blank" href="
19     https://github.com/mjnaderi/Sharif-Judge/blob/docs/v1.4/add_assignment.md#
20     coefficient-rule">?</a>)</label><br>
21 +     <label for="form_late_rule">Coefficient rule (<a target="_blank" href="
22     https://github.com/ftisunpar/Sharif-Judge/blob/docs/v1.4/add_assignment.md#
23     coefficient-rule">?</a>)</label><br>
24 <span class="form_comment medium clear" style="display: block;">PHP script
25     without <?php ?> tags</span>
26 <textarea id="form_late_rule" name="late_rule" rows="20" class="sharif_input
27     add_text">{{ edit ? edit_assignment.late_rule : set_value('late_rule',
28     default_late_rule) }}</textarea>
29 {{ form_error('late_rule', '<div class="shj_error">', '</div>')}}
30
31 @@ -187,10 +187,10 @@
32 <th rowspan="2">Score</th>
33 <th colspan="3" style="border-bottom: 1px solid #BDBDBD">Time Limit (ms)</th>
34 <th rowspan="2">Memory<br>Limit (kB)</th>
35 -     <th rowspan="2">Allowed<br>Languages (<a target="_blank" href="https://
36     github.com/mjnaderi/Sharif-Judge/blob/docs/v1.4/add_assignment.md#allowed-
37     languages">?</a>)</th>
38 -     <th rowspan="2">Diff<br>Command (<a target="_blank" href="https://github.
39     com/mjnaderi/Sharif-Judge/blob/docs/v1.4/add_assignment.md#diff-command">?</a
40     >)</th>
41 -     <th rowspan="2">Diff<br>Argument (<a target="_blank" href="https://github.
42     com/mjnaderi/Sharif-Judge/blob/docs/v1.4/add_assignment.md#diff-arguments">?</

```



```

1      a>></th>
2      -      <th rowspan="2">Upload<br>Only (<a target="_blank" href="https://github.
3      com/mjnaderi/Sharif-Judge/blob/docs/v1.4/add_assignment.md#upload-only">?</a>)
4      </th>
5      +      <th rowspan="2">Allowed<br>Languages (<a target="_blank" href="https://
6      github.com/ftisunpar/Sharif-Judge/blob/docs/v1.4/add_assignment.md#allowed-
7      languages">?</a></th>
8      +      <th rowspan="2">Diff<br>Command (<a target="_blank" href="https://github.
9      com/ftisunpar/Sharif-Judge/blob/docs/v1.4/add_assignment.md#diff-command">?</a
10     ></th>
11     +      <th rowspan="2">Diff<br>Argument (<a target="_blank" href="https://github.
12     com/ftisunpar/Sharif-Judge/blob/docs/v1.4/add_assignment.md#diff-arguments
13     ">?</a></th>
14     +      <th rowspan="2">Upload<br>Only (<a target="_blank" href="https://github.
15     com/ftisunpar/Sharif-Judge/blob/docs/v1.4/add_assignment.md#upload-only">?</a
16     ></th>
17     <th rowspan="2"></th>
18     </tr>
19     <tr>
20

```

5.4.5 Hapus Perbedaan antara *Admin* dan *Student* pada PDF *Download*

Issue dengan kode unik #7 mengatakan agar mengubah beberapa bagian hasil implementasi kode pada kelas *controller Assignments.php*. Kode tersebut terdapat pada fungsi unduh *assignment* dengan ketentuan yang telah dirancang pada sub bab 4.3. Bagian kode `$this->user->level == 0` berfungsi untuk mengizinkan pengguna dengan *role admin* dapat mengunduh *assignment*. Bagian tersebut dihilangkan agar adanya transparansi antara dosen dan peserta, sehingga dosen saat membuat *assignment* lebih yakin bahwa soalnya tidak akan bisa diunduh. Listing 5.7 menunjukkan perubahan kode program yang dilakukan di *Assignment.php*

Listing 5.7: Perubahan kode program pada *Assignments.php*

```

29
30 @@ -111,13 +111,13 @@ class Assignments extends CI_Controller
31 $pdf_files = glob($pattern);
32     if ( ! $pdf_files )
33         show_error("File not found");
34 -     elseif (!$this->assignment_model->assignment_info($assignment_id)['open']
35         && $this->user->level == 0 )
36 +     elseif (!$this->assignment_model->assignment_info($assignment_id)['open
37         '])
38         show_error('Selected assignment has been closed. ');
39     elseif ( ! $this->assignment_model->is_participant($this->
40     assignment_model->assignment_info($assignment_id)['participants'], $this->user
41     ->username) )

```

```

1         show_error('You are not registered for submitting.');
```

```

2 -         elseif ( shj_now() > $finishtime + $extratime && $this->user->level == 0
3     )
4 +         elseif ( shj_now() > $finishtime + $extratime)
5             show_error('Selected assignment has finished.');
```

```

6 -         elseif ( shj_now() < $starttime && $this->user->level == 0 )
7 +         elseif ( shj_now() < $starttime)
8             show_error('Selected assignment has not started.');
```

```

9
10 // Download the file to browser
11
```

5.4.6 Download Excel Tidak Berfungsi pada Halaman *Submission*

Salah satu asisten dosen mata kuliah Desain & Analisis Algoritma mengatakan bahwa fitur untuk mengunduh *excel* pada halaman *Submission* tidak berfungsi. Perangkat lunak *Sharif Judge* yang terkini memiliki fitur yang dapat mengunduh *excel* pada halaman *All Submission*, *Final Submission* dan *Users*. Fitur ini berfungsi untuk memuat seluruh daftar *All Submission*, *Final Submission* dan *Users* ke dalam format *excel*. Agar fitur ini dapat berjalan dengan baik, *Sharif Judge* menggunakan *library* bantuan yaitu *PHPExcel*.

Persoalan di atas terjadi karena versi PHP yang digunakan tidak lagi mendukung *library PHPExcel*. Dalam pengembangannya, *PHPExcel* sudah tidak lagi digunakan. Para pengguna disarankan untuk migrasi ke *library* penerusnya yaitu *PhpSpreadsheet* atau alternatif lainnya¹.

Persoalan di atas dapat diselsaikan dengan mengubah *library PHPExcel* menjadi *PhpSpreadsheet* sehingga fitur mengunduh *excel* pada halaman *All Submission*, *Final Submission* dan *Users* dapat berfungsi kembali. *PhpSpreadsheet* adalah *library* yang ditulis dalam PHP. *Library PhpSpreadsheet* menyediakan sekumpulan kelas yang memungkinkan pengguna untuk membaca dan menulis ke berbagai format *file spreadsheet*, seperti *Excel* dan *LibreOffice Calc* [9]. Menginstall *library PhpSpreadsheet* dapat dilakukan dengan menggunakan *Composer* dan menjalankan perintah "*composer require phpoffice/phpspreadsheet*".

Rancangan algoritma yang digunakan untuk mengubah *library PHPExcel* menjadi *PhpSpreadsheet* yaitu

1. Menginstall *Composer* pada perangkat lunak *Sharif Judge*.
2. Menambahkan *library PhpSpreadsheet* menggunakan *Composer*.
3. Mengubah fungsi yang menggunakan *library PHPExcel* menjadi menggunakan *library PhpSpreadsheet*

Dari rancangan algoritma yang diterapkan, terdapat perubahan kode pada kelas *controller Submissions.php* dan *controller Users.php*. Perubahan kode di *controller Submissions.php* dan

¹Adrien Crivelli, "PHPExcel - DEPRECATED" terakhir diubah 25 Desember 2017. <https://github.com/PHPOffice/PHPExcel>

1 *controller Users.php* untuk mengubah fungsi yang menggunakan kelas *PHPExcel* menjadi menggu-
 2 nakan kelas *PhpSpreadsheet*. Listing 5.8 menunjukkan perubahan kode program yang dilakukan di
 3 *Submission.php*

Listing 5.8: Perubahan kode program pada *Submission.php*

```

4
5 @@ -6,6 +6,13 @@
6 */
7 defined('BASEPATH') OR exit('No direct script access allowed');
8
9 + use PhpOffice\PhpSpreadsheet\Spreadsheet;
10 + use PhpOffice\PhpSpreadsheet\IOFactory;
11 + use PhpOffice\PhpSpreadsheet\Writer\Xlsx;
12 + use PhpOffice\PhpSpreadsheet\Style\Fill;
13 + use PhpOffice\PhpSpreadsheet\Style\Border;
14 + use PhpOffice\PhpSpreadsheet\Style\Alignment;
15 +
16 class Submissions extends CI_Controller
17 {
18
19 @@ -56,10 +63,10 @@ class Submissions extends CI_Controller
20 $now = shj_now_str(); // current time
21
22 // Load PHPEXcel library
23 - $this->load->library('phpexcel');
24 + $phpspreadsheet = new Spreadsheet();
25
26 // Set document properties
27 - $this->phpexcel->getProperties()->setCreator('SharIF Judge')
28 + $phpspreadsheet->getProperties()->setCreator('SharIF Judge')
29 ->setLastModifiedBy('SharIF Judge')
30 ->setTitle('SharIF Judge Users')
31 ->setSubject('SharIF Judge Users')
32
33 @@ -69,8 +76,8 @@ class Submissions extends CI_Controller
34 $output_filename = 'judge_'. $view.'_submissions';
35
36 // Set active sheet
37 - $this->phpexcel->setActiveSheetIndex(0);
38 - $sheet = $this->phpexcel->getActiveSheet();
39 + $phpspreadsheet->setActiveSheetIndex(0);
40 + $sheet = $phpspreadsheet->getActiveSheet();
41
42 // Add current assignment, time, username filter, and problem filter to document

```

```

1 $sheet->fromArray(array('Assignment:', $this->user->selected_assignment['name']),
2     null, 'A1', true);
3
4 @@ -97,7 +104,7 @@ class Submissions extends CI_Controller
5 $sheet->getStyle('A6:'.$highest_column.'6')->applyFromArray(
6     array(
7         'fill' => array(
8             - 'type' => PHPExcel_Style_Fill::FILL_SOLID,
9             + 'fillType' => Fill::FILL_SOLID,
10            'color' => array('rgb' => '173C45')
11        ),
12        'font' => array(
13
14 @@ -182,7 +189,7 @@ class Submissions extends CI_Controller
15 $sheet->getStyle('A'.$i.':'.$highest_column.$i)->applyFromArray(
16     array(
17         'fill' => array(
18             - 'type' => PHPExcel_Style_Fill::FILL_SOLID,
19             + 'fillType' => Fill::FILL_SOLID,
20            'color' => array('rgb' => (($i%2)?'F0F0F0':'FAFAFA'))
21        )
22    )
23
24 @@ -192,7 +199,7 @@ class Submissions extends CI_Controller
25 // Set text align to center
26 $sheet->getStyle( $sheet->calculateWorksheetDimension() )
27 ->getAlignment()
28 - ->setHorizontal(PHPExcel_Style_Alignment::HORIZONTAL_CENTER);
29 + ->setHorizontal(Alignment::HORIZONTAL_CENTER);
30
31 // Making columns autosize
32 for ($i=2;$i<count($header);$i++)
33
34 @@ -203,7 +210,7 @@ class Submissions extends CI_Controller
35 array(
36     'borders' => array(
37         'outline' => array(
38             - 'style' => PHPExcel_Style_Border::BORDER_THIN,
39             + 'borderStyle' => Border::BORDER_THIN,
40            'color' => array('rgb' => '444444'),
41        ),
42    )

```

```

1
2 @@ -219,7 +226,7 @@ class Submissions extends CI_Controller
3 header('Content-Type: application/vnd.openxmlformats-officedocument.spreadsheetml.
4     sheet');
5 header('Content-Disposition: attachment;filename="'. $output_filename. ' '.$ext
6     .'");
7 header('Cache-Control: max-age=0');
8 - $objWriter = PHPExcel_IOFactory::createWriter($this->phpexcel, ($ext==='xlsx
9     '?'Excel2007':'Excel5'));
10 + $objWriter = IOFactory::createWriter($phpspreadsheet, ucfirst($ext));
11 $objWriter->save('php://output');
12 }
13

```

14

15 Listing 5.9 menunjukan perubahan kode program yang dilakukan di *Users.php*

Listing 5.9: Perubahan kode program pada *Users.php*

```

16
17 @@ -6,6 +6,13 @@
18 */
19 defined('BASEPATH') OR exit('No direct script access allowed');
20
21 + use PhpOffice\PhpSpreadsheet\Spreadsheet;
22 + use PhpOffice\PhpSpreadsheet\IOFactory;
23 + use PhpOffice\PhpSpreadsheet\Writer\Xlsx;
24 + use PhpOffice\PhpSpreadsheet\Style\Fill;
25 + use PhpOffice\PhpSpreadsheet\Style\Border;
26 + use PhpOffice\PhpSpreadsheet\Style\Alignment;
27 +
28 class Users extends CI_Controller
29 {
30
31 @@ -142,10 +149,10 @@ class Users extends CI_Controller
32 $now = shj_now_str(); // current time
33
34 // Load PHPExcel library
35 - $this->load->library('phpexcel');
36 + $phpspreadsheet = new Spreadsheet();
37
38 // Set document properties
39 - $this->phpexcel->getProperties()->setCreator('SharIF Judge')
40 + $phpspreadsheet->getProperties()->setCreator('SharIF Judge')
41 ->setLastModifiedBy('SharIF Judge')
42 ->setTitle('SharIF Judge Users')

```

```

1 ->setSubject('SharIF Judge Users')
2
3 @@ -155,8 +162,8 @@ class Users extends CI_Controller
4 $output_filename = 'sharifjudge_users';
5
6 // Set active sheet
7 - $this->phpexcel->setActiveSheetIndex(0);
8 - $sheet = $this->phpexcel->getActiveSheet();
9 + $phpspreadsheet->setActiveSheetIndex(0);
10 + $sheet = $phpspreadsheet->getActiveSheet();
11
12 // Add current time to document
13 $sheet->fromArray(array('Time:',$now), null, 'A1', true);
14
15 @@ -170,7 +177,7 @@ class Users extends CI_Controller
16 $sheet->getStyle('A3:'. $highest_column.'3')->applyFromArray(
17 array(
18 'fill' => array(
19 - 'type' => PHPExcel_Style_Fill::FILL_SOLID,
20 + 'fillType' => Fill::FILL_SOLID,
21 'color' => array('rgb' => '173C45')
22 ),
23 'font' => array(
24
25 @@ -205,7 +212,7 @@ class Users extends CI_Controller
26 $sheet->getStyle('A'.$i.':'. $highest_column.$i)->applyFromArray(
27 array(
28 'fill' => array(
29 - 'type' => PHPExcel_Style_Fill::FILL_SOLID,
30 + 'fillType' => Fill::FILL_SOLID,
31 'color' => array('rgb' => (($i%2)?'F0F0F0':'FAFAFA'))
32 )
33 )
34
35 @@ -215,7 +222,7 @@ class Users extends CI_Controller
36 // Set text align to center
37 $sheet->getStyle( $sheet->calculateWorksheetDimension() )
38 ->getAlignment()
39 - ->setHorizontal(PHPExcel_Style_Alignment::HORIZONTAL_CENTER);
40 + ->setHorizontal(Alignment::HORIZONTAL_CENTER);
41
42 // Making columns autosize

```

```

1  for ($i=2;$i<count($header);$i++)
2
3  @@ -226,7 +233,7 @@ class Users extends CI_Controller
4  array(
5  'borders' => array(
6  'outline' => array(
7      'style' => PHPExcel_Style_Border::BORDER_THIN,
8      'borderStyle' => Border::BORDER_THIN,
9  'color' => array('rgb' => '444444'),
10 ),
11 )
12
13 @@ -244,9 +251,9 @@ class Users extends CI_Controller
14 header('Content-Type: application/vnd.openxmlformats-officedocument.spreadsheetml.
15     sheet');
16 header('Content-Disposition: attachment;filename="'. $output_filename.'.'.$ext
17     .'");
18 header('Cache-Control: max-age=0');
19 -     $objWriter = PHPExcel_IOFactory::createWriter($this->phpexcel, ($ext==='
20     xlsx'? 'Excel2007': 'Excel5'));
21 +     $objWriter = IOFactory::createWriter($phpspreadsheet, ucfirst($ext));
22 $objWriter->save('php://output');
23 }
24

```

Setelah melakukan perubahan pada beberapa kelas, fungsi unduh *excel* pada halaman *All Submission*, *Final Submission* dan *Users* dapat berjalan dengan baik. *File excel* yang diunduh bisa dibuka menggunakan *Microsoft Excel 365*. [Gambar 5.32](#) menunjukkan isi dari *excel judge_all_submissions.xlsx* yang diunduh dari halaman *All Submission*. [Gambar 5.33](#) menunjukkan isi dari *excel sharifjudge_users.xlsx* yang diunduh dari halaman *Users*.

PROTECTED VIEW Be careful—files from the Internet can contain viruses. Unless you need to edit, it's safer to stay in Protected View. Enable Editing

Assignment:

	A	B	C	D	E	F	G	H	I	J	K
1	ssignment	09stack(L)									
2	Time:	-05-03 22:46:34									
3	name Fil	No filter									
4	blem Fil	No filter									
5											
6	Final	Submit ID	Username	Name	Problem	Submit Time	Score	Delay (HH:MM)	Coefficient	Final Score	Language
7	*	358	i17034	Christopher William Sanjaya	3 (Postfix)	2018-04-08 22:04:43	100	No Delay	100	100	Java
8		357	i17034	Christopher William Sanjaya	3 (Postfix)	2018-04-08 20:58:24	40	No Delay	100	40	Java
9	*	356	i17006	Yovannanta Jong	3 (Postfix)	2018-04-05 22:16:54	60	No Delay	100	60	Java
10	*	355	i17006	Yovannanta Jong	2 (Tanda Kurung)	2018-04-05 21:50:44	100	No Delay	100	100	Java
11	*	354	i17043	Kelvin Adrian Darmawan	3 (Postfix)	2018-04-05 20:49:13	100	No Delay	100	100	Java
12		353	i17043	Kelvin Adrian Darmawan	3 (Postfix)	2018-04-05 20:47:48	0	No Delay	100	0	Java
13	*	352	i17036	Reynard Rafferty Susilo	3 (Postfix)	2018-04-05 19:19:23	60	No Delay	100	60	Java
14		351	i17036	Reynard Rafferty Susilo	3 (Postfix)	2018-04-05 19:16:12	0	No Delay	100	0	Java
15	*	350	i16040	Frederick	1 (Tumpukan Kartu)	2018-04-05 13:19:33	100	No Delay	100	100	Java
16		349	i16040	Frederick	1 (Tumpukan Kartu)	2018-04-05 13:16:26	0	No Delay	100	0	Java
17	*	348	i15065	Kenny Gohan Siahaan	2 (Tanda Kurung)	2018-04-05 13:08:32	60	No Delay	100	60	Java
18	*	347	i15064	Johanna Prima Christanti	3 (Postfix)	2018-04-05 13:03:37	0	No Delay	100	0	Java
19	*	346	i17010	William Oktavianus Kurniawan	3 (Postfix)	2018-04-05 12:37:42	100	No Delay	100	100	Java
20	*	345	i17074	Yosef Panji Pangestu	2 (Tanda Kurung)	2018-04-05 11:45:22	60	No Delay	100	60	Java
21		344	i17074	Yosef Panji Pangestu	2 (Tanda Kurung)	2018-04-05 11:43:59	60	No Delay	100	60	Java
22	*	343	i16021	Regen Renaldo	3 (Postfix)	2018-04-05 10:44:51	0	No Delay	100	0	Java
23	*	342	i16055	Antonius Susanto	3 (Postfix)	2018-04-05 10:36:02	100	No Delay	100	100	Java
24		341	i16055	Antonius Susanto	3 (Postfix)	2018-04-05 10:27:38	100	No Delay	100	100	Java
25		340	i16055	Antonius Susanto	3 (Postfix)	2018-04-05 10:18:45	100	No Delay	100	100	Java
26		339	i16055	Antonius Susanto	3 (Postfix)	2018-04-05 10:18:27	0	No Delay	100	0	Java

Gambar 5.32: Isi Excel *judge_all_submissions.xlsx*

PROTECTED VIEW Be careful—files from the Internet can contain viruses. Unless you need to edit, it's safer to stay in Protected View. Enable Editing

Time:

	A	B	C	D	E	F	G	H	I	J
1	Time:	-05-03 22:46:46								
2										
3	#	User ID	Username	Display Name	Email	Role	First Login	Last Login		
4	1	1	admin		it@unpar.ac.id	admin	2018-01-21 17:38:30	2018-04-28 00:15:54		
5	2	3	stillmen	Stillmen Vallian	stillmen.v@gmail.com	admin	2018-01-26 11:22:51	2018-05-03 16:55:39		
6	3	5	pascal	Pascal	pascal@unpar.ac.id	admin	2018-02-01 08:43:58	2018-05-03 13:08:44		
7	4	102	husnulhakim	Husnul Hakim	husnulhakim@unpar.ac.id	admin	2018-01-31 10:35:16	2018-05-03 13:01:32		
8	5	103	vania	Vania	vania@unpar.ac.id	admin	2018-02-01 10:24:16	2018-05-03 13:48:07		
9	6	104	natalia	Natalia	natalia@unpar.ac.id	admin	2018-02-01 10:24:10	2018-05-03 22:03:41		
10	7	105	marshal	Marshal	marshal@unpar.ac.id	admin	2018-02-06 09:05:02	2018-04-17 09:02:06		
11	8	106	i14056	Andre	7314056@student.unpar.ac.id	instructor	2018-03-18 13:53:21	2018-03-18 13:53:21		
12	9	107	i16012	Apsari Ayusya Cantika	7316012@student.unpar.ac.id	instructor	2018-04-05 10:23:33	2018-04-10 07:44:15		
13	10	108	i15051	Hengky Suryahanadi	7315051@student.unpar.ac.id	instructor	Never	Never		
14	11	109	i15057	Henry Wijaya	7315057@student.unpar.ac.id	instructor	2018-03-19 19:55:13	2018-03-19 19:55:13		
15	12	110	i15009	Himawan Saputrautama	7315009@student.unpar.ac.id	instructor	2018-03-19 18:07:12	2018-04-12 13:16:40		
16	13	111	i16070	Irwan Hardyanto	7316070@student.unpar.ac.id	instructor	2018-04-10 10:04:59	2018-05-03 13:36:10		
17	14	112	i16082	Ivan Kristanto TEST	7316082@student.unpar.ac.id	instructor	2018-03-18 21:34:23	2018-03-20 09:07:05		
18	15	113	i16010	Kevin Renaldi Halim	7316010@student.unpar.ac.id	instructor	2018-04-17 21:46:24	2018-04-30 09:17:46		
19	16	114	i15048	Vincent Joelsinatra	7315048@student.unpar.ac.id	instructor	Never	Never		
20	17	2	i16011	(٧٠٠) ٧٠٠٠٠٠	7316011@student.unpar.ac.id	student	2018-01-21 17:44:36	2018-02-05 10:32:06		
21	18	6	i12090	Delta Dirgantara Riyadi Putra	7312090@student.unpar.ac.id	student	2018-02-01 13:11:55	2018-04-19 13:07:18		
22	19	7	i13030	Fadel Muhammed Amien	7313030@student.unpar.ac.id	student	2018-02-01 13:14:45	2018-05-03 13:07:29		
23	20	8	i13031	Alvin Kresna Wibisono	7313031@student.unpar.ac.id	student	2018-01-29 16:44:03	2018-05-03 13:06:08		
24	21	9	i13053	Soluturon Fernando	7313053@student.unpar.ac.id	student	2018-02-01 13:11:56	2018-04-19 13:21:49		
25	22	10	i13063	Alinna Margaretha	7313063@student.unpar.ac.id	student	2018-02-01 13:11:46	2018-05-03 13:08:38		
26	23	11	i14019	Michael Stefanus Walah	7314019@student.unpar.ac.id	student	2018-01-30 09:48:41	2018-05-03 13:22:16		

Gambar 5.33: Isi Excel *sharifjudge_users.xlsx*

5.5 Masukan dari Pengguna *GitHub*

- Pada saat skripsi ini dibuat, seluruh perkembangan perangkat lunak *SharIF Judge* dicommit pada repositori *GitHub*. Repositori yang digunakan bersifat *public*, sehingga seluruh pengguna *GitHub* dapat mengakses repositori tersebut. Pengguna *GitHub* dengan *username* *wojck13* memberikan masukan dengan membuat *issue* dengan judul "*Some development proposals*" dan berkode unik #6.

Berikut isi masukan dari *username* wojcik13

“Hello,

As I see, you’re developing and adding some functions to Sharif Judge, well, I’m newbie here and I don’t know how to add some functions to repo. If You will have time to add them, I would be grateful.

So Ideas as following:

- *IMO Sharif Judge should have ability to freeze scoreboard, as it is in other OJ’s (DOMJudge). Usually in contest organized in Poland, it’s enabled in half of contest time.*
- *As far as I’m concerned, Sharif should have better developed permissions system. I mean, to make some groups of students, so student from group A, cannot see contest assigned to group B, and vice versa. (My idea is to add students categories, and in assignment_edit view change participants box to Categories of students allowed to see contest)*
- *What do You think about adding switch, which is changing judging-method? One method is already programmed by mjnaderi, and the second method is ICPC-based*
- *Last suggestion is to precisely set publishing contests, so at setted time contest become visible, and deactivation time, so contest become invisible.*

I’m curious, what is Your opinion about my proposals.

Kindly regards

Wojtek Wasilewski”

BAB 6

KESIMPULAN DAN SARAN

Bab ini membahas tentang kesimpulan berdasarkan hasil dari analisis, implementasi dan pengujian perangkat lunak yang telah dibuat, serta saran-saran untuk pengembangan selanjutnya.

6.1 Kesimpulan

Berdasarkan hasil dari analisis, implementasi dan pengujian perangkat lunak *SharIF Judge* yang telah dibuat, maka diperoleh kesimpulan sebagai berikut:

1. Perangkat lunak *Sharif Judge* telah dikembangkan dengan mengimplementasi kebutuhan-kebutuhan yang dapat dilihat pada [Tabel 3.1](#). Kebutuhan-kebutuhan tersebut yaitu:
 - Membatasi pengaksesan soal (deskripsi & PDF)
 - Mensupport *file* dengan ekstensi TXT
 - Menambahkan parameter "*Display Name*" pada pendaftaran peserta
 - Menambahkan fitur "*Lock Student's Display Name*"
 - Menambahkan fitur "*Archived Assignment*"
 - Menambahkan halaman *Logs*
 - Menambahkan halaman *Hall of Fame*
 - Integrasi *login* ke *server RADIUS*
 - Branding Teknik Informatika
 - Mengubah *library PHPExcel* menjadi *library PHPSpreadsheet*
 - Mengubah *method shell_exec("rm ...")* menjadi *unlink()*
 - Menambahkan *method* rekoneksi ke *database*
2. Sudah berhasil diimplementasikan 12 dari 13 fitur yang ditentukan. Fitur *upload file* dengan ekstensi JAR tidak diimplementasi, dikarenakan keterbatasan waktu.
3. Struktur *database* yang lama tidak sesuai dengan *Sharif Judge* hasil implementasi, sehingga diperlukan pembaruan struktur database.

1 **6.2** **Saran**

2 Berdasarkan hasil pengembangan yang dilakukan, berikut adalah saran-saran untuk pengembangan
3 selanjutnya:

- 4 1. Mengimplementasi fitur *upload file* dengan ekstensi JAR (*Java multi class*).
- 5 2. Mengimplementasi fitur hasil dari masukan pengguna *GitHub* yang terdapat pada [sub bab 5.5](#).

DAFTAR REFERENSI

- [1] Wasik, S., Antczak, M., Badura, J., Laskowski, A., dan Sternal, T. (2018) A survey on online judge systems and their applications. *ACM Computing Surveys*, **51**, 3:1–3:34.
- [2] Naderi, M. J. (2014) Sharif judge. <https://github.com/mjnaderi/Sharif-Judge/>. 6 Oktober 2017.
- [3] of Technology, B. C. I. (2017) Codeigniter documentation. https://codeigniter.com/user_guide/overview/index.html. 6 Oktober 2017.
- [4] Naderi, M. J. (2014) Sharif judge documentation. <https://github.com/mjnaderi/Sharif-Judge/tree/docs/v1.4>. 6 Oktober 2017.
- [5] Potencier, F. (2009) The flexible, fast, and secure template engine for php. <https://twig.symfony.com/>. 12 Maret 2018.
- [6] Aiken, A. (2010) A system for detecting software similarity. <http://theory.stanford.edu/~aiken/moss/>. 25 Februari 2018.
- [7] Smith, B. (2009) The gnu operating system and the free software movement. <https://www.gnu.org/>. 22 Februari 2018.
- [8] Phillips, D. (2016) Dappphp - radius - a pure php radius client based on the sysco/al implementation. <https://github.com/dappphp/radius>. 10 April 2018.
- [9] PHPOffice (2010) Phpspreadsheet - a pure php library for reading and writing spreadsheet files. <https://github.com/PHPOffice/PhpSpreadsheet>. 10 April 2018.

LAMPIRAN A

KODE PROGRAM HALAMAN *24-HOUR LOGS*

Kode Program kelas *model* halaman *24-hour Logs*.

Listing A.1: *Logs_model.php*

```
1 <?php
2 /**
3  * SharIF Judge online judge
4  * @file Logs_model.php
5  * @author Stillmen Vallian <stillmen.v@gmail.com>
6  */
7 defined('BASEPATH') OR exit('No direct script access allowed');
8
9 class Logs_model extends CI_Model
10 {
11     public function __construct()
12     {
13         parent::__construct();
14     }
15
16     // -----
17
18     /**
19      * Mencatat Logs (Tabel shj_logins)
20      *
21      *
22      *
23      */
24     public function insert_to_logs($username, $ip_address)
25     {
26         $query = $this->db->get('logins')->result_array();
27
28         //menghapus timestamp user yang lebih dari 24 jam
29         foreach ($query as $row)
30         {
31             $temptime=strtotime('+24_hour', strtotime($row['timestamp']));
32             if ($temptime < shj_now()) {
33                 # delete
34                 $this->db->where('timestamp', $row['timestamp']);
35                 $this->db->delete('logins');
36             }
37         }
38
39         $result = $this->db->query("SELECT_*_FROM_shj_logins_WHERE_username='".$username."'_AND_ip_address!='".$ip_address."'_
40                                ORDER_BY_timestamp_DESC")->row();
41         if ($result == NULL) {
42             $logins = array(
43                 'username' => $username,
44                 'ip_address' => $ip_address
45             );
46             $this->db->insert('logins', $logins);
47         }
48         else{
49             $get_last_login_id = $result -> login_id;
50             $logins = array(
51                 'username' => $username,
52                 'ip_address' => $ip_address,
53                 'last_24h_login_id' => $get_last_login_id
54             );
55             $this->db->insert('logins', $logins);
56         }
57     }
58
59     // -----
60
61     /**
62      * Get All Logs
63      *
64      * Returns an array of all logs
65      *
66      * @return mixed
67      */
68     public function get_all_logs()
69     {
70         return $this->db->order_by('login_id', 'desc')->get('logins')->result_array();
71     }
72 }
```

Kode Program kelas *view* halaman 24-hour Logs.

Listing A.2: *logs.twig*

```

1  {#
2  # SharIF Judge
3  # file: logs.twig
4  # author: Stillmen Vallian <stillmen.v@gmail.com>
5  #}
6  {% set selected = 'logs' %}
7  {% extends 'templates/base.twig' %}
8  {% block icon %}fa-book{% endblock %}
9  {% block title %}24-hour Log{% endblock %}
10 {% block head_title %}24-hour Log{% endblock %}
11
12
13
14 {% block main_content %}
15
16 <p style="text-align:justify">Use this table to detect account lendings between students in a pre-seated exam environment. In a
    pre-seated environment, student should have logged in from only one IP address within the exam time period. Whenever a user
    logs in from different IP address in the past 24 hours, last column of this table will show link to the previous login that
    happened. In such case, proctor can analyse from those two logins, whether the same login has been used in different devices
    (meaning the account had been lent).</p>
17
18 <table class="sharif_table">
19   <thead>
20     <tr>
21       <th>#</th>
22       <th>Login ID</th>
23       <th>Username</th>
24       <th>IP Address</th>
25       <th>Login Time</th>
26       <th>Log from different IP (< 24 hours)</th>
27     </tr>
28   </thead>
29   {% for log in logs %}
30     <tr>
31       <td>{{ loop.index }}</td>
32       <td id="{{ log.login_id }}">{{ log.login_id }}</td>
33       <td>{{ log.username }}</td>
34       <td>{{ log.ip_address }}</td>
35       <td>{{ log.timestamp }}</td>
36       <td><a href="#{{ log.last_24h_login_id }}">{{ log.last_24h_login_id }}</a></td>
37     </tr>
38   {% endfor %}
39 </table>
40 {% endblock %} {# main_content #}

```

Kode Program kelas *controller* halaman 24-hour Logs.

Listing A.3: *Logs.php*

```

1  <?php
2  /**
3   * SharIF Judge online judge
4   * @file Logs.php
5   * @author Stillmen Vallian <stillmen.v@gmail.com>
6   */
7  defined('BASEPATH') OR exit('No direct script access allowed');
8
9  class Logs extends CI_Controller
10 {
11
12     public function __construct()
13     {
14         parent::__construct();
15         if ( ! $this->session->userdata('logged_in')) // if not logged in
16             redirect('login');
17         if ( $this->user->level <= 2) // permission denied
18             show_404();
19     }
20
21     // -----
22
23     public function index()
24     {
25
26         $data = array(
27             'logs' => $this->logs_model->get_all_logs()
28         );
29
30         $this->twig->display('pages/admin/logs.twig', $data);
31     }
32
33     // -----
34
35 }

```


LAMPIRAN B

KODE PROGRAM HALAMAN *HALL OF FAME*

Kode Program kelas *model* halaman *Hall of Fame*.

Listing B.1: *Hof_model.php*

```

1 <?php
2 /**
3  * SharIF Judge online judge
4  * @file Hof_model.php
5  * @author Stillmen Vallian <stillmen.v@gmail.com>
6  */
7 defined('BASEPATH') OR exit('No direct script access allowed');
8
9 class Hof_model extends CI_Model
10 {
11     public function __construct()
12     {
13         parent::__construct();
14     }
15
16     // -----
17
18     /**
19      * Get data for Hall of Fame
20      *
21      * @return mixed
22      */
23     public function get_all_final_submission()
24     {
25         $this->load->model('user_model');
26         //include upload only file: remove " AND file_type!='txt' AND file_type!='pdf' AND file_type!='zip' ";
27         $datas = $this->db->query("SELECT username, CEILING(SUM(pre_score*_coefficient_/10000)) AS totalscore FROM shj_submissions_
28             WHERE is_final=1 AND file_type!='txt' AND file_type!='pdf' AND file_type!='zip' GROUP BY username ORDER BY totalscore_
29             DESC")->result_array();
30         foreach ($datas as $key => $data) {
31             $user_id = $this->user_model->username_to_user_id($data['username']);
32             $datas[$key]['display_name'] = $this->user_model->get_user($user_id)->display_name;
33         }
34         return $datas;
35     }
36
37     // -----
38
39     /**
40      * Get details assignments & problems for selected user
41      *
42      * @return mixed
43      */
44     public function get_all_user_assignments($username)
45     {
46         $this->load->model('assignment_model');
47         $details = $this->db->query("SELECT assignment, problem, CEILING((pre_score*_coefficient_/10000)) AS score FROM
48             shj_submissions WHERE is_final=1 AND username='$username' AND file_type!='txt' AND file_type!='pdf' AND file_type!='
49             zip' ORDER BY assignment ASC")->result_array();
50         foreach ($details as $key => $detail) {
51             $assignment_id = $detail['assignment'];
52             $problem_id = $detail['problem'];
53             $details[$key]['assignment'] = $this->assignment_model->assignment_info($assignment_id)['name'];
54             $details[$key]['scoreboard'] = $this->assignment_model->assignment_info($assignment_id)['scoreboard'];
55             $details[$key]['problem'] = $this->assignment_model->problem_info($assignment_id, $problem_id)['name'];
56         }
57         return $details;
58     }
59 }
60
61 }
```

Kode Program kelas *view* halaman *Hall of Fame*.

Listing B.2: *halloffame.twig*

```

1 {#
2 # SharIF Judge
```

```

3  # file: halloffame.twig
4  # author: Stillmen Vallian <stillmen.v@gmail.com>
5  #}
6  {% set selected = 'halloffame' %}
7  {% extends 'templates/base.twig' %}
8  {% block icon %}fa fa-list-alt fa-lg{% endblock %}
9  {% block title %}Hall of Fame{% endblock %}
10 {% block head_title %}Hall of Fame{% endblock %}
11 {% block other_assets %}
12   <link rel='stylesheet' type='text/css' href='{{ base_url("assets/snippet/jquery.snippet.css") }}'/>
13   <link rel='stylesheet' type='text/css' href='{{ base_url("assets/snippet/themes/github.css") }}'/>
14   <script type='text/javascript' src='{{ base_url("assets/snippet/jquery.snippet.js") }}'/></script>
15   <link rel='stylesheet' type='text/css' href='{{ base_url("assets/reveal/reveal.css") }}'/>
16   <script type='text/javascript' src='{{ base_url("assets/reveal/jquery.reveal.js") }}'/></script>
17 {% endblock %}
18
19
20 {% block main_content %}
21 <div style="height:15px"></div>
22 <table class="sharif_table">
23   <thead>
24     <tr>
25       <th>#</th>
26       <th>Rank</th>
27       <th>Username</th>
28       <th>Display Name</th>
29       <th>Total Score</th>
30     </tr>
31   </thead>
32   {% set tempTotalScore = 0 %}
33   {% set tempLoop = 0 %}
34   {% for hof in hofs %}
35     <tr class="hof_details" style="cursor:pointer;">
36       <td>{{ loop.index }}</td>
37       {% if loop.index == 1 %}
38         <td>1</td>
39         {% set tempTotalScore = hof.totalscore %}
40         {% set tempLoop = tempLoop+1 %}
41       {% elseif tempTotalScore == hof.totalscore %}
42         <td>{{ tempLoop }}</td>
43       {% else %}
44         {% set tempTotalScore = hof.totalscore %}
45         {% set tempLoop = tempLoop+1 %}
46         <td>{{ tempLoop }}</td>
47       {% endif %}
48       <td class="username">{{ hof.username }}</td>
49       <td class="display_name">{{ hof.display_name }}</td>
50       <td>{{ hof.totalscore }}</td>
51     </tr>
52   {% endfor %}
53 </table>
54 {% endblock %} {# main_content #}
55
56 {% block body_end %}
57 <div id="shj_modal" class="reveal-modal_xlarge">
58   <div class="modal_inside">
59     <div style="text-align:center;">Loading<br><img src='{{ base_url('assets/images/loading.gif') }}'/></div>
60   </div>
61   <a class="close-reveal-modal">&#215;</a>
62 </div>
63 {% endblock %}

```

Kode Program kelas *controller* halaman *Hall of Fame*.

Listing B.3: *Halloffame.php*

```

1  <?php
2  /**
3   * SharIF Judge online judge
4   * @file Halloffame.php
5   * @author Stillmen Vallian <stillmen.v@gmail.com>
6   */
7  defined('BASEPATH') OR exit('No direct script access allowed');
8
9  class Halloffame extends CI_Controller
10 {
11
12     public function __construct()
13     {
14         parent::__construct();
15         if ( ! $this->session->userdata('logged_in')) // if not logged in
16             redirect('login');
17         $this->load->model('hof_model');
18     }
19
20     // -----
21
22     public function index()
23     {
24         $data = array(
25             'hofs' => $this->hof_model->get_all_final_submission()
26         );
27         $this->twig->display('pages/halloffame.twig', $data);
28     }
29
30     // -----
31
32

```

```
33 | /**
34 |  * Controller for shows the details of the score
35 |  * Called by ajax request
36 |  */
37 | public function hof_details()
38 | {
39 |     if ( ! $this->input->is_ajax_request() )
40 |         show_404();
41 |     $username = $this->input->post('username');
42 |
43 |     $json_result = $this->hof_model->get_all_user_assignments($username);
44 |
45 |     $this->output->set_header('Content-Type:_application/json;_charset=utf-8');
46 |     echo json_encode($json_result);
47 | }
48 | }
```


LAMPIRAN C

KODE PROGRAM *SHJ_FUNCTIONS.JS*

Kode Program *shj_functions.js*.

Listing C.1: *shj_functions.js*

```
1  /**
2   * SharIF Judge
3   * @file shj_functions.js
4   * @author Mohammad Javad Naderi <mjnaderi@gmail.com>
5   */
6
7  // These words are used in countdown timer
8  shj.time_words = ['day', 'days', 'hour', 'hours', 'minute', 'minutes', 'second', 'seconds'];
9
10 // selectText is used for "Select All" when viewing a submitted code
11 jQuery.fn.selectText = function(){
12     var doc = document
13         , element = this[0]
14         , range, selection
15         ;
16     if (doc.body.createTextRange) {
17         range = document.body.createTextRange();
18         range.moveToElementText(element);
19         range.select();
20     } else if (window.getSelection) {
21         selection = window.getSelection();
22         range = document.createRange();
23         range.selectNodeContents(element);
24         selection.removeAllRanges();
25         selection.addRange(range);
26     }
27 };
28
29 shj.html_encode = function(value) {
30     return $('<div/>').text(value).html();
31 }
32
33 shj.supports_local_storage = function() {
34     try {
35         return 'localStorage' in window && window['localStorage'] !== null;
36     } catch(e){
37         return false;
38     }
39 }
40
41 shj.loading_start = function()
42 {
43     $('#top_bar,.shj-spinner').css('display', 'block');
44 }
45
46 shj.loading_finish = function()
47 {
48     $('#top_bar,.shj-spinner').css('display', 'none');
49 }
50
51 shj.loading_error = function()
52 {
53     noty({
54         text: 'An_error_encountered_while_processing_your_request._Check_your_network_connection.',
55         layout: 'bottomRight',
56         type: 'error',
57         timeout: 3500
58     });
59 }
60
61 shj.loading_failed = function(message)
62 {
63     noty({
64         text: 'Request_failed._Server_says:_ ' + message,
65         layout: 'bottomRight',
66         type: 'error',
67         timeout: 3500
68     });
69 }
70
71 shj.sync_server_time = function () {
72     $.ajax({
73         type: 'POST',
74         url: shj.site_url + 'server_time',
```

```

75     data: {
76         shj_csrf_token: shj.csrf_token
77     },
78     success: function (response) {
79         shj.offset = moment(response).diff(moment());
80     }
81 });
82 }
83
84 shj.update_clock = function(){
85     if (Math.abs(moment().diff(shj.time))>3500){
86         //console.log('moment: '+moment()+' time: '+time+' diff: '+Math.abs(moment().diff(time)));
87         shj.sync_server_time();
88     }
89     shj.time = moment();
90     var now = moment().add('milliseconds', shj.offset);
91     $('#timer').html('Server_Time: '+now.format('MMM_DD_--HH:mm:ss'));
92     var countdown = shj.finish_time.diff(now);
93     if (countdown<=0 && countdown + shj.extra_time.asMilliseconds()>=0){
94         countdown = countdown + shj.extra_time.asMilliseconds();
95         $('#div#extra_time').css("display","block");
96     }
97     else
98         $('#div#extra_time').css("display","none");
99     if (countdown<=0){
100         countdown=0;
101     }
102
103     countdown = Math.floor(moment.duration(countdown).asSeconds());
104     var seconds = countdown%60; countdown=(countdown-seconds)/60;
105     var minutes = countdown%60; countdown=(countdown-minutes)/60;
106     var hours = countdown%24; countdown=(countdown-hours)/24;
107     var days = countdown;
108     $('#time_days').html(days);
109     $('#time_hours').html(hours);
110     $('#time_minutes').html(minutes);
111     $('#time_seconds').html(seconds);
112     if(days==1)
113         $('#days_label').html(shj.time_words[0]);
114     else
115         $('#days_label').html(shj.time_words[1]);
116     if(hours==1)
117         $('#hours_label').html(shj.time_words[2]);
118     else
119         $('#hours_label').html(shj.time_words[3]);
120     if(minutes==1)
121         $('#minutes_label').html(shj.time_words[4]);
122     else
123         $('#minutes_label').html(shj.time_words[5]);
124     if(seconds==1)
125         $('#seconds_label').html(shj.time_words[6]);
126     else
127         $('#seconds_label').html(shj.time_words[7]);
128 }
129
130 shj.sidebar_open = function(time){
131     if (time==0){
132         $('#.sidebar_text').css('display', 'inline-block');
133         $('##sidebar_bottom_p').css('display', 'block');
134         $('##side_bar').css('width', '173px');
135         $('##main_container').css('left', '173px');
136     }
137     else{
138         $('##side_bar').animate({width: '173px'}, time, function(){
139             $('#.sidebar_text').css('display', 'inline-block');
140             $('##sidebar_bottom_p').css('display', 'block');
141         });
142         $('##main_container').animate({'left': '173px'}, time*1.7);
143     }
144     $('#i#collapse').removeClass("fa-caret-square-o-right");
145     $('#i#collapse').addClass("fa-caret-square-o-left");
146 }
147
148 shj.sidebar_close = function(time){
149     if (time==0){
150         $('#.sidebar_text').css('display', 'none');
151         $('##sidebar_bottom_p').css('display', 'none');
152         $('##side_bar').css('width', '48px');
153         $('##main_container').css('left', '48px');
154     }
155     else{
156         $('#.sidebar_text').css('display', 'none');
157         $('##sidebar_bottom_p').css('display', 'none');
158         $('##side_bar').animate({width: '48px'}, time);
159         $('##main_container').animate({'left': '48px'}, time*1.7);
160     }
161     $('#i#collapse').removeClass("fa-caret-square-o-left");
162     $('#i#collapse').addClass("fa-caret-square-o-right");
163 }
164
165 shj.toggle_collapse = function(){
166     if (shj.sidebar == "open"){
167         shj.sidebar = "close";
168         shj.sidebar_close(200);
169         if (shj.supports_local_storage())
170             localStorage.shj_sidebar = 'close';
171         else
172             $.cookie('shj_sidebar','close',{path: '/', expires: 365});
173     }

```

```

174     else if (shj.sidebar == "close"){
175         shj.sidebar = "open";
176         shj.sidebar_open(200);
177         if (shj.supports_local_storage())
178             localStorage.shj_sidebar = 'open';
179         else
180             $.cookie('shj_sidebar', 'open', {path: '/', expires: 365});
181     }
182 }
183
184
185
186 // Notifications
187 shj.notif_check_time = null;
188 shj.check_notifs = function () {
189     if (shj.notif_check_time == null)
190         shj.notif_check_time = moment().add('milliseconds', shj.offset - (shj.notif_check_delay * 1000));
191     $.ajax({
192         type: 'POST',
193         url: shj.site_url+'notifications/check',
194         data: {
195             time: shj.notif_check_time.format('YYYY-MM-DD_HH:mm:ss'),
196             shj_csrf_token: shj.csrf_token
197         },
198         success: function (data) {
199             if (data == "new_notification") {
200                 noty({
201                     text: 'New_Notification',
202                     layout: 'bottomRight',
203                     type: 'information',
204                     closeWith: ['click', 'button'],
205                     animation: {
206                         open: {height: 'toggle'},
207                         close: {height: 'toggle'},
208                         easing: 'swing',
209                         speed: 300
210                     }
211                 });
212                 alert("New_Notification");
213             }
214         }
215     });
216     shj.notif_check_time = moment().add('milliseconds', shj.offset);
217 }
218
219
220
221
222 /**
223  * Notifications
224  */
225 $(document).ready(function () {
226     $('._ttl_n').click(function(){
227         var id = $(this).parents('.notif').data('id');
228         window.location = shj.site_url+'notifications#number'+id;
229     });
230     $('._edt_n').click(function () {
231         var id = $(this).parents('.notif').data('id');
232         window.location = shj.site_url+'notifications/edit/'+id;
233     });
234     $('._del_n').click(function () {
235         var notif = $(this).parents('.notif');
236         var id = $(notif).data('id');
237         noty({
238             text: 'Are_you_sure_you_want_to_delete_this_notification?',
239             layout: 'center',
240             type: 'confirm',
241             animation: {
242                 open: {height: 'toggle'},
243                 close: {height: 'toggle'},
244                 easing: 'swing',
245                 speed: 300
246             },
247             buttons: [
248                 {addClass: 'btn_shj-red', text: 'Yes,_Delete', onClick: function ($noty) {
249                     $noty.close();
250                     $.ajax({
251                         type: 'POST',
252                         url: shj.site_url + 'notifications/delete',
253                         data: {
254                             id: id,
255                             shj_csrf_token: shj.csrf_token
256                         },
257                         beforeSend: shj.loading_start,
258                         complete: shj.loading_finish,
259                         error: shj.loading_error,
260                         success: function (response) {
261                             if (response.done) {
262                                 notif.animate({backgroundColor: '#FF7676'}, 1000, function () {
263                                     notif.remove();
264                                 });
265                                 noty({text: 'Notification_deleted', layout: 'bottomRight', type: 'success', timeout: 5000});
266                             }
267                             else
268                                 shj.loading_failed(response.message);
269                         }
270                     });
271                 }
272             ]
273         });
274     });
275 }

```

```

273         {addClass: 'btn_shj-blue', text: 'No, Don\'t Delete', onClick: function ($noty) {
274             $noty.close();
275         }}
276     ]
277     });
278 });
279
280 if ( shj.check_for_notifications )
281     window.setInterval(shj.check_notifs, (shj.notif_check_delay*1000));
282
283 });
284
285
286
287
288 /**
289  * Scrollbars
290  */
291 $(document).ready(function(){
292     $('#scroll-wrapper').nanoScroller({
293         contentClass: 'scroll-content'
294     });
295     $('#main_content').resize(function(){
296         // update the scrollbar
297         $('#scroll-wrapper').nanoScroller();
298     });
299     $('#widget_contents_container').resize(function(){
300         // update the scrollbar
301         $('#scroll-wrapper').nanoScroller();
302     });
303 });
304
305
306
307
308 /**
309  * Sidebar
310  */
311 $(document).ready(function () {
312     if (shj.supports_local_storage())
313         shj.sidebar = localStorage.shj_sidebar;
314     else
315         shj.sidebar = $.cookie('shj_sidebar');
316
317     if (shj.sidebar != 'open' && shj.sidebar != 'close') {
318         shj.sidebar = 'open';
319         if (shj.supports_local_storage())
320             localStorage.shj_sidebar = 'open';
321         else
322             $.cookie('shj_sidebar', 'open', {path: '/', expires: 365});
323     }
324     if (shj.sidebar == "open")
325         shj.sidebar_open(0);
326     else
327         shj.sidebar_close(0);
328
329     $("#shj-collapse").click(shj.toggle_collapse);
330
331     // update the clock and countdown timer every 1 second
332     shj.update_clock();
333     window.setInterval(shj.update_clock, 1000);
334 });
335
336
337
338
339
340 /**
341  * Top Bar
342  */
343 $(document).ready(function () {
344     $("#top_bar").hoverIntent({
345         over: function () {
346             $(this).children(".top_menu").show();
347             $(this).addClass('shj-white');
348         },
349         out: function () {
350             $(this).children(".top_menu").hide();
351             $(this).removeClass('shj-white');
352         },
353         selector: '.top-object.shj-menu'
354     });
355     $(".select_assignment").click(
356         function () {
357             var id = $(this).children('i').addBack('i').data('id');
358             $.ajax({
359                 type: 'POST',
360                 url: shj.site_url + 'assignments/select',
361                 data: {
362                     assignment_select: id,
363                     shj_csrf_token: shj.csrf_token
364                 },
365                 beforeSend: shj.loading_start,
366                 complete: shj.loading_finish,
367                 error: shj.loading_error,
368                 success: function (response) {
369                     if (response.done)
370                         {
371                             var checkboxes = $(".select_assignment").children('i').addBack('i');

```



```

372 |         checkboxes.removeClass('fa-check-square-o_color6').addClass('fa-square-o');
373 |         checkboxes.filter("[data-id='" + id + "']").removeClass('fa-square-o').addClass('fa-check-square-o_color6'
374 |         );
375 |         $('<assignment_name>').html($('<top_object_[data-id="' + id + '"]').parents('assignment_block').children('
376 |         .assignment_item').html());
377 |         shj.finish_time = moment(response.finish_time);
378 |         shj.extra_time = moment.duration(parseInt(response.extra_time, 10), 'seconds');
379 |         shj.update_clock();
380 |     }
381 |     else
382 |         shj.loading_failed(response.message);
383 |     }
384 |     });
385 | });
386 |
387 |
388 |
389 |
390 |
391 | /**
392 |  * "Users" page
393 |  */
394 | $(document).ready(function(){
395 |     $('<delete_user>').click(function(){
396 |         var row = $('<this>').parents('tr');
397 |         var user_id = row.data('id');
398 |         var username = row.children('<un>').html();
399 |         noty({
400 |             text: 'Are_you_sure_you_want_to_delete_this_user?<br>User_ID:<+user_id+<br>Username:<+username+<br><i_class="
401 |             splashy-warning_triangle"></i>All_submissions_of_this_user_will_be_deleted.',
402 |             layout: 'center',
403 |             type: 'confirm',
404 |             animation: {
405 |                 open: {height: 'toggle'},
406 |                 close: {height: 'toggle'},
407 |                 easing: 'swing',
408 |                 speed: 300
409 |             },
410 |             buttons: [
411 |                 {addClass: 'btn_shj-red', text: 'Yes,<Delete', onClick: function($noty) {
412 |                     $noty.close();
413 |                     $.ajax({
414 |                         type: 'POST',
415 |                         url: shj.site_url+'users/delete',
416 |                         data: {
417 |                             user_id: user_id,
418 |                             shj_csrf_token: shj.csrf_token
419 |                         },
420 |                         beforeSend: shj.loading_start,
421 |                         complete: shj.loading_finish,
422 |                         error: shj.loading_error,
423 |                         success: function(response){
424 |                             if (response.done)
425 |                             {
426 |                                 row.animate({backgroundColor: 'FF7676'},1000, function(){row.remove();});
427 |                                 noty({text: 'User:<+username+<deleted.', layout:'bottomRight', type: 'success', timeout: 5000});
428 |                             }
429 |                             else
430 |                                 shj.loading_failed(response.message);
431 |                         }
432 |                     });
433 |                 },
434 |                 {addClass: 'btn_shj-blue', text: 'No,<Don't_Delete', onClick: function($noty){$noty.close();}}
435 |             ]
436 |         });
437 |     });
438 |     $('<delete_submissions>').click(function(){
439 |         var row = $('<this>').parents('tr');
440 |         var user_id = row.data('id');
441 |         var username = row.children('<un>').html();
442 |         noty({
443 |             text: 'Are_you_sure_you_want_to_delete_this_user's_submissions?<br>User_ID:<+user_id+<br>Username:<+username,
444 |             layout: 'center',
445 |             type: 'confirm',
446 |             animation: {
447 |                 open: {height: 'toggle'},
448 |                 close: {height: 'toggle'},
449 |                 easing: 'swing',
450 |                 speed: 300
451 |             },
452 |             buttons: [
453 |                 {addClass: 'btn_shj-red', text: 'Yes,<Delete', onClick: function($noty) {
454 |                     $noty.close();
455 |                     $.ajax({
456 |                         type: 'POST',
457 |                         url: shj.site_url+'users/delete_submissions',
458 |                         data: {
459 |                             user_id: user_id,
460 |                             shj_csrf_token: shj.csrf_token
461 |                         },
462 |                         beforeSend: shj.loading_start,
463 |                         complete: shj.loading_finish,
464 |                         error: shj.loading_error,
465 |                         success: function(response){
466 |                             if (response.done)

```

```

467         notify({text: 'Submissions_of_user_'+username+'_deleted_successfully.', layout: 'bottomRight', type:
468             'success', timeout: 5000});
469         else
470             shj.loading_failed(response.message);
471     });
472     },
473     {addClass: 'btn_shj-blue', text: 'No_Don\'t_Delete', onClick: function($noty){$noty.close();}}
474 ],
475     });
476 });
477 });
478 });
479
480
481
482
483 /**
484  * "Hall of Fame" page
485  */
486 shj.modal_open = false;
487 $(document).ready(function(){
488     $('.hof_details').click(function () {
489         var row = $(this).closest("tr"); // Find the row
490         var username = row.find(".username").text();
491         var display_name = row.find(".display_name").text();
492         var view_code_request = $.ajax({
493             cache: true,
494             type: 'POST',
495             url: shj.site_url+'halloffame/hof_details',
496             data: {
497                 username: username,
498                 shj_csrf_token: shj.csrf_token
499             },
500             success: function (response) {
501                 var currentAssignment = '';
502                 var prevAssignment = '';
503                 var temp = '';
504
505                 for (var i = 0; i < response.length; i++) {
506                     if (response[i].scoreboard == 0) {
507                         temp = temp + '-----<br><b>' + response[i].assignment + '</b>'
508                             + '<br>Scoreboard_Disabled!<br>';
509                     }
510                     else{
511                         if (i == 0) {
512                             temp = temp + '<b>' + response[i].assignment + '</b><br>' + response[i].problem + '␣' + response[i]
513                                 .score + '<br>';
514                             prevAssignment = response[i].assignment;
515                         }
516                         else{
517                             currentAssignment = response[i].assignment;
518                             var index = currentAssignment.localeCompare(prevAssignment); //comparing previous assignment's name
519                                 with current assignment's name
520                             if (index == 0) {
521                                 temp = temp + response[i].problem + '␣' + response[i].score + '<br>';
522                                 prevAssignment = currentAssignment;
523                             }
524                             else{
525                                 temp = temp + '-----<br><b>' + currentAssignment + '</b>'
526                                     + '<br>' + response[i].problem + '␣' + response[i].score + '<br>';
527                                 prevAssignment = currentAssignment;
528                             }
529                         }
530                     }
531                 }
532                 temp = temp + '</b>';
533                 $('.modal_inside').html('<pre>code-column'+temp+'</pre>');
534                 $('.modal_inside').prepend('<p><code>Hall_of_Fame_Details␣Username:␣'+username+'␣Display_Name:␣'+display_name
535                     + '</code></p>');
536             }
537         });
538         if (!shj.modal_open) {
539             shj.modal_open = true;
540             $('#shj_modal').reveal({
541                 animationspeed: 300,
542                 on_close_modal: function () {
543                     view_code_request.abort();
544                 },
545                 on_finish_modal: function () {
546                     $(".modal_inside").html('<div>style="text-align:center;">Loading<br><img_src="'+shj.base_url+'assets/
547                         images/loading.gif"/></div>');
548                     shj.modal_open = false;
549                 }
550             });
551         }
552     });
553 });
554
555 /**
556  * Set dir="auto" for all input elements
557  */
558 $(document).ready(function(){
559     $('input').attr('dir', 'auto');
560 });

```

LAMPIRAN D

KODE PROGRAM *SECRETS.PHP*

Kode Program *secrets.php*.

Listing D.1: *secrets.php*

```
1 <?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');
2
3 // Leave empty if you prefer just password.
4 // The password will be authenticated with this orded:
5 // -- saved password from the database
6 // -- your preferred authentication method.
7 $config['shj_authenticate'] = 'radius';
8
9 $config['shj_radius']= [
10     "server" => "127.0.0.1",
11     "secret" => "i-have-no-secret"
12 ];
13 $config['shj_mail'] = [
14     'protocol' => 'smtp',
15     'smtp_host' => 'ssl://smtp.mailgun.org',
16     'smtp_port' => 465,
17     'smtp_user' => '',
18     'smtp_pass' => '',
19     'mailtype' => 'html',
20     'charset' => 'utf-8'
21 ];
```