

TUGAS AKHIR

PEMUTARAN ULANG KETIKAN MAHASISWA PADA
SHARIF JUDGE



Andreas Ronaldi

NPM: 6182101026

PROGRAM STUDI INFORMATIKA
FAKULTAS SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2025

FINAL PROJECT

STUDENT KEYSTROKE REPLAY ON SHARIF JUDGE



Andreas Ronaldi

NPM: 6182101026

**DEPARTMENT OF INFORMATICS
FACULTY OF SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
2025**

LEMBAR PENGESAHAN

PEMUTARAN ULANG KETIKAN MAHASISWA PADA SHARIF JUDGE

Andreas Ronaldi

NPM: 6182101026

Bandung, 20 Juni 2025

Menyetuju,

Pembimbing

Pascal Alfadian, Nugroho, M.Comp.

Ketua Tim Penguji

Anggota Tim Penguji

Prof. Dr. Veronica Sri Moertini

Pascal Alfadian, Nugroho, M.Comp.

Mengetahui,

Ketua Program Studi

Mariskha Tri Adithia, P.D.Eng

PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa tugas akhir dengan judul:

PEMUTARAN ULANG KETIKAN MAHASISWA PADA SHARIF JUDGE

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

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

Dinyatakan di Bandung,
Tanggal 20 Juni 2025

Andreas Ronaldi
NPM: 6182101026

ABSTRAK

Pada Universitas Katolik Parahyangan, digunakan SharIF Judge sebagai Online Judge yang telah dimodifikasi untuk mendukung kebutuhan pembelajaran. Namun, sistem ini masih memiliki keterbatasan dalam pengawasan, terutama saat ujian online, di mana potensi kecurangan seperti *copy-paste* meningkat. Untuk mempermudah pengawasan, tugas akhir sebelumnya telah mengintegrasikan Integrated Development Environment atau IDE ke dalam SharIF Judge, memungkinkan seluruh proses pengerjaan tugas dilakukan dalam satu platform.

Meskipun demikian, IDE tersebut tidak dapat mengawasi aktivitas pengguna jika terjadinya kecurangan. Oleh karena itu, dalam tugas akhir ini, dikembangkan fitur perekaman ketikan dan pemutaran ulang ketikan pada IDE SharIF Judge. Fitur ini bertujuan memudahkan pengawasan, mendeteksi kecurangan, dan memberikan bukti jika terjadi pelanggaran.

Fitur ini akan dirancang untuk merekam seluruh aktivitas mahasiswa yaitu perubahan kode, navigasi keluar *web page*, dan aksi pada IDE seperti menyimpan kode program. Setelah itu, data akan disimpan dalam format JSON agar dapat divisualisasikan kembali kepada pengajar.

Implementasi fitur ini diuji secara fungsional dan eksperimental. Hasil pengujian menunjukkan bahwa sistem mampu merekam aktivitas mahasiswa secara rinci dan memutar ulang proses pengerjaan secara akurat. Analisis terhadap data rekaman juga mengungkapkan pola-pola pengerjaan yang wajar maupun mencurigakan, seperti tingkat perubahan kode, keberadaan debugging, waktu jeda berpikir, serta indikasi *copy-paste*.

Sistem ini memberikan dukungan pengawasan tambahan yang efektif, meskipun belum menggunakan metode pendektsian otomatis. Kesimpulan dari penelitian ini adalah bahwa fitur pemutaran ulang ketikan dapat membantu dosen dalam melakukan evaluasi yang lebih objektif. Saran pengembangan selanjutnya mencakup otomatisasi deteksi pola, efisiensi penyimpanan, dan peningkatan antarmuka pengguna.

Kata-kata kunci: *Online Judge, Integrated Development Environment, Perekaman Penekanan Tombol*

ABSTRACT

At Parahyangan Catholic University, SharIF Judge is used as an Online Judge that has been modified to support learning needs. However, this system has limitations in monitoring especially during online exams, where the potential for cheating such as copy-pasting will increases. To fix the problem, the previous final project integrated an Integrated Development Environment or IDE into SharIF Judge, allowing the entire task completion process to be conducted within a single platform.

Nevertheless, the IDE could not monitor user activity in cases of cheating. Therefore, in this final project, a keystroke recording and playback feature was developed for the SharIF Judge IDE. This feature aims to simplify monitoring, detect cheating, and provide evidence of violations.

The feature is designed to record all student activities, including code changes, navigation away from the web page, and actions within the IDE such as saving program code. The data is then stored in JSON format so it can be visualized for instructors.

The implementation of this feature was tested functionally and experimentally. The test results showed that the system was able to record student activities in detail and replay the work process accurately. Analysis of the recorded data also revealed reasonable and suspicious work patterns, such as the level of code change, the presence of debugging, thinking pauses, and indications of copy-paste.

This system provides effective additional supervision support, although it does not yet use an automatic detection method. The conclusion of this study is that the typing replay feature can help lecturers in conducting more objective evaluations. Suggestions for further development include pattern detection accuracy, storage efficiency, and user interface improvements.

Keywords: Online Judge, Integrated Development Environment, Keystroke Logging

*Tugas akhir ini dipersembahkan kepada Tuhan Yang Maha Esa,
Keluarga, dan rekan-rekan sesama mahasiswa*

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa atas berkat dan penyertaan-Nya, penulis dapat menyelesaikan tugas akhir ini dengan judul "Pemutaran Ulang Ketikan Mahasiswa Pada SharIF Judge". Penulis juga ingin menyampaikan rasa terima kasih yang sebesar-besarnya kepada berbagai pihak yang telah memberikan dukungan, bimbingan, dan bantuan selama proses pengerjaan. Secara khusus, penulis berterima kasih kepada:

1. Kedua orang tua dan keluarga tercinta yang senantiasa memberikan doa, dukungan, dan motivasi tanpa henti.
2. Bapak Pascal Alfadian, Nugroho, M.Comp., selaku dosen pembimbing yang telah meluangkan waktu, memberikan bimbingan, saran, dan arahan berharga selama penyusunan skripsi ini.
3. Kepada seluruh teman-teman yang telah menemani, membantu, dan memberi dukungan selama pengerjaan tugas akhir ini.
4. Semua pihak yang tidak dapat disebutkan satu per satu, yang telah berkontribusi baik secara langsung maupun tidak langsung.

Akhir kata, penulis ingin mengucapkan permohonan maaf atas kekurangan-kekurangan yang terdapat pada tugas akhir ini. Penulis berharap agar tugas akhir ini dapat bermanfaat bagi pembaca yang hendak melakukan penelitian serupa.

Bandung, Juni 2025

Penulis

DAFTAR ISI

KATA PENGANTAR	xv
DAFTAR ISI	xvii
DAFTAR GAMBAR	xix
1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	3
1.3 Tujuan	3
1.4 Batasan Masalah	3
1.5 Metodologi	4
1.6 Sistematika Pembahasan	4
2 LANDASAN TEORI	5
2.1 SharIF Judge	5
2.1.1 Instalasi	5
2.1.2 Users	6
2.2 CodeIgniter 3	6
2.2.1 Model-View-Controller	7
2.2.2 CodeIgniter URLs	9
2.2.3 <i>Helpers</i>	9
2.3 Twig	9
2.4 Integrated Development Environment	10
2.5 Ace	10
2.5.1 Perekaman Event	11
2.6 Chart.js	13
2.7 Code Churn	14
2.7.1 Code Churn Rate	15
2.8 Deteksi Kecurangan dalam Pengerjaan Tugas Pemrograman	15
2.8.1 Perubahan Kode yang sedikit	15
2.8.2 Navigasi yang tidak wajar	15
2.8.3 Tidak Membutuhkan Waktu untuk Berpikir	15
2.8.4 Tidak Ada Aktivitas <i>Debugging</i>	16
2.8.5 <i>Copy-Pasting</i>	16
3 ANALISIS	17
3.1 Analisis Sistem Kini	17
3.1.1 Model, View, Controller	17
3.1.2 Assets	35
3.1.3 Penyimpanan Kode Submission	37
3.1.4 Antrean Penilaian Kode	37
3.1.5 Editor Kode	37

3.2	Analisis Pola Kecurangan pada Rekaman Ketikan	39
3.2.1	Analisis Perubahan Kode	39
3.2.2	Analisis Aktivitas Navigasi	39
3.2.3	Analisis Pengukuran Jeda Berpikir	39
3.2.4	Analisis Deteksi Debugging	39
3.2.5	Analisis <i>Copy-Pasting</i>	39
3.3	Analisis Sistem Usulan	39
3.3.1	Fitur perekaman perubahan atau event	40
3.3.2	Fitur penyimpanan rekaman perubahan	41
3.3.3	Fitur melihat daftar rekaman	41
3.3.4	Fitur pemutaran ulang rekaman	42
4	PERANCANGAN	43
4.1	Rancangan Penyimpanan Rekaman	43
4.2	Rancangan Event yang direkam	44
4.3	Rancangan Antarmuka	46
4.3.1	Sistem Rekaman	46
4.3.2	Sistem Pemutaran ulang	46
4.4	Rancangan Perubahan Kode	47
4.4.1	Merekam perubahan atau event	47
4.4.2	Menyimpan rekaman	47
4.4.3	Melihat daftar rekaman	48
4.4.4	Pemutaran ulang rekaman	48
5	IMPLEMENTASI DAN PENGUJIAN	51
5.1	Implementasi	51
5.1.1	Merekam Peristiwa pada IDE	51
5.1.2	Menyimpan Rekaman pada Sistem	53
5.1.3	Melihat Daftar Rekaman	54
5.1.4	Pemutaran Ulang Rekaman	55
5.2	Pengujian Fungsional	57
5.3	Pengujian Eksperimental	57
5.3.1	Lingkungan pengujian	57
5.3.2	Eksperimen	58
6	KESIMPULAN DAN SARAN	65
6.1	Kesimpulan	65
6.2	Saran	65
DAFTAR REFERENSI		67
A	KODE PROGRAM	69
B	HASIL HALAMAN	97
C	FILE DOCKER EKSPERIMEN	99

DAFTAR GAMBAR

1.1	Sistem Tradisional Pemberian Tugas	1
1.2	Sistem Integrasi oleh <i>Online Judge</i>	2
1.3	Tampilan Awal SharIF Judge	2
2.1	<i>Flow Chart</i> CodeIgniter	6
2.2	Hasil Halaman Web <i>Library Ace</i>	12
2.3	Hasil Halaman Web <i>Library Chart.js</i>	14
3.1	Struktur MVC pada SharIF Judge	18
3.2	Struktur Kelas Model pada SharIF Judge	18
3.3	Struktur Direktori View pada SharIF Judge	22
3.4	Struktur Kelas Controller pada SharIF Judge	24
3.5	Halaman Assignments	25
3.6	Halaman Dashboard	26
3.7	Halaman Hall of Fame	26
3.8	Halaman Install	27
3.9	Halaman Login	27
3.10	Halaman 24-Hour Log	28
3.11	Halaman Moss	28
3.12	Halaman Notifications	29
3.13	Halaman Problems	29
3.14	Halaman Profile	30
3.15	Halaman Queue	31
3.16	Halaman Rejudge	31
3.17	Halaman Scoreboard	32
3.18	Halaman Settings	32
3.19	Halaman All Submissions	33
3.20	Halaman Final Submissions	33
3.21	Halaman Submit	35
3.22	Halaman Users	35
3.23	Editor Kode SharIF Judge	38
3.24	Usecase analisis sistem usulan	40
3.25	Sequence Diagram Fitur Perekaman Perubahan	40
3.26	Sequence Diagram Fitur Penyimpanan Rekaman	41
3.27	Sequence Diagram Membuka Halaman Rekaman	42
3.28	Sequence Diagram Membuka Halaman Rekaman	42
4.1	Tabel SharIF Judge Saat Ini	45
4.2	Rancangan Tabel Penyimpanan Rekaman	45
4.3	Halaman Submit	46
4.4	Rancangan Antarmuka Halaman Daftar Rekaman	46
4.5	Rancangan Antarmuka Halaman Pemutaran Ulang	47

5.1	Representasi warna untuk setiap jenis peristiwa pada histogram	59
5.2	Bagan Histogram Perubahan Kode Program	60
5.3	Bagan Heatmap Perubahan Lokasi Kode Program	60
5.4	Bagan Histogram Perubahan Input dan Aksi <i>Execute</i>	61
5.5	Bagan Histogram Perubahan Navigasi	61
5.6	Bagan Histogram Perubahan	62
5.7	Bagan Histogram Perubahan	62
B.1	Halaman Daftar Rekaman	97
B.2	Halaman Rekaman	98

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Institusi yang memberikan pendidikan, perlu memiliki cara untuk mengetahui pemahaman pelajarannya. Salah satu caranya adalah dengan memberikan tugas. Tugas merupakan sebuah bentuk penilaian oleh pengajar kepada pelajarnya [1]. Tugas diberikan kepada pelajar untuk membantu pelajar mendalami materi yang sudah diberikan sebelumnya oleh pengajar dan juga untuk melihat seberapa jauh pemahaman pelajar terhadap materi yang sudah diberikan.

Pada bidang informatika, banyak materi pembelajaran yang dapat diberikan. Salah satu pembelajaran utama dalam bidang informatika adalah keterampilan pemrograman. Oleh karena itu, diperlukan sebuah sistem untuk melatih keterampilan pemrograman yaitu dengan memberikan tugas menulis kode program sesuai dengan petunjuk yang diberikan dan program tersebut dapat berjalan sesuai dengan petunjuk [2]. Secara tradisional, tugas ini diberikan dengan cara pengajar menyiapkan dan mendistribusikan tugas tersebut kepada pelajar, kemudian dikumpulkan kembali hasil program pekerjaan pelajar, dan pengajar akan menilai kode program sesuai ketepatan dengan program yang diinginkan secara manual seperti Gambar 1.1. Karena menilai kode program mencakup keluaran program dan juga analisis kode, maka proses tersebut memerlukan waktu yang cukup lama. Meskipun demikian, cara tradisional masih dapat digunakan jika jumlah siswanya sedikit.

Namun, semakin banyak kode program yang harus diperiksa maka semakin banyak waktu yang dibutuhkan dan semakin banyak pula kesalahan yang diakibatkan oleh manusia. Salah satu masalah lain yang muncul juga adalah pelajar tidak dapat mengetahui apakah kode program berada pada jalur yang benar dalam menemukan solusi tugas tersebut.

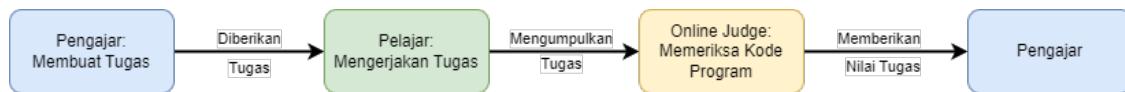


Gambar 1.1: Sistem Tradisional Pemberian Tugas

Pemberian tugas menulis kode program memiliki banyak masalah. Oleh karena itu, dibutuhkannya sistem baru untuk memberikan tugas kepada pelajar bidang informatika. Sistem baru yang dimaksud tentunya untuk melakukan penilaian secara otomatis. Sebuah sistem yang mengambil kode program pelajar dan memberikan sebuah nilai numerik yang menandakan hasil dari kode program tersebut [3]. Suatu hal yang menarik, Tugas kode program dapat dibagi menjadi 2 jenis yaitu tugas individu dan tugas kelompok. Pada tugas kelompok merupakan tugas yang ditanggung oleh banyak pelajar, biasanya program yang dibuat memiliki antarmuka dan harus diperiksa oleh pengguna khusus yang mengetahui fitur-fitur yang dibutuhkan. Sedangkan tugas individu merupakan sebuah tugas yang diberikan untuk satu individu, biasanya program yang dibuat bersifat algoritmik dan tidak memerlukan antarmuka untuk dijalankan. Program algoritmik sebuah jenis program yang dibuat berdasarkan algoritma untuk menyelesaikan masalah tertentu. Algoritma sendiri adalah langkah-langkah dalam pemecahan masalah secara sistematis [4]. Algoritma itu

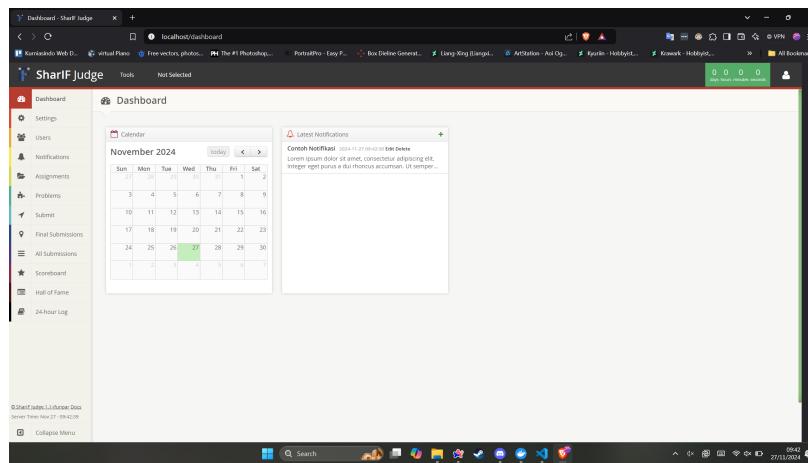
seperti resep makanan, dimana akan ada bahan-bahan yang dibutuhkan dan serangkaian langkah untuk membuat suatu makanan yang dijelaskan.

Sebagian besar program yang bersifat algoritmik hanya perlu mengambil *input* dari *input* standar seperti angka, huruf, dan sebuah kata atau kalimat dengan format yang sudah ditentukan, seolah-olah *input* ini merupakan *output* dari program lain. Kemudian program algoritmik akan memproses *input* tersebut dalam komputer dan mengeluarkan hasil komputasinya dalam format yang sudah ditentukan untuk dibaca oleh program lain dan memanfaatkan hasil komputasi tersebut. Singkatnya, program algoritmik itu seperti filter antar program. Dengan ini, sistem penilaian secara otomatis dapat dibuat dengan membuat sebuah program yang mengambil kode program, memasukkan *input* sesuai format ke dalam program tersebut, membaca hasil keluaran program, dan menilai hasil keluaran program tersebut [3]. Sistem penilaian otomatis ini diberikan nama *Online Judge*. Terlebih lagi sistem ini dapat dilakukan secara *offline* maupun *online*. Gambar 1.2 menunjukkan bagaimana *online judge* berintegrasi dengan sistem pemberian tugas yang sudah ada.



Gambar 1.2: Sistem Integrasi oleh *Online Judge*

Tugas pemrograman sudah menjadi keseharian dalam pembelajaran pada bidang informatika. Termasuk pada perguruan tinggi pada bidang informatika, maka *online judge* menjadi sebuah kebutuhan, termasuk di Universitas Katolik Parahyangan atau yang biasa disebut UNPAR. *Online Judge* yang digunakan oleh UNPAR dinamakan SharIF-Judge [5] yang merupakan hasil modifikasi oleh Stillmen Vallian terhadap Sharif-Judge [6] buatan Mohammad Javad Naderi yang dibuat menggunakan *framework* CodeIgniter dan Bash. Gambar 1.3 merupakan tampilan halaman utama setelah memasuki situs web SharIF-Judge dalam *browser* yaitu halaman *Dashboard*.



Gambar 1.3: Tampilan Awal SharIF Judge

Ujian juga merupakan bentuk penilaian oleh pengajar kepada pelajarnya. Tentunya pelajar maupun mahasiswa ingin memperoleh nilai yang memuaskan dalam ujiannya. Banyak cara yang dilakukan oleh pelajar maupun mahasiswa untuk memperoleh nilai tersebut, salah satunya adalah dengan melakukan kecurangan yaitu *copy-paste* atau menyalin jawaban teman atau rekan mereka [1]. Praktik ini diperparah jika ujian dilakukan secara *online*, dikarenakan pelajar dapat mengakses berbagai fasilitas di internet. Oleh karena itu, diperlukan sebuah sistem pada sistem *online judge* untuk mengawasi saat terjadinya saat ujian *online*.

Pada saat siswa mengerjakan tugas maupun ujian pembuatan kode program, umumnya pekerjaan kode tersebut dilakukan pada aplikasi eksternal seperti *Visual Studio Code* atau *notepad*. Hal ini juga terjadi pada sistem dalam UNPAR dimana mahasiswa akan membuat kode program pada aplikasi eksternal. Ini membuat pengawasan saat pembuatan kode program lebih sulit untuk

dilakukan, terlebih jika ujian dilakukan secara *online*. Maka dari itu, Nicholas Aditya Halim memodifikasi SharIF Judge agar semua sistem pemberian tugas seperti pada Gambar 1.2 dapat dilakukan dalam sistem yang sama yaitu pada SharIF Judge. Sistem yang bangun oleh Nicholas Aditya Halim adalah “Implementasi editor kode pada Sharif Judge” [7], dimana SharIF Judge ditambahkan sebuah *Integrated Development Environment* atau yang disebut dengan IDE. IDE merupakan sebuah sistem yang memiliki kemampuan untuk membuat kode dalam editor kode dan menjalankan kode program tersebut. Dengan adanya IDE, seluruh proses pembuatan kode program dapat dilakukan dalam SharIF Judge. Maka dari itu, seluruh proses sistem pemberian tugas dapat dilakukan dalam satu sistem saja, yaitu SharIF Judge.

Walaupun begitu, pada dasarnya IDE tidak dapat mengawasi jika terjadinya praktik *copy-paste*. Maka dari itu pada tugas akhir ini, IDE pada SharIF Judge akan dimodifikasi untuk menangani hal tersebut dengan ditambahkan fitur untuk merekam semua ketikan atau kejadian dalam editor kode dalam IDE. Lalu ketikan atau kejadian dalam editor dapat diputar kembali seperti rekaman. Fitur ini akan membuat pengawasan terhadap kegiatan kuliah lebih mudah untuk pengawas dan dapat menjadi bukti kecurangan jika dibutuhkan. Dalam tugas akhir ini juga, akan membahas analisis sederhana mengenai pola yang muncul pada data rekaman pemutaran ulang.

1.2 Rumusan Masalah

Rumusan Masalah yang akan dibahas pada tugas akhir ini adalah:

1. Bagaimana merencanakan dan mengimplementasikan perekaman dan pemutaran ulang ketikan mahasiswa pada IDE SharIF-Judge?
2. Bagaimana cara menyimpan data rekaman pemutaran ulang mahasiswa?
3. Bagaimana menganalisis data rekaman pemutaran ulang mahasiswa?
4. Aktivitas atau kecurangan apa saja yang dapat terdeteksi melalui data rekaman, dan apa saja yang tidak dapat dideteksi?

1.3 Tujuan

Tujuan yang ingin dicapai skripsi ini adalah sebagai berikut:

1. Merencanakan dan mengimplementasikan perekaman dan pemutaran ulang ketikan mahasiswa pada IDE SharIF-Judge.
2. Mencari cara penyimpanan data efektif dan sesuai dengan sistem yang sudah ada dan mengimplementasikannya pada perekaman dan pemutaran ulang ketikan.
3. Menganalisis data rekaman hasil eksperimen mengenai pola yang muncul dengan memvisualisasikan data menjadi bagan-bagan seperti *histogram*.
4. Melakukan analisis terhadap data rekaman untuk mengamati pola pengerjaan dan potensi kecurangan.

1.4 Batasan Masalah

Pada pengerjaan tugas akhir ini terhadap batasan sebagai berikut:

- Perangkat lunak SharIF Judge hanya digunakan pada lingkungan mahasiswa yang menguasai hal mengenai pemrograman.
- Pengujian perangkat lunak dilakukan dalam sistem tertutup di mana akan diawasi.
- Fitur yang dikembangkan hanya mencakup aktivitas yang terjadi di dalam halaman IDE SharIF Judge.
- Rekaman yang dibuat hanya mencakup perubahan kode, *input/output*, eksekusi program, dan interaksi pada halaman (seperti navigasi tab).
- Kecurangan yang terjadi di luar sistem, seperti menggunakan perangkat lain, menggunakan dua layar, atau menjiplak dengan cara manual dari luar sistem, tidak dapat dideteksi.

- Sistem tidak menggunakan AI atau algoritma khusus untuk mendeteksi kecurangan, melainkan hanya menyediakan data rekaman yang bisa dianalisis oleh pengajar.

1.5 Metodologi

Metodologi pelaksanaan tugas akhir ini adalah sebagai berikut:

1. Melakukan studi mengenai komponen yang diperlukan untuk membuat sistem perekaman dan pemutaran ulang ketikan pada IDE berbasis web.
2. Merancang sistem perekaman dan pemutaran ulang ketikan berbasis web untuk SharIF Judge
3. Mengimplementasikan IDE berbasis web pada SharIF Judge.
4. Melakukan pengujian dan eksperimen.
5. Menganalisis hasil eksperimen.
6. Menulis dokumen tugas akhir.

1.6 Sistematika Pembahasan

Sistematika pembahasan skripsi ini adalah sebagai berikut:

- **Bab 1:** Pendahuluan
Membahas latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi, dan sistematika pembahasan.
- **Bab 2:** Landasan Teori
Membahas teori-teori yang berhubungan dengan penelitian ini, yaitu SharIF Judge, CodeIgniter 3, Twig, IDE, Ace, dan ChartJS.
- **Bab 3:** Analisis
Membahas analisis terhadap perangkat lunak SharIF Judge dan IDE pada SharIF Judge.
- **Bab 4:** Perancangan
Membahas perancangan fitur yang akan diimplementasikan pada SharIF Judge.
- **Bab 5:** Implementasi dan Pengujian
Membahas implementasi fitur pada SharIF Judge dan pengujian yang dilakukan beserta dengan analisis hasil pengujian.
- **Bab 6:** Kesimpulan dan Saran
Membahas kesimpulan dari penelitian ini dan saran untuk penelitian berikutnya.

BAB 2

LANDASAN TEORI

2.1 SharIF Judge

SharIF Judge merupakan modifikasi dari *open source* bernama Sharif Judge, sebuah situs web judge gratis dengan kemampuan mengkompilasi bahasa C, C++, Java, dan Python. Sharif Judge dibuat oleh Mohammad Javad Naderi dengan interface web berbahasa PHP menggunakan *framework* CodeIgniter 3 dan BASH [6]. Modifikasi dilakukan untuk menambahkan fitur pada Sharif Judge dan juga untuk menyesuaikan sesuai dengan kebutuhan Teknik Informatika UNPAR.

2.1.1 Instalasi

Beberapa prasyarat yang diperlukan untuk menjalankan SharIF Judge pada sebuah *server* Linux adalah sebagai berikut:

- *Webserver* dengan PHP versi 5.3 atau lebih dengan `mysqli` extension
- PHP Command Line Interface (CLI)
- *Database* MySQL atau PostgreSQL
- PHP harus memiliki akses untuk menjalankan *shell commands* dengan fungsi `shell_exec`
- Kemampuan untuk mengompilasi dan menjalankan kode yang dikumpulkan (`gcc`, `g++`, `javac`, `java`, `python2`, dan `python3`)
- Perl

Setelah semua prasyarat terpenuhi, berikut langkah-langkah instalasi SharIF Judge:

1. Unduh versi terakhir dari Sharif Judge dan menempatkannya pada direktori publik.
2. Pindahkan direktori `system` dan `application` ke luar direktori publik. Kemudian simpan alamatnya pada `index.php`.
3. Buat sebuah *Database* MySQL atau PostgreSQL.
4. Atur pengaturan koneksi *database* pada `application/config/database.php`.
5. Atur pengaturan RADIUS dan SMTP pada `application/config/secrets.php` jika dibutuhkan.
6. Atur agar direktori `application/cache/Twig` dapat ditulis oleh php.
7. Buka halaman utama SharIF Judge pada *browser* dan ikuti proses instalasi.
8. Log in dengan akun admin
9. Pindahkan direktori `tester` dan `assignments` ke luar direktori publik. Kemudian simpan alamatnya pada halaman pengaturan.

2.1.2 Users

Pada SharIF Judge, pengguna dibagi menjadi 4 *role*. Role yang tersedia adalah sebagai berikut:

1. *admin*
2. *head instructor*
3. *instructor*
4. *student*

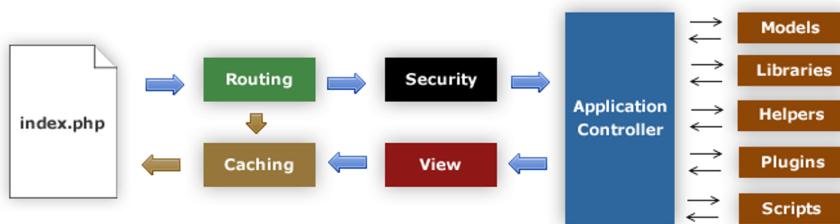
Setiap *role* memiliki akses pada fitur yang berbeda sesuai dengan *role*-nya. Tabel 2.1 merupakan aksi-aksi yang dapat dilakukan untuk setiap pengguna pada SharIF Judge.

Tabel 2.1: *Tabel fitur untuk setiap role*

Aksi	Admin	Head Instructor	Instructor	Student
Mengubah <i>Settings</i>	✓	✗	✗	✗
Mengelola Pengguna	✓	✗	✗	✗
Mengelola <i>Assignment</i>	✓	✓	✗	✗
Mengelola Notifikasi	✓	✓	✗	✗
<i>Rejudge</i>	✓	✓	✗	✗
Mengelola <i>Queue</i>	✓	✓	✗	✗
Mendeteksi Kode yang Mirip	✓	✓	✗	✗
Melihat Semua <i>Submission</i>	✓	✓	✓	✗
Mengunduh Kode Final	✓	✓	✓	✗
Memilih <i>Assignment</i>	✓	✓	✓	✓
Submit Kode	✓	✓	✓	✓

2.2 CodeIgniter 3

CodeIgniter 3 adalah sebuah *framework opensource* untuk mempermudah pengguna dalam membangun sebuah aplikasi *website* menggunakan bahasa PHP. CodeIgniter 3 bertujuan untuk membantu pengguna dalam membangun sebuah aplikasi *website* lebih cepat dengan menyediakan *library* yang beragam dengan fungsi yang umum digunakan dan tampilan dan *logic* yang simpel. Gambar 2.1 merupakan bagaimana data mengalir pada sistem CodeIgniter.



Gambar 2.1: *Flow Chart CodeIgniter*

Berikut merupakan penjelasan sederhana dari *flow chart* sistem CodeIgniter 3:

1. *index.php* berfungsi sebagai *front controller* yang akan melakukan inisiasi *resource* utama untuk menjalankan CodeIgniter.
2. Router memeriksa *request* HTTP dan menentukan tindakan selanjutnya sesuai *request* tersebut.
3. Jika tersedia, *cache* akan langsung dikirimkan ke *browser* melewati proses eksekusi biasa.
4. Sebelum *controller* dimuat, seluruh *request* HTTP dan data dari pengguna akan disaring terlebih dahulu untuk keamanan sistem.
5. *Controller* memuat *model*, *library* utama, dan *resource* lainnya yang diperlukan.
6. *View* akhir lalu dikirim ke *browser* untuk dilihat.

2.2.1 Model-View-Controller

CodeIgniter merupakan framework berbasis arsitektur Model-View-Controller atau yang selanjutnya akan disebut dengan MVC. MVC adalah pendekatan *software* yang memisahkan *logic* aplikasi dan tampilannya. Pendekatan ini membuat *website* hanya memiliki sedikit *script* karena tampilan *website* terpisah dari *scripting* PHP. Berikut merupakan penjelasan mengenai struktur MVC:

Model

Model mewakili struktur data pada sistem untuk mengambil, memasukkan, dan memperbarui data pada *database*. *Model* dapat dibuat dengan membuat sebuah kelas yang mewarisi `CI_Model` dan diletakkan pada `application/models/`.

Kode 2.1: Contoh *model*

```

1 class Blog_model extends CI_Model {
2
3     public $title;
4     public $content;
5     public $date;
6
7     public function get_last_ten_entries()
8     {
9         $query = $this->db->get('entries', 10);
10        return $query->result();
11    }
12
13    public function insert_entry()
14    {
15        $this->title    = $_POST['title'];
16        $this->content  = $_POST['content'];
17        $this->date     = time();
18
19        $this->db->insert('entries', $this);
20    }
21
22    public function update_entry()
23    {
24        $this->title    = $_POST['title'];
25        $this->content  = $_POST['content'];
26        $this->date     = time();
27
28        $this->db->update('entries', $this, array('id' => $_POST['id']));
29    }
30}
31

```

Kode 2.1 merupakan contoh model kelas bernama `Blog_model` pada CodeIgniter. *Model* `Blog_model` dapat mengambil, menambahkan, dan memperbarui *database* bernama ‘entries’. File *model* tersebut akan disimpan pada `application/models/Blog_model`. Selanjutnya, pengguna dapat memanggil *Model* tersebut pada *file controller* (akan dijelaskan pada bagian [Controller](#)) untuk memuat model pada Kode 2.1 dengan menggunakan notasi sebagai berikut:

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

Untuk memanggil fungsi pada model tersebut, notasi yang digunakan adalah sebagai berikut:

```
$this->Blog_model->get_last_ten_entries();
```

Notasi di atas akan memuat model dengan nama `Blog_model` dan akan memanggil fungsi bernama `get_last_ten_entries`. Notasi untuk memanggil sebuah fungsi dalam model hanya dapat dilakukan jika notasi untuk memuat sebuah model sudah dilakukan sebelumnya.

View

View adalah informasi yang akan ditunjukkan kepada pengguna. Biasanya *view* merupakan sebuah halaman web, tetapi pada CodeIgniter, *view* dapat berupa pecahan halaman seperti *header*, *footer*, *sidebar*, dan lainnya. Pecahan halaman tersebut dapat dimasukkan secara fleksibel ke dalam *view* lainnya apabila dibutuhkan.

Kode 2.2: Contoh *view*

```

1 <html>
2 <head>
3     <title>My Blog</title>
4 </head>
5 <body>
6     <h1>Welcome to my Blog!</h1>
7 </body>
8 </html>

```

Kode 2.2 merupakan contoh dari *file view* pada CodeIgniter. File akan disimpan pada direktori `application/views/`. Untuk dapat diperlihatkan dibutuhkan penggalian halaman pada *file controller* dengan cara sebagai berikut:

```
$this->load->view('name');
```

Notasi di atas akan mengembalikan halaman *view* dengan nama `name` yang terletak pada direktori `application/views/name.php` dan menampilkannya kepada pengguna.

Controller

Controller adalah bagian utama dari aplikasi CodeIgniter, berfungsi sebagai perantara antara *model*, *view*, dan *resources* lainnya yang dibutuhkan untuk memproses HTTP *request* dan membuat sebuah halaman web. Kelas *Controller* akan mewarisi `CI_Controller` dan disimpan pada `application/controllers/`. Contoh *controller* ditunjukkan pada Kode 2.3.

Kode 2.3: Contoh *controller*

```

1 <?php
2 class Blog extends CI_Controller {
3
4     public function index()
5     {
6         echo 'Hello_World!';
7     }
8
9     public function comments()
10    {
11        echo 'Look_at_this!';
12    }
13}

```

Kode 2.3 berfungsi dalam mengembalikan string sesuai dengan fungsi *controller* yang dipanggil. Nama file *controller* pada direktori `application/controllers/blog.php` dan metode di atas akan dijadikan segmen pada URL seperti berikut:

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

URL di atas akan mengembalikan sebuah teks ‘Hello World!’.

Kode 2.4: Contoh memuat *model* dan menampilkan *view*

```

1 class Blog_controller extends CI_Controller {
2     public function blog()
3     {
4         $this->load->model('blog');
5
6         $data['query'] = $this->blog->get_last_ten_entries();
7
8         $this->load->view('blog', $data);
9     }
10}

```

Pada CodeIgniter, *model* dan *view* hanya dapat dimuat melalui controller. Seperti contoh, Kode 2.4 akan memuat *model blog* dan mengambil data dari *database* melalui model `blog`, lalu menampilkan *view* bernama `blog` yang memuat data tersebut.

2.2.2 CodeIgniter URLs

URL pada CodeIgniter menggunakan *segment-based approach* dibandingkan dengan *query string approach* yang biasanya dipakai. *Segment-based approach* dirancang untuk *search-engine* dan dapat mempermudah pengguna juga. Berikut merupakan contoh dari URL CodeIgniter:

```
example.com/news/article/my_article
```

Struktur URL pada CodeIgniter juga mengikuti pendekatan MVC (Referensi 2.2.1) dan biasanya memiliki struktur pemanggilan fungsi pada controller sebagai berikut:

```
example.com/class/function/params
```

1. Segmen pertama mewakili kelas *controller* yang ingin dipanggil.
2. Segmen berikutnya mewakili fungsi kelas atau *method* yang ingin di panggil.
3. Segmen ketiga dan selanjutnya mewakili *identifier* atau pengenal dan variable-variable lain yang akan dikirimkan ke *controller*.

2.2.3 Helpers

Helpers merupakan sebuah kumpulan fungsi untuk membantu dalam sebuah kategori tertentu. *File helpers* terdapat pada direktori `system\helpers` atau `application\helpers`. Penggunaan *helpers* dalam *CodeIgniter* adalah dengan memuat file helpers dalam fungsi kelas *Controller* serupa dengan cara pengguna model, yaitu dengan menggunakan notasi seperti berikut ini:

```
$this->load->helper('name')
```

Setelah *helper* dimuat dalam fungsi, maka kumpulan fungsi dalam *file helper* dapat langsung dipanggil dalam kode setelah notasi memuat *helper* dipanggil.

2.3 Twig

Twig merupakan sebuah *template engine* untuk PHP. Ada beberapa *expression*, *expression*, atau *statement* yang ditemukan pada template Twig adalah sebagai berikut:

- Pewarisan *Template*
- Struktur Kontrol (menggunakan kondisional, *looping*)
- Filter
- Variable pada PHP

Pada saat template dievaluasi, semua *variable* atau *expression* akan dibuang menjadi value dan *tag* yang mengontrol logika template. Selain itu twig dapat mengembalikan data sesuai yang tertulis pada *file* seperti normal teks dan HTML *tag*.

Kode 2.5: Contoh template Twig

```

1  {% extends "base.html" %} 
2  {% block navigation %} 
3  <ul id="navigation"> 
4  {% for item in navigation %} 
5  <li> 
6  <a href="{{item.href}}> 
7  {% if item.level == 2 %}&nbsp;&nbsp;{% endif %} 
8  {{ item.caption|upper }} 
9  </a> 
10 </li> 
11 {% endfor %} 
12 </ul> 
13 {% endblock navigation %}
```

Kode 2.5 merupakan contoh sebuah template Twig. Terdapat dua jenis *delimiter*, yaitu `{% ... %}` dan `{{ ... }}`. *Delimiter* `{% ... %}` digunakan untuk menjalankan sebuah *statement* seperti *for-loops*, sedangkan *delimiter* `{{ ... }}` digunakan untuk mengubah sebuah *variable* atau *expression* menjadi sebuah HTML biasa dengan nilai *variable* atau *expression* tersebut.

2.4 Integrated Development Environment

Integrated Development Environment (IDE) merupakan sebuah aplikasi yang menyediakan berbagai peralatan yang diperlukan untuk membantu pengembangan perangkat lunak. Beberapa peralatan umum yang dimiliki oleh sebuah IDE adalah sebagai berikut:

- *Editor*
Editor teks sebagai tempat untuk mengetik kode, dapat dilengkapi dengan berbagai fitur seperti *syntax highlighting* (menampilkan teks dengan warna yang berbeda untuk meningkatkan keterbacaan kode sesuai dengan bahasa pemrograman yang digunakan) dan *word completion* (menampilkan prediksi kata yang sedang atau akan diketik oleh pengguna).
- *Complier*
Digunakan untuk menterjemahkan kode program yang dibuat pada editor teks ke dalam sebuah program yang dapat dijalankan oleh komputer.
- *Execution*
Menjalankan kode program yang sudah dikompilasi dari *Editor*, dengan input jika dibutuhkan, dan mengembalikan hasilnya kepada pengguna.

2.5 Ace

Ace merupakan *library* yang menyediakan sebuah editor kode yang dapat dimasukkan ke dalam sebuah halaman web dan dikembangkan menggunakan bahasa *JavaScript*. Ace memiliki kemampuan yang sama seperti editor kode pada umumnya. Berikut merupakan beberapa fitur utama yang dimiliki dan dapat diaktivasikan pada *library* Ace:

- *Syntax highlighting* untuk bahasa pemrograman.
- Automatic indent dan outdent.
- Kemampuan *cut*, *copy*, dan *paste*.
- Kemampuan *drag and drop* teks menggunakan mouse.
- Banyak *Cursors* dan *selections*
- *Line wrapping*
- *Code folding*

Untuk mengintegrasikan *library* Ace dalam sebuah halaman web, Ace perlu ditanam dalam halaman web. Salah satu cara untuk menanam Ace ke dalam halaman web adalah dengan mem-*build library* atau mengunduh hasil dari *build* direktori bernama *src* versi *pre-packaged* yang disediakan oleh Ace. Hasil dari *building library* Ace adalah sebuah direktori yang dapat ditaruh dalam direktori lokal. Dalam direktori tersebut, terdapat file *JavaScript* bernama *ace.js* yang dapat dipanggil dalam halaman web untuk menanam *library* Ace dalam halaman web. Cara untuk menanamkan file tersebut sama dengan cara untuk memasukkan file *JavaScript* yaitu dengan cara seperti berikut:

```
<script src="/ace-builds/ace.js" type="text/javascript"></script>
```

Setelah *library* Ace ditanam untuk mengakses berbagai macam fitur yang disediakan, maka kelas yang disediakan Ace dapat dipanggil. Berikut merupakan beberapa kelas penting yang terdapat pada *library* Ace adalah sebagai berikut:

- **Ace**
Kelas **Ace** merupakan kelas utama untuk menyiapkan editor kode Ace pada *browser*. Ace memiliki fungsi utama yang penting yaitu fungsi **edit** yang akan membuat sebuah editor dalam halaman web pada element beridentitas argumen yang diberikan saat dipanggil. Fungsi **edit** akan mengembalikan kelas **Editor**.
- **Editor**
Entri utama untuk fungsionalitas *library* Ace. Editor sendiri merepresentasikan editor kode yang dibuat pada halaman web. Editor juga menjadi kelas utama untuk mengakses kelas-kelas yang berhubungan dengan editor kode dengan mengakses kelas variable **Editor** seperti **session**

dalam editor kode. Kelas `Editor` sendiri dapat dikonfigurasikan sesuai dengan fungsi yang disediakan seperti `setTheme` yang mengubah warna editor kode sesuai dengan `theme` yang dipilih.

- **EditSession**

Sebuah kelas yang menyimpan semua status dalam editor seperti isi editor, `selection`, dan lain-lain. Kelas ini dinamakan `EditSession`, tetapi untuk mengakses dari kelas `Editor`, variable `EditSession` dinamakan `session`.

- **Anchor**

Menangani posisi `pointer` pada dokumen. Saat teks dimasukkan atau dihapus, posisi `anchor` akan diperbarui sesuai dengan teks yang dimasukkan atau dihapus dari dokumen.

- **Document**

Menyimpan teks dokumen.

- **Range**

Kelas ini digunakan di berbagai tempat untuk mengindikasikan suatu wilayah di dalam editor. Kelas ini menyimpan posisi baris awal dan kolom awal, serta baris akhir dan kolom akhir.

- **Selection**

Kelas ini menyimpan posisi yang di pilih oleh pengguna dalam editor.

- **Commands**

Kelas ini digunakan untuk menjalankan perintah pada sebuah editor. Contoh perintah yang sudah ada dalam editor yaitu `insert`, `copy`, `paste`.

Kode 2.6: Contoh kode penggunaan Ace

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <title>ACE in Action</title>
5 <style type="text/css" media="screen">
6   #editor {
7     position: absolute;
8     top: 0;
9     right: 0;
10    bottom: 0;
11    left: 0;
12  }
13 </style>
14 </head>
15 <body>
16
17 <div id="editor">function foo(items) {
18   var x = "All this is syntax highlighted";
19   return x;
20 }</div>
21
22 <script src="/ace-builds/ace.js" type="text/javascript"></script>
23 <script>
24   var editor = ace.edit("editor");
25   editor.setTheme("ace/theme/monokai");
26   editor.session.setMode("ace/mode/javascript");
27 </script>
28 </body>
29 </html>

```

Kode 2.6 merupakan cara penggunaan Ace pada sebuah div dengan id `editor`. Ace juga memiliki beberapa konfigurasi, seperti terlihat pada Kode 2.6, yaitu editor menggunakan tema `monokai` dan menggunakan `syntax highlighting` untuk bahasa pemrograman `JavaScript`. Gambar 2.2 menunjukkan hasil halaman web yang dibuka dalam `browser` menggunakan Kode 2.6.

2.5.1 Perekaman Event

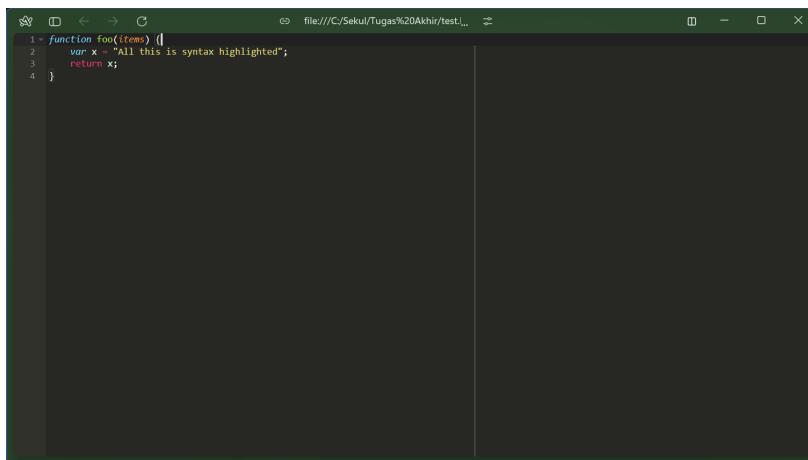
Pada `library` Ace, disediakannya fungsi `event listener` atau pendengar `event` atau kejadian yang berhubungan dengan kelas-kelas yang disediakan. Pada `event listener` ini akan disediakannya sebuah fungsi `callback` yang akan dipanggil saat `event` tersebut terjadi. Berikut merupakan beberapa `event listener` dalam kelas-kelas yang disediakan oleh `library` Ace:

- **Editor**

Pada editor sendiri disediakannya satu `event listener` yaitu `mouseup` yang akan mendengarkan saat melepaskan tombol pada tetikus atau `mouse`.

- **EditSession**

Pada kelas `session` ada satu `event listener` yaitu `change` yang akan mendengarkan perubahan



Gambar 2.2: Hasil Halaman Web *Library Ace*

pada isi atau kode pada editor kode. Pada fungsi *callback* yang akan dijalankan oleh *event listener* ini akan diberikan objek *JavaScript* bernama **delta** yang menunjukkan perubahan dan lokasi terjadi perubahan pada editor kode.

- **Selection**

Pada kelas *selection* ada beberapa *event listener* yaitu sebagai berikut:

- **changeCursor** : Mendengarkan perubahan pada kursor atau *anchor* dalam editor kode.
- **changeSelection** : Mendengarkan perubahan pemilihan isi kode dalam editor kode.

- **Commands**

Pada kelas ini tersedia dua *event listener* yaitu **exec** dan **afterExec**. **exec** akan mendengarkan saat perintah akan dijalankan pada editor kode, sedangkan **afterExec** akan mendengarkan perintah yang sudah selesai dijalankan pada editor kode. Pada fungsi *callback* yang akan dijalankan oleh *event listener* ini akan diberikan perintah yang dijalankan oleh kelas **Commands**.

Untuk menggunakan fungsi *event listener* pada kelas yang diinginkan, dibutuhkan fungsi **on** pada kelas tersebut. Fungsi **on** memiliki dua parameter yaitu nama *event* yang ingin didengar (**exec** atau **change**) dan sebuah fungsi *callback* yang akan dijalankan saat *event* terjadi. Kode 2.7 merupakan perubahan kode yang dilakukan dalam tag **<script>** pada Kode 2.6 agar perubahan isi editor dapat didengar dan akan mengeluarkan perubahan yang dilakukan pada editor kode.

Kode 2.7: Contoh kode event listener

```

1 <script src="/ace-builds/ace.js" type="text/javascript"></script>
2 <script>
3   var editor = ace.edit("editor");
4   editor.setTheme("ace/theme/monokai");
5   editor.session.setMode("ace/mode/javascript");
6
7   editor.session.on("change", (delta) => {
8     console.log(delta);
9     // Contoh Keluaran :
10    // {
11    //   action: "insert"
12    //   end: {row: 3, column: 5}
13    //   lines: ['a']
14    //   start: {row: 3, column: 4}
15    // }
16  });
17 </script>

```

Kode 2.7 akan menggunakan *event listener* **change** dalam kelas **EditSession**, dengan mengakses kelas **EditSession** melalui **editor** yang dinamakan **session**. Pada kelas tersebut akan dijalankan fungsi **on** dengan parameter “**change**” dan sebuah fungsi anonimous sebagai fungsi *callback* yang akan dicetak ke **console** isi perubahan pada editor kode.

2.6 Chart.js

Chart.js merupakan sebuah *library JavaScript open-source* untuk membuat visualisasi data bagan interaktif berbasis **canvas** dalam halaman web [8]. Chart.js memiliki fitur-fitur yang dapat digunakan untuk mendukung dan mempermudah visualisasi data dalam halaman web. Berikut merupakan beberapa fitur yang dimiliki oleh Chart.js:

- Chart.js menyediakan berbagai tipe bagan yang dapat digunakan dan juga memiliki opsi penyesuaian yang sering digunakan.
- Chart.js memiliki konfigurasi bawaan yang membuat visualisasi bagan sudah bagus.
- Chart.js mudah untuk diintegrasikan dalam sebuah halaman web.
- Chart.js menggunakan *canvas HTML5 rendering* yang membuat pembuatan bagan sangat cepat terutama jika data yang dimasukkan sangat besar.

Identik dengan *library* Ace untuk menintegrasikan *library* Chart.js dalam sebuah halaman web, *library* Chart.js dapat *dibuild* dan dimasukkan ke dalam direktori projek dan menambahkan file *JavaScript* bernama **chart.js** dalam direktori *dist* ke dalam halaman web menggunakan cara yang identik dengan cara memasukkan file *JavaScript* pada umumnya yaitu dengan cara sebagai berikut:

```
<script src="/chartjs/dist/chart.js" type="text/javascript"></script>
```

Setelah itu *library* Chart.js dapat digunakan dengan membuat sebuah kelas *JavaScript* baru bernama **Chart**. Untuk membuat kelas **Chart** dibutuhkan 2 argumen yaitu elemen *canvas* dalam HTML halaman web dan sebuah objek *JavaScript* yang dapat diisi dengan opsi-opsi yang diinginkan dengan menspesifikasi *key* dan *value* yang sesuai dengan opsi yang diinginkan. Berikut merupakan beberapa *key* dan *value* yang dapat digunakan ada dalam opsi *library* Chart.js:

- **type**
type hanya menerima sebuah kata yang menjadi tipe utama bagan yang dibuat oleh *library* Chart.js, tetapi tipe ini dapat berubah mengikuti data yang diberikan. Berikut merupakan beberapa tipe-tipe yang ada dalam *library* Chart.js:
 - **bar**
Bagan **bar** menyediakan cara untuk memvisualisasikan data sebagai batang vertikal maupun horizontal. Bagan ini biasanya digunakan untuk menunjukkan data tren dan perbandingan beberapa set data secara berdampingan.
 - **line**
Bagan **line** adalah cara memvisualisasikan data sebagai titik data yang disambungkan dengan garis. Identik dengan Bagan **bar**, Bagan **line** juga digunakan untuk menunjukkan data tren dan perbandingan beberapa set data secara berdampingan.
- **data**
data sendiri menerima *value* objek *JavaScript* dengan isi **labels** dan **dataset**. Kedua *key* menerima sebuah *array* dengan isi yang berbeda. **labels** hanya menerima sebuah *array* berisi teks untuk label data horizontal atau vertikal. **dataset** menerima *array* primitive type, *array* dengan isi *array*, dan *array objek*. *objek* dalam **dataset** menerima *key* **data** dan juga memiliki beberapa *key* opsi yaitu **label** untuk melabelkan data dalam horizontal maupun vertikal. *Key* **data** menerima *array* dengan isi *array objek* dengan data yang ingin divisualisasikan.
- **options**
Key options merupakan fitur utama dari kelas **Chart** dan hanya menerima sebuah objek *JavaScript* yang memiliki banyak *key* untuk menyesuaikan bagan yang dibuat oleh *library* Chart.js. Salah satu *key* dalam **options** adalah **scales** yang digunakan untuk mengatur data yang ditampilkan untuk aksis X dan Y. Salah satu pengaturannya adalah untuk menumpuk data dengan data yang sama di aksis yang sama yaitu dengan menggunakan *key* **stacked**.

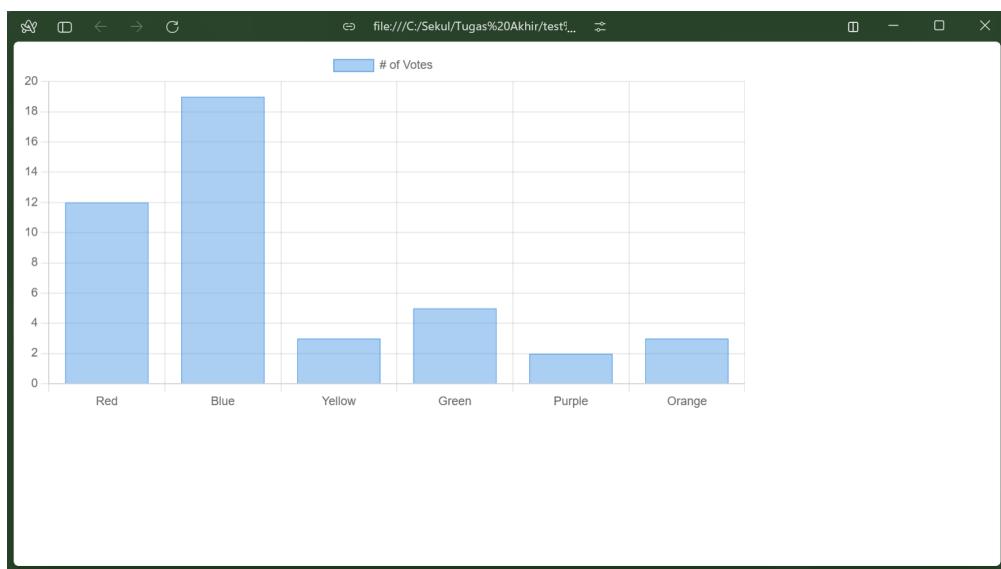
Kode 2.8: Contoh kode penggunaan Chart.js

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <body>
4 <div>
5   <canvas id="myChart"></canvas>
6 </div>
7
8 <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
9
10<script>
11   const ctx = document.getElementById('myChart');
12
13   new Chart(ctx, {
14     type: 'bar',
15     data: {
16       labels: ['Red', 'Blue', 'Yellow', 'Green', 'Purple', 'Orange'],
17       datasets: [
18         {
19           label: '# of Votes',
20           data: [12, 19, 3, 5, 2, 3],
21           borderWidth: 1
22         }
23       ],
24       options: {
25         scales: {
26           y: {
27             beginAtZero: true
28           }
29         }
30       }
31     });
32 </script>
33 </body>
34 </html>

```

Kode 2.8 merupakan contoh penggunaan *library* Chart.js pada sebuah *canvas* dengan id `myChart`. *Key options* pada contoh ini menggunakan `beginAtZero` yang membuat data dimulai dari nol. Gambar 2.3 merupakan hasil halaman web yang dibuka dalam *browser* dengan Kode 2.8.

Gambar 2.3: Hasil Halaman Web *Library* Chart.js

2.7 Code Churn

Code Churn merujuk pada perubahan yang dilakukan pada kode sumber dalam suatu perangkat lunak selama periode pengembangan tertentu. Perubahan ini dapat mencakup penambahan, penghapusan, atau perubahan pada baris kode. Konsep Code Churn digunakan untuk mengukur tingkat dinamika pengembangan perangkat lunak serta mengidentifikasi potensi ketidakstabilan atau area yang memerlukan perhatian lebih dalam basis kode.

Kode yang baru diubah biasanya mengalami beberapa perubahan. Frekuensi perubahan kode dalam periode waktu tertentu dapat bervariasi karena beberapa faktor dan pergantian kode dapat menjadi baik atau buruk tergantung pada kapan dan mengapa hal itu terjadi.

Seperti contohnya, teknisi sering menguji, menulis ulang, dan memeriksa beberapa solusi untuk suatu masalah, khususnya saat mereka berekspeten dengan solusi untuk tugas yang sedang dikerjakan. Dalam hal ini, pergantian kode adalah hal yang baik, karena merupakan hasil dari pemecahan masalah yang kreatif.

2.7.1 Code Churn Rate

Code Churn Rate adalah metrik yang menghitung frekuensi perubahan kode dalam suatu rentang waktu tertentu, biasanya dinyatakan dalam persentase atau jumlah perubahan per satuan waktu. Metrik ini membantu tim pengembang dan manajer proyek untuk memahami seberapa sering suatu kode dimodifikasi dan apakah perubahan tersebut bersifat wajar atau justru menunjukkan masalah dalam pengembangan. Rumus dasar untuk menghitung Code Churn Rate adalah:

$$\text{Code Churn Rate} = \frac{\text{Jumlah Kode yang Berubah}}{\text{Hasil Kode}} \times 100\%$$

2.8 Deteksi Kecurangan dalam Pengerjaan Tugas Pemrograman

Dalam konteks penilaian tugas pemrograman, terutama pada ujian berbasis online judge, kecurangan akademik dapat terjadi dalam berbagai bentuk, seperti plagiarisme, kolusi, atau penggunaan sumber eksternal secara tidak sah. Untuk mengidentifikasi perilaku mencurigakan, diperlukan pendekatan sistematis yang memanfaatkan analisis pola aktivitas pengguna selama proses pengodean. Subbab ini membahas lima indikator utama kecurangan dalam pemrograman, yaitu sebagai berikut:

2.8.1 Perubahan Kode yang sedikit

Code churn rate dapat mengukur seberapa sering kode diubah atau ditulis ulang selama proses pengembangan. Pada kondisi normal, mahasiswa cenderung melakukan banyak percobaan dan revisi, sehingga nilai code churn rate biasanya tinggi karena adanya penghapusan, penambahan, atau perbaikan kode. Namun, jika code churn rate terlalu rendah atau mendekati nol, hal ini dapat mengindikasikan kecurangan, seperti menyalin kode dari sumber eksternal tanpa melakukan modifikasi signifikan. Sebaliknya, code churn rate yang sangat tinggi juga perlu diperhatikan karena bisa menunjukkan kebingungan atau ketidakfisienan dalam menyelesaikan masalah.

2.8.2 Navigasi yang tidak wajar

Aktivitas navigasi yang tidak wajar, seperti sering berpindah tab, membuka aplikasi lain, atau kehilangan fokus dari IDE dalam waktu singkat, dapat menjadi tanda bahwa mahasiswa mencari bantuan dari luar. Pada pengembangan kode yang alami, mahasiswa biasanya fokus pada IDE dalam jangka waktu tertentu sebelum beralih ke referensi dokumentasi. Jika sistem merekam frekuensi perpindahan tab yang tinggi dalam waktu singkat, terutama saat ujian berlangsung, hal ini patut dicurigai sebagai upaya melakukan pencarian jawaban.

2.8.3 Tidak Membutuhkan Waktu untuk Berpikir

Proses pengerjaan masalah pemrograman yang alami biasanya melibatkan jeda untuk menganalisis, merancang solusi, atau melakukan pengujian pada kode program. Jika aktivitas menunjukkan ketikan yang sangat cepat tanpa jeda berarti, terutama pada kode yang kompleks, hal ini dapat mengindikasikan bahwa mahasiswa tidak benar-benar memahami kode yang ditulis. Kecurangan seperti *copy-paste* atau penggunaan bantuan AI sering kali menghasilkan kode yang tiba-tiba muncul tanpa melalui proses berpikir yang lama.

2.8.4 Tidak Ada Aktivitas *Debugging*

Debugging adalah bagian penting dalam pemrograman, di mana mahasiswa menguji, memperbaiki kesalahan, dan menyesuaikan input dan output kode program. Jika rekaman menunjukkan sedikit upaya *debugging* seperti jarang menjalankan program, tidak mengubah input percobaan, tidak banyak menulis ulang pada bagian tertentu, tidak menjalankan input yang sama untuk melihat kesalahan yang terjadi dan mendapatkan solusi sempurna hal ini dapat menandakan bahwa kode tidak dikembangkan secara mandiri. Kode yang didapat dari sumber lain biasanya dapat langsung berjalan tanpa banyak penyesuaian, berbeda dengan kode yang ditulis sendiri yang biasanya melalui beberapa iterasi perbaikan.

2.8.5 *Copy-Pasting*

Salah satu tanda kecurangan yang paling jelas adalah adanya *copy-paste*, di mana blok kode panjang tiba-tiba muncul dalam editor tanpa proses pengetikan bertahap.

BAB 3

ANALISIS

3.1 Analisis Sistem Kini

Seperti yang sudah dibahas pada subbab 2.1, SharIF Judge merupakan sebuah website judge yang dimodifikasi sesuai dengan kebutuhan Teknik Informatika UNPAR. Analisis diawali dengan MVC aplikasi SharIF Judge. Berikut merupakan hasil eksplorasi SharIF Judge yang telah dilakukan:

3.1.1 Model, View, Controller

SharIF Judge menggunakan *framework* CodeIgniter 3 yang berbasis arsitektur Model-View-Controller seperti yang dijelaskan pada subbab 2.2.1. Gambar 3.1 merupakan kelas diagram struktur MVC pada SharIF Judge. Berikut merupakan hasil eksplorasi dari struktur MVC pada SharIF Judge:

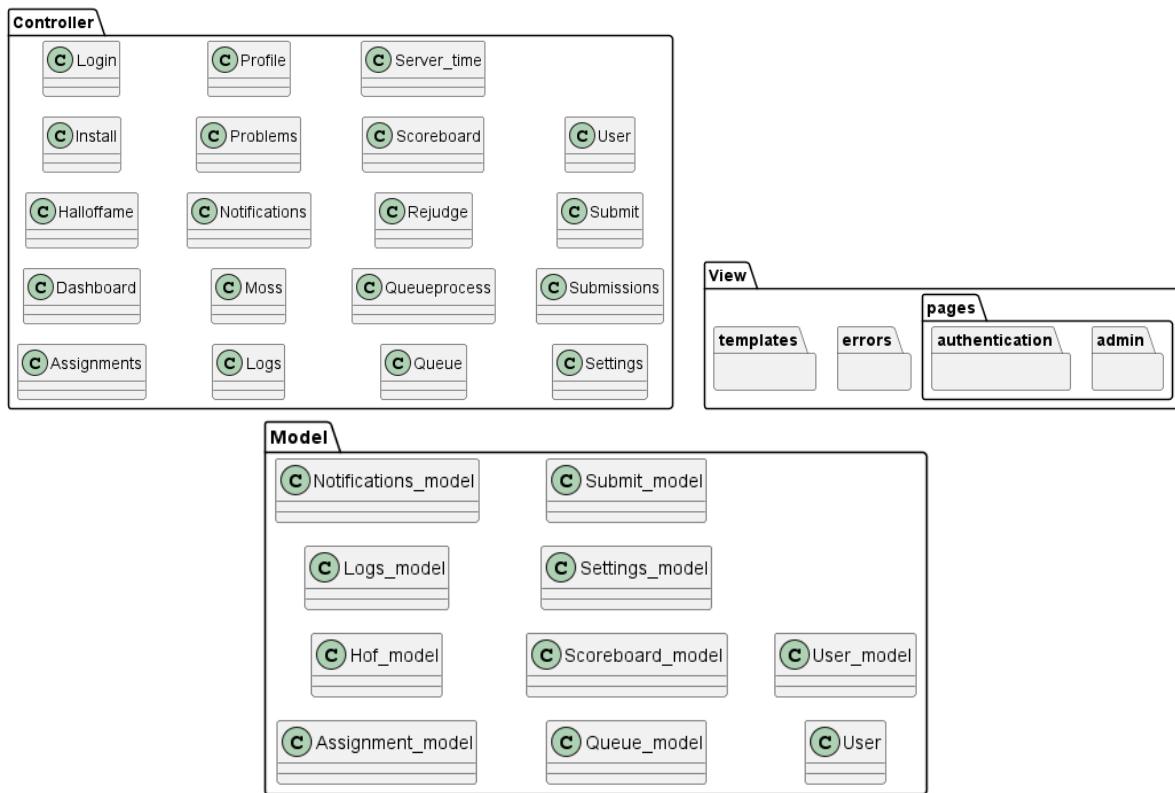
Model

Analisis MVC akan dimulai dengan *model* yang berada pada direktori `application/models`. Direktori *Model* berisi kelas-kelas yang digunakan untuk mengelola dan mengembalikan data dari *database*. Gambar 3.2 merupakan struktur kelas *model* dalam SharIF Judge. Berikut merupakan penjelasan dari kelas *model* dan fungsi-fungsinya yang terdapat pada SharIF Judge:

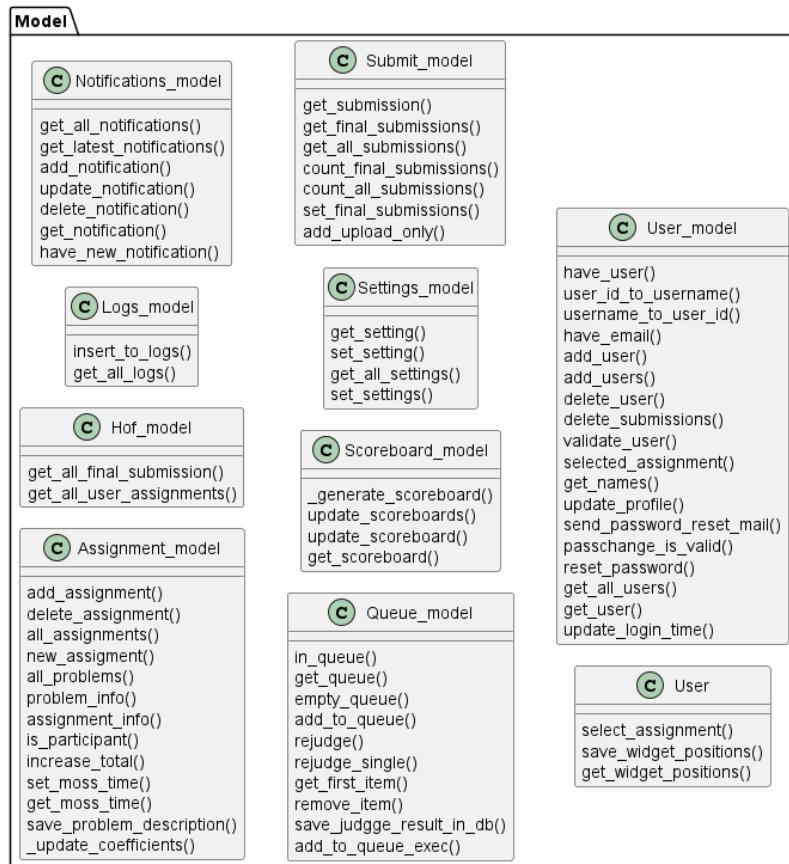
- `Assignment_model.php`

Model `Assignment_model.php` digunakan untuk mengelola tabel `assignments` dan mengembalikan informasi dari database yang digunakan dalam halaman `assignment` dan `problem`. Fungsi yang dimiliki adalah sebagai berikut:

- `add_assignment($id, $edit)`
Menambahkan atau memperbarui sebuah *assignment*.
- `delete_assignment($assignment_id)`
Menghapus sebuah *assignment*.
- `all_assignments()`
Mengembalikan daftar semua *assignment* dan informasinya.
- `new_assignment_id()`
Mendapatkan nomor terkecil dan dapat digunakan sebagai *id assignment* terbaru.
- `all_problems($assignment_id)`
Mengembalikan daftar semua *problems* dari sebuah *assignment*.
- `problem_info($assignment_id, $problem_id)`
Mengembalikan semua informasi sebuah *problem*
- `assignment_info($assignment_id)`
Mengembalikan semua informasi sebuah *assignment*
- `is_participant($participants, $username)`
Mengembalikan *boolean* apakah `$username` terdapat dalam `$participants`.
- `increase_total_submits($assignment_id)`
Menambahkan jumlah *total submits* sebanyak satu pada sebuah *assignment*.



Gambar 3.1: Struktur MVC pada SharIF Judge



Gambar 3.2: Struktur Kelas Model pada SharIF Judge

- `set_moss_time($assignment_id)`
Memperbarui “*Moss Update Time*” pada sebuah *assignment*.
- `get_moss_time($assignment_id)`
Mengembalikan “*Moss Update Time*” pada sebuah assignment.
- `save_problem_description($assignment_id, $problem_id, $text, $type)`
Menambahkan atau memperbarui deskripsi pada sebuah *problem*.
- `_update_coefficients($a_id, $extra_time, $finish_time, $new_late_rule)`
Memperbarui koefisien dari sebuah *assignment*.
- **Hof_model.php**
Model ini digunakan untuk mengembalikan informasi yang digunakan dalam *hall of fame* dari tabel *submissions*. Fungsi yang dimiliki adalah sebagai berikut:
 - `get_all_final_submission()`
Mengembalikan seluruh total nilai *final submission* untuk semua pengguna.
 - `get_all_user_assignments($username)`
Mengembalikan nilai *final submission* pada semua problem untuk pengguna tertentu.
- **Logs_model.php**
Model ini berfungsi untuk mengelola tabel *logins* dan mengembalikan catatan *login*. Fungsi yang dimiliki adalah sebagai berikut:
 - `insert_to_logs($username, $ip_address)`
Mencatat *login* pengguna dan menghapus catatan jika melebihi 24 jam.
 - `get_all_logs()`
Mengembalikan semua catatan *login*.
- **Notifications_model.php**
Model ini digunakan untuk mengelola tabel *notifications*. Fungsi yang dimiliki oleh model *Notifications_model.php* adalah sebagai berikut:
 - `get_all_notifications()`
Mengembalikan semua *notifications*.
 - `get_latest_notifications()`
Mengembalikan 10 *notifications* terbaru.
 - `add_notification($title, $text)`
Menambahkan *notification* baru.
 - `update_notification($id, $title, $text)`
Memperbarui sebuah *notification*.
 - `delete_notification($id)`
Menghapus sebuah *notification*.
 - `get_notification($notif_id)`
Mengembalikan sebuah *notification*.
 - `have_new_notification($time)`
Mengembalikan nilai *boolean* yang menunjukkan apakah terdapatnya *notification* baru.
- **Queue_model.php**
Model ini digunakan untuk mengelola tabel *queue* dan menampilkan data *queue*. Fungsi yang dimiliki adalah sebagai berikut:
 - `in_queue($username, $assignment, $problem)`
Fungsi ini digunakan untuk mengembalikan sebuah *boolean* yang menyatakan bahwa *username* masih memiliki *queue* dalam sebuah *problem*.
 - `get_queue()`
Mengambil semua *submission queue*.
 - `empty_queue()`
Menghapus semua *queue*.
 - `add_to_queue($submit_info)`
Menambahkan sebuah *submission* ke dalam *queue*.

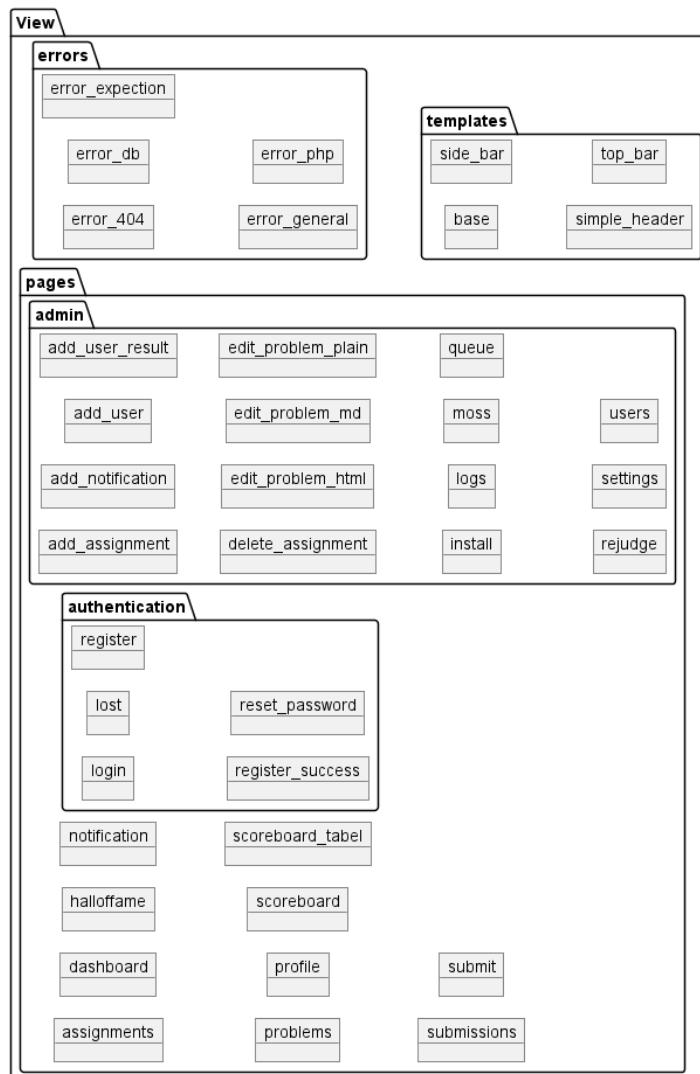
- `rejudge($assignment_id, $problem_id)`
Fungsi ini digunakan untuk menambahkan seluruh *submissions* dalam sebuah *problem* ke dalam *queue* untuk dinilai ulang.
 - `rejudge_single($submission)`
menambahkan sebuah *submission* ke dalam *queue* untuk dinilai ulang.
 - `get_first_item()`
Mengembalikan *item* pertama dalam tabel *queue*.
 - `remove_item($username, $assignment, $problem, $submit_id)`
Menghapus sebuah *item* tertentu dalam tabel *queue*.
 - `save_judge_result_in_db ($submission, $type)`
Menyimpan hasil penilaian *judge* ke dalam *database*.
 - `add_to_queue_exec($submit_info)`
Fungsi ini digunakan untuk menambahkan sebuah *dummy submission* dari IDE yang digunakan hanya untuk dijalankan ke dalam *queue*.
- **Scoreboard_model.php**
Model ini digunakan untuk mengelola tabel **scoreboard**. Fungsi yang dimiliki oleh *model* **Scoreboard_model.php** adalah sebagai berikut:
 - `_generate_scoreboard($assignment_id)`
Menghasilkan *scoreboard* untuk sebuah *assignment* dari nilai akhir semua *submission*.
 - `update_scoreboards()`
Memperbarui *scoreboard* untuk semua *assignment*.
 - `update_scoreboard($assignment_id)`
Memperbarui *scoreboard* untuk sebuah *assignment*.
 - `get_scoreboard($assignment_id)`
Mengembalikan *scoreboard* pada sebuah *assignment*.
 - **Settings_model.php**
Model ini digunakan untuk mengelola tabel **settings**. Fungsi yang dimiliki oleh *model* **Settings_model.php** adalah sebagai berikut:
 - `get_setting($key)`
Mengembalikan nilai dari sebuah `$key` pada tabel **settings**.
 - `set_setting($key, $value)`
Memperbarui nilai dari pada *setting* `$key`.
 - `get_all_settings()`
Mengembalikan seluruh *settings*.
 - `set_settings($settings)`
Memperbarui seluruh nilai perubahan *settings*.
 - **Submit_model.php**
Model ini digunakan untuk mengelola tabel **submission**. Fungsi yang dimiliki oleh *model* **Submit_model.php** adalah sebagai berikut:
 - `get_submission($username, $assignment, $problem, $submit_id)`
Mengembalikan sebuah baris data *submission* tertentu.
 - `get_final_submissions($a_id, $u_vl, $uname, $p_num, $fil_u, $fil_prob)`
Mengembalikan seluruh *final submission* pada sebuah *assignment*. pengguna dengan role *student* hanya dapat melihat *final submission* dirinya sendiri.
 - `get_all_submissions($a_id, $u_vl, $uname, $p_num, $fil_u, $fil_prob)`
Mengembalikan seluruh *submission* pada sebuah *assignment*. pengguna dengan role *student* hanya dapat melihat *submission* dirinya sendiri.
 - `count_final_submissions($a_id, $u_vl, $uname, $fil_u, $fil_prob)`
Mengembalikan jumlah *final submission* pada sebuah *assignment*.
 - `count_all_submissions($a_id, $u_vl, $uname, $fil_u, $fil_prob)`
Mengembalikan jumlah *submission* pada sebuah *assignment*.

- `set_final_submission($username, $assignment, $problem, $submit_id)`
Memperbaharui sebuah *submission* menjadi *final submission*.
- `add_upload_only($submit_info)`
Menyimpan hasil *upload only problem* ke dalam tabel *database*.
- **User.php**
Model ini digunakan untuk menyimpan *settings* sebuah pengguna. Fungsi yang dimiliki oleh *model User.php* adalah sebagai berikut:
 - `select_assignment($assignment_id)`
Menyimpan *assignment* yang dipilih oleh pengguna.
 - `save_widget_positions($positions)`
Menyimpan posisi *widget* sebuah pengguna.
 - `get_widget_positions()`
Mendapatkan posisi *widget* sebuah pengguna.
- **User_model.php**
Model *User_model.php* digunakan untuk mengelola tabel *users*. Fungsi yang dimiliki oleh *model User_model.php* adalah sebagai berikut:
 - `have_user($username)`
Mengembalikan sebuah *boolean* yang menyatakan *\$username* sudah ada pada *database*.
 - `user_id_to_username($user_id)`
Mengembalikan *username* dari *\$user_id*.
 - `username_to_user_id($username)`
Mengembalikan *user id* dari *\$username*.
 - `have_email($email, $username)`
Mengembalikan *boolean* yang menyatakan jika pengguna memiliki *email* pada *database*.
 - `add_user($username, $email, $display_name, $password, $role)`
Menambahkan satu pengguna baru ke dalam *database*.
 - `add_users($text, $send_mail, $delay)`
Menambahkan banyak pengguna baru ke dalam *database*.
 - `delete_user($user_id)`
Menghapus sebuah pengguna dalam *database*.
 - `delete_submissions($user_id)`
Mendelete semua *submissions* yang di *submit* oleh sebuah pengguna.
 - `validate_user($username, $password)`
Mengembalikan sebuah *boolean* yang menyatakan bahwa *\$password* dan *\$username*
 - `selected_assignment($username)`
Mengembalikan *assignment* yang dipilih oleh *\$username*.
 - `get_names()`
Mengembalikan semua *display name* pada tabel *users*.
 - `update_profile($user_id)`
Memperbaharui nama, email, password, atau role sebuah pengguna.
 - `send_password_reset_mail($email)`
Mengirimkan *link reset password* ke email pengguna yang dapat dipakai selama 1 jam.
 - `passchange_is_valid($passchange_key)`
Fungsi ini digunakan untuk mengembalikan sebuah *boolean* yang menyatakan bahwa *link reset password* yang diberikan dapat digunakan atau tidak.
 - `reset_password($passchange_key, $newpassword)`
Memperbaharui *password* dengan divalidasinya *password change key*.
 - `get_all_users()`
Mengembalikan seluruh pengguna pada tabel *users*.
 - `get_user($user_id)`
Mengembalikan sebuah pengguna yang memiliki id *\$user_id*.

- `update_login_time($username)`
Memperbaharui catatan *login* untuk sebuah pengguna.

View

View merupakan tampilan yang menjadi perantara antara pengguna dan *sistem*. Pada SharIF Judge, *View* disimpan pada direktori `application/views` dan dibagi menjadi 3 direktori terpisah yaitu errors, pages, dan template. Gambar 3.3 merupakan struktur direktori *view* beserta *view* yang terdapat pada direktorinya dalam SharIF Judge.



Gambar 3.3: Struktur Direktori View pada SharIF Judge

Berikut merupakan penjelasan mengenai direktori penyimpanan untuk *view* pada SharIF Judge.

- **errors**

Pada direktori `errors`, berisi tampilan halaman *error* jika terjadi error pada penggunaan SharIF Judge. Berikut merupakan *views* yang terdapat pada direktori `errors`:

- `error_404`
- `error_db`
- `error_exception`
- `error_general`
- `error_php`

- **pages**

Pada direktori *pages*, berisi tampilan halaman-halaman utama. *pages* juga memiliki dua direktori selain halaman-halama. Berikut merupakan *views* dan direktori pada *pages*:

- **pages/admin**

Direktori *admin* berisi tampilan halaman khusus untuk *role admin*. Berikut merupakan *views* yang terdapat pada direktori *admin*:

- * **add_assignment.twig**
 - * **add_notification.twig**
 - * **add_user.twig**
 - * **add_user_result.twig**
 - * **delete_assignment.twig**
 - * **edit_problem_html.twig**
 - * **edit_problem_md.twig**
 - * **edit_problem_plain.twig**
 - * **install.twig**
 - * **logs.twig**
 - * **moss.twig**
 - * **queue.twig**
 - * **rejudge.twig**
 - * **settings.twig**
 - * **users.twig**

- **pages/authentication**

Direktori *authentication* berisi tampilan halaman khusus untuk *authentication* seperti halaman direktori *Login*. Berikut merupakan *views* yang terdapat pada direktori *admin*:

- * **login.twig**
 - * **lost.twig**
 - * **register.twig**
 - * **register_success.twig**
 - * **reset_password.twig**

- **assignments.twig**
- **dashboard.twig**
- **halloffame.twig**
- **notification.twig**
- **problems.twig**
- **profile.twig**
- **scoreboard.twig**
- **scoreboard_tabel.twig**
- **submissions.twig**
- **submit.twig**

- **templates**

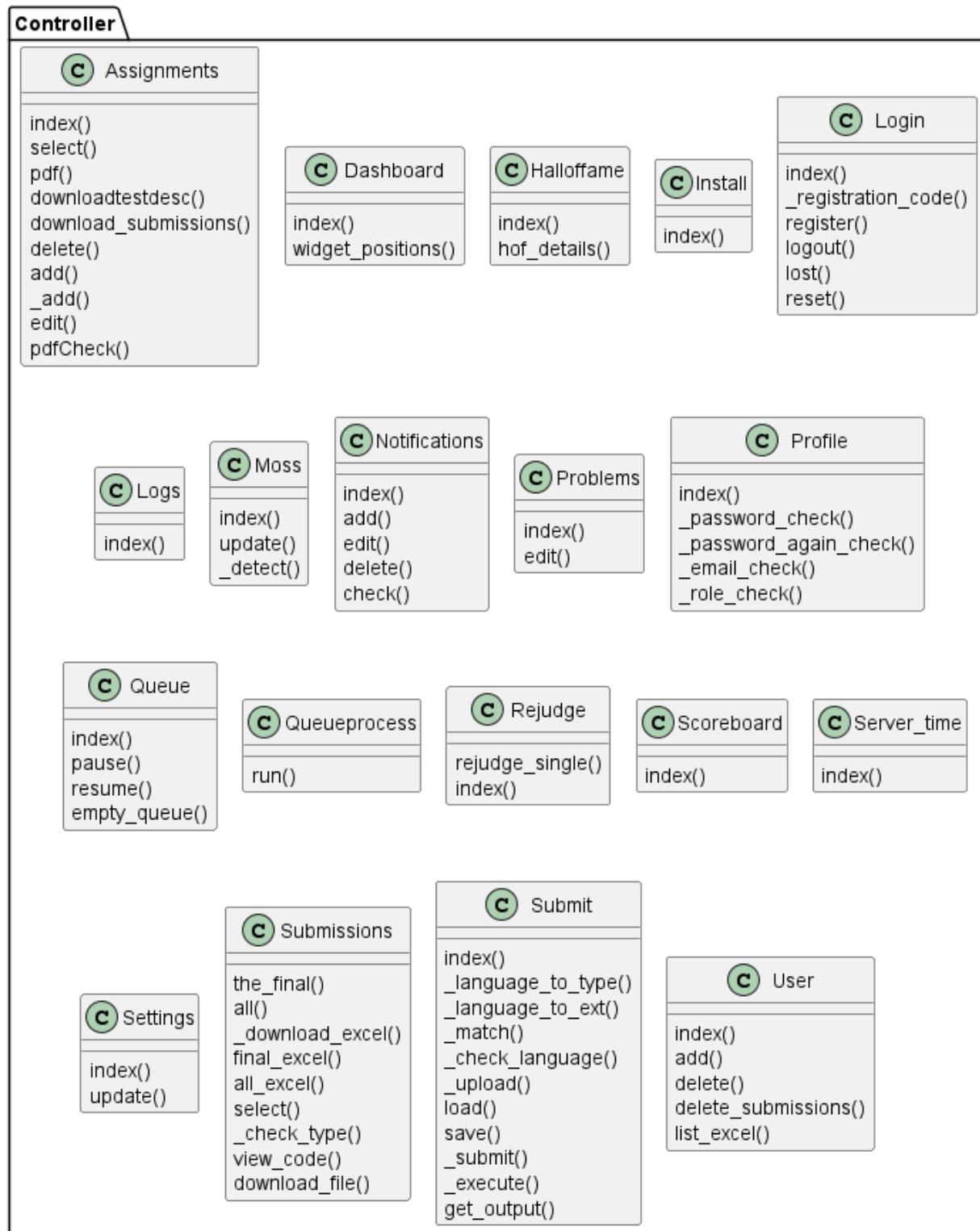
Pada direktori *templates*, berisi tampilan yang digunakan berulang oleh halaman utama seperti *header*, *sidebar*, dan *base*. Berikut merupakan *views* yang terdapat pada *templates*:

- **base.twig**
 - **side_bar.twig**
 - **simple_header.twig**
 - **top_bar.twig**

Controller

Pada bagian analisis MVC terakhir, terdapat *controller* yang berada pada direktori *application/controller*. Seperti yang dijelaskan pada subbab 2.2.1, *Controller* digunakan sebagai perantara antara *model*, *view*, dan *resources* lainnya yang dibutuhkan saat membuat sebuah web page. Direktori

controller berisi kelas-kelas yang akan mengolah data yang didapat pada *model* dan menyatukan data tersebut ke dalam *views* yang akan ditampilkan kepada pengguna. Pada setiap kelas *controller*, terdapat fungsi `index()` yang menjadi fungsi utama saat kelas di akses oleh pengguna. Gambar 3.4 merupakan struktur kelas *controller* yang terdapat pada SharIF Judge.



Gambar 3.4: Struktur Kelas Controller pada SharIF Judge

Berikut merupakan file *controller* dan penjelasan fungsinya pada SharIF Judge:

- **Assignments.php**

Berikut fungsi dengan penjelasannya pada *controller* **Assignments.php**:

- **select()**

Memilih *assignment* yang ditampilkan pada *top bar* menggunakan *ajax request*.

- **pdf(\$assignment_id, \$problem_id, \$no_download)**

Mengunduh *assignment* atau *problem* dalam bentuk PDF ke browser.

- **downloadtestsdesc(\$assignment_id)**

Mengunduh dan mencompress data uji dan deskripsi sebuah *assignment*.

- **download_submissions(\$type, \$assignment_id)**

Mengunduh semua *final submission* pada semua *assignment*.

- **delete(\$assignment_id)**

Menghapus sebuah *assignment*.

- **add()**

Mendapatkan *input* dari pengguna untuk menambah atau memperbarui *assignment*.

- **_add()**

Menambahkan atau memperbarui sebuah *assignment*.

- **edit(\$assignment_id)**

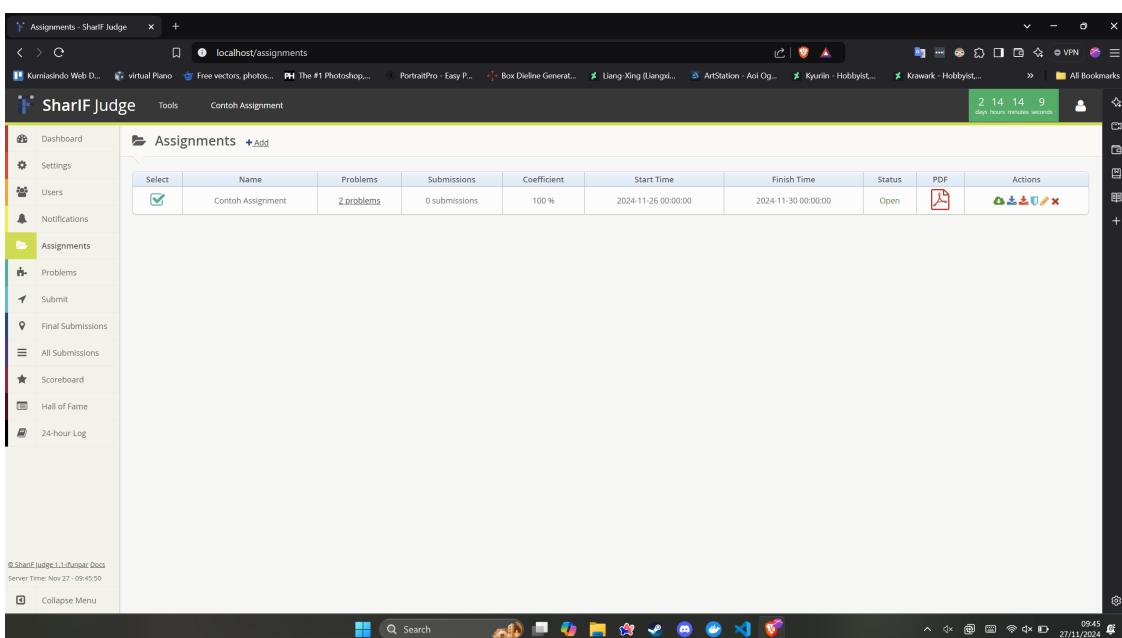
Menandai *assignment* yang akan di *edit* dan memanggil fungsi *add*.

- **pdfCheck(\$assignment_id, \$problem_id)**

Melakukan validasi ketersediaan pdf pada sebuah *assignment* atau pada sebuah *problem*.

- **index()**

Mengambil data dari **Assignment_model** dan menyimpan data dan mengembalikan *views assignments.twig*. Gambar 3.5 menunjukkan hasil halaman Assignment.



Gambar 3.5: Halaman Assignments

- **Dashboard.php**

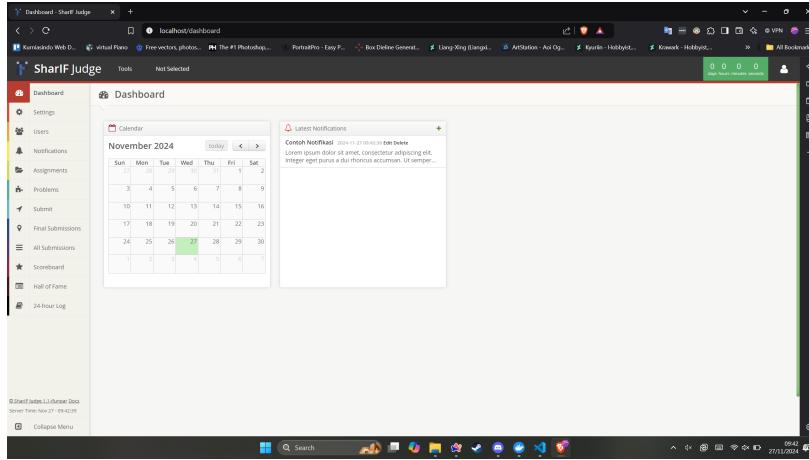
Berikut fungsi dengan penjelasannya pada *controller* **Dashboard.php**:

- **widget_positions()**

menggunakan *ajax request* untuk menyimpan posisi *widget*.

- **index()**

Mendapatkan data dari beberapa model yaitu `Assignment_model`, `Settings_model`, `User`, dan `Notifications_model`. Data akan dimasukkan ke dalam `dashboard.twig` yang akan dikembalikan ke pengguna. Gambar 3.6 menunjukkan hasil halaman Dashboard yang dapat diakses oleh semua *role*.



Gambar 3.6: Halaman Dashboard

- **HallofFame.php**

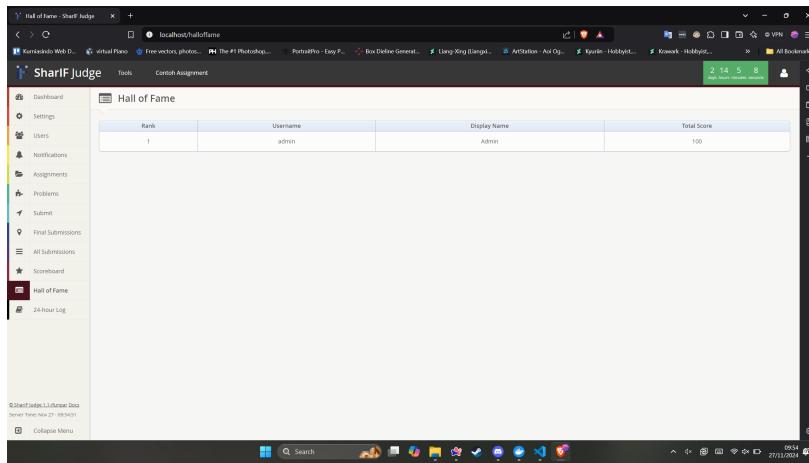
Berikut fungsi dengan penjelasannya pada controller `Haloffame.php`:

- **hof_details()**

Menampilkan nilai akhir semua *problem* dan *assignments* pada sebuah pengguna.

- **index()**

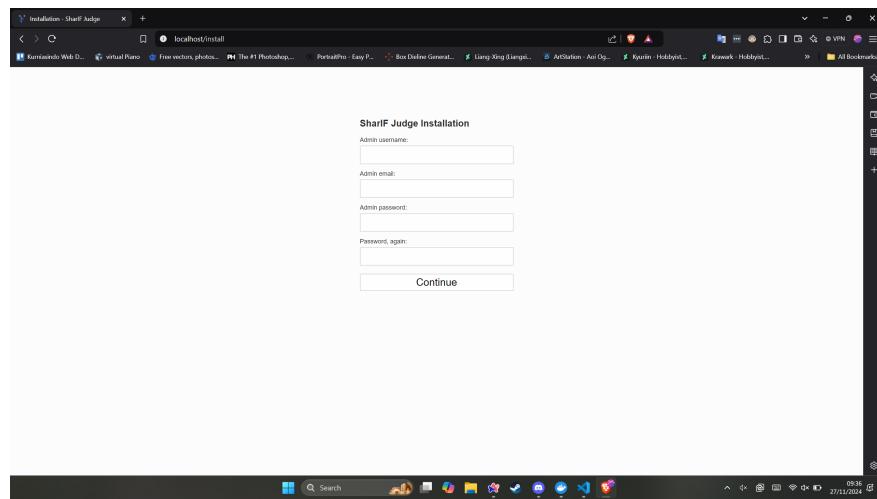
Mendapatkan data dari `Hof_model` dan mengembalikan `view halloffame.twig`. Gambar 3.7 menunjukkan hasil halaman Hall of Fame yang dapat diakses oleh semua *role*.



Gambar 3.7: Halaman Hall of Fame

- **Install.php**

Pada controller `Install.php` hanya ada satu fungsi yang menangani pembuatan seluruh tabel pada *database* yang dibutuhkan oleh SharIF Judge. Setelah membuat *database* akan mengembalikan `view install.twig` yang dapat diisi oleh pengguna tentang data pengguna dengan role *admin* saat *form* di kirim. Gambar 3.8 menunjukkan hasil halaman Install.



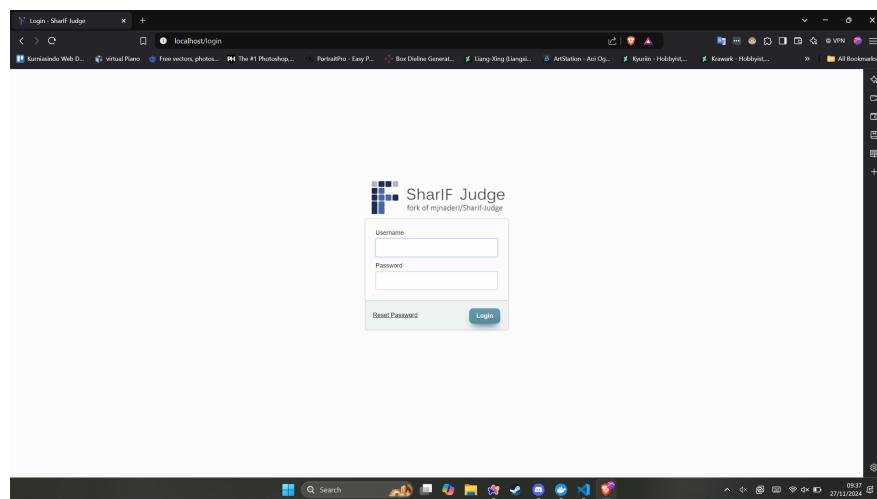
Gambar 3.8: Halaman Install

- **Login.php**

Berikut fungsi dengan penjelasannya pada *controller Login.php*:

- **index()**

Mengembalikan *view login.twig* dan memeriksa username dan password pada *form* saat dikirim. Gambar 3.9 menunjukkan hasil halaman Login.



Gambar 3.9: Halaman Login

- **_registration_code(\$code)**

Melakukan validasi kode registrasi.

- **register()**

Menunjukkan halaman *register.twig* dan membuat pengguna baru.

- **logout()**

Melakukan *Log out* dan mengalihkan ke halaman *login*.

- **lost()**

Mengirimkan email *reset password*.

- **reset(\$passchange_key)**

Melakukan *reset password* dengan halaman *reset_password.twig*.

- **Logs.php**

Pada *controller Logs.php* hanya memiliki satu fungsi yaitu *index()*, dimana fungsi tersebut akan mendapatkan data dari *Logs_model* dan menampilkan halaman *logs.twig*. Gambar

3.10 menunjukkan halaman Log yang dinamakan halaman 24-Hour Log.

#	Login ID	Username	IP Address	Login Time	Log from different IP (> 24 hours)
1	2	admin	172.20.0.1	2024-11-27 02:48:17	
2	1	admin	172.20.0.1	2024-11-27 02:38:45	

Gambar 3.10: Halaman 24-Hour Log

- **Moss.php**

Berikut fungsi dengan penjelasannya pada controller *Moss.php*:

- *index()*

Mengambil data dan memasukkannya ke dalam *view moss.twig*. Gambar 3.11 merupakan hasil halaman *moss*. Fungsi *_detect* juga akan dijalankan saat *form* terkirim.

Moss user id
You have not entered your Moss user id.
Read [this page](#) and register for Moss, then find your user id in the script sent to your email by Moss and enter your user id here.
Your Moss user id:

Detect similar submissions of assignment "Contoh Assignment":
You can send final submissions of assignment "Contoh Assignment" to Moss by clicking on this button.
Zip and PDF files will not be sent.
It may take a minute. Please be patient.
Detect similar codes!

Moss results for assignment "Contoh Assignment":
Links will expire after some time. (last update: Never)
• Problem 1: Link Not Found.
• Problem 2: Link Not Found.

Gambar 3.11: Halaman Moss

- *update(\$assignment_id)*

Memperbaharui *settings* dari masukkan *moss_userid* pengguna.

- *_detect(\$assignment_id)*

Melakukan pemeriksaan kesamaan kode dengan Moss.

- **Notifications.php**

Berikut fungsi dengan penjelasannya pada controller *Notifications.php*:

- *add()*

Menambahkan atau memperbaharui sebuah *notification*.

- *edit(\$notif_id)*

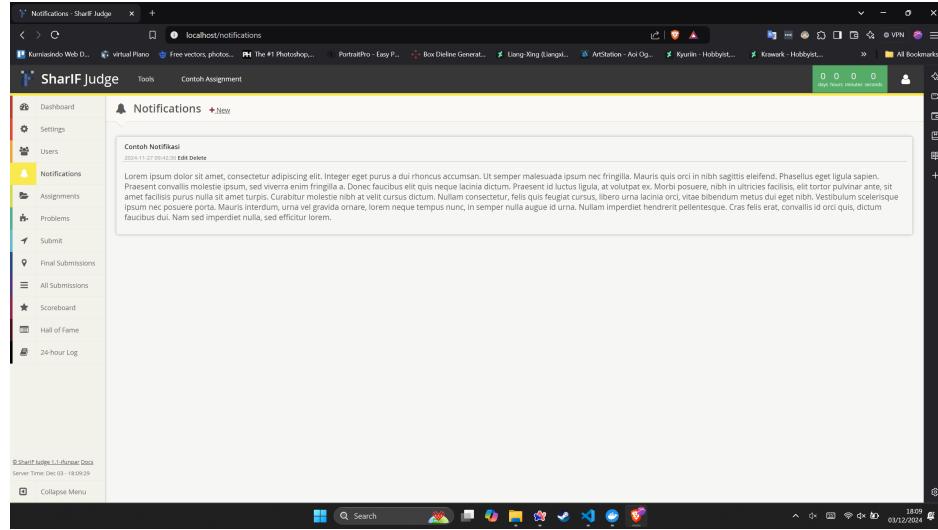
Menandai *notification* yang akan di *edit* dan memanggil fungsi *add*.

- *delete()*

Menghapus sebuah *notification*.

- **check()**
menggunakan *ajax request* untuk mengetahui ketersediaan *notification* baru.
- **index()**

Mendapatkan data dari dua model yaitu **Assignment_model** dan **Notifications_model**. Data akan dimasukkan ke dalam *view notifications.twig* yang akan dikembalikan ke pengguna. Gambar 3.12 menunjukkan hasil halaman *Notifications*.



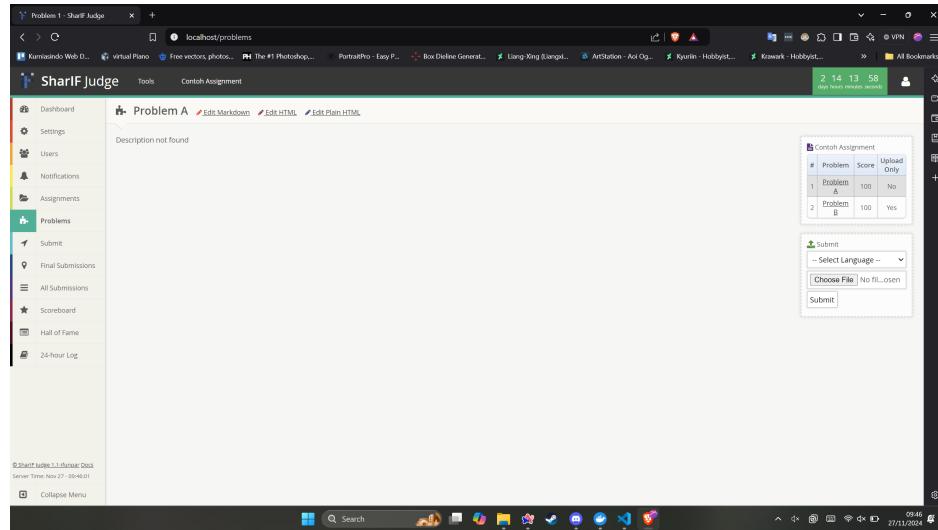
Gambar 3.12: Halaman Notifications

• Problems.php

Berikut fungsi dengan penjelasannya pada *controller Notifications.php*:

- **edit()**
Memperbaharui deskripsi sebuah *problem* dalam bentuk *html* atau *markdown*.
- **index()**

Mendapatkan data *problem* dari berbagai *model* sesuai dengan *assignment* yang dipilih dan menyimpan data tersebut pada halaman *problems.twig* yang akan ditampilkan ke pengguna. Gambar 3.13 menunjukkan hasil halaman *Problems*.



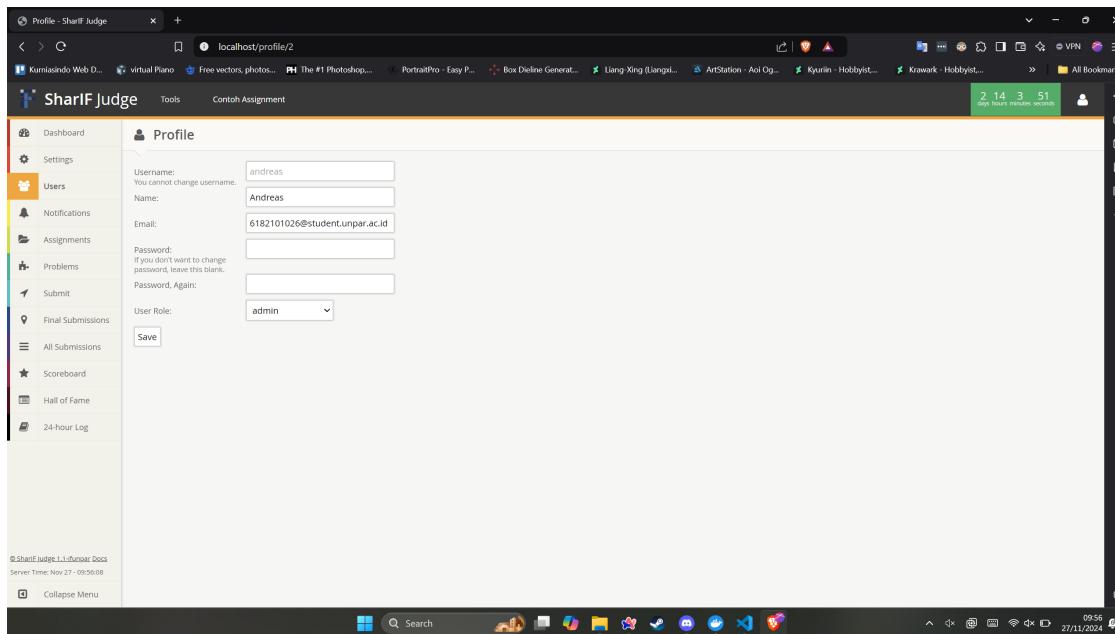
Gambar 3.13: Halaman Problems

- **Profile.php**

Berikut fungsi dengan penjelasannya pada *controller Profile.php*:

- **index()**

Mendapatkan data dari berbagai *model* terutama dari pengguna yang akan dimasukkan ke dalam *view profile.twig*. Fungsi ini juga menangani pengiriman *form* pembaharuan data pengguna pengguna. Gambar 3.14 menunjukkan hasil halaman Profile.



Gambar 3.14: Halaman Profile

- **_password_check(\$str)**

Melakukan validasi *input password*.

- **_password_again_check(\$str)**

Melakukan validasi *input tulisan pengulangan password*.

- **_email_check(\$str)**

Melakukan validasi ketersediaan email pada *database*.

- **_role_check(\$str)**

Melakukan validasi *role* pengguna saat ingin mengubah *role* pengguna.

- **Queue.php**

Berikut fungsi dengan penjelasannya pada *controller Profile.php*:

- **pause()**

Memberhentikan proses *queue*.

- **resume()**

Melanjutkan proses *queue*.

- **empty_queue()**

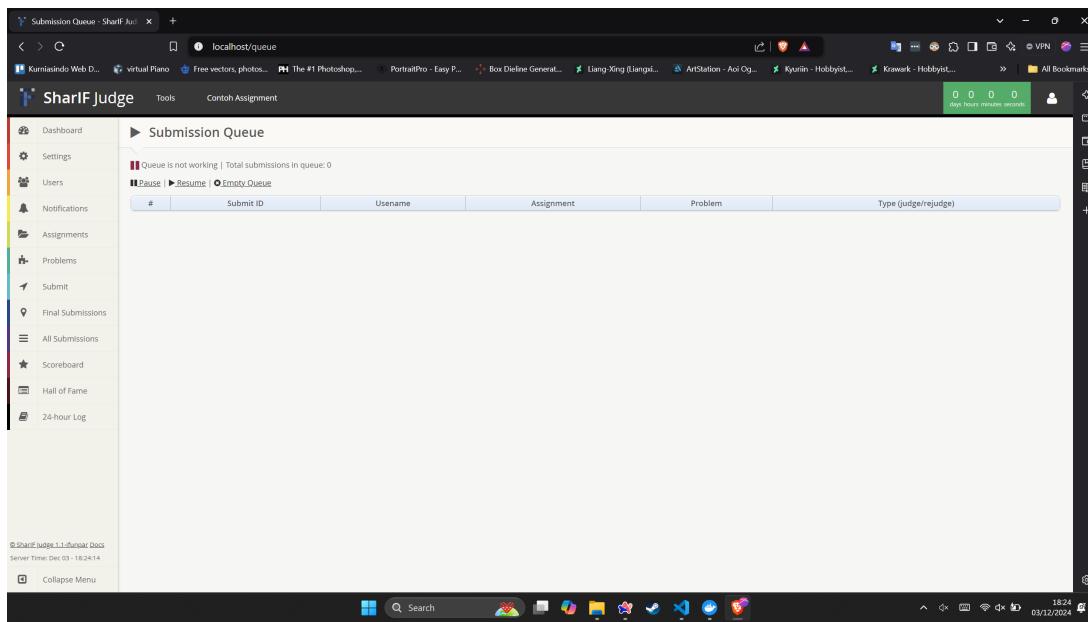
Menghapus semua *queue* yang ada.

- **index()**

Mendapatkan data dari *model Queue*, *Assignments_model*, dan *Settings_model* yang dipakai dalam *view queue.twig* dan ditampilkan kepada pengguna. Gambar 3.15 menunjukkan hasil halaman Queue.

- **Queueprocess.php**

Controller Queueprocess.php hanya memiliki satu fungsi yaitu *run()* yang akan menjalankan *queue* satu per satu menggunakan *bash*.



Gambar 3.15: Halaman Queue

- **Rejudge.php**

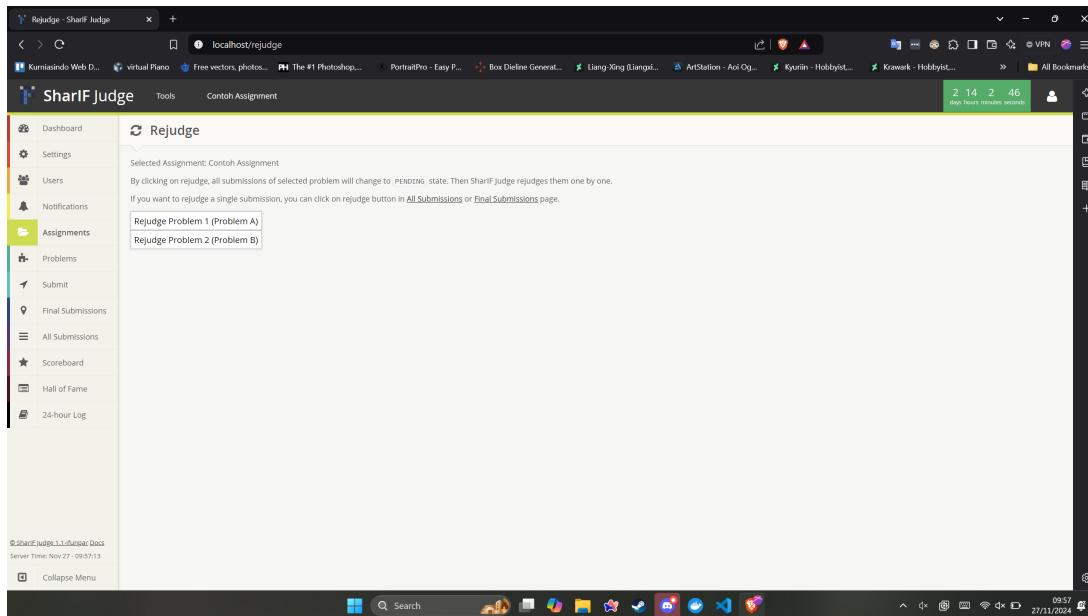
Berikut fungsi dengan penjelasannya pada *controller Profile.php*:

- **rejudge_single()**

Melakukan *rejudge* untuk satu buah *submission*.

- **index()**

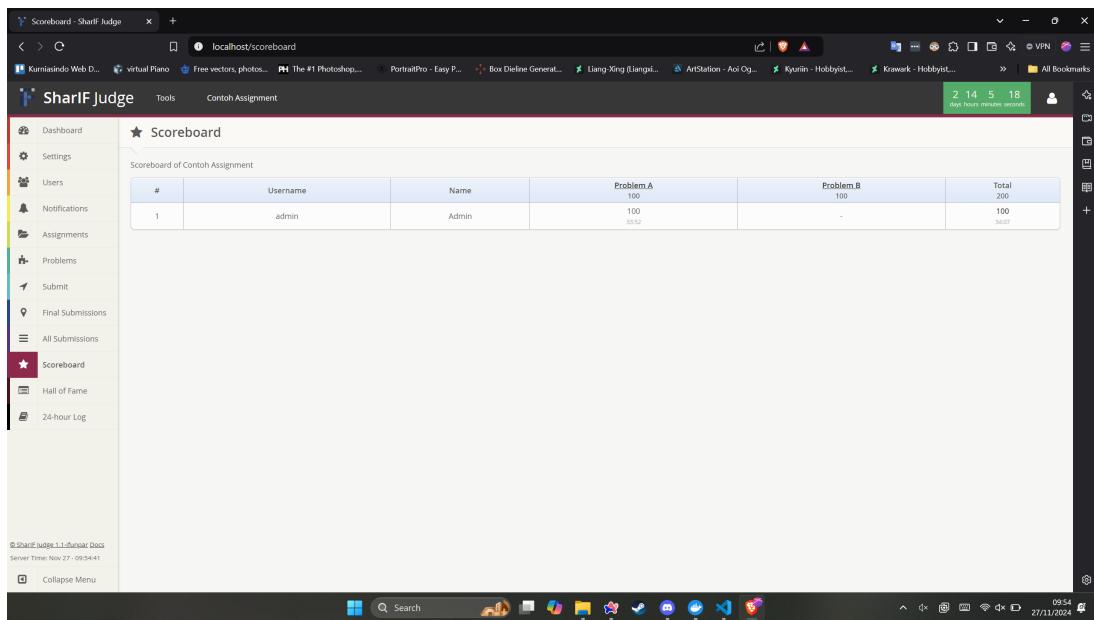
Mendapatkan data dan menampilkan *rejudge.twig*. Fungsi ini juga dapat melakukan *rejudge* pada sebuah *problem* tertentu. Gambar 3.16 menunjukkan halaman Rejudge.



Gambar 3.16: Halaman Rejudge

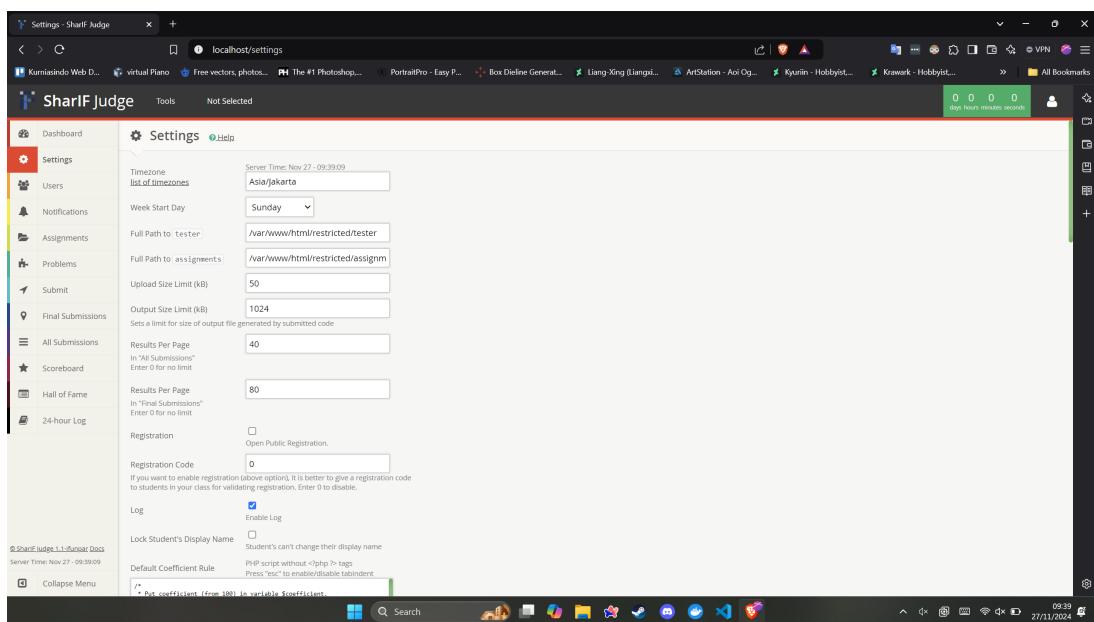
- **Scoreboard.php**

Controller Queueprocess.php hanya memiliki satu fungsi yaitu *index()* yang akan menampilkan *view scoreboard.twig* dengan data yang didapat dari model *Scoreboard_model*. Gambar 3.17 menunjukkan hasil halaman Scoreboard.



Gambar 3.17: Halaman Scoreboard

- **Server_time.php**
Controller Queueprocess.php hanya memiliki satu fungsi yaitu `index()` yang akan mencetak waktu pada *server*, waktu akan digunakan untuk sinkronisasi waktu.
- **Settings.php**
Berikut fungsi dengan penjelasannya pada *controller Settings.php*:
 - `update()`
Memperbaharui *settings* dari masukkan pengguna.
 - `index()`
Mendapatkan data dari **Settings_model** dan menampilkan *view settings.twig*. Jika terdapat *error setting* pada sistem, akan ditampilkan juga pada *view* tersebut. Gambar 3.18 menunjukkan hasil halaman Users.



Gambar 3.18: Halaman Settings

Final	ID	Username	Name	Problem	Submit Time	Score				Language	Status	Code	Log	Actions
						Score	Delay %	Final Score	Language					
✓	4	admin	Admin	2	2024-11-27 09:54:16	0	No delay 100%	0	Java	Uploaded	<button>Code</button>	<button>Log</button>	<button>...</button>	<button>↻</button>
✓	3	admin	Admin	1	2024-11-27 09:52:35	100	No delay 100%	100	Java	<button>100</button>	<button>Code</button>	<button>Log</button>	<button>...</button>	<button>↻</button>
○	2	admin	Admin	1	2024-11-27 09:51:48	100	No delay 100%	100	C++	<button>100</button>	<button>Code</button>	<button>Log</button>	<button>...</button>	<button>↻</button>
○	1	admin	Admin	1	2024-11-27 09:51:33	0	No delay 100%	0	C++	<button>0</button>	<button>Code</button>	<button>Log</button>	<button>...</button>	<button>↻</button>

Gambar 3.19: Halaman All Submissions

- **Submissions.php**

Berikut fungsi dengan penjelasannya pada *controller Submissions.php*:

- **all()**

Fungsi ini akan mendapatkan data dari *Submit_model* untuk mendapatkan seluruh *submission* dan menampilkan halaman *submission.twig* berisi semua *submission*. Gambar 3.19 menunjukkan halaman All Submissions .

- **the_final()**

Fungsi ini akan mendapatkan data dari *Submit_model* untuk mendapatkan *final submission* dan menampilkan halaman *submission.twig* berisi *final submission*. Gambar 3.20 menunjukkan halaman Final Submissions .

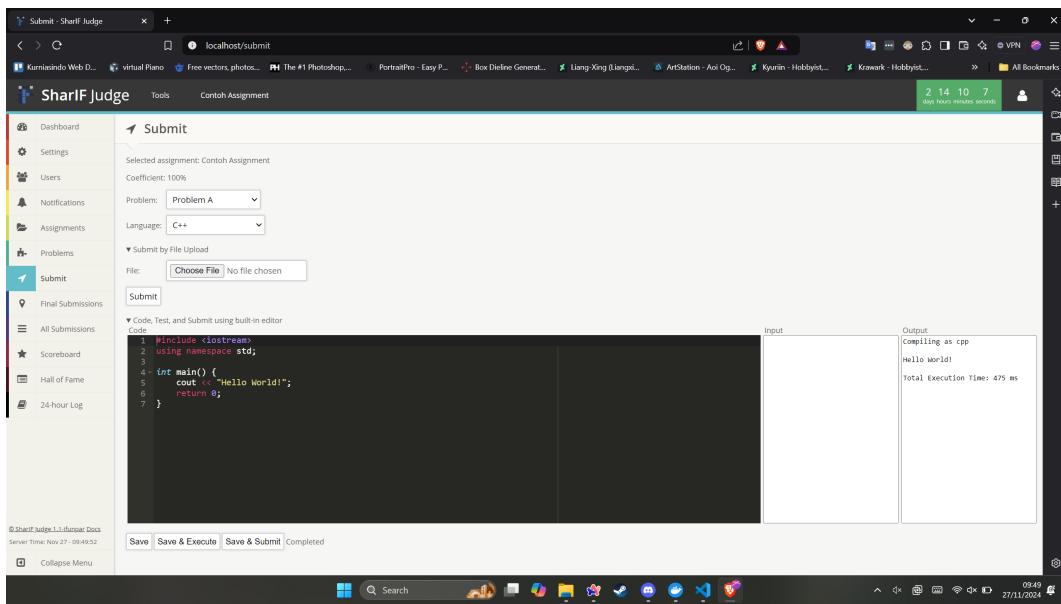
#	ID	Username	Name	Problem	Submit Time	Score				Language	Status	Code	Log	Actions
						Score	Delay %	Final Score	Language					
1	3	admin	Admin	1	2024-11-27 09:52:35	100	No delay 100%	100	Java	<button>100</button>	<button>Code</button>	<button>Log</button>	<button>...</button>	<button>↻</button>
2	4	admin	Admin	2	2024-11-27 09:54:16	0	No delay 100%	0	Java	Uploaded	<button>Code</button>	<button>Log</button>	<button>...</button>	<button>↻</button>

Gambar 3.20: Halaman Final Submissions

- **_download_excel(\$view)**

Fungsi ini menggunakan *library PHPExcel* untuk membuat sebuah *file excel* dari *submissions* yang akan diunduh pengguna.

- `final_excel()`
Menggunakan fungsi `_download_excel` untuk mendownload *final submission*.
 - `all_excel()`
Menggunakan fungsi `_download_excel` untuk mendownload seluruh *submission*.
 - `select()`
Fungsi ini digunakan sebagai fungsi yang dipanggil oleh *ajax request* untuk memilih *submission* yang akan dikumpulkan atau menjadi *final*.
 - `_check_type($type)`
Melakukan validasi tipe *submission* yang dikumpulkan.
 - `view_code()`
Digunakan untuk melihat kode, melihat hasil kode, atau melihat *log* sebuah *submission*.
 - `download_file()`
Mengunduh *file* kode sebuah *submission*.
- `Submit.php`
Berikut fungsi dengan penjelasannya pada *controller* `Submit.php`:
 - `_language_to_type($language)`
Mengembalikan kode singkat dari `$language` dipilih.
 - `_language_to_ext($language)`
Mengembalikan extensi file dari `$language` yang dipilih.
 - `_match($type, $extension)`
Melakukan validasi untuk `$type` dan `$extension` agar sesuai.
 - `_check_language($str)`
Melakukan validasi sudah dipilihannya bahasa.
 - `_upload()`
Menyimpan jawaban pengguna yang dikirim dan menambahkannya ke dalam *queue* untuk dinilai jika bukan *upload only problem*.
 - `load($problem_id)`
Mendapatkan isi file dan menyimpan isi file ke editor kode.
 - `save($type)`
Fungsi ini digunakan untuk menyimpan isi editor kode ke dalam *server* dan menjalankan atau mengumpulkan jika diinginkan.
 - `_submit($data, $problem_id, $language, $user_dir)`
Menambahkan kode ke dalam *submission* untuk dinilai.
 - `_execute($data, $problem_id, $language, $user_dir)`
Menambahkan kode ke dalam *queue* untuk dijalankan saja.
 - `get_output($problem_id)`
Mendapatkan keluaran dari kode yang telah dijalankan sebagai hasil eksekusi.
 - `index()`
Fungsi ini akan mendapatkan data dari *model* `Assignments_model` untuk mendapatkan *problem* dan data lainnya. Semua data tersebut akan dimasukkan dalam *view* `submit.twig`. Gambar 3.21 menunjukkan hasil halaman Submit. Halaman ini terdapat editor kode yang sudah diimplementasikan [7].
 - `User.php`
Berikut fungsi dengan penjelasannya pada *controller* `User.php`:
 - `add()`
Menambahkan pengguna baru ke dalam *sistem*.
 - `delete()`
Menghapus sebuah pengguna.
 - `delete_submissions()`
Menghapus seluruh *submissions* dari sebuah pengguna.
 - `list_excel()`



Gambar 3.21: Halaman Submit

Menggunakan *library* PHPExcel untuk membuat sebuah file excel dari seluruh daftar pengguna yang akan diunduh pengguna.

- **index()**

Fungsi ini akan mendapatkan data dari `User_model` dan menunjukkan `view users.twig` menggunakan data tersebut. Gambar 3.22 menunjukkan hasil halaman Users. Pada halaman ini terdapat daftar seluruh pengguna yang terdaftar pada SharIF Judge. Pengguna dapat membuat, memperbarui, dan menghapus pengguna.

#	User ID	Username	Display Name	Email	Role	First Login	Last Login	Actions
1	1	admin	Admin	admin@unpar.ac.id	admin	2024-11-27 09:38:45	2024-11-27 09:38:45	
2	2	andreas	Andreas	6182101026@student.unpar.ac.id	admin	Never	Never	

Gambar 3.22: Halaman Users

3.1.2 Assets

Pada SharIF Judge terdapat direktori bernama *Assets* yang menjadi tempat untuk menyimpan seluruh kebutuhan dari sisi pengguna seperti gambar logo dan *library JavaScript*. Berikut merupakan isi dari direktori *assets* beserta dengan kegunaannya dalam aplikasi SharIF Judge:

- Direktori **ace**
Direktori ini berisi hasil *build library* Ace, menghasilkan file *JavaScript* yang berfungsi untuk menambahkan editor kode pada aplikasi.
- Direktori **font**
Direktori ini berisi font khusus dan juga memiliki *library JavaScript* bernama font-awesome yang digunakan untuk menyimpan icon berformat svg dalam tampilan SharIF Judge.
- Direktori **fullcalendar**
Direktori ini berisi hasil *build library* FullCalendar, menghasilkan file *JavaScript* dan *css* yang berfungsi untuk memasukkan kalender dalam tampilan SharIF Judge.
- Direktori **gridster**
Direktori ini berisi hasil *build plugin* Gridster yang menggunakan *library* jQuery, menghasilkan file *JavaScript* dan *css* yang berfungsi untuk memasukkan *layout grid* yang dapat diubah dengan menggunakan sifat mouse *drag and drop*.
- Direktori **images**
Direktori ini berisi gambar kustom yang digunakan oleh SharIF Judge seperti logo universitas dan banner universitas dalam situs web.
- Direktori **js**
Direktori ini berisi hasil *build library* jQuery, taboverride, moment, *plugins* jQuery untuk membantu *JavaScript* khusus yang dipakai dalam SharIF Judge. Direktori ini juga memiliki file *JavaScript* khusus yang dipakai dalam halaman SharIF Judge yaitu sebagai berikut:
 - **shj_functions.js**
File *JavaScript* **shj_functions** digunakan untuk seluruh sistem umum dalam SharIF Judge seperti waktu server, sidebar *toggle*, loading dan berbagai macam fungsi yang digunakan dalam SharIF Judge.
 - **shj_submissions.js**
File *JavaScript* **shj_submissions** digunakan untuk menangani berbagai fitur untuk halaman all submissions dan final submissions seperti menampilkan kode program dan memeriksa ulang hasil kode dalam judge.
 - **shj_submit.js**
File *JavaScript* **shj_submit** digunakan untuk menangani berbagai fitur untuk halaman submit terutama fungsi aksi pada IDE yaitu aksi *save*, *execute*, *submit*, dan memuat kode lama ke dalam editor kode.
- Direktori **nano_scroller**
Direktori ini berisi hasil *build plugin* nanoScrollerJS untuk *library* jQuery, menghasilkan file *JavaScript* dan *css* yang berfungsi untuk membangun *scrollbar* pada SharIF Judge.
- Direktori **noty**
Direktori ini berisi *plugin* noty untuk *library* jQuery, menghasilkan file *JavaScript* dan *css* yang berfungsi untuk menampilkan pesan atau notifikasi kepada pengguna pada SharIF Judge.
- Direktori **pdfjs**
Direktori ini berisi hasil *build library* pdfjs, menghasilkan file *JavaScript* yang berfungsi untuk menampilkan file pdf pada SharIF Judge.
- Direktori **reveal**
Direktori ini berisi hasil *build plugin* Reveal yang sudah dimodifikasi untuk *library* jQuery, menghasilkan file *JavaScript* dan *css* yang berfungsi untuk menampilkan *modal* konfirmasi.
- Direktori **snippet**
Direktori ini berisi hasil *build plugin* Snippet yang sudah dimodifikasi untuk *library* jQuery, berfungsi untuk sebagai *Syntax Highlighter* untuk teks kode dalam halaman Hall of Fame, Problems, dan Submissions.
- Direktori **styles**
Direktori ini berisi seluruh *Cascading Style Sheets* atau *css file* khusus yang digunakan untuk memperindah halaman web dalam SharIF Judge.

- Direktori `tinymce`

Direktori ini berisi hasil *build library* tinymce yang berfungsi untuk memasukkan editor teks pada halaman *Submission*, *Final Submission* dalam SharIF Judge.

3.1.3 Penyimpanan Kode Submission

Pada SharIF Judge, Kode akan disimpan pada lokasi `Assignment` yang dapat diubah pada halaman `Settings`. Berikut merupakan format penyimpanan sebuah kode:

```
assignment_<a_id>/p<p_id>/<nama user>/<nama file>-<s_id>. <file ext>
```

Penjelasan untuk format di atas adalah sebagai berikut:

- `<a_id>`
id pada *assignment*.
- `<p_id>`
id pada *problem*.
- `<nama user>`
Nama dari pengguna yang mengumpulkan kode/file.
- `<nama file>`
Nama file yang dikumpulkan, `editor` jika mengumpulkan menggunakan editor kode.
- `<s_id>`
id pada *submission*.
- `<file ext>`
Extensi file kode yang dikumpulkan.

Sebagai contoh, pengguna bernama `kenzhi` mengumpulkan kode dengan nama file `probA.java` ke dalam *problem* pertama dari *assignment* dengan id 5. `kenzhi` sudah melakukan pengumpulan pada *problem* yang sama sebanyak 5 kali dan *submission* kali ini akan menjadi nomor 6, sehingga *submission id* adalah 6. Maka kode pengguna akan disimpan pada alamat:

```
assignment_5/p1/kenzhi/probA-6.java
```

3.1.4 Antrean Penilaian Kode

Pada SharIF Judge, Kode yang dikumpulkan akan dijalankan satu per satu pada antrean menggunakan `bash`. Berikut merupakan cara SharIF Judge menilai kode dari awal pengumpulan pada sistem:

1. *Controller Submit* akan menyimpan kode ke dalam *file* pada direktori sesuai pada [3.1.3](#).
2. *Controller Submit* akan memasukkan data *submission* ke dalam *model Queue_model*.
3. *Model Queue_model* akan menyimpan data *submission* pada *database submission* dan menambahkan data *queue*.
4. Selanjutnya *Controller Submit* akan memanggil fungsi `process_the_queue()` yang akan menjalankan fungsi `run()` pada *controller Queueprocess*.
5. *Queueprocess* akan menjalankan `tester.sh` pada direktori `tester` dengan data dari *queue*.
6. Selanjutnya `tester.sh` akan menilai kode program sesuai dengan *input* dan *output* yang diinginkan. Lalu penilaian akan dibaca dan disimpan oleh *controller Queueprocess*.
7. Terakhir *Queueprocess* akan menyimpan hasil penilaian pada *database submission* dan menghapus data *queue* menggunakan *Queue_model*.

3.1.5 Editor Kode

Integrated Development Environment (IDE) pada SharIF Judge dirancang dengan antarmuka yang terdiri atas empat komponen utama, yaitu sebagai berikut:

- Editor Kode

Editor kode ditandai dengan warna merah dalam Gambar 3.23 merupakan area utama tempat pengguna menulis dan mengedit kode program. Editor ini dilengkapi dengan fitur syntax highlighting dan auto-indent untuk mendukung produktivitas pengguna.

- Editor Input

Editor Input ditandai dengan warna biru dalam Gambar 3.23 berfungsi sebagai tempat pengguna memasukkan data uji yang akan digunakan saat menjalankan program melalui aksi Execute. Input yang dimasukkan akan diproses sebagai *standard input* atau *stdin* saat kode dijalankan.

- Editor Output

Editor Output ditandai dengan warna hijau dalam Gambar 3.23 menampilkan hasil eksekusi kode program, termasuk keluaran, pesan kesalahan jika ada, dan waktu komputasi. Konten pada bagian ini dihasilkan secara otomatis setiap aksi Execute dilakukan.

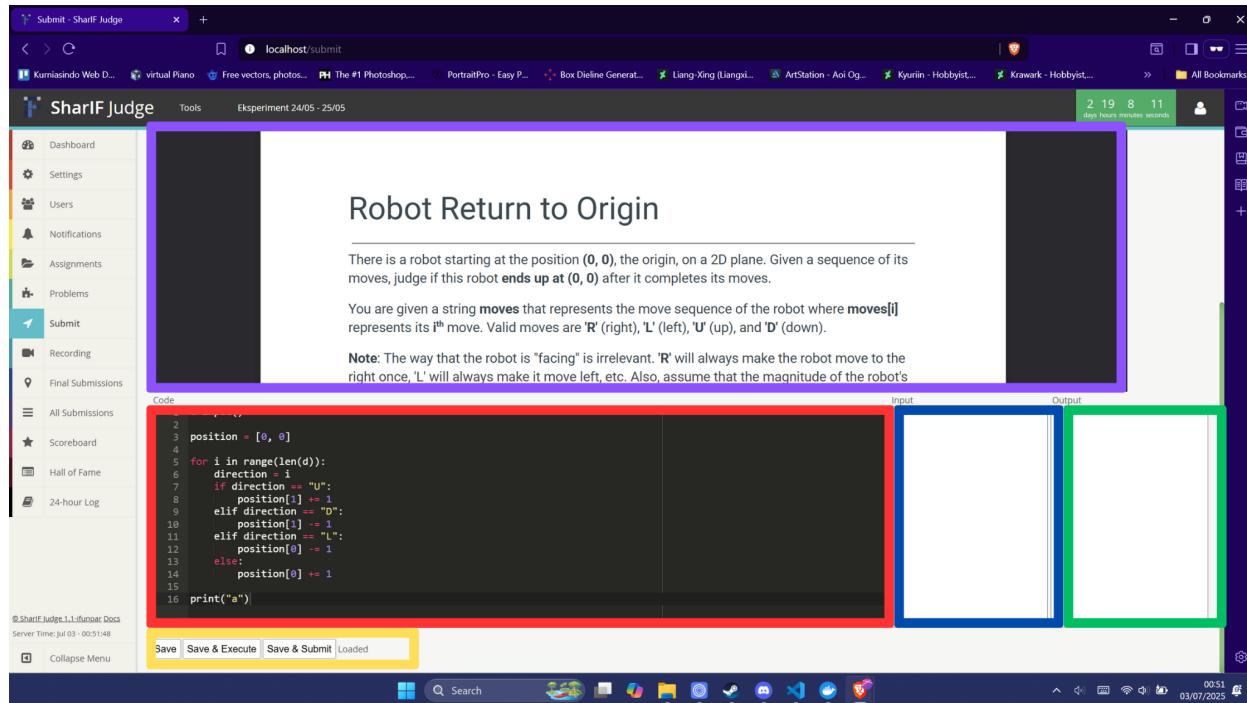
- Tombol Aksi

Tombol Aksi ditandai dengan warna kuning dalam Gambar 3.23 menyediakan tiga tombol aksi interaksi yaitu sebagai berikut:

- PDF Viewer

PDF Viewer ditandai dengan warna kuning dalam Gambar 3.23 merupakan sebuah komponen dalam IDE yang digunakan untuk menampilkan soal atau instruksi tugas kepada mahasiswa.

- **Save**: Menyimpan sementara kode program tanpa menjalankan atau mengumpulkannya.
- **Save & Execute**: Menyimpan kode sekaligus menjalankannya dengan input yang tersedia di Editor Input. Hasilnya ditampilkan di Editor Output.
- **Save & Submit**: Menyimpan dan mengumpulkan kode sebagai submission untuk dinilai oleh SharIF Judge.



Gambar 3.23: Editor Kode SharIF Judge

Fitur Execute dan Submit merupakan inti dari alur kerja SharIF Judge. Execute memungkinkan pengguna menguji kode secara mandiri, sementara Submit mengirim kode ke sistem penilaian otomatis untuk evaluasi akhir. Desain warna yang konsisten ini memudahkan pengguna dalam mengidentifikasi fungsi masing-masing komponen selama sesi pemrograman.

3.2 Analisis Pola Kecurangan pada Rekaman Ketikan

Berdasarkan indikator yang dijelaskan pada Bagian 2.8, sistem pemutaran ulang SharIF Judge akan mendeteksi pola perilaku yang mencurigakan. Bagian ini menjelaskan implementasi dari indikator-indikator tersebut serta metode analisis yang digunakan untuk mengidentifikasi kecurangan.

3.2.1 Analisis Perubahan Kode

Sistem menghitung code churn rate dengan membandingkan jumlah karakter yang ditambahkan, dihapus selama sesi penggerjaan. Rekaman ketikan yang menunjukkan perubahan minimal seperti angka code churn rate mendekati angka satu, akan ditandai sebagai potensi kecurangan. Tetapi code churn rate ini akan dibandingkan dengan keseluruhan code churn rate pada permasalahan tersebut agar lebih memiliki toleransi jika permasalahan yang diberikan merupakan permasalahan yang mudah.

3.2.2 Analisis Aktivitas Navigasi

Sistem akan merekam setiap peristiwa kehilangan fokus pada IDE, termasuk perpindahan tab browser. Pola kecurangan ditandai dengan frekuensi navigasi tinggi dalam waktu per jam yang berkorelasi.

3.2.3 Analisis Pengukuran Jeda Berpikir

Durasi jeda antara ketikan dianalisis untuk mengidentifikasi ketidakwajaran. Pada pemrograman alami, jeda lebih dari 10 detik sering muncul saat mahasiswa merancang solusi atau menguji logika. Rekaman yang menunjukkan ketikan berkelanjutan tanpa jeda atau jeda sangat singkat seperti 5 detik dan sering dianggap sebagai indikasi copy-paste. Sebaliknya juga jika jeda panjang juga dapat menjadi potensi kecurangan. Tetapi pengukuran ini juga akan dibandingkan dengan rata-rata pada permasalahan agar lebih toleransi.

3.2.4 Analisis Deteksi Debugging

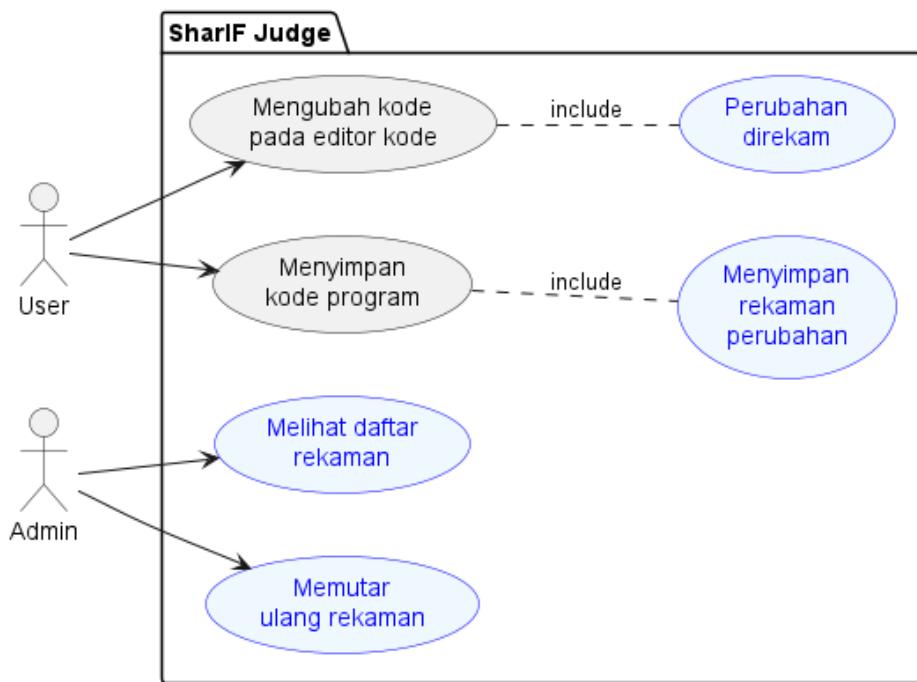
Aktivitas debugging dapat diindikasikan dari frekuensi eksekusi kode, perubahan input, modifikasi pada area spesifik, dan perubahan hasil output dengan input yang sama. Semua aktivitas tersebut akan direkam dan jika tidak ada indikasi debugging tersebut, kecurangan diduga terjadi.

3.2.5 Analisis *Copy-Pasting*

Jika pada rekaman terdapat pemasukkan kode yang besar tanpa melakukan penghapusan dengan besar yang sama, maka dapat diduga bahwa dia mendapatkan kode eksternal.

3.3 Analisis Sistem Usulan

Pembuatan sistem pemutaran ulang ketikan membutuhkan 4 fitur baru yaitu fitur perekaman perubahan, fitur penyimpanan rekaman perubahan, fitur melihat daftar rekaman yang ada, dan fitur untuk memutar ulang rekaman. Gambar 3.24 menunjukkan bagaimana fitur baru akan berinteraksi dengan sistem SharIF Judge dan pengguna. Berikut merupakan penjelasan mengenai fitur-fitur yang akan ditambahkan pada SharIF Judge untuk membangun sistem perekaman ketikan ditandai dengan warna biru dalam bagan *usecase* tersebut.



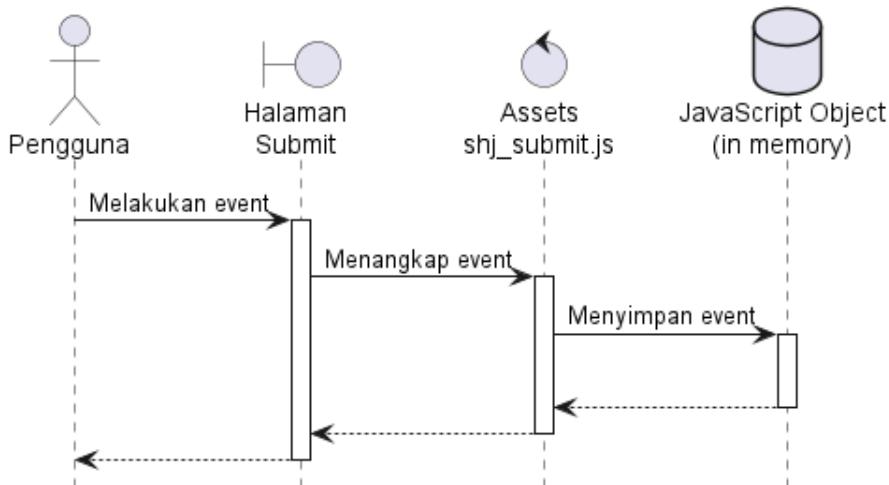
Gambar 3.24: Usecase analisis sistem usulan

3.3.1 Fitur perekaman perubahan atau event

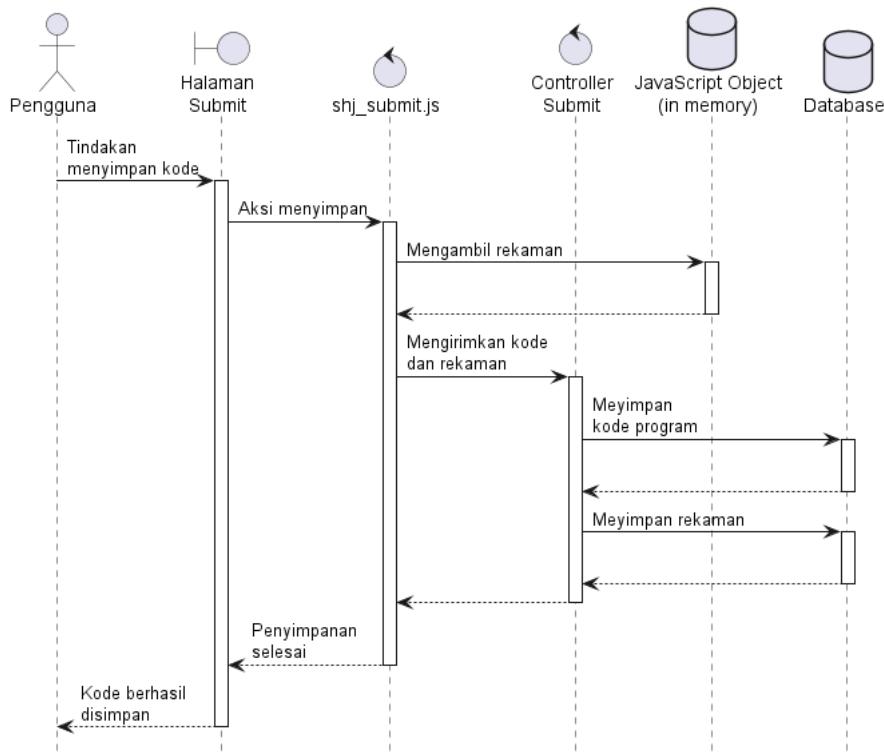
Fitur perekaman pada editor kode bukan hanya perubahan text melainkan pada seluruh kejadian atau *event* yang terjadi pada editor kode seperti contohnya adalah perubahan posisi kursor maupun pilihan pada kode. Fitur perekaman perubahan akan otomatis oleh browser dengan bantuan *JavaScript* yang ada pada browser pengguna dan akan dijalankan saat sebuah *event* terjadi pada editor kode. Fitur ini membutuhkan perubahan pada *Assets shj_submit.js* untuk merekam event yang dibutuhkan.

Sequence Diagram

Gambar 3.25 merupakan sebuah *sequence diagram* yang menunjukkan bagaimana fitur perekaman perubahan atau event akan berintegrasi dengan sistem IDE akan bekerja pada SharIF Judge.



Gambar 3.25: Sequence Diagram Fitur Perekaman Perubahan



Gambar 3.26: Sequence Diagram Fitur Penyimpanan Rekaman

3.3.2 Fitur penyimpanan rekaman perubahan

Fitur penyimpanan rekaman perubahan akan dilakukan secara otomatis saat pengguna melakukan tindakan menyimpan kode program. Fitur penyimpanan rekaman akan berintegrasi dengan fitur penyimpanan kode program yang sudah ada. Pada fitur ini daftar rekaman akan diperbarui dengan adanya rekaman baru atau perubahan pada *file* rekaman. Fitur ini membutuhkan perubahan pada *Assets shj_submit.js*, *Controller Submit*, *Model* baru untuk *Rekaman* untuk mengirimkan data rekaman, mendapatkan data rekaman dan mencatat data rekaman.

Sequence Diagram

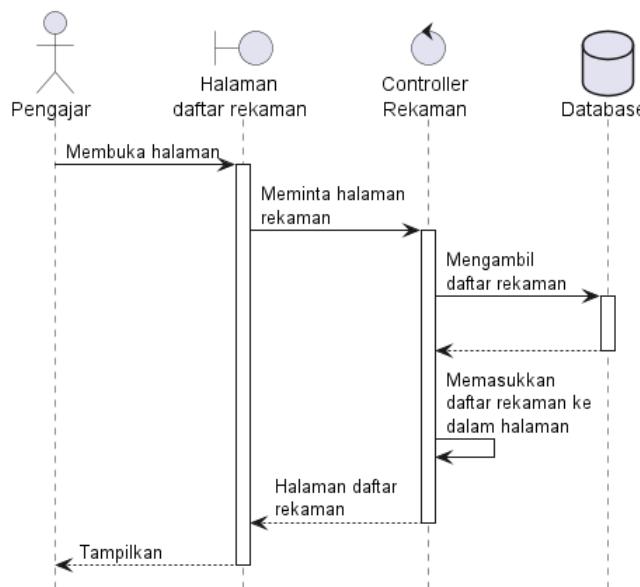
Gambar 3.26 merupakan sebuah *sequence diagram* yang menunjukkan bagaimana fitur penyimpanan rekaman perubahan akan berintegrasi dengan sistem IDE akan bekerja pada SharIF Judge.

3.3.3 Fitur melihat daftar rekaman

Pada sistem pemutaran ulang ketikan dibutuhkannya sebuah halaman baru yang akan dinamakan halaman rekaman. Pada halaman ini akan ditampilkan daftar rekaman untuk *assignment* yang dipilih pada halaman *Assignment*. Fitur ini akan dijalankan pada saat halaman rekaman dimuat ke dalam browser oleh SharIF Judge, dimana data yang dimasukkan ke dalam halaman rekaman adalah daftar rekaman tersebut dan beberapa data yang dibutuhkan. Fitur ini membutuhkan *View* dan *Controller* baru untuk membuat sebuah halaman baru untuk menunjukkan list rekaman yang ada. Fitur ini juga membuatkan penambahan pada *Model Recording* untuk mendapatkan daftar rekaman yang akan ditampilkan.

Sequence Diagram

Gambar 3.27 merupakan *sequence diagram* yang menunjukkan bagaimana sistem akan bekerja saat user akan membuka halaman rekaman pada SharIF Judge.



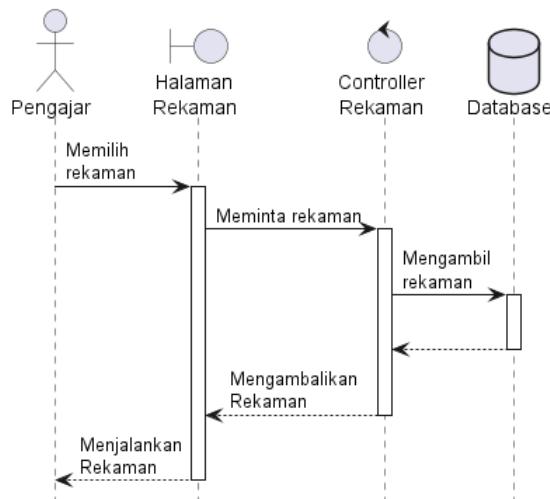
Gambar 3.27: Sequence Diagram Membuka Halaman Rekaman

3.3.4 Fitur pemutaran ulang rekaman

Fitur pemutaran ulang rekaman akan membuat satu buah rekaman dalam daftar rekaman dalam halaman rekaman dapat ditekan oleh pengguna untuk menandakan bahwa sebuah rekaman dipilih untuk dijalankan. Saat sebuah rekaman dipilih, browser akan meminta data rekaman kepada SharIF Judge dan dengan bantuan *JavaScript* akan melakukan pemutaran ulang rekaman pada sebuah editor kode yang tidak dapat diubah dalam halaman rekaman. Fitur ini membatkan penambahan ada *Controller* Rekaman dan penambahan halaman baru atau *View* baru untuk menunjukkan sebuah rekaman. Fitur ini juga membutuhkan *Assets* baru untuk memutar ulang dari data rekaman yang didapat.

Sequence Diagram

Gambar 3.28 merupakan *sequence diagram* yang menunjukkan bagaimana sistem SharIF Judge bekerja dari pemilihan rekaman hingga pemutaran ulang rekaman akan terjadi.



Gambar 3.28: Sequence Diagram Membuka Halaman Rekaman

BAB 4

PERANCANGAN

Bab ini membahas tentang perancangan untuk implementasi sistem perekaman ulang dalam SharIF-Judge. Perancangan akan dilakukan dengan mempertimbangkan aspek fungsionalitas, antarmuka pengguna, penyimpanan data, event yang direkam serta perubahan kode yang diperlukan pada sistem yang sudah ada.

4.1 Rancangan Penyimpanan Rekaman

Rekaman yang akan disimpan akan berupa sebuah daftar *event* yang terjadi. Dalam *JavaScript*, daftar tersebut akan menjadi *array* yang berisi *event-event* yang terjadi. Rekaman juga akan menyimpan waktu dimana rekaman dimulai, awal kode yang ada dalam editor kode dan juga awal posisi kursor dalam IDE. Rekaman akan menjadi sebuah objek *JavaScript* yang berisi sebagai berikut:

1. `timestart`: Waktu awal rekaman dimulai.
2. `start_value`: Isi awal dalam editor kode.
3. `start_cursor`: Posisi awal kursor dalam editor kode.
4. `events`: Daftar *event* yang terjadi.

Event yang akan direkam juga membutuhkan beberapa data yang harus disimpan yaitu: waktu *event* terjadi, *event* yang terjadi, dan muatan *event* yang terjadi. Maka *event* juga akan disimpan dalam bentuk objek *JavaScript*. Berikut merupakan format sebuah *event*:

```
{time: <time>, event: <event>, payload: <payload>}
```

Berikut merupakan penjelasan tentang format penyimpanan perekaman.

- <time> akan menunjukkan pada milidetik berapa event terjadi setelah waktu awal perekaman dimulai. sedangkan <event> dan <payload> merupakan data event yang terjadi.
- <event> merupakan *event* yang terjadi pada <time>, contohnya adalah pengguna melakukan perubahan pada editor kode dengan menambahkan huruf ‘a’, maka *event* yang terjadi merupakan *insert*. Semua event yang ditanggap akan dijelaskan pada sub Bagian 4.4.1.
- <payload> merupakan muatan *event* yang terjadi. Muatan akan disesuaikan dengan *event* yang terjadi. Sebagai contoh untuk event *insert* di atas, maka isi dari *event* tersebut adalah huruf ‘a’, dan posisi kursor dalam editor kode dimana huruf tersebut dimasukkan.

Berikut contoh hasil untuk sebuah *event insert* pada penjelasan di atas:

```
{time: 1203, event: "insert", payload: {data: "a", start: [10, 9]}}
```

Penyimpanan rekaman juga akan disimpan pada direktori yang sama dengan penyimpanan kode submission seperti yang dijelaskan pada Bagian 3.1.3 dengan nama file `record`. Meskipun file rekaman disimpan sebagai berkas JSON di direktori tersebut, informasi mengenai file tersebut juga dicatat ke dalam database agar sistem dapat menampilkan daftar rekaman yang tersedia untuk setiap problem. Informasi yang disimpan adalah pengguna yang melakukan rekaman, waktu perekaman, serta assignment dan problem yang terkait.

Karena sistem SharIF Judge tidak menggunakan relasi antar tabel yang dapat dilihat dari Gambar 4.1, maka rancangan tabel penyimpanan rekaman ini juga dibuat tanpa *foreign key*. Setiap kolom menyimpan data secara langsung seperti ID assignment atau ID problem dalam bentuk nilai biasa (bukan relasi). Hal ini dilakukan untuk menyesuaikan dengan struktur database SharIF Judge yang sudah ada dan menjaga kompatibilitas sistem. Rancangan struktur tabel dapat dilihat pada Gambar 4.2.

4.2 Rancangan Event yang direkam

Untuk mendukung fitur pemutaran ulang ketikan pada IDE SharIF Judge, sistem perlu merekam berbagai jenis interaksi pengguna selama proses penggerjaan tugas pemrograman berlangsung. Oleh karena itu, dirancanglah sejumlah event yang akan ditangkap dan dicatat secara real-time pada IDE. Berikut merupakan *event* yang akan ditangkap oleh *JavaScript* dalam halaman Submit:

- **Perubahan isi kode pada editor kode.**

Event ini merupakan interaksi utama dalam proses pemutaran ulang, karena isi kode adalah peristiwa utama dari proses penggerjaan. Tanpa merekam perubahan kode secara kronologis, sistem tidak dapat memperlihatkan perubahan kode solusi yang dibuat oleh mahasiswa.

- **Perubahan posisi kursor pada editor kode.**

Posisi kursor penting agar sistem tidak hanya menampilkan isi kode, tetapi juga memperlihatkan lokasi interaksi pengguna seperti proses penggerjaan aslinya.

- **Perubahan fokus pada web page.**

Fokus halaman direkam agar bisa diketahui apakah mahasiswa tetap di halaman atau pindah ke halaman atau aplikasi lain. Hal ini memperkuat validitas rekaman sebagai alat pemantauan perilaku.

- **Pergantian tab dalam browser.**

Pergantian tab perlu ditampilkan dalam pemutaran ulang agar pengajar dapat melihat jika terdapat pola perpindahan ke luar halaman yang mencurigakan, terutama saat penggerjaan berlangsung dalam waktu terbatas.

- **Perubahan fokus pada PDF Viewer.**

Perubahan fokus pada PDF Viewer perlu direkam karena saat mahasiswa membuka atau membaca soal, fokus pada halaman utama akan berubah. Perubahan ini direkam karena pada saat mahasiswa berganti fokus pada PDF Viewer, fokus pada halaman utama akan dianggap hilang yang seharusnya tidak.

- **Perubahan isi pada editor *input* dalam IDE.**

Input merupakan bagian dari proses pengujian program. Dengan merekamnya, pemutaran ulang dapat menampilkan bagaimana mahasiswa bereksperimen terhadap berbagai kasus uji.

- **Perubahan isi pada editor *output* dalam IDE.**

Output membantu merekonstruksi respon dari *input* setelah aksi *Execute* dilakukan. Dalam sistem, hal ini menunjukkan hasil dari setiap eksekusi, termasuk output yang keliru atau tidak sesuai harapan.

- **Aksi men-*Save*.**

Tindakan menyimpan kode menandakan momen penting dalam penggerjaan.

- **Aksi men-*Save & Execute*.**

Pemutaran ulang yang menampilkan momen eksekusi kode memberikan gambaran apakah mahasiswa sering mencoba kodennya, yang berkaitan dengan proses alur debugging.

- **Aksi men-*Save & Submit*.**

Aksi submit merupakan akhir dari proses penggerjaan. Merekam dan menampilkannya dalam sistem sangat penting untuk menandai titik akhir penggerjaan.

judge shj_queue	judge shj_problems	judge shj_assignments
# id : int(11) unsigned	# assignment : smallint(4) unsigned	# id : int(11) unsigned
# submit_id : int(11) unsigned	# id : smallint(4) unsigned	# name : varchar(50)
# username : varchar(20)	# name : varchar(50)	# problems : smallint(4) unsigned
# assignment : smallint(4) unsigned	# score : int(11)	# total_submits : int(11) unsigned
# problem : smallint(4) unsigned	# is_upload_only : tinyint(1)	# open : tinyint(1)
# type : varchar(8)	# c_time_limit : int(11) unsigned	# scoreboard : tinyint(1)
judge shj_scoreboard	# python_time_limit : int(11) unsigned	# javaexceptions : tinyint(1)
# assignment : smallint(4) unsigned	# java_time_limit : int(11) unsigned	# description : text
# scoreboard : text	# memory_limit : int(11) unsigned	# start_time : datetime
judge shj_settings	# allowed_languages : text	# finish_time : datetime
# shj_key : varchar(50)	# diff_cmd : varchar(20)	# extra_time : int(11)
# shj_value : text	# diff_arg : varchar(20)	# late_rule : text
judge shj_sessions	judge shj_logins	# participants : text
# session_id : varchar(40)	# login_id : int(11) unsigned	# moss_update : varchar(30)
# ip_address : varchar(45)	# username : varchar(20)	# archived_assignment : tinyint(1)
# user_agent : varchar(120)	# ip_address : varchar(15)	
# last_activity : int(10) unsigned	# timestamp : timestamp	
# user_data : text	# last_24h_login_id : int(11)	
judge shj_notifications	judge shj_users	judge shj_submissions
# id : int(11) unsigned	# id : int(11) unsigned	# submit_id : int(11) unsigned
# title : varchar(200)	# username : varchar(20)	# username : varchar(20)
# text : text	# password : varchar(100)	# assignment : smallint(4) unsigned
# time : datetime	# display_name : varchar(40)	# problem : smallint(4) unsigned
	# email : varchar(40)	# is_final : tinyint(1)
	# role : varchar(20)	# time : datetime
	# passchange_key : varchar(60)	# status : varchar(100)
	# passchange_time : datetime	# pre_score : int(11)
	# first_login_time : datetime	# coefficient : varchar(6)
	# last_login_time : datetime	# file_name : varchar(100)
	# selected_assignment : smallint(4) unsigned	# main_file_name : varchar(100)
	# dashboard_widget_positions : varchar(500)	# file_type : varchar(6)

Gambar 4.1: Tabel SharIF Judge Saat Ini

judge shj_recording	
# rec_id : int(11) unsigned	
# upload_at : datetime	
# assignment : smallint(4) unsigned	
# problem : smallint(4) unsigned	
# username : varchar(20)	

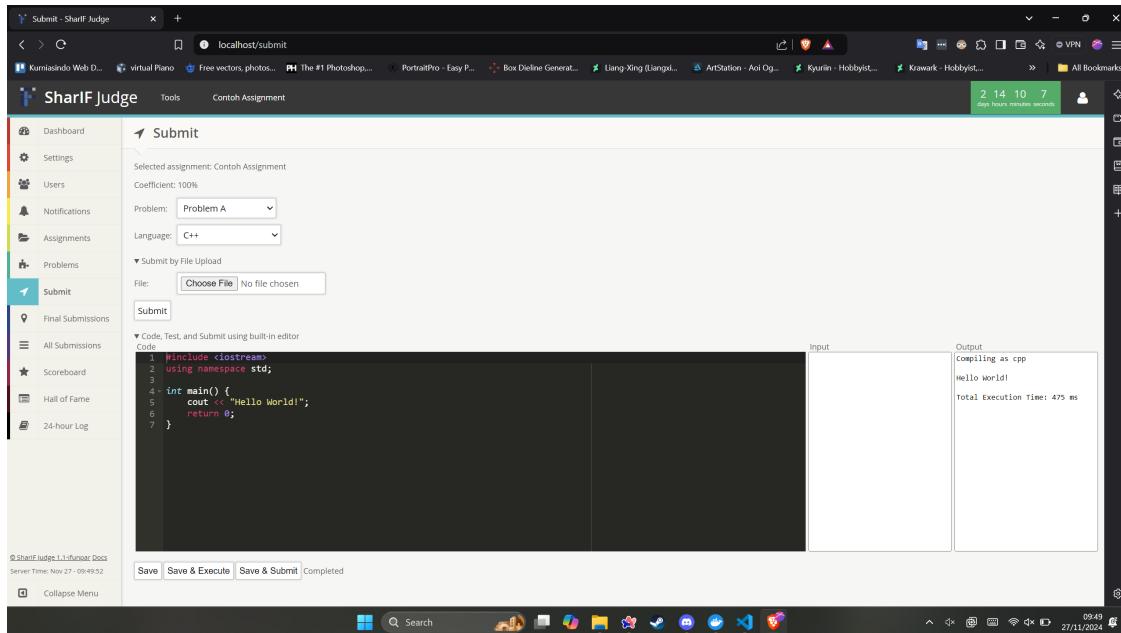
Gambar 4.2: Rancangan Tabel Penyimpanan Rekaman

4.3 Rancangan Antarmuka

Rancangan antarmuka sistem ini dibagi menjadi dua komponen utama yaitu sistem perekaman yang terintegrasi dengan halaman **Submit** yang sudah ada dan sistem pemutaran ulang yang membutuhkan pengembangan halaman baru.

4.3.1 Sistem Rekaman

Seluruh sistem rekaman akan diimplementasikan dalam halaman Submit tidak memerlukan perubahan pada antarmuka. Gambar 4.3 menunjukkan halaman Submit dan IDE yang sudah diimplementasikan oleh Nicholas Aditya Halim [7].



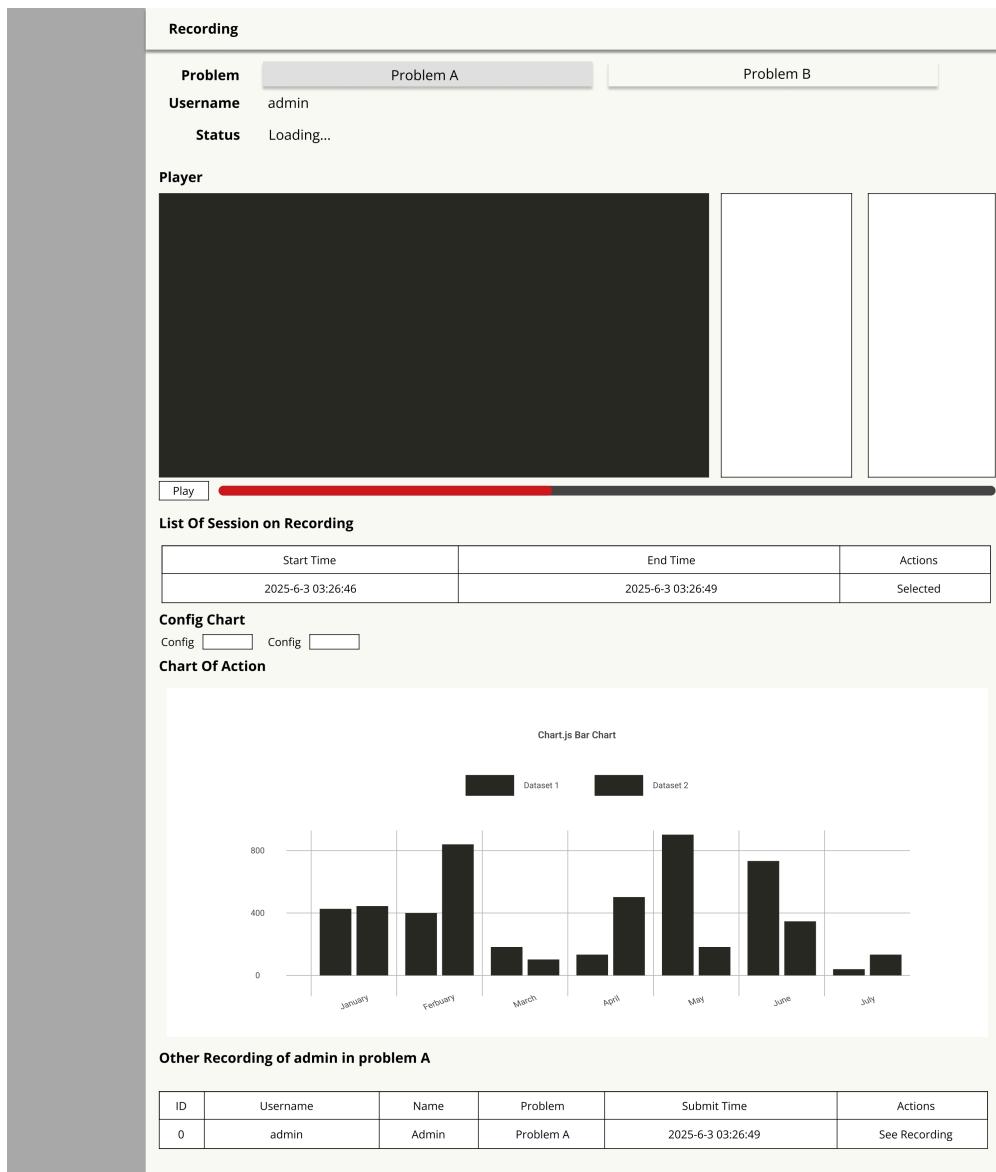
Gambar 4.3: Halaman Submit

4.3.2 Sistem Pemutaran ulang

Pada sistem pemutaran ulang dibutuhkan dua halaman baru yaitu halaman untuk menunjukkan daftar rekaman dalam sistem dan halaman untuk pemutaran ulang sebuah rekaman. Gambar 4.4 merupakan rancangan antarmuka untuk halaman daftar rekaman dan Gambar 4.5 merupakan rancangan antarmuka untuk halaman pemutaran ulang.

Recording					
Problem	All	Problem A	Problem B		
ID	Username	Name	Problem	Submit Time	Actions
0	admin	Admin	Problem A	2025-6-3 03:26:49	See Recording

Gambar 4.4: Rancangan Antarmuka Halaman Daftar Rekaman



Gambar 4.5: Rancangan Antarmuka Halaman Pemutaran Ulang

4.4 Rancangan Perubahan Kode

Untuk mengimplementasikan fitur yang telah diusulkan pada Bagian 3.3, diperlukan beberapa perubahan pada kode sumber SharIF-Judge. Berikut merupakan rancangan untuk setiap fitur:

4.4.1 Merekam perubahan atau event

Untuk menambahkan fitur ini, diperlukan perubahan pada bagian *JavaScript* yaitu *assets/js/shj_submit.js* dalam halaman *Submit*. Dimana *JavaScript* tersebut akan menjalankan perekaman secara otomatis saat pengguna memilih *problem* yang ada dalam *assignment* yang dipilih.

4.4.2 Menyimpan rekaman

Fitur ini akan dilakukan setiap aksi menyimpan kode, menjalankan kode dengan tes kasus, dan mengumpulkan kode melalui IDE. Data rekaman juga akan disimpan ke dalam sistem dan *database*. Untuk itu, perlu dilakukan perubahan sebagai berikut:

- *Controller Submit:*
 - Fungsi `save($type)`:
Fungsi ini akan diubah agar dapat menangani data rekaman yang dikirim oleh pengguna. Data tersebut akan disimpan dalam direktori yang sama dengan kode program.
 - Fungsi `_submit($data, $problem_id, $language, $user_dir)`:
Fungsi diubah agar untuk setiap *submit*, rekaman dari *save* sebelumnya akan diubah menjadi rekaman untuk *submit* tersebut.
- *Assets shj_submit.js*:
Pada file *assets* ini akan ditambahkan data rekaman yang dimuat ke dalam fungsi aksi menyimpan kode, menjalankan kode dengan tes kasus, dan mengumpulkan kode melalui IDE, rekaman juga akan disimpan ke dalam sistem.

4.4.3 Melihat daftar rekaman

Untuk melihat semua daftar rekaman yang terjadi, maka dibutuhkannya *database* untuk menyimpan dan mendapatkan daftar rekaman yang sudah disimpan dalam sistem. Setelah itu dibutuhkannya juga halaman baru dalam SharIF-Judge, maka perubahan yang dilakukan adalah sebagai berikut:

- *Controller Submit*:
 - Fungsi `save($type)`:
Fungsi ini menambahkan *metadata* rekaman pengguna ke dalam *database*. *Metadata* yang dimaksud adalah *id problem*, *id assignment*, dan pengguna yang direkam. Metadata dalam database digunakan untuk mendapatkan daftar rekaman yang belum disubmit pada sebuah *problem* dalam *assignment* beserta dengan nama pengguna rekaman.
 - Fungsi `_submit($data, $problem_id, $language, $user_dir)`:
Perubahan untuk fitur melihat daftar rekaman akan menambahkan *metadata* ke dalam *database* sebagai daftar rekaman yang sudah di submit. *Metadata* yang dimaksud adalah *id problem*, *id assignment*, dan pengguna yang direkam. Metadata dalam database digunakan untuk mendapatkan daftar rekaman yang sudah disubmit pada sebuah *problem* dalam *assignment* disertai dengan pengguna yang direkam.
- *Controller Recording*:
Sebuah *controller* baru yang menangani segala hal mengenai sistem pemutaran ulang dalam SharIF-Judge. Fungsi yang dibutuhkan agar fitur ini berjalan hanyalah fitur untuk menunjukkan daftar rekaman dalam sistem dengan mengambil data dari *model Recording* dan menyimpannya dalam *view recording*. Setelah itu, *controller* akan menunjukkan *view* tersebut kepada pengguna yang sudah login dan memiliki akses *instructor* atau lebih tinggi.
- *Model Recording*:
Sebuah *model* baru yang menangani segala hal mengenai penyimpanan dan pengambilan data rekaman dalam *database*. Fungsi-fungsi yang direncanakan untuk diimplementasikan dalam *model Recording* adalah sebagai berikut:
 - Fungsi `get_recording()`:
Fungsi digunakan untuk mendapatkan seluruh daftar rekaman dalam database. Fungsi ini dapat menyaring daftar rekaman berdasarkan *assignment*, *problem*, dan pengguna.
 - Fungsi `add_recording()`:
Fungsi ini digunakan untuk menyimpan sebuah rekaman ke dalam database.
- *View Recording_list*:
Sebuah *view* baru yang menampilkan daftar rekaman yang ada dalam sistem. *View* ini akan digunakan oleh *Controller Recording*.

4.4.4 Pemutaran ulang rekaman

Untuk dapat memutar ulang rekaman diperlukan beberapa perubahan kode dalam SharIF-Judge. Berikut merupakan rencana perubahan kode dalam SharIF-Judge:

- *Controller Recording:*

Controller Recording merupakan sebuah *controller* baru untuk menangani segala hal mengenai sistem pemutaran ulang dalam SharIF-Judge. Untuk fitur pemutaran ulang rekaman, diperlukan dua fungsi pada *controller* yaitu sebagai berikut:

- Fungsi `index()`:

Fungsi ini digunakan untuk menunjukkan *view Recording* kepada pengguna.

- Fungsi `download_record()`:

Fungsi ini digunakan untuk mendapatkan *file* rekaman dalam sistem. Fungsi ini akan dipanggil menggunakan AJAX dalam *assets Recording.js*.

- *Model Recording:*

Sebuah *model* baru yang menangani segala hal mengenai penyimpanan dan pengambilan data rekaman dalam *database*. Fungsi yang dibutuhkan oleh fitur pemutaran ulang rekaman adalah fungsi `get_recording()` untuk mendapatkan seluruh daftar rekaman sebuah pengguna dalam database berdasarkan *problem* dan *assignment*.

- *View Recording:*

Sebuah *view* baru yang menampilkan rekaman sebuah pengguna yang ada dalam sistem. *View* ini akan digunakan oleh *Controller Recording*.

- *Assets Recording.js:*

Assets Recording.js merupakan sebuah *assets JavaScript* baru untuk digunakan oleh *view Recording* sebagai *script* yang akan dijalankan oleh *browser* pengguna.

BAB 5

IMPLEMENTASI DAN PENGUJIAN

Bab ini akan membahas mengenai implementasi serta pengujian sistem perekaman ulang yang diterapkan dalam SharIF-Judge.

5.1 Implementasi

Bagian ini menjelaskan hasil implementasi sistem pemutaran ulang pada SharIF-Judge berdasarkan perancangan pada Bab 3. Pada saat implementasi juga dilakukan penyesuaian pada perancangan yang sudah dibuat untuk mengatasi kendala yang dialami pada saat implementasi.

5.1.1 Merekam Peristiwa pada IDE

Fitur merekam peristiwa pada editor kode diimplementasikan untuk menangkap seluruh interaksi pengguna terhadap IDE dan SharIF-Judge pada saat pengguna menyelesaikan sebuah masalah dalam *assignment*. Data yang direkam akan digunakan untuk memutar ulang penyelesaian yang dilakukan oleh pengguna. Fitur ini diimplementasikan dengan memanfaatkan *Library Ace*, *event hooks* pada *JavaScript*. Implementasi ini akan memerlukan penyesuaian pada bagian *JavaScript* yaitu file `assets/js/shj_submit.js` yang dimuat pada halaman *Submit*.

Berikut merupakan alur sistem perekaman peristiwa:

1. Inisialisasi Perekam

Pada alur ini, semua perekaman akan diinisialisasi dengan menjalankan sebuah fungsi dinamakan `recordStart`. Fungsi ini akan memanggil seluruh *event hooks* dan *event listener* dalam *Library ace* agar dinyalakan.

Dalam menjalankan fungsi *event listener*, dibutuhkan dua argumen yaitu event yang akan dipanggil dan sebuah *callback function* yang akan dieksekusi ketika *event* tersebut terjadi.

Implementasi akan dibuat 2 objek *JavaScript* yang menjadi fungsi *event listener* yaitu objek pemanggilan *event listener* dan objek menyimpan *callback function* yang dipakai oleh *event listener* masing-masing. Kode 5.1 merupakan objek *callback function* yang diimplementasikan.

Kode 5.1: objek *callback function*

```
1 const handlers = {
2   editor_change: (e) =>
3     recordEvent(e.action, {
4       data: e.lines,
5       start: e.start,
6       end: e.end,
7     }),
8   ...
9 }
```

Kode 5.2 merupakan objek pemanggilan *event listener* yang dipanggil pada saat inisialisasi dan mendapatkan fungsi *callback function* dari objek *handlers* pada Kode 5.1.

Kode 5.2: objek *event listener*

```
1 const addListener = {
2   editor_change: () => editor.session.on("change", handlers.editor_change),
3   ...
4 }
```

Setelah itu objek `addListener` akan dipanggil oleh fungsi `recordStart`. Kode 5.3 menunjukkan fungsi yang dipanggil saat inisialisasi.

Kode 5.3: Beberapa *event listener* yang dipanggil

```
1 addListener.editor_change();
```

Fungsi `recordStart` akan dijalankan setiap kali pengguna mengubah *problem* yang dipilih dalam halaman Submit untuk memulai rekaman peristiwa.

2. Penyimpanan sebuah rekaman

Untuk setiap perekaman yang dibutuhkan, dijalankan sebuah fungsi yang mendeteksi perubahan tersebut dan menjalankan sebuah *callback function* saat terjadi perubahan tersebut. Fungsi ini dipanggil pada saat inisialisasi. *Callback function* tersebut akan mendapatkan argumen sesuai dengan perubahan yang dideteksi. Pada contohnya untuk perubahan teks pada editor kode yaitu fungsi `onchange`, argumen yang diberikan merupakan teks yang dimasukan yaitu contohnya adalah ‘A’, dan juga posisi dimana teks tersebut dimasukkan dalam editor kode. Kode 5.4 merupakan contoh argumen yang diberikan.

Kode 5.4: Contoh argumen yang diberikan oleh fungsi `onchange`

```
1 {
2   data: ["A"],
3   start: {row: 0, column: 1},
4   end: {row: 0, column: 2}
5 }
```

Setelah itu data akan disimpan dalam sebuah objek *JavaScript* seperti yang sudah dijelaskan pada Bab 4.1. data argumen akan disimpan menggunakan key ‘args’ atau ‘payload’.

3. Penyimpanan data rekaman

Selanjutnya sebuah *event* atau rekaman yang sudah dicatat dan menjadi sebuah objek *JavaScript* bernama `recording`. seluruh event rekaman akan simpan dalam sebuah *array* dalam `recording` dengan key ‘events’ seperti yang sudah dijelaskan pada Bab 4.1. `recording` juga memiliki waktu dimulainya rekaman, isi awal editor kode, posisi awal kursor dalam editor kode. `recording` juga memiliki fungsi `init` untuk meninisialisasi seluruh objek `recording`.

Kode 5.5: Contoh argumen yang diberikan oleh fungsi `onchange`

```
1 const recording = {
2   events: [],
3   startTime: -1,
4   startValue: "",
5   startSelection: [],
6
7   init: () => {
8     recording.events = [];
9     recording.startTime = Date.now();
10    recording.startValue = editor.getValue();
11    recording.startSelection = getSelection(editor);
12  },
13};
```

Kode 5.5 merupakan `recording` pada saat keadaan kosong. Pada objek `recording`, `events` merupakan sebuah array dengan isi sebuah rekaman event, `startTime` merupakan waktu awal rekaman dimulai, `startValue` merupakan isi awal dalam editor kode, dan `startSelection` merupakan posisi awal kursor dalam editor kode.

Berikut merupakan peristiwa yang akan direkam oleh sistem perekaman:

- `editor.change`: Peristiwa ini akan menangkap perubahan isi teks dalam editor kode.
- `editor.changeCursor`: Peristiwa ini akan menangkap perubahan kursor dalam editor kode.
- `editor.changeSelection`: Peristiwa ini akan menangkap setiap perubahan yang terjadi pada *selection* kursor dalam editor kode.
- `window.focus`: Peristiwa ini akan menangkap pengguna *focus* pada SharIF Judge web page.
- `window.blur`: Peristiwa ini akan menangkap pengguna pada saat pengguna tidak *focus* pada SharIF Judge web page atau *focus* pada aplikasi lain atau web page lain.
- `window.visibilitychange`: Peristiwa ini akan menangkap pengguna yang mengubah *tab* dari SharIF Judge ke *tab* lain dalam browser.

- `pdf_viewer.focusin`: Peristiwa ini akan menangkap pengguna pada saat pengguna *focus* pada PDF Viewer dalam IDE.
 - `pdf_viewer.focusout`: Peristiwa ini akan menangkap pengguna pada saat pengguna tidak *focus* pada PDF Viewer dalam IDE.
 - `editor_input.input`: Peristiwa ini akan menangkap perubahan isi editor input dalam IDE.
 - `editor_output.change`: Peristiwa akan menangkap *output* saat terjadinya aksi *Execute*.
 - `save.click`: Peristiwa ini akan menangkap aksi *Save* yang dilakukan pengguna.
 - `execute.click`: Peristiwa ini akan menangkap aksi *Save & Execute* yang dilakukan pengguna.
 - `submit.click`: Peristiwa ini akan menangkap aksi *Save & Submit* yang dilakukan pengguna.
- Seluruh alur sistem perekaman peristiwa dalam SharIF-Judge akan ditambahkan ke dalam file `assets/js/shj_submit.js`. Kode perubahan terdapat pada Lampiran A.

Perbaikan Implementasi

Pada saat mengujian fungsi penyimpanan rekaman, dalam tahap alur penyimpanan data rekaman dan juga bagaimana inisialisasi perekaman akan diubah dari perancangan Bab 4.4.1. Hal ini dikarenakan saat pengguna memilih ulang masalah atau memuat ulang halaman Submit, maka semua events yang sudah direkam akan hilang. Maka dari itu berikut merupakan perubahan pada alur fitur sistem perekaman ketikan:

1. Inisialisasi Perekam

Alur inisialisasi akan diubah agar dapat memuat events yang sebelumnya sudah disimpan dalam objek *JavaScript* bernama `befRecording`. Oleh karena itu, dibutuhkannya penambahan fungsi pada *Controller Submit.php* yaitu fungsi untuk mengambil data rekaman sebelumnya bernama `load_rec` yang mengambil argumen pengenal masalah yang dipilih oleh pengguna. Kode penambahan terdapat di Lampiran A

2. Penyimpanan data rekaman

Pada objek `recording`, `startValue` dan `startSelection` tidak dibutuhkan karena isi awal dari editor kode dan juga posisi awal kursor dalam editor kode akan selalu berisi dengan nilai kosong yaitu teks kosong dan posisi di baris dan kolom pertama.

5.1.2 Menyimpan Rekaman pada Sistem

Fitur penyimpanan rekaman pada sistem bertujuan untuk menyimpan data secara permanen ke dalam database dan *server* agar dapat diputar kembali di lain waktu. Berikut merupakan alur fitur penyimpanan rekaman pada sistem:

1. Mengirimkan Data ke *Server*

Dalam halaman Submit, pengguna memiliki 3 aksi penting dalam IDE SharIF-Judge yaitu: `save`, `execute`, dan `submit`. Alur ini akan mengirimkan data rekaman pada saat pengguna melakukan aksi tersebut. Data yang dikirim merupakan objek `recording` yang sudah jadikan sebagai teks JSON dengan menggunakan fungsi `JSON.stringify`.

2. Menyimpan Data Dalam File Sistem

File akan disimpan dalam direktori yang sama dengan penyimpanan kode *submission* yang dijelaskan pada Bagian 3.1.3. File akan diisi secara langsung oleh data rekaman dan tidak diubah oleh *server*. Penamaan file dapat dibagi berdasarkan aksi yang membuat pengguna mengirimkan data rekaman. Untuk aksi `save` dan `execute`, file dengan data rekaman akan disimpan dengan nama `recording`. Untuk aksi `submit`, file akan disimpan dengan nama `recording` dilanjutkan dengan sebuah '-' dan `submit id` yang dibuat. File tersebut memiliki tipe data yang sama yaitu JSON karena itu extensi file yang digunakan adalah `.json`.

3. Menyimpan Data Dalam Database

Saat penyimpanan data ke dalam file sistem berhasil, maka penyimpanan ke dalam database akan dilakukan. Data yang akan disimpan kedalam database akan digunakan untuk mendaftar

rekaman yang ada dalam sistem, maka data yang akan disimpan bukan data rekaman melainkan data statistik. Berikut merupakan data yang akan disimpan ke dalam Database:

- **rec_id**: pengenal rekaman yang sama dengan *submit id*.
- **username**: nama pengguna yang mengirimkan data rekaman.
- **problem_id**: pengenal masalah yang pengguna kerjakan.
- **assignment_id**: pengenal tugas yang pengguna kerjakan.
- **upload_at**: waktu sistem penyimpanan data rekaman.

Untuk membuat databasenya sendiri, dibutuhkan penambahan tabel bernama tabel **recording** yang memiliki lima atribut di atas. Kode 5.6 menunjukkan pembuatan tabel baru menggunakan *CodeIgniter* dalam SharIF-Judge.

Kode 5.6: Kode membuat database pada SharIF-Judge

```

1 // create table 'recording'
2 $fields = array(
3     'rec_id'      => array('type' => 'INT', 'constraint' => 11, 'unsigned' => TRUE),
4     'upload_at'   => array('type' => $DATETIME),
5     'assignment'  => array('type' => 'SMALLINT', 'constraint' => 4, 'unsigned' => TRUE),
6     'problem'     => array('type' => 'SMALLINT', 'constraint' => 4, 'unsigned' => TRUE),
7     'username'    => array('type' => 'VARCHAR', 'constraint' => 20),
8 );
9 $this->dbforge->add_field($fields);
10 if (! $this->dbforge->create_table('recording', TRUE))
11     show_error("Error creating database table " . $this->db->dbprefix('recording'));
12 // ADD Unique constraint
13 $this->db->query(
14     "ALTER TABLE {$this->db->dbprefix('recording')}
15         ADD CONSTRAINT {$this->db->dbprefix('srup_unique')} UNIQUE (rec_id, username, assignment, problem);"
16 );

```

Mengikuti arsitektural *CodeIgniter*, untuk menambahkan sebuah data ke dalam database perlu menggunakan sebuah *Model*. Oleh karena itu, dibutuhkannya *model* baru bernama **Recording_model.php** yang ditambahkan fungsi **add_recording()** yang memiliki argumen yaitu seluruh data yang ingin disimpan dalam database.

Untuk aksi *save* dan *execute* dimana tidak adanya *submit id* maka akan menjadi angka nol ('0') pada pengenal rekamannya atau **rec_id** jika aksinya merupakan *submit*.

Untuk alur pengiriman data ke *server*, dibutuhkannya penambahan kode ke dalam **assets/js/shj_submit.js**. Agar dapat menyimpan dibutuhkannya perubahan dalam kode *Controller Submit* pada fungsi **save(\$type)** dan fungsi **_submit()**. Kode pembahannya terdapat pada Lampiran A.

Perbaikan Implementasi

Pada pengujian yang sama dengan Bab 5.1.1, dibutuhkannya perubahan pada alur pengiriman data ke server. Karena *events* yang sudah di *save* dapat terhapus karena pengguna memilih ulang masalah dan memuat ulang halaman *Submit*, yang membuat rekaman inisialisasi dan menghapus rekaman lama. Maka pada saat mengirimkan data **recording**, akan disertakan juga data **befRecording** yang sudah diambil pada saat inisialisasi. Format pengiriman data juga akan berubah dikarenakan adanya rekaman yang lama menjadi sebuah *key* dan *value* karena hanya dua value yang harus disimpan yaitu *events* sebagai *value* dan *startTime* sebagai *key*. Maka format ini menjadi format keseluruhan events yang terjadi dan dapat disatukan dengan format yang sama menggunakan *spread operator* agar seluruh rekaman lama digabungkan. Kode 5.7 merupakan kode untuk mengirimkan teks JSON dengan mengabungkan kedua rekaman menggunakan *spread operator*.

Kode 5.7: objek *callback function*

```

1 JSON.stringify({
2     ...befRecording,
3     [recording.startTime]: recording.events
4 }),

```

5.1.3 Melihat Daftar Rekaman

Fitur ini digunakan untuk melihat daftar rekaman mahasiswa yang tersimpan dalam sistem, pengguna juga dapat melihat isi rekaman yang terdapat dalam daftar rekaman tersebut. Fitur ini dibutuhkan dua tahap untuk diimplementasikan yaitu sebagai berikut:

- Pengambilan Data Rekaman

Fitur pengambilan data rekaman digunakan agar bagian depan SharIF-Judge dapat meminta daftar rekaman yang ada pada bagian belakang SharIF-Judge. Oleh karena ini merupakan sebuah halaman baru dalam SharIF-Judge, maka dibutuhkannya sebuah *Controller* baru bernama *Recording.php* yang menampilkan sebuah halaman baru yang dapat diakses melalui rute */recording/all/* yang menggunakan fungsi baru dalam *Recording_model.php* yaitu fungsi untuk mendapatkan daftar rekaman dinamakan *all_user_recordings*. Kode pertambahan pada *Model Recording.php* berada pada Lampiran A.

Berikut fungsi yang akan diimplementasikan dalam *Controller Recording.php*:

1. *__construct()*

Fungsi ini akan memuat seluruh kebutuhan *Model* dan *Helper* ke dalam *Recording.php*, fungsi *construct* juga akan membatasi akses oleh pengguna dibawah *instructor*. Fungsi ini juga mendapatkan *params url* yang dikirim oleh pengguna.

2. *all()*

Fungsi ini akan mengambil beberapa data yang dibutuhkan oleh antarmuka SharIF-Judge yaitu *assignment* yang dipilih oleh pengguna menggunakan *assignment_model*, *problem* yang ada dalam *assignment* tersebut menggunakan *assignment_model*, dan daftar *recording* yang tersimpan dalam sistem menggunakan *recording_model*. Setelah itu, server akan menempatkan seluruh data tersebut ke dalam *view* baru bernama *recording_list.twig* untuk ditampilkan kepada pengguna.

Seluruh alur untuk mengimplementasikan fitur pengambilan data rekaman dalam *Controller Recording.php* terdapat dalam Lampiran A.

- Antarmuka dan Tampilan Data

Antarmuka yang diimplementasikan akan dibuat serupa dengan perancangan pada Bab 4.3.2 yang diimplementasikan ke dalam SharIF-Judge. Data yang dikirim oleh *Controller Recording.php* juga dapat ditampilkan menggunakan *Library twig* tanpa menbutuhkan *JavaScript* maupun *php* dalam antarmukanya. Gambar B.1 menunjukkan implementasi antarmuka beserta data yang terdapat dalam sistem. Untuk kode keseluruhan antarmuka menggunakan *Library twig* dapat dilihat pada Lampiran A.

Perbaikan Implementasi

Untuk fitur ini, tidak dilakukan perubahan pada implementasi yang sudah ada karena dianggap sudah memenuhi kebutuhan pengguna.

5.1.4 Pemutaran Ulang Rekaman

Fungsi pemutaran ulang rekaman menggunakan data rekaman yang sudah disimpan dalam sistem untuk memvisualisasikan proses penyelesaian masalah pengguna secara kronologis. Fitur pemutaran ulang ini membutuhkan yaitu sebagai berikut:

- Implementasi Antarmuka

Untuk menambahkan sebuah halaman baru dalam SharIF-Judge dibutuhkannya juga fungsi baru pada *Controller Recording.php*. Fungsi baru akan dinamakan *index* untuk menampilkan sebuah *view* baru bernama *recording.twig*. Fungsi *index* akan menampilkan halaman baru itu melalui rute */recording/*. Penamaan *index* itu agar rute tidak memerlukan */index* pada akhir rute karena jika rute *function* (Bab 2.2.2) akan otomatis mengarah pada fungsi *index* dalam kelas tersebut.

Fungsi *index* akan mengirim data daftar rekaman pengguna lainnya dalam masalah yang dipilih. Data tersebut akan dipakai oleh *recording.twig* untuk menambahkan daftar rekaman pengguna lainnya pada *assignment* dan *problem* yang sama. Gambar B.2 merupakan antarmuka yang diimplementasikan serupa dengan perancangan pada Bab 4.3.2.

- Implementasi Memuat Data Rekaman

Data rekaman yang akan diambil sudah disimpan (Bab 5.1.2) dalam sistem. Tetapi data rekaman tidak akan dikirim oleh *Controller* pada saat halaman *recording* dimuat, melainkan menggunakan AJAX pada halaman *recording*. Maka dibutuhkannya fungsi baru pada *Controller Recording.php* dan sebuah assets *JavaScript* baru bernama *shj_function.js*. Fungsi baru dalam *Controller Recording.php* akan dinamakan *download_record* yang memiliki argumen *assignment_id*, *problem_id*, dan *rec_id*. Fungsi tersebut akan mengambil *file* rekaman dalam file sistem dengan menggunakan argumen untuk mendapatkan lokasi dan nama *file* rekaman (Bab 4.1) SharIF-Judge dan mengirimkan file tersebut secara langsung. File juga akan dirikim dengan header *Content-Type: application/json* dan *Content-Disposition: attachment; filename="rec.json"*.

Fungsi *download_record* dipanggil pada saat halaman *recording* dimuat dan *JavaScript* *shj_function.js* dengan fungsi baru yaitu *getRecording*. Fungsi *getRecording* akan meninisialisasi editor kode dalam antarmuka, mendapatkan data rekaman melalui fungsi *download_record*, dan menformat data rekaman agar lebih mudah untuk di putar ulang. Berikut merupakan format data tambahan yang akan diubah oleh fungsi *getRecording*:

- *events*: Data rekaman
- *eventsIndex*: sebuah map dengan *key* waktu saat sebuah *events* terjadi dan *value index* waktu saat sebuah *events* terjadi.
- *indexEvents*: sebuah map dengan *key index* waktu saat sebuah *events* terjadi dan *value* waktu saat sebuah *events* terjadi.
- *presumIndexDuration*: menkalkulasikan panjang rekaman sebelum rekaman selesai.
- *length*: panjang data rekaman
- *duration*: durasi dari seluruh rekaman yang ada pada data rekaman

Data akan disimpan dalam sebuah objek *JavaScript* dinamakan *recording* dan akan dipakai pada saat menjalankan rekaman dan untuk menampilkan histogram *events* yang terjadi.

- Implementasi Menjalankan Rekaman

Fitur menjalankan rekaman akan menggunakan data bedasarkan data yang didapatkan oleh AJAX yang dijelaskan pada bagian 5.1.4. Fitur menjalankan rekaman akan membutuhkan penambahan kode pada *JavaScript* *shj_function.js* yang akan menjalankan fungsi *play* atau *stop* untuk menjalankan atau memberhentikan rekaman oleh pengguna.

Fungsi menjalankan atau mematikan rekaman dibagi menjadi dua yaitu fungsi rekaman dalam IDE dengan data rekaman dinamakan *Recording* dan fungsi timer yang digunakan untuk memberitahu kepada pengguna progress waktu pemutaran rekaman dinamakan *Timer*. Fungsi *Recording* menggunakan fungsi dalam *Library Ace* dan fungsi *JavaScript* untuk memperbarui IDE antarmuka berdasarkan *event* yang dipanggil, fungsi *setTimeout* dalam *JavaScript* akan digunakan untuk menjalankan *event* selanjutnya bedasarkan perbedaan waktu antara event sekarang dan event selanjutnya dengan memanggil fungsi *playRecording* dengan *event* selanjutnya. Sedangkan fungsi *Timer* menggunakan fungsi *setInterval* yang akan dijalankan berulang untuk setiap detiknya dan memperbarui progress waktu dalam antarmuka berdasarkan waktu yang sudah lewat.

- Menampilkan Histogram Events yang Terjadi

Pada fungsi *getRecording* setelah memuat data rekaman dan menformat data rekaman tersebut, fungsi *setUpChart* akan dipanggil dan membuat data grafik histogram. Data histogram akan dimuat menggunakan *Library Chart.js*.

Perbaikan Implementasi

Untuk fitur ini, tidak dilakukan perubahan pada implementasi yang sudah ada karena dianggap sudah memenuhi kebutuhan pengguna.

5.2 Pengujian Fungsional

Pengujian fungsional dilakukan secara lokal. Berikut merupakan pengujian terhadap fitur-fitur sistem pemutaran ulang dalam SharIF Judge:

Tabel 5.1: Tabel Pengujian Fungsional

No.	Aksi Pengguna	Reaksi yang diharapkan	Reaksi
1	Memilih masalah dalam Halaman Submit	Sistem memulai rekaman	sesuai
2	Melakukan Aksi <i>Save</i>	Menyimpan rekaman sesi dalam sistem	sesuai
3	Melakukan Aksi <i>Save & Execute</i>	Menyimpan rekaman sesi dalam sistem	sesuai
4	Melakukan Aksi <i>Save & Submit</i>	Menyimpan rekaman sesi dalam sistem	sesuai
5	Membuka Halaman Daftar Rekaman	Daftar Rekaman ditampilkan	sesuai
6	Memilih Rekaman dalam Halaman Rekaman	Memuka halaman Rekaman	sesuai
7	Menyaring Daftar Rekaman dengan User	Daftar rekaman menampilkan hanya daftar rekaman user	sesuai
8	Menyaring Daftar Rekaman dengan Problem	Daftar rekaman menampilkan hanya daftar rekaman untuk problem saringan	sesuai
9	Menyaring Daftar Rekaman dengan User dan Problem	Daftar rekaman menampilkan hanya daftar rekaman user pada problem saringan	sesuai
10	Membuka Halaman Rekaman	Rekaman ditampilkan	sesuai
11	Menjalankan Rekaman	Rekaman diputar ulang	sesuai
12	Memberhentikan Rekaman yang berjalan	Putaran ulang rekaman berhenti	sesuai

5.3 Pengujian Eksperimental

Pada Bagian ini dilakukannya pengujian terhadap sistem Perekaman Ulang pada SharIF Judge.

5.3.1 Lingkungan pengujian

Pengujian sistem perekaman ulang akan dilakukan pada sebuah server VPS atau *Virtual Private Server*. Pada server akan dijalankannya *docker* agar sistem aplikasi akan identik dengan pada saat sistem sedang diimplementasikan. Kode C.1 dan Kode C.2 merupakan file *docker-compose* dan *Dockerfile* yang digunakan membangun sistem SharIF Judge dalam server VPS.

Tabel 5.2 menunjukkan spesifikasi perangkat keras yang digunakan saat pengujian.

Tabel 5.2: Perangkat Keras Lingkungan Pembangunan

Parameter	Nilai
<i>Processor</i>	<i>AMD EPYC 9354P 2 vCPU</i>
<i>Random Access Memory (RAM)</i>	8 GB
<i>Storage</i>	100 GB <i>SSD</i>

Tabel 5.3 menunjukkan spesifikasi perangkat lunak yang digunakan saat pengujian.

Tabel 5.3: Perangkat Lunak Lingkungan Pembangunan

Parameter	Nilai
Sistem Operasi	<i>Debian Version 12</i>
Bahasa Pemrograman	<i>PHP, JavaScript, CSS, dan HTML</i>
Framework	<i>CodeIgniter 3.1.13</i>
Code Editor	<i>Visual Studio Code 1.99.3</i>
Perangkat Lunak Pendukung	<i>Docker Version 20.10.24+dfsg1</i> <i>Debian 11-slim</i> <i>MySQL 5.7</i> <i>phpMyAdmin 5.2.1</i> <i>PHP 7.3.33</i>

5.3.2 Eksperimen

Pengujian dilakukan untuk mengetahui dampak dan masalah yang mungkin muncul jika sistem perekaman aktivitas ditambahkan ke dalam IDE pada SharIF Judge. Selain itu, hasil dari pengujian ini juga akan digunakan untuk melakukan analisis sederhana terhadap kemungkinan penerapan sistem deteksi kecurangan berdasarkan data rekaman.

Eksperimen dilakukan dengan melibatkan beberapa responden, yaitu mahasiswa aktif dan alumni yang sudah menyelesaikan studinya. Masing-masing peserta diminta untuk mengerjakan tiga soal pemrograman dengan tingkat kesulitan berbeda, yaitu mudah, sedang, dan sulit. Seluruh pengerjaan dilakukan dalam satu hari dan menggunakan sistem IDE SharIF Judge yang sudah terintegrasi dengan fitur perekaman.

Dalam eksperimen ini, peserta diberi kebebasan untuk melakukan kecurangan pada salah satu soal dengan cara apa pun, misalnya menyalin jawaban dari sumber lain atau melihat pembahasan di internet. Sistem tidak membatasi bentuk kecurangan, agar rekaman yang dihasilkan mencerminkan kondisi yang mungkin terjadi pada ujian aslinya. Kecurangan dianggap terjadi apabila peserta melihat atau menggunakan cara penyelesaian soal dari luar sistem. Hasil rekaman dari setiap peserta kemudian dianalisis untuk melihat pola pengerjaan normal dan pola yang mengindikasikan adanya kecurangan.

Berikut merupakan masalah yang ditemukan pada saat pengujian sistem pemutaran ulang:

- Fitur menampilkan soal pada IDE

Fitur ini tidak dapat dilakukan karena permasalahan dengan *networking* dalam server VPS yang menjadikan tidak bisanya mengakses PDF permasalahan dalam SharIF Judge dan menghasilkan *status 404*. Fitur ini diabaikan karena fitur ini tidak mengganggu pengujian yang sedang berjalan, tetapi untuk peserta melihat soal dengan cara mendownload soal dan melihatnya pada aplikasi lain yang akan berdampak pada analisis hasil eksperimen.

- Fitur Menyimpan Rekaman pada Sistem

Fitur ini tidak mengubah database pada saat pengguna melakukan aksi *submit*. Pada saat aksi *submit* dilakukan *file* rekaman dibuat menjadi *file* rekaman yang sudah di-*submit* tetapi pada daftar rekaman dalam database itu sendiri belum dihapus dan pada saat rekaman diambil, tidak ditemukannya rekaman tersebut dan mengembalikan error pada pengguna. Untuk menyelesaikan masalah ini, diperlukan sebuah fungsi baru dalam *Model Recording_model*. Kode 5.8 merupakan fungsi yang ditambahkan pada *Model Recording_model*.

Kode 5.8: Fungsi tambahan pada *Recording model*

```

1 public function remove_saveonly_recording($assignment_id, $problem_id, $username) {
2     $this->db->delete('recording', array(
3         'assignment' => $assignment_id,
4         'problem' => $problem_id,
5         'username' => $username,
6         'rec_id' => 0)

```

```
7 } );
8 }
```

Fungsi ini akan digunakan pada *Controller Recording* pada saat fungsi `_submit()` dipanggil. Setelah pengujian berakhir, peserta diminta untuk mengisi beberapa pertanyaan mengenai permasalahan yang dikerjakan dan juga beberapa pertanyaan mengenai pengalaman mengerjakan permasalahan pemrograman. Tetapi karena tidak ada eksperimen untuk fitur melihat perekaman, tidak ada umpan balik untuk fitur melihat rekaman.

Analisis Hasil Pengujian

Sebelum membahas pola-pola yang terlihat dalam histogram hasil rekaman peserta, perlu dijelaskan bahwa setiap jenis *event* atau peristiwa yang direkam ditampilkan dalam warna yang berbeda-beda. Gambar 5.1 menunjukkan representasi warna untuk masing-masing *event* yang direkam dalam histogram. Warna digunakan untuk membedakan jenis aktivitas yang dilakukan oleh peserta selama mengerjakan soal. Berikut adalah daftar *event* dan warna yang digunakan secara umum:

- Penambahan kode ditampilkan dengan warna hijau.
- Penghapusan kode ditampilkan dengan warna merah.
- Halaman aktif (fokus ke halaman IDE) menggunakan warna biru terang.
- Halaman tidak aktif (keluar dari fokus halaman) menggunakan warna biru gelap.
- Fokus ke PDF Viewer ditandai dengan warna kuning terang.
- Keluar dari PDF Viewer ditandai dengan warna kuning gelap.
- Perubahan pada bagian input menggunakan warna ungu terang.
- Perubahan pada bagian output menggunakan warna ungu gelap.
- Menyimpan kode ditampilkan dengan warna hitam.
- Menjalankan program menggunakan warna abu-abu.
- Mengumpulkan kode (`submit`) ditampilkan dengan warna putih.



Gambar 5.1: Representasi warna untuk setiap jenis peristiwa pada histogram

Analisis ini dilakukan dengan melihat data yang rekaman yang sudah di proses dan menjadi bagan histogram dalam sistem dan bagan lainnya yang dibutuhkan pada proses analisis. Ada beberapa pola yang dapat dilihat pada saat melihat histogram hasil rekaman peserta pengujian. Berikut merupakan pola yang dapat dilihat pada:

- Pola Pergantian Kode

Pola ini dapat dilihat pada *events editor.change* dan hasil kode yang dibuat. Pola ini didasari dengan adanya *Code Churn* atau bisa disebut dengan penggerjaan ulang kode. *Code Churn* ini terjadi pada saat peserta atau mahasiswa menghapus atau menulis ulang kode program [9]. Pada *Code Churn* ada yang dinamakan *Code Churn Rate* adalah sebuah tingkat pergantian kode selama sedang mengerjakan kode program tersebut. *Code Churn Rate* dapat menilai seberapa banyak kode berubah dalam suatu tim, module, file maupun fungsi. Pada pola ini digunakannya *Code Churn Rate* dalam tingkat file sebagai sebuah penilaian yang dapat dilakukan dengan menggunakan rumus sebagai berikut:

$$(inserted_kode + removed_kode) / total_kode$$

`inserted.kode` dihitung dengan menjumlahkan total karakter yang di masukkan dalam editor. Begitu juga dengan `removed.kode` dihitung dengan menjumlahkan total karakter yang dihapus dalam editor. `total.kode` dihitung dengan menjumlahkan jumlah total karakter pada hasil kode program akhir. Berikut contoh perhitungan untuk *Code Churn Rate*:



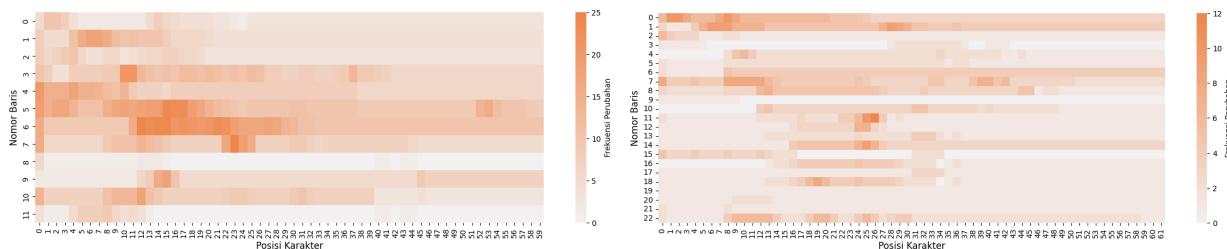
Gambar 5.2: Bagan Histogram Perubahan Kode Program

Pada Gambar 5.2 sebelah kiri peserta mengerjakan dengan tidak melakukan kecurangan. Terjadi 1436 karakter yang dimasukkan dan 2240 karakter yang dihapus pada editor dengan total kode akhir memiliki panjang 362 karakter. Maka menggunakan rumus di atas, *Code Churn Rate* pada peserta ini menjadi 10.15. Dengan nilai *Code Churn Rate* yang tinggi maka peserta ini banyak mengubah kode program.

Sebaliknya pada Gambar 5.2 sebelah kanan peserta menggunakan AI untuk mengerjakan dengan men-copy hasil kode AI ke dalam editor. Terjadi 1723 karakter yang dimasukkan dan 28 karakter yang dihapus pada editor dengan total kode akhir memiliki panjang 1771 karakter. Maka menggunakan rumus di atas, *Code Churn Rate* pada peserta ini menjadi 0.99. Dengan nilai *Code Churn Rate* yang rendah maka peserta ini tidak banyak mengubah kode program. Maka Pola Pergantian Kode ini dinilai dengan *Code Churn Rate*, semakin tinggi nilai tersebut maka semakin tinggi juga perubahan kode yang terjadi. Jika banyak perubahan kode maka dimungkinkan bahwa peserta tidak melakukan kecurangan pada saat pengerjaan.

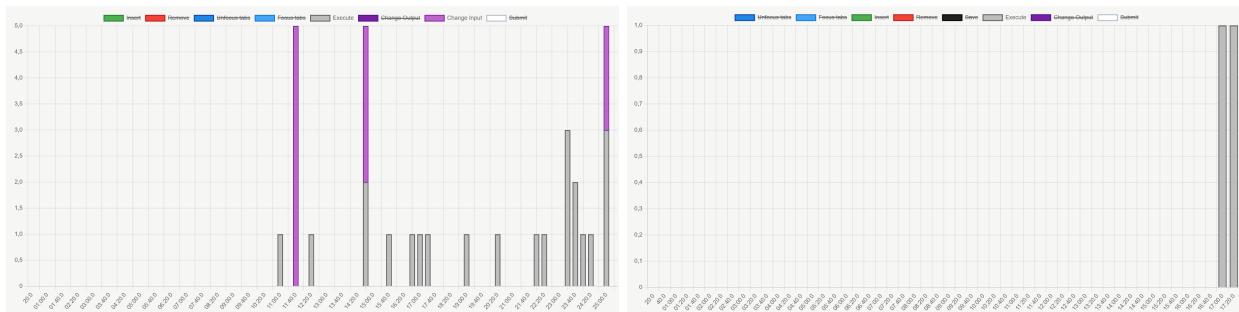
- Pola *Debugging*

Pola ini dapat dilihat pada `events editor.change`, `editor.changeCursor`, `editor.changeSelection`, `editor_input.input`, dan `editor_output.change`. Pengguna yang melakukan *debugging* pada saat mengerjakan akan lebih sering mengubah kode pada lokasi yang sama. Gambar 5.3 sebelah kiri menunjukkan bahwa frekuensi lebih tinggi akan berwarna lebih merah dan menandakan pola debugging pada posisi tertentu yang dapat dilihat karena terjadinya banyak perubahan pada posisi yang sama dibandingkan dengan Gambar 5.3 sebelah kanan yang memiliki frekuensi maksimum 12 pada warna yang paling merah.



Gambar 5.3: Bagan Heatmap Perubahan Lokasi Kode Program

Peserta yang melakukan *debugging* cenderung lebih sering mengubah bagian input dan mengeksekusi program dibandingkan peserta yang tidak melakukan *debugging*. Gambar 5.4 sebelah kiri menunjukkan bahwa terdapat banyak aktivitas perubahan input dan eksekusi program selama pengerjaan soal. Sebaliknya, pada Gambar 5.4 sebelah kanan, jumlah perubahan input dan eksekusi program jauh lebih sedikit. Perbedaan ini menunjukkan bahwa peserta pada gambar kiri kemungkinan besar sedang melakukan *debugging*, sedangkan peserta pada gambar kanan lebih langsung mengerjakan tanpa banyak percobaan atau pengujian.

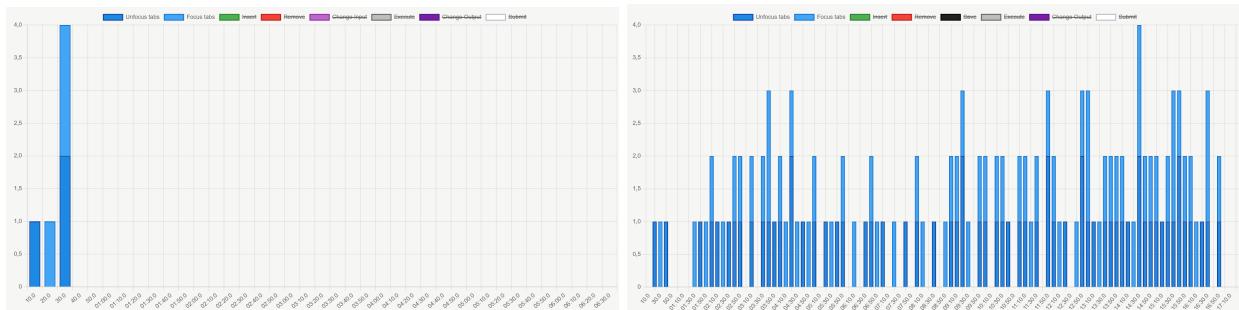


Gambar 5.4: Bagan Histogram Perubahan Input dan Aksi *Execute*

Pola *Debugging* ini mendukung Pola Pergantian Kode untuk mengetahui apakah peserta melakukan kecurangan. Tetapi tidak dapat mengetahui pasti peserta yang melakukan kecurangan.

- Pola Perubahan Navigasi

Pola ini dapat terlihat pada *events* `window.focus`, `window.blur`, dan `window.visibilitychange`. Pola ini dilihat dengan membandingkan seberapa sering pengguna mengubah navigasi dalam sesi mengerjakan kode program. Gambar 5.5 menunjukkan perbedaan bagan histrogram *event* `window.focus`, `window.blur`, dan `window.visibilitychange` pada peserta yang men-*copy* kode program dari jawaban *online* pada saat mengerjakan soal dan peserta yang mengerjakan soal hanya pada website SharIF Judge.



Gambar 5.5: Bagan Histogram Perubahan Navigasi

Pola Perubahan Navigasi dapat diukur dengan membatasi perubahan *event* tersebut dalam waktu tertentu. Jika nilai melebihi batas, maka peserta dapat diduga melakukan kecurangan.

- Pola Berpikir

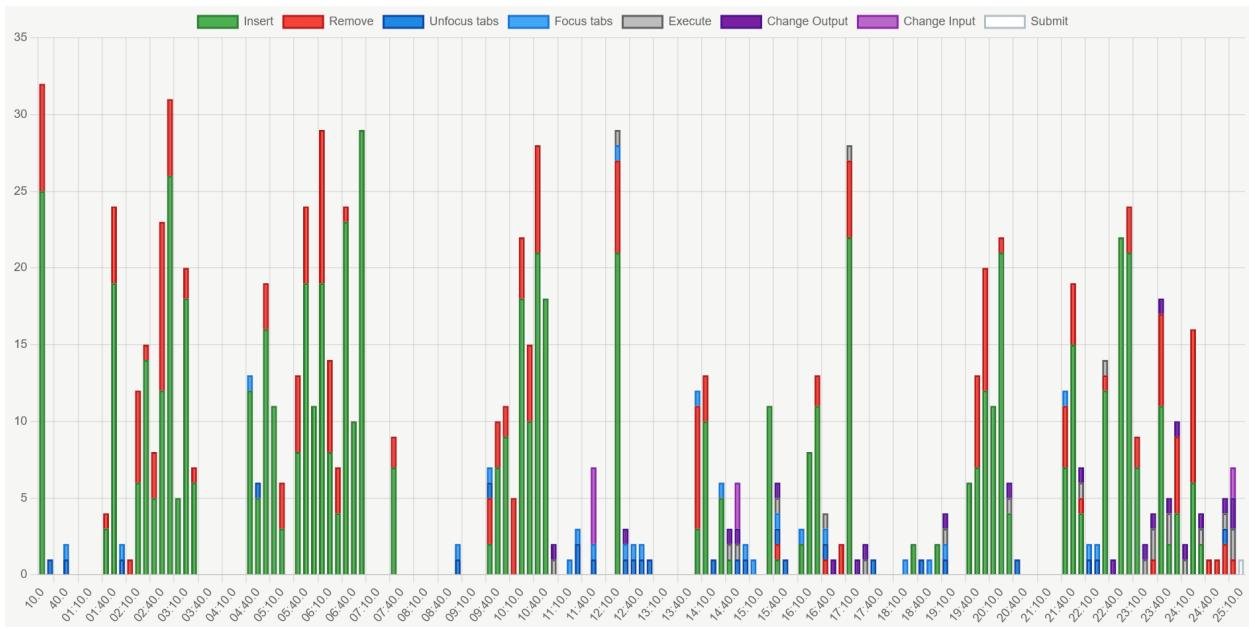
Pola ini dapat dilihat pada keseluruhan *events* yang terjadi. Pada Pola ini dapat dilihat dari saat pemberhentian peserta sebelum melakukan perubahan yang kompleks terhadap kode program dan juga keseringannya terjadinya pemberhentian.

Gambar 5.6 memiliki histogram yang dipisah setiap sepuluh detik dan memiliki waktu yang cukup lama sebelum melakukan perubahan yang cukup besar. Sebaliknya Gambar 5.7 menunjukkan peserta yang men-*copy* kode dari sumber jawaban *online*. Bagan ini menunjukkan lebih seringnya berhenti untuk melihat jawaban terlebih dahulu dan durasi pemberhentianya lebih kecil dibandingkan bagan histogram pertama.

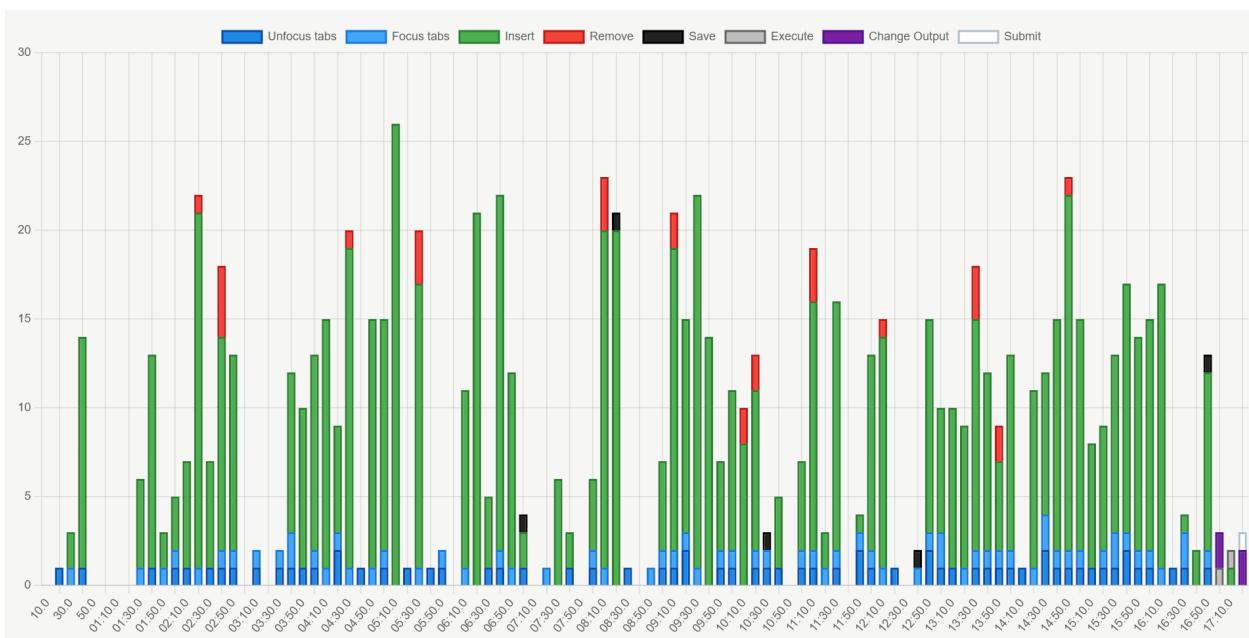
Pola ini dinilai dengan memberikan batas frekuensi berpikir dan lamanya berpikir dalam suatu waktu. Jika nilai melebihi batas, maka peserta dapat diduga melakukan kecurangan.

- Pola *Copy-Paste*

Pola ini merupakan pola sederhana dimana jika terjadinya penambahan kode yang panjang atau besar tetapi tidak melakukan penghapusan yang panjang atau besar terlebih dahulu, maka akan diduga melakukan kecurangan.



Gambar 5.6: Bagan Histogram Perubahan



Gambar 5.7: Bagan Histogram Perubahan

Tetapi pola-pola tersebut dapat juga mendapatkan *false positive* karena banyaknya *variable* yang harus diperhatikan dalam menentukan tidakan kecurangan itu sendiri. *Variable* yang dimaksudkan antara lain adalah kesulitan masalah dan pengalaman programing. Jika kesulitan masalah mudah maka peserta dapat mengerjakan dengan mudah dan tidak memerlukannya *debugging*, begitu juga jika pengalaman programing yang tinggi.

BAB 6

KESIMPULAN DAN SARAN

Bab ini akan membahas kesimpulan dari hasil analisis, perancangan, implementasi, pengujian, dan saran-saran untuk pengembangan yang dapat dipertimbangkan.

6.1 Kesimpulan

Berdasarkan proses dan hasil analisis, perancangan, implementasi, dan pengujian perangkat lunak yang telah dibuat, maka diperoleh kesimpulan sebagai berikut:

1. Implementasi fitur merekam dan memutar ulang ketikan pada SharIF judge berhasil dikembangkan dengan memanfaatkan fitur dalam *CodeIgniter*, *library twig* sebagai tampilan, *library ace* sebagai editor kode, *library Chart.js* sebagai pembuat bagan dan fitur pada *JavaScript*.
2. Pemutaran ulang rekaman dapat memberikan gambaran yang jelas mengenai proses berpikir mahasiswa, termasuk aktivitas seperti pengetikan kode, eksekusi program, perubahan tab, hingga pola interaksi lainnya dalam menyelesaikan soal. Dengan adanya rekaman ini, dosen dapat menilai apakah mahasiswa menyelesaikan tugasnya secara mandiri atau tidak.
3. Eksperimen membuktikan bahwa pola penggerjaan mahasiswa yang normal dan yang terindikasi melakukan kecurangan dapat dibedakan melalui histogram interaksi dan analisis terhadap perilaku seperti copy-paste kode, kecepatan ketikan, frekuensi berpikir (pause sebelum mengetik ulang), dan intensitas debugging.
4. Analisis data rekaman menunjukkan potensi besar dalam mendeteksi kecurangan secara manual, walaupun sistem belum secara otomatis memberikan penilaian atau keputusan. Hal ini membuka peluang untuk pengembangan sistem pendekripsi otomatis berbasis pola.
5. Sistem ini dibatasi pada aktivitas yang terekam dalam halaman IDE, sehingga kecurangan di luar sistem seperti menggunakan perangkat lain, tidak dapat dideteksi. Namun demikian, sistem ini sudah menyediakan data yang cukup kaya untuk dilakukan analisis awal terhadap dugaan kecurangan.

6.2 Saran

Berdasarkan proses dan hasil analisis, perancangan, implementasi, dan pengujian perangkat lunak yang telah dibuat, maka diperoleh saran-saran sebagai berikut:

1. Pengembangan fitur analisis otomatis terhadap pola interaksi, yang dapat memberikan notifikasi kepada dosen apabila ditemukan indikasi kuat terhadap kecurangan, misalnya pola copy-paste mendadak.
2. Optimalisasi penyimpanan rekaman, mengingat setiap aktivitas mahasiswa disimpan dalam bentuk JSON yang dapat menjadi besar. Kompresi data atau sistem penyimpanan dapat dipertimbangkan agar lebih efisien.
3. Pada saat pengujian, banyak peserta yang menyampaikan keluhan mengenai perilaku IDE terutama untuk mengetahui kesalahan dalam kode program berbahasa *Python*. Maka IDE dapat dioptimasi agar kesalahan dalam kode program dapat dimunculkan.

4. Menambahkan sebuah pilihan pada saat pembuatan tugas atau *assignment* untuk menyalakan *auto-complete* kode dalam SharIF Judge.
5. Perluasan cakupan rekaman aktivitas, misalnya dengan merekam tangkapan layar atau memperluas deteksi tab yang dibuka pengguna, selama masih dalam ranah yang etis dan tidak melanggar privasi mahasiswa.
6. Melanjutkan eksperimen dengan skala yang lebih besar, agar dapat ditentukan batas-batas kuantitatif yang lebih akurat dalam mengidentifikasi pola-pola interaksi mahasiswa yang mencurigakan.

DAFTAR REFERENSI

- [1] Prihatini, F. N. dan Indudewi, D. (2016) Kesadaran dan Perilaku Plagiarisme dikalangan Mahasiswa(Studi pada Mahasiswa Fakultas Ekonomi Jurusan Akuntansi Universitas Semarang). *Dinamika Sosial Budaya*, **18**, 68–75.
- [2] Önder Demir, Aykut Soysal, Ahmet Arslan, Burcu Yürekli, dan Özgür Yilmazel (2010) Automatic grading system for programming homework. *Proceedings of the Annual International Conference on Computer Science Education: Innovation & Technology CSEIT 2010 & Proceedings of the Annual International Conference on Software Engineering SE 2010*, **18**, 68–75.
- [3] Kurnia, A., Lim, A., dan Cheang, B. (2001) Online judge. *Computers & Education*, **18**, 299–315.
- [4] IDCloudHost (2020) Algoritma pemrograman beserta contohnya. <https://idcloudhost.com/blog/algoritma-pemrograman-pengertian-fungsi-cara-kerja-dan-contohnya/>. 6 Desember 2024.
- [5] Vallian, S. (2018) Kustomisasi Sharif Judge Untuk Kebutuhan Program Studi Teknik Informatika. Skripsi. Universitas Katolik Parahyangan, Indonesia.
- [6] Version 1.4 (2014) *Sharif Judge Documentation*. Mohammad Javad Naderi. Tehran, Iran.
- [7] Halim, N. A. (2021) Implementasi Editor Kode pada SharIF Judge. Skripsi. Universitas Katolik Parahyangan, Indonesia.
- [8] Version (2025) *Chart.js Documentation*. Chart.js Contributors. Open Source Project.
- [9] Ghufran, H. (2022) What is code churn? <https://www.hatico.io/blog/code-churn-rate/>. 2 Juni 2025.

LAMPIRAN A

KODE PROGRAM

Kode A.1: Install.php

```

1 diff --git a/application/controllers/Install.php b/application/controllers/Install.php
2 index 2d87d1f..389e195 100644
3 --- a/application/controllers/Install.php
4 +++ b/application/controllers/Install.php
5 @@ -45,7 +45,7 @@ class Install extends CI_Controller
6
7     // Use InnoDB engine for MySQL database
8     if ($this->db->dbdriver === 'mysql' || $this->db->dbdriver === 'mysqli')
9     -
10        $this->db->query('SET storage_engine=InnoDB;');
11    +
12        $this->db->query('SET default_storage_engine=InnoDB;');
13
14     // Creating Tables:
15     // sessions, submissions, assignments, notifications, problems, queue, scoreboard, settings, users
16     @@ -86,9 +86,30 @@ class Install extends CI_Controller
17         $this->dbforge->add_key(array('assignment', 'submit_id'));
18         if ( ! $this->dbforge->create_table('submissions', TRUE))
19             show_error("Error creating database table ".$this->db->dbprefix('submissions'));
20
21     +
22     +
23     // create table 'recording'
24     $fields = array(
25         'rec_id'      => array('type' => 'INT', 'constraint' => 11, 'unsigned' => TRUE),
26         'upload_at'   => array('type' => $DATETIME),
27         'assignment'  => array('type' => 'SMALLINT', 'constraint' => 4, 'unsigned' => TRUE),
28         'problem'    => array('type' => 'SMALLINT', 'constraint' => 4, 'unsigned' => TRUE),
29         'username'   => array('type' => 'VARCHAR', 'constraint' => 20),
30     );
31     $this->dbforge->add_field($fields);
32     if ( ! $this->dbforge->create_table('recording', TRUE))
33         show_error("Error creating database table " . $this->db->dbprefix('recording'));
34     // ADD Unique constraint
35     $this->db->query(
36         "ALTER TABLE {$this->db->dbprefix('recording')}"
37         "ADD CONSTRAINT {$this->db->dbprefix('srup_unique')} UNIQUE (rec_id, username, assignment, problem);"
38     );
39
40     // create table 'assignments'
41     $fields = array(
42         'id'          => array('type' => 'INT', 'constraint' => 11, 'unsigned' => TRUE, 'auto_increment' => TRUE),

```

Kode A.2: main.css

```

1 diff --git a/assets/styles/main.css b/assets/styles/main.css
2 index 57a6f61..1b3d981 100644
3 --- a/assets/styles/main.css
4 +++ b/assets/styles/main.css
5 @@ -432,6 +432,7 @@ div#extra_time i {
6     .color-notifications {border-color: #FAE80B!important;}
7     .color-assignments {border-color: #CCDB29!important;}
8     .color-problems {border-color: #0DB297!important;}
9     +.color-recording {border-color: #0d99b2!important;}
10    .color-submit {border-color: #35C0CB!important;}
11    .color-final_submissions {border-color: #22478A!important;}
12    .color-all_submissions {border-color: #543680!important;}
13 @@ -445,6 +446,7 @@ div#extra_time i {
14     .color-notifications.selected i {background-color: #FAE80B!important; border-color: #FAE80B!important;}
15     .color-assignments.selected i {background-color: #CCDB29!important; border-color: #CCDB29!important;}
16     .color-problems.selected i {background-color: #0DB297!important; border-color: #0DB297!important;}
17     +.color-recording.selected i {background-color: #0d99b2!important; border-color: #0d99b2!important;}
18     .color-submit.selected i {background-color: #35C0CB!important; border-color: #35C0CB!important;}
19     .color-final_submissions.selected i {background-color: #22478A!important; border-color: #22478A!important;}
20     .color-all_submissions.selected i {background-color: #543680!important; border-color: #543680!important;}

```

Kode A.3: recording.css

```

1 /* Overall */
2
3 .title_section {
4     margin: 0;

```

```
5     display: flex;
6     align-items: center;
7     font-size: 1.5em;
8 }
9 .title_normal {
10    font-weight: normal;
11    justify-content: flex-start;
12 }
13 /* Loader */
14 .loader {
15     border: 4px solid rgba(255, 255, 255, 0.3);
16     border-top: 4px solid #3498db;
17     border-radius: 50%;
18     width: 40px;
19     height: 40px;
20     animation: spin_2s_linear_infinite;
21     -webkit-animation: spin_2s_linear_infinite; /* Safari */
22 }
23 /* Safari */
24 @webkit-keyframes spin {
25     0% {
26         -webkit-transform: rotate(0deg);
27     }
28     100% {
29         -webkit-transform: rotate(360deg);
30     }
31 }
32 @keyframes spin {
33     0% {
34         transform: rotate(0deg);
35     }
36     100% {
37         transform: rotate(360deg);
38     }
39 }
40 /* Recording Header */
41 div#wrap_selector {
42     display: grid;
43     grid-template-columns: minmax(100px, auto) 1fr;
44     column-gap: 20px;
45     row-gap: 8px;
46     margin-bottom: 8px;
47 }
48 h2.title_selector {
49     justify-content: flex-end;
50 }
51 table#wrap_sel_problem {
52     table-layout: fixed;
53     width: 100%;
54     height: 32px;
55 }
56 td.sel_problem {
57     text-align: center;
58     border-bottom: 2px solid #dddad0;
59     border-left: 2px solid #dddad0;
60     background: #faf9f5;
61 }
62 td.selected {
63     background: #e0e0e0;
64 }
65 td.sel_problem > a {
66     display: flex;
67     justify-content: center;
68     align-items: center;
69     width: 100%;
70     height: 100%;
71     text-decoration: none;
72     /* text-align: center; */
73     /* vertical-align: middle; */
74 }
75 div#recording_wrap {
76     display: grid;
77     /* width: ; */
78     min-height: 50vh;
79     gap: 8px;
80 }
81 /* IDE In Recording */
82 div#ide_wrap {
83     display: flex;
84     /* width: calc(100vw - 200px); */
85     height: 50vh;
86     padding: 0;
87     border: 0;
88     position: relative;
```

```

104 }
105 fieldset#editor_wrap {
106   width: 70%;
107   height: 100%;
108   padding: 0;
109   border: 0;
110 }
111 }
112 div#code_editor {
113   width: 100%;
114   height: 100%;
115   padding: 0;
116   border: 0;
117 }
118 }
119 fieldset#in_wrap {
120   width: 15%;
121   height: 100%;
122   padding: 0;
123   border: 0;
124 }
125 }
126 fieldset#out_wrap {
127   width: 15%;
128   height: 100%;
129   padding: 0;
130   border: 0;
131 }
132 }
133 textarea.in_out {
134   resize: none;
135   width: 100%;
136   height: 100%;
137 }
138 }
139 /* Status */
140 #status_wrapper {
141   position: absolute;
142   display: flex;
143   /* display: none; */
144   align-items: center;
145   justify-content: center;
146   top: 0;
147   bottom: 0;
148   left: 0;
149   right: 0;
150   background-color: rgba(0, 0, 0, 0.3);
151   z-index: 10;
152   border-radius: 8px;
153   font-size: 5em;
154   font-weight: 900;
155   color: #fff;
156 }
157 }
158 }
159 /* Input Player */
160 div.wrap_input_player {
161   display: flex;
162   flex-direction: row;
163   align-items: center;
164   gap: 8px;
165 }
166 }
167 }
168 div.wrap_input_player > input {
169   flex: 1;
170 }
171 }
172 }
173 #wrap_input_range {
174   width: 100%;
175   display: flex;
176   flex-direction: row;
177 }
178 }
179 .wrap_range {
180   width: 100%;
181 }
182 }
183 .range {
184   appearance: none;
185   background: #444;
186   cursor: pointer;
187   height: 8px;
188   margin: 0;
189   transition: 0.1s ease-in;
190   vertical-align: bottom;
191   width: 100%;
192   border-radius: 8px;
193   margin-bottom: 4px;
194 }
195 .wrap_range:hover .range,
196 .wrap_range.hover .range {
197   height: 16px;
198   margin-bottom: 0px;
199 }
200 }
201 .range::-webkit-slider-thumb {
202   appearance: none;

```

```

203     background:#aeaea;
204     border-radius:8px;
205     box-shadow:inset 0 0 0 5px #aeaea;
206     height:0;
207     transition:0.1s_ease-in;
208     width:0;
209 }
210 .wrap_range:hover.range::-webkit-slider-thumb,
211 .wrap_range.hover.range::-webkit-slider-thumb{
212   width:16px;
213   height:16px;
214 }
215 /* List Recording */
216 .sel_recording {
217   cursor:pointer;
218 }
219 .sel_selected {
220   font-weight:bold;
221   text-decoration:underline;
222 }
223 /* Chart */
224 #wrap_config_chart {
225   display:grid;
226   gap:16px;
227   grid-template-columns:100px_1fr;
228 }
```

Kode A.4: Recording.php

```

1 <?php
2 /**
3  * SharIF Judge online judge
4  * @file Recording.php
5  * @author Andreas Ronaldi <andreasronaldi25@gmail.com>
6  */
7 */
8 defined('BASEPATH') or exit('No direct script access allowed');
9
10 class Recording extends CI_Controller
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 <= 1) // permission denied
18       show_404();
19
20     $this->load->model('recording_model');
21
22     $input = $this->uri->uri_to_assoc();
23
24     // var_dump($input);
25
26     $this->filter_user = $this->filter_problem = NULL;
27     if (array_key_exists('user', $input) && $input['user'])
28       $this->filter_user = $this->form_validation->alpha_numeric($input['user'])? $input['user']:NULL;
29     if (array_key_exists('problem', $input) && $input['problem'])
30       $this->filter_problem = is_numeric($input['problem'])? $input['problem']:NULL;
31   }
32
33   // For seeing all recording
34   public function all()
35   {
36     $assignment_id = $this->user->selected_assignment['id'];
37
38     if ($assignment_id == 0)
39       show_error('No_assignment_selected.');
40
41     $problem = $this->assignment_model->all_problems($assignment_id);
42     $assignment = $this->assignment_model->assignment_info($assignment_id);
43     $recordings = $this->recording_model->all_user_recordings($assignment_id, $this->filter_problem, $this->filter_user);
44     $names = $this->user_model->get_names();
45
46     foreach ($recordings as &$item) {
47       $item['name'] = $names[$item['username']];
48     }
49
50     $data = array(
51       'all_problems' => $problem,
52       'assignment' => $assignment,
53       'recordings' => $recordings,
54       'filter_problem' => $this->filter_problem,
55       'filter_user' => $this->filter_user,
56     );
57
58     $this->twig->display('pages/recording_list.twig', $data);
59   }
60
61   public function index($assignment_id = NULL, $problem_id = 1, $username = NULL, $rec_id = NULL)
62   {
63     // If no assignment is given
64     if ($assignment_id === NULL)
65       redirect('recording/all');
```

```

66|     if ($assignment_id == 0)
67|         show_error('No_assignment_selected.');
68|
69|     $assignment = $this->assignment_model->assignment_info($assignment_id);
70|
71|     $data = array(
72|         'all_problems' => $this->assignment_model->all_problems($assignment_id),
73|         'assignment' => $assignment,
74|         'filter_user' => $username,
75|         'filter_problem' => $problem_id,
76|     );
77|
78|     if ($username !== NULL) {
79|         $list_rec = $this->recording_model->get_user_recordings($assignment_id, $problem_id, $username);
80|         $data['list_recording'] = $list_rec;
81|
82|         if ($rec_id === NULL && $list_rec) {
83|             $rec_id = $list_rec[0]['rec_id'];
84|         }
85|
86|         $data['rec_id'] = $rec_id;
87|     }
88|
89|     $this->twig->display('pages/recording.twig', $data);
90}
91|
92|
93| public function download_record($assignment_id, $problem_id, $username, $rec_id)
94| {
95|     $assignment_root = rtrim($this->settings_model->get_setting('assignments_root'), '/');
96|     $file_path = $assignment_root.'/assignment'.$assignment_id.'/'.$problem_id.'/'.$username;
97|     $rec_path = $file_path.'/'.$RECORD_FILE_NAME.'.'.$RECORD_FILE_EXT;
98|
99|     if ($rec_id !== "0") {
100|         $rec_path = $file_path.'/'.$RECORD_FILE_NAME.'-'.$rec_id.'.'.$RECORD_FILE_EXT;
101|     }
102|
103|     $this->load->helper('file');
104|     $this->load->helper('url');
105|
106|     if (!file_exists($rec_path)) {
107|         throw new Exception("File $rec_path does not exist");
108|     }
109|     if (!is_readable($rec_path)) {
110|         throw new Exception("File $rec_path is not readable");
111|     }
112|
113|     $file = glob($rec_path);
114|     $content = file_get_contents($file[0]);
115|     header('Content-Type: application/json');
116|     header('Content-Disposition: attachment; filename="rec.json"');
117|     die($content);
118}
119}

```

Kode A.5: recording.twig

```

1  {#
2  # SharIF Judge
3  # file: recording.twig
4  # author: Andreas Ronaldi <andreasronaldi25@gmail.com>
5  #}
6  {% set selected = 'recording' %}
7  {% extends 'templates/base.twig' %}
8  {% block icon %}fa-video-camera
9  {% endblock %}
10 | {% block title %}Recording
11 | {% endblock %}
12 | {% block head_title %}Recording
13 | {% endblock %}
14 |
15 |
16 | {% block other_assets %}
17 |     <link rel="stylesheet" type='text/css' href="{{ _base_url('assets/styles/recording.css') }}"/>
18 |
19 |     <script src="{{ base_url('assets/ace/ace.js') }}></script>
20 |     <script src="{{ base_url('assets/ace/ext-language_tools.js') }}></script>
21 |
22 |     <script>
23 |         var rec_path='{{ assignment.id }}/{{ filter_problem }}/{{ filter_user }}/{{ rec_id }}';
24 |     </script>
25 |
26 |     <script src="{{ _base_url("assets/chartjs/dist/chart.umd.js") }}></script>
27 |
28 |     <script type='text/javascript' src="{{ _base_url("assets/js/shj_recording.js") }}></script>
29 | {% endblock %}
30 |
31 | {% block main_content %}
32 | <div id="main_rec">
33 |     {% if all_problems|length == 0 %}
34|         <p style="text-align:center;">Nothing to show...</p>
35 |     {% else %}
36 |         <div id="wrap_selector">
37 |             <h2 class="title_section_title_selector">Problem</h2>
38 |             <table id="wrap_sel_problem">
39 |                 <tr>
40 |                     {% for one_problem in all_problems %}
41 |                         <td class="sel_problem{{ _filter_problem == one_problem.id ? '_selected' }}>

```

```

42          <a dir="auto" href="{{_site_url("recording/#{{assignment.id}}/{{one_problem.id}}/{{filter_user}}")}}>{{
43              one_problem.name }}</a>
44      </td>
45      {% endfor %}
46  </tr>
47 </table>
48
49 <h2 class="title_section_title_selector">Username</h2>
50 <div id="wrap_sel_user">
51     <h2 class="title_section_title_normal">{{ filter_user }}</h2>
52 </div>
53 <h2 class="title_section_title_selector">Status</h2>
54 <h2 id="status_rec" class="title_section_title_normal"></h2>
55 </div>
56
57 {% if list_recording is not empty %}
58 <div id="recording_wrap">
59
60     {%# Chart #}
61     <h2 class="title_section">Player</h2>
62     <div id="ide_wrap">
63         <fieldset id="editor_wrap">
64             <legend>Code</legend>
65             <div id="code_editor"></div>
66         </fieldset>
67         <fieldset id="in_wrap">
68             <legend>Input</legend>
69             <textarea id="editor_input" class="in_out" readonly></textarea>
70         </fieldset>
71         <fieldset id="out_wrap">
72             <legend>Output</legend>
73             <textarea id="editor_output" class="in_out" readonly></textarea>
74         </fieldset>
75         <div id="status_wrapper">
76             Status
77         </div>
78     </div>
79
80     {%# Player #}
81     <div class="wrap_input_player">
82         {%# For Playing Currently %}
83         <button type="button" class="sharif_input" id="rec_btn">Play</button>
84         <span id="timer_player">00:00</span>
85
86         <div class="wrap_range">
87             <input type="range" min="0" max="0" value="0" class="range" id="range_player">
88         </div>
89     </div>
90
91     {%# List of saved %}
92     <h2 class="title_section">List Of Work on Recording</h2>
93     <table class="sharif_table">
94         <thead>
95             <tr>
96                 <th>Start Time</th>
97                 <th>End Time</th>
98                 <th>Actions</th>
99             </tr>
100        </thead>
101        <tbody id="tbody_saved">
102        </tbody>
103    </table>
104
105    {%# Chart #}
106    <h2 class="title_section">Config Chart</h2>
107    <div class="wrap_config_chart">
108        <label for="config_chart_type">Chart Type</label>
109        <select id="config_chart_type" name="config_chart_type">
110            <option value="bar">Bar</option>
111            <option value="line">Line</option>
112        </select>
113        <label for="config_chart_stack">Stack</label>
114        <input type="checkbox" value="stack" id="config_chart_stack" name="config_chart_stack" checked></input>
115        <label for="config_chart_divider">Divider<label>
116        <input min="0" value="1" id="config_chart_divider" name="config_chart_divider" type="number"></input>
117        <label for="config_chart_time">Time<label>
118        <select id="config_chart_time" name="config_chart_time"></select>
119    </div>
120
121    <h2 class="title_section">Chart Of Action</h2>
122    <canvas id="recording_chart"></canvas>
123
124    <h2 class="title_section">Other Recording of {{ filter_user }} in problem {{ filter_problem }}</h2>
125    <table class="sharif_table">
126        <thead>
127            <tr>
128                <th>ID</th>
129                <th>Username</th>
130                <th>Submit Time</th>
131                <th>Actions</th>
132            </tr>
133        </thead>
134        <tbody>
135            {% for recording in list_recording %}
136            <tr>
137                <td>{{ recording.rec_id }}</td>
138                <td>{{ recording.username }}</td>
139                <td>{{ recording.upload_at }}</td>

```

```

140
141          <td>
142              <a dir="auto" href="{{site_url("recording/#{assignment.id}/#{filter_problem}/#{recording.username})}}>See Recording</a>
143          </td>
144      {% endfor %}
145  </tbody>
146 </table>
147 </div>
148 {% else %}
149 Nothing Here...
150 {% endif %}
151 {% endif %}
152 </div>
153 {% endblock %}
154 {# main_content #}

```

Kode A.6: recording_list.twig

```

1  {#
2  # SharIF Judge
3  # file: recording_list.twig
4  # author: Andreas Ronaldi <andreasronaldi25@gmail.com>
5  #}
6  {% set selected = 'recording' %}
7  {% extends 'templates/base.twig' %}
8  {% block icon %}fa-video-camera
9  {% endblock %}
10 {% block title %}Recording
11 {% endblock %}
12 {% block head_title %}Recording
13 {% endblock %}
14
15
16 {% block other_assets %}
17     <link rel="stylesheet" type='text/css' href="{{base_url('assets/styles/recording.css')}}"/>
18
19     <script>
20         var test = '{{ all_problems | json_encode(constant('JSON_PRETTY_PRINT')) }}';
21         console.log(test)
22     </script>
23 {% endblock %}
24
25
26 {% block main_content %}
27     {% if all_problems|length == 0 %}
28         <p style="text-align: center;>Nothing to show...</p>
29     {% else %}
30         <div id="wrap_selector">
31             <h2 class="title_section_title_selector">Problem</h2>
32             <table id="wrap_sel_problem">
33                 <tr>
34                     <td class="sel_problem{{_.filter_problem_is_null_?'selected'}}">
35                         <a dir="auto" href="{{site_url('recording/all/~(filter_user?/user/~filter_user))}}>All</a>
36                     </td>
37                     {% for one_problem in all_problems %}
38                         <td class="sel_problem{{_.filter_problem_==one_problem.id_?'selected'}}">
39                             <a dir="auto" href="{{site_url('recording/all/~(filter_user?/user/~filter_user)~/problem/~one_problem.id)}}>{{ one_problem.name }}</a>
40                         </td>
41                     {% endfor %}
42                 </tr>
43             </table>
44         </div>
45
46         <table class="sharif_table">
47             <thead>
48                 <tr>
49                     <th>ID</th>
50                     <th>Username</th>
51                     <th>Name</th>
52                     <th>Problem</th>
53                     <th>Submit Time</th>
54                     <th>Actions</th>
55                 </tr>
56             </thead>
57             {% for recording in recordings %}
58                 <tr>
59                     <td>{{ recording.rec_id }}</td>
60                     <td><a dir="auto" href="{{site_url('recording/all/user/~recording.username~(filter_problem?/problem/~filter_problem))}}>{{ recording.username }}</a></td>
61                     <td>{{ recording.name }}</td>
62                     <td><a dir="auto" href="{{site_url('recording/all/~(filter_user?/user/~filter_user)~/problem/~recording.problem)}}>{{ recording.problem }}</a></td>
63                     <td>{{ recording.upload_at }}</td>
64                     {% Action #}
65                     <td>
66                         <a dir="auto" href="{{site_url("recording/#{assignment.id}/#{recording.problem}/#{recording.username})}}>See Recording</a>
67                     </td>
68                 </tr>
69             {% endfor %}
70         </table>
71
72         {% if username != NULL %}
73             Please used /see_recording/...
74         {% endif %}
75

```

```

76    {% endif %}
77  {% endblock %}
78  {% main_content %}
```

Kode A.7: Recording_model.php

```

1 <?php
2
3 /**
4  * SharIF Judge online judge
5  * @file Submit_model.php
6  * @author Andreas Ronaldi <andreasronaldi25@gmail.com>
7  */
8 defined('BASEPATH') or exit('No direct script access allowed');
9
10 class Recording_model extends CI_Model
11 {
12     public function __construct()
13     {
14         parent::__construct();
15     }
16
17     public function all_user_recordings($assignment_id, $filter_problem = NULL, $filter_user = NULL) {
18         $arr['assignment'] = $assignment_id;
19         if ($filter_problem !== NULL)
20             $arr['problem'] = $filter_problem;
21         if ($filter_user !== NULL)
22             $arr['username'] = $filter_user;
23
24         return $this->db
25             ->order_by('upload_at_asc')
26             ->get_where('recording', $arr)
27             ->result_array();
28     }
29
30     public function get_user_recordings($assignment_id, $problem_id, $username) {
31         $arr['assignment'] = $assignment_id;
32         $arr['problem'] = $problem_id;
33         $arr['username'] = $username;
34
35         return $this->db
36             ->order_by('upload_at_asc')
37             ->get_where('recording', $arr)
38             ->result_array();
39     }
40
41     public function add_recording($rec_info) {
42         $this->db->replace('recording', $rec_info);
43     }
44
45     public function remove_saveonly_recording($assignment_id, $problem_id, $username) {
46         $this->db->delete('recording', array(
47             'assignment' => $assignment_id,
48             'problem' => $problem_id,
49             'username' => $username,
50             'rec_id' => 0
51         ));
52     }
53 }
```

Kode A.8: routes.php

```

1 diff --git a/application/config/routes.php b/application/config/routes.php
2 index 060bcbb8..a6c2d54 100644
3 --- a/application/config/routes.php
4 +++ b/application/config/routes.php
5 @@ -1,5 +1,5 @@
6 <?php
7 -defined('BASEPATH') OR exit('No direct script access allowed');
8 +defined('BASEPATH') or exit('No direct script access allowed');
9
10 /*
11 | -----
12 @@ -52,15 +52,19 @@ defined('BASEPATH') OR exit('No direct script access allowed');
13 $route['default_controller'] = 'dashboard';
14 $route['register'] = "login/register";
15 $route['logout'] = "login/logout";
16 $route['submissions/final']="submissions/the_final";
17 $route['submissions/final(.*)']="submissions/the_final/$1";
18 $route['profile/(:num)'] = "profile/index/$1";
19 $route['moss/(:num)'] = "moss/index/$1";
20 $route['problems/(:num)'] = "problems/index/$1";
21 $route['problems/(:num)/(:num)'] = "problems/index/$1/$2";
22 $route['rejudge/(:num)'] = "rejudge/index/$1";
23 +$route['recording/(:num)/(:num)/(:any)'] = "recording/index/$1/$2/$3";
24 +$route['recording/(:num)/(:num)/(:any)/(:num)'] = "recording/index/$1/$2/$3/$4";
25 $route['404_override'] = '';
26 $route['translate_uri_dashes'] = FALSE;
```

Kode A.9: shj_recording.js

```

1 /**
2  * SharIF Judge
```

```

3  * @file shj_recording.js
4  * author: Andreas Ronaldi <andreasronaldi25@gmail.com>
5  *
6  *     Javascript codes for "Recording" page
7  */
8
9 $(document).ready(() => {
10    // ##### Variable #####
11    // ##### Variable #####
12    // ##### Variable #####
13    const editor = ace.edit("code_editor");
14    const Range = ace.Range;
15
16    let funcTimeoutRecording = null;
17    let funcIntervalTimer = null;
18
19    let curChart = null;
20    let isRecPlaying = false;
21
22    let confTimer = {
23        delay: 1,
24    };
25
26    let confChart = {
27        type: "bar",
28        autoDivider: 20, // How many section does the auto divider will giveout (default: 10)
29        divider: 1,
30        time: "s",
31        stack: true,
32    };
33
34    let confPlayer = {
35        durationNext: 500, // in ms
36        mergeDuration: 0, // if time difference smaller than value, then merge the events
37    };
38
39    editor.setOptions({
40        theme: "ace/theme/monokai",
41        fontSize: "11pt",
42        readOnly: true,
43        enableLiveAutocompletion: true,
44        enableBasicAutocompletion: true,
45        enableSnippets: true,
46    });
47
48    // Object for playing record
49    const recording = {
50        events: {}, // Map -> time events to list of events
51        eventsIndex: {}, // Map -> index to time events
52        indexEvents: {}, // Map -> time events to index
53        presumIndexDuration: {}, // Map -> index to duration before timestamp event
54        length: -1, // Length of list of saved events
55        curEvents: -1, // Currently selected index
56        duration: 0, // Duration of events
57        save_state: [], // Saved state to use in playback
58
59        reset: () => {
60            recording.events = {};
61            recording.eventsIndex = {};
62            recording.indexEvents = {};
63            recording.presumIndexDuration = {};
64            recording.length = -1;
65            recording.curEvents = -1;
66            recording.duration = 0;
67            recording.save_state = [];
68        },
69    };
70
71    const handlersIncludeInGoState = {
72        insert: true,
73        remove: true,
74        cursor_selection: true,
75        sel_selection: true,
76        focus: false,
77        blur: false,
78        visibility: false,
79        pdf_focus: false,
80        pdf_blur: false,
81        input_change: true,
82        output_change: true,
83        save: false,
84        submit: false,
85        execute: false,
86    };
87
88    const handlers = {
89        insert: (args) => {
90            editor.session.replace(
91                new ace.Range(
92                    args.start.row,
93                    args.start.column,
94                    args.start.row,
95                    args.start.column
96                ),
97                args.data.join("\n")
98            );
99        },
100        remove: (args) => {
101            editor.session.remove({ start: args.start, end: args.end });
101

```

```

102 },
103 cursor_selection: (args) => {
104     setSelection(editor, args);
105 },
106 sel_selection: (args) => {
107     setSelection(editor, args);
108 },
109 focus: (args) => {
110     setStatus("User_is_focus_now");
111     setTitle("User_is_focus_now");
112 },
113 blur: (args) => {
114     setStatus("User_is_not_focus_on_website_now");
115     setTitle("User_is_not_focus_on_website_now");
116 },
117 visibility: (args) => {
118     if (args) {
119         setStatus("Open_ShariF-Judge");
120         setTitle("User_is_on_ShariF-Judge_now");
121     } else {
122         setStatus("Switching_tabs");
123         setTitle("User_is_on_other_tabs_now");
124     }
125 },
126 pdf_focus: (args) => {
127     setStatus("User_is_focus_on_pdf_viewer_now");
128     setTitle("User_is_focus_on_pdf_viewer_now");
129 },
130 pdf_blur: (args) => {
131     setStatus("User_is_not_focus_on_pdf_viewer_now");
132     setTitle("User_is_not_focus_on_pdf_viewer_now");
133 },
134 input_change: (args) => $("#editor_input").val(args),
135 output_change: (args) => $("#editor_output").val(args),
136 save: (args) => {
137     setStatus("User_just_saved");
138     setTitle("User_just_saved");
139 },
140 submit: (args) => {
141     setStatus("User_just_submit!");
142     setTitle("User_just_submit!");
143 },
144 execute: (args) => {
145     setStatus("User_is_running_the_program.");
146     setTitle("User_is_running_the_program.");
147 },
148 };
149
150 const mult = {
151     s: 1000,
152     m: 1000 * 60,
153     h: 1000 * 60 * 60,
154     d: 1000 * 60 * 60 * 24,
155     w: 1000 * 60 * 60 * 24 * 7,
156     y: 1000 * 60 * 60 * 24 * 7 * 365.25,
157 };
158
159 const nameInChart = {
160     insert: "Insert",
161     remove: "Remove",
162     cursor_selection: "Cursor_Change",
163     sel_selection: "Selection_Change",
164     focus: "Focus_tabs",
165     blur: "Unfocus_tabs",
166     visibility: "Change_tabs",
167     pdf_focus: "Focus_PDF_Viewer",
168     pdf_blur: "Unfocus_PDF_Viewer",
169     input_change: "Change_Input",
170     output_change: "Change_Output",
171     save: "Save",
172     submit: "Submit",
173     execute: "Execute",
174 };
175
176 // ##### Main Method #####
177 // #####
178 // #####
179
180 const getRecording = () => {
181     setTitle("Loading...");
182     setLoading(true);
183     disabledInput(true);
184     recording.reset();
185     emptyEditor();
186
187     $.ajax({
188         type: "GET",
189         url: shj.site_url + "recording/download_record/" + rec_path,
190         cache: false,
191         success: (data) => {
192             recording.events = data;
193
194             $("select#rec_selection").empty();
195
196             recording.length = 0;
197
198             Object.keys(data).forEach((c, i, a) => {
199                 // Push to table of saved
200                 $(<tr>

```

```

201         <td>${convTimeToEpoch(c)}</td>
202         <td>${convTimeToEpoch(parseInt(c) + data[c][data[c].length - 1].time)}</td>
203         <td id="sel_${c}" class="sel_recording">
204             Select
205         </td>
206     </tr>').appendTo("tbody#tbody_saved");
207
208     $('#sel_${c}').click(() => {
209         playOrStop(true);
210         setValueTimer(recording.presumIndexDuration[i]);
211         setSelectedSaveTime(c);
212     });
213
214     recording.eventsIndex[c] = i;
215     recording.indexEvents[i] = c;
216     recording.presumIndexDuration[i] = recording.duration;
217     recording.length++;
218
219     recording.duration += data[c][data[c].length - 1].time;
220
221     if (recording.curEvents === -1) {
222         recording.curEvents = c;
223     }
224
225     recording.duration += confPlayer.durationNext;
226 });
227
228 $('#range_player').attr("max", recording.duration);
229 $('#sel_${recording.curEvents)').text("Selected");
230 $('#sel_${recording.curEvents}').addClass("sel_selected");
231 disabledInput(false);
232 setTitle("Ready!");
233 setLoading(false);
234 setStatus("Ready!");
235 setSelectedSaveTime(recording.curEvents);
236 },
237 error: function (error) {
238     console.error(error);
239     setTitle("Error")
240 },
241 });
242 };
243
244 const playOrStop = (stopRec = isRecPlaying) => {
245     if (stopRec) {
246         // if recording is playing -> stop
247         stop();
248         $("#rec_btn").text("Play");
249     } else {
250         // if recording is not played -> played in the time in input.
251         play($("#range_player").val());
252         $("#rec_btn").text("Stop");
253     }
254
255     isRecPlaying = !stopRec;
256 };
257
258 const play = (time = 0) => {
259     emptyEditor();
260
261     let left = 0;
262     let right = recording.length - 1;
263     let eventIndex = 0;
264
265     // lower bound
266     while (left <= right) {
267         let middle = left + Math.floor((right - left) / 2);
268
269         if (recording.presumIndexDuration[middle] > time) {
270             right = middle - 1;
271         } else {
272             eventIndex = middle;
273             left = middle + 1;
274         }
275     }
276
277     eventIndex = recording.indexEvents[eventIndex];
278     setSelectedSaveTime(eventIndex);
279
280     let timeEvent =
281         time - recording.presumIndexDuration[recording.eventsIndex[eventIndex]];
282     let events = recording.events[eventIndex];
283     left = 0;
284     right = events.length - 1;
285     let eventsIndex = events.length - 1;
286
287     // upper bound
288     while (left <= right) {
289         let middle = left + Math.floor((right - left) / 2);
290
291         if (events[middle].time < timeEvent) {
292             left = middle + 1;
293         } else {
294             eventsIndex = middle;
295             right = middle - 1;
296         }
297     }
298
299     while (

```

```

300        eventsIndex > 0 &&
301        events[eventsIndex].time - events[eventsIndex - 1].time <=
302            confPlayer.mergeDuration
303    ) {
304        eventsIndex--;
305    }
306
307    handlerForLast = {};
308
309    // from index 0 of selected save time to eventsIndex, run all insert and remove
310    for (let index = 0; index < eventsIndex; index++) {
311        const event = events[index];
312        if (handlersIncludeInGoState[event.event]) {
313            handlers[event.event](event.args);
314        } else if (events[eventsIndex] - event.time <= confPlayer.mergeDuration) {
315            handlerForLast[event.event] = () => handlers[event.event](event.args);
316        }
317    }
318
319    // Run all handler that set to false the last handler that the event is
320    Object.values(handlerForLast).forEach((f) => {
321        f();
322    });
323
324    startTimer();
325
326    // Start after time to eventsIndex
327    if (events[eventsIndex].time < timeEvent)
328        funcTimeoutRecording = setTimeout(() => {
329            startRecording(eventsIndex);
330            , timeEvent - events[eventsIndex].time);
331    else
332        funcTimeoutRecording = setTimeout(() => {
333            startRecording(eventsIndex);
334            , events[eventsIndex].time - timeEvent);
335    };
336
337    // Methods for plays
338    const startRecording = (index) => {
339        let events = recording.events[recording.curEvents];
340        if (index >= events.length) {
341            if (playNextRecording())
342                setStatus("Playing_Next_Session");
343                setTitle("Playing_Next_Session");
344                return;
345            } else {
346                funcTimeoutRecording = setTimeout(() => {
347                    setStatus("Finish...");
348                    setTitle("Finish...");
349                    playOrStop(true);
350                    , confPlayer.durationNext);
351                    return;
352                }
353            }
354
355            let event = events[index];
356            let timeDiff =
357                (index + 1 < events.length
358                    ? events[index + 1].time
359                    : events[index].time) - event.time;
360
361            handlers[event.event](event.args);
362
363            while (index + 1 < events.length && timeDiff <= confPlayer.mergeDuration) {
364                index++;
365                event = events[index];
366                timeDiff = events[index + 1].time - event.time;
367                handlers[event.event](event.args);
368            }
369
370            funcTimeoutRecording = setTimeout(() => {
371                startRecording(index + 1);
372                , timeDiff);
373        };
374
375        const startTimer = () => {
376            let offset = 0;
377
378            function delta() {
379                var now = Date.now(),
380                    d = now - offset;
381
382                    offset = now;
383                    return d;
384            }
385
386            const update = () => {
387                let val = parseInt($('#range_player').val());
388                val += delta();
389                $('#range_player').val(val);
390                updateTimerRange();
391            };
392
393            if (!funcIntervalTimer) {
394                offset = Date.now();
395                funcIntervalTimer = setInterval(update, confTimer.delay);
396            }
397        };
398    }

```

```

399 const playNextRecording = () => {
400   if (recording.eventsIndex[recording.curEvents] < recording.length - 1) {
401     setSelectedSaveTime(
402       recording.indexEvents[recording.eventsIndex[recording.curEvents] + 1],
403       false
404     );
405   }
406   funcTimeoutRecording = setTimeout(() => {
407     emptyEditor();
408   }
409   funcTimeoutRecording = setTimeout(() => {
410     startRecording();
411   }, recording.events[recording.curEvents][0].time);
412   confPlayer.durationNext);
413   return true;
414 }
415 funcTimeoutRecording = setTimeout(() => {}, confPlayer.durationNext);
416 return false;
417 };
418 }
419 const stop = () => {
420   stopTimer();
421   stopRecording();
422 };
423
424 // Methods for stop
425 const stopTimer = () => {
426   clearTimeout(funcIntervalTimer);
427   funcIntervalTimer = null;
428 };
429
430 const stopRecording = () => {
431   clearTimeout(funcTimeoutRecording);
432   funcTimeoutRecording = null;
433 };
434
435 // #####
436 // ##### Chart Method #####
437 // #####
438
439 const setUpChart = (
440   index = recording.curEvents,
441   type = confChart.type ? confChart.type : "bar",
442   divider = confChart.divider ? confChart.divider : 1,
443   time = confChart.time ? confChart.time : "s",
444   stack = confChart.stack != undefined ? confChart.stack : true,
445   step = confChart.step != undefined ? confChart.step : false,
446   fill = confChart.fill != undefined ? confChart.fill : false
447 ) => {
448   if (curChart != null) curChart.destroy();
449
450   let times = calcTimeForChart(recording.events[index], divider, time);
451   let data = formatToChartData(recording.events[index], times);
452
453   const ctx = document.getElementById("recording_chart");
454
455   const dataChart = {
456     labels: times.map((c) => c.time),
457     datasets: Object.entries(data).map(([k, v]) => {
458       return {
459         label: nameInChart[k],
460         data: v,
461         fill: fill,
462         stepped: step,
463       };
464     }),
465   };
466
467   const configChart = {
468     type: type,
469     options: {
470       scales: {
471         x: {
472           stacked: stack,
473         },
474         y: {
475           stacked: stack,
476         },
477       },
478     },
479   };
480
481   const chart = new Chart(ctx, {
482     ...configChart,
483     data: dataChart,
484   });
485
486   curChart = chart;
487   return chart;
488 };
489
490 const formatToChartData = (arrEvent, arrTimes) => {
491   let res = {};
492
493   let idxTimes = 0;
494
495   arrEvent.forEach((e) => {
496     while (e.time > arrTimes[idxTimes].ms) {
497

```

```

498         idxTimes++;
499     }
500     if (!res[e.event]) {
501         res[e.event] = {};
502     }
503     if (!res[e.event][arrTimes[idxTimes].time]) {
504         res[e.event][arrTimes[idxTimes].time] = 0;
505     }
506     res[e.event][arrTimes[idxTimes].time]++;
507 }
508 });
509
510     return res;
511 };
512
513 const calcTimeForChart = (arrEvent, divider, time) => {
514     let res = [];
515
516     let dev = calcTimeToMilliSec(divider, time);
517     let max = arrEvent[arrEvent.length - 1].time;
518     let i = 1;
519
520     for (; dev * i < max; i++) {
521         res.push({
522             time: Math.floor(divider * i * 100) / 100 + time,
523             ms: dev * i,
524         });
525     }
526
527     res.push({
528         time: Math.floor(divider * i * 100) / 100 + time,
529         ms: dev * i,
530     });
531
532     return res;
533 };
534
535 );
536
537 const calcTimeToMilliSec = (divider, time) => {
538     if (mult[time]) {
539         return divider * mult[time];
540     }
541
542     return divider;
543 };
544
545 // ##### Set Up Config Chart #####
546 // #####
547 // #####
548
549 const setUpTimeDividerSelector = (selectID) => {
550     Object.keys(mult).forEach((c) => {
551         $(
552             `` +
555             `').appendTo(`#${selectID}`);
556     });
557
558 // #####
559 // ##### Listener Config Chart #####
560 // #####
561 // #####
562
563 $("#config_chart_type").on("change", (e) => {
564     let val = $("#config_chart_type").val();
565     confChart.type = val;
566     setUpChart();
567 });
568
569 $("#config_chart_stack").on("click", (e) => {
570     let val = $("#config_chart_stack").prop("checked");
571     confChart.stack = val;
572     setUpChart();
573 });
574
575 $("#config_chart_divider").on("input", (e) => {
576     let val = $("#config_chart_divider").val();
577     if (val > 0) {
578         confChart.divider = val;
579         setUpChart();
580     }
581 });
582
583 $("#config_chart_time").on("change", (e) => {
584     let val = $("#config_chart_time").val();
585     confChart.time = val;
586     setUpChart();
587 });
588
589 // #####
590 // ##### Listener #####
591 // #####
592
593 $("#rec_btn").click(() => {
594     playOrStop();
595 });
596

```

```

597  $("#" + range_player).on("mousemove", function () {
598      updateTimerRange();
599  });
600
601  $("#" + range_player).on("change", function () {
602      let isPlaying = isRecPlaying;
603      if (isPlaying) playOrStop(true);
604      updateTimerRange();
605      if (isPlaying) playOrStop(false);
606  });
607
608 // ##### Misc Method #####
609 // #####
610 // #####
611
612  function setSelection(editor, data) {
613      let x = data;
614      let start = { row: x[0], column: x[1] };
615      let end = x.length == 2 ? start : { row: x[2], column: x[3] };
616      let isBackwards = Range.comparePoints(start, end) > 0;
617      editor.selection.fromJSON(
618          isBackwards
619          ?
620              start: end,
621              end: start,
622              isBackwards: true,
623          :
624              start: start,
625              end: end,
626              isBackwards: true,
627          );
628      );
629  }
630
631
632  const emptyEditor = () => {
633      editor.session.setValue("", -1);
634  };
635
636  const disabledInput = (bool) => {
637      $("#rec_play").prop("disabled", bool);
638      $("#rec_stop").prop("disabled", bool);
639  };
640
641 // Convert Timestamp to Epoch
642  const convTimeToEpoch = (timestamp) => {
643      var date = new Date(parseInt(timestamp));
644
645      var year = date.getFullYear();
646      var month = date.getMonth() + 1;
647      var day = date.getDate();
648      var hours = date.getHours();
649      var minutes = date.getMinutes();
650      var seconds = date.getSeconds();
651
652      return (
653          year +
654          "_" +
655          month +
656          "_" +
657          day +
658          "_" +
659          (hours < 10 ? "0" : "") +
660          hours +
661          ":" +
662          (minutes < 10 ? "0" : "") +
663          minutes +
664          ":" +
665          (seconds < 10 ? "0" : "") +
666          seconds
667      );
668  };
669
670  const convTimeToHMMSS = (ms, showZero = true, showMS = false) => {
671      console.log(ms);
672
673      let seconds = ms / 1000;
674      const hours = parseInt(seconds / 3600).toFixed(0);
675      seconds = seconds % 3600;
676      const minutes = parseInt(seconds / 60).toFixed(0);
677      seconds = (seconds % 60).toFixed(0);
678      const sec = parseInt(seconds / 60).toFixed(0);
679      ms = (seconds % 60).toFixed(0);
680
681      console.log(ms, hours, minutes, sec);
682
683      return (
684          (hours > 0 ? (hours < 10 ? "0" : "") + hours + ":" : "") +
685          (showZero || minutes > 0
686              ? (minutes < 10 ? "0" : "") + minutes + ":" +
687                  :
688              (seconds < 10 ? "0" : "") +
689              seconds +
690              (showMS ? ms : ""))
691      );
692  };
693
694  const setTitle = (title) => {
695      $("#status_rec").text(title);

```

```

696  };
697
698 const setSelectedSaveTime = (time, stop = true) => {
699   if (stop) playOrStop(true);
700   $('#sel_${recording.curEvents}').text("Select");
701   $('#sel_${recording.curEvents}').removeClass("sel_selected");
702
703   recording.curEvents = time;
704   $('#sel_${time}').text("Selected");
705   $('#sel_${time}').addClass("sel_selected");
706
707   let duration =
708     recording.events[time][recording.events[time].length - 1].time;
709
710   let divider = (duration / confChart.autoDivider / 1000).toFixed(2);
711   let times = "s";
712
713   $('#config_chart_divider').val(divider);
714   $('#config_chart_time').val(times);
715
716   setUpChart(time, confChart.type, divider, times);
717 }
718
719 const setStatus = (text = "...") => {
720   const status = $("#status_wrapper");
721
722   status.show();
723   status.text(text);
724
725   setTimeout(() => {
726     status.hide();
727   }, 1000);
728 }
729
730 const setLoading = (bool) => {
731   const mainEle = $("#recording_wrap");
732
733   if (bool) {
734     mainEle.hide();
735   } else {
736     mainEle.show();
737   }
738 }
739
740 const setValueTimer = (value) => {
741   $('#range_player').val(value);
742   updateTimerRange();
743 }
744
745 const updateTimerRange = (
746   idRange = "#range_player",
747   idSpan = '#timer_player'
748 ) => {
749   const val = $(idRange).val() / recording.duration * 100;
750   const ms = $(idRange).val();
751
752   $(idSpan).text(convTimeToHHMMSS(ms));
753
754   $(idRange).css(
755     "background",
756     "linear-gradient(to_right, #cc181e_0%, #cc181e_ +
757       val +
758       "%, #444_ +
759       val +
760       "%, #444_100%)"
761   );
762 }
763
764 // ##### Runner #####
765 // ##### Runner #####
766 // #####
767
768 getRecording();
769
770 // Runner Config
771 setUpTimeDividerSelector("config_chart_time");
772 });

```

Kode A.10: shj_submit.js

```

1 diff --git a/assets/js/shj_submit.js b/assets/js/shj_submit.js
2 index 7a70670..babaf43 100644
3 --- a/assets/js/shj_submit.js
4 +++ b/assets/js/shj_submit.js
5 @@ -5,181 +5,747 @@
6   *      Javascript codes for "Submit" page
7   */
8
9 -$ (document).ready(function(){
10 -   var editor = ace.edit("code_editor");
11 -   editor.setOptions({
12 -     theme: "ace/theme/monokai",
13 -     fontSize: "11pt"
14 -   });
15 -
16 -   function disableEditor(bool) {
17 -     $("#editor_save").prop("disabled", bool);
18 -     $("#editor_execute").prop("disabled", bool);

```

```

19| -     $("#editor_submit").prop("disabled", bool);
20| -     $("#editor_input").prop("disabled", bool);
21| -     editor.setReadOnly(bool);
22| - }
23|
24| - function loadCode(problem_id){
25| -     $("#editor_input").val("");
26| -     $("#editor_output").val("");
27|
28| -     if(problem_id == 0){
29| -         disableEditor(true);
30| -         editor.setValue("");
31| -         $("#ajax_status").html("Select problem and language");
32| -     }
33| -     else{
34| -         disableEditor(true);
35| -         $.ajax({
36| -             url: shj.site_url + 'submit/load/' + problem_id,
37| -             cache: false,
38| -             success: function (data){
39| -                 data = JSON.parse(data);
40| -                 editor.setValue(data.content);
41| -                 $("#ajax_status").html(data.message);
42| -             },
43| -             error: function (error){
44| -                 console.error(error);
45| -             },
46| -         });
47| -     }
48| - }
49|
50| - $("select#problems").change(function(){
51| -     var v = $(this).val();
52| -     loadCode(v);
53| -     $('select#languages').empty();
54| -     $(<option value="0" selected="selected">-- Select Language --</option>).appendTo('select#languages');
55| -     for (var i=0;i<shj.p[v].length;i++)
56| -         $(<option value="'+shj.p[v][i]+'>'+'shj.p[v][i]'+</option>).appendTo('select#languages');
57| - });
58|
59| - $("select#languages").change(function(){
60| -     if(this.value.toLowerCase().includes("java")){
61| -         editor.session.setMode("ace/mode/java");
62| -         disableEditor(false);
63| -     }
64| -     else if(this.value.toLowerCase().includes("python")){
65| -         editor.session.setMode("ace/mode/python");
66| -         disableEditor(false);
67| -     }
68| -     else if(this.value.toLowerCase().includes("c")){
69| -         editor.session.setMode("ace/mode/c_cpp");
70| -         disableEditor(false);
71| -     }
72| -     else if(this.value.toLowerCase().includes("txt")){
73| -         editor.session.setMode("ace/mode/plain_text");
74| -         disableEditor(false);
75| -         $("#editor_execute").prop("disabled", true);
76| -         $("#editor_input").prop("disabled", true);
77| -     }
78| -     else{
79| -         editor.session.setMode("ace/mode/plain_text");
80| -         disableEditor(true);
81| -     }
82| - });
83|
84| - $("#editor_save").click(function(){
85| -     disableEditor(true);
86| -     $.ajax({
87| -         type: "POST",
88| -         url: shj.site_url + 'submit/save',
89| -         data: {
90| -             shj_csrf_token: shj.csrf_token,
91| -             code_editor: editor.getValue(),
92| -             problem_id: $("select#problems").val(),
93| -             language: $("select#languages").val(),
94| -         },
95| -         cache: false,
96| -         success: function(data){
97| -             data = JSON.parse(data);
98| -             $("#ajax_status").html(data.message);
99| -             disableEditor(false);
100| -         },
101| -         error: function (error){
102| -             console.error(error);
103| -             disableEditor(false);
104| -         },
105| -     });
106| - });
107|
108| - $("#editor_submit").click(function(){
109| -     disableEditor(true);
110| -     $.ajax({
111| -         type: "POST",
112| -         url: shj.site_url + 'submit/save/submit',
113| -         data: {
114| -             shj_csrf_token: shj.csrf_token,
115| -             code_editor: editor.getValue(),
116| -             problem_id: $("select#problems").val(),
117| -             language: $("select#languages").val(),

```

```

118|     },
119|     cache: false,
120|     success: function(data){
121|         data = JSON.parse(data);
122|         $("#ajax_status").html(data.message);
123|         disableEditor(false);
124|         if(data.status){
125|             window.location.href = shj.site_url + 'submissions/all';
126|         }
127|     },
128|     error: function (error){
129|         console.error(error);
130|         disableEditor(false);
131|     },
132| });
133| });
134|
135| $("#editor_execute").click(function(){
136|     disableEditor(true);
137|     $.ajax({
138|         type: "POST",
139|         url: shj.site_url + 'submit/save/execute',
140|         data: {
141|             shj_csrf_token: shj.csrf_token,
142|             code_editor: editor.getValue(),
143|             editor_input: $('#textareagriditor_input').val(),
144|             problem_id: $('#select#problems').val(),
145|             language: $('#select#languages').val(),
146|         },
147|         cache: false,
148|         success: function(data){
149|             data = JSON.parse(data);
150|             $("#ajax_status").html(data.message);
151|             if(data.status){
152|                 (function update() {
153|                     $.ajax({
154|                         url: shj.site_url + 'submit/get_output/' + $('#select#problems').val(),
155|                         cache: false,
156|                         success: function (data){
157|                             data = JSON.parse(data);
158|                             $('#textareagriditor_output').val(data.content);
159|                             if(!data.status){
160|                                 setTimeout(update, 1000);
161|                             }
162|                             else{
163|                                 $("#ajax_status").html("Completed");
164|                                 disableEditor(false);
165|                             }
166|                         },
167|                         error: function (error){
168|                             console.error(error);
169|                             disableEditor(false);
170|                         },
171|                     })
172|                 })();
173|             }
174|             else{
175|                 disableEditor(false);
176|             }
177|         },
178|         error: function (error){
179|             console.error(error);
180|             disableEditor(false);
181|         },
182|     });
183| });
184|
185| loadCode($('#select#problems').val());
186| });
187\ No newline at end of file
188+$(document).ready(function () {
189+    var editor = ace.edit("code_editor");
190+
191+    editor.setOptions({
192+        theme: "ace/theme/monokai",
193+        fontSize: "11pt",
194+        enableLiveAutocompletion: true,
195+        enableBasicAutocompletion: true,
196+        enableSnippets: true,
197+    });
198+
199+    function disableEditor(bool) {
200+        $("#editor_save").prop("disabled", bool);
201+        $("#editor_execute").prop("disabled", bool);
202+        $("#editor_submit").prop("disabled", bool);
203+        $("#editor_input").prop("disabled", bool);
204+        editor.setReadOnly(bool);
205+    }
206+
207+    function loadCode(problem_id) {
208+        $("#editor_input").val("");
209+        $("#editor_output").val("");
210+
211+        if (problem_id == 0) {
212+            disableEditor(true);
213+            editor.setValue("");
214+            $("#ajax_status").html("Select problem and language");
215+        } else {
216+            disableEditor(true);

```

```

217| +         $.ajax({
218| +             url: shj.site_url + "submit/load/" + problem_id,
219| +             cache: false,
220| +             success: function (data) {
221| +                 data = JSON.parse(data);
222| +                 editor.setValue(data.content);
223| +                 $("#ajax_status").html(data.message);
224| +             },
225| +             error: function (error) {
226| +                 console.error(error);
227| +             },
228| +         });
229| +     }
230| + }
231| +
232| + const canItBeDisabled = () => {
233| +     if (!isRetrieved) {
234| +         isRetrieved = true;
235| +     } else {
236| +         disableEditor(false);
237| +     }
238| + }
239| +
240| + const loadBeforeRec = async (problem_id) => {
241| +
242| +     const funcLoad = async (data) => {
243| +         data = JSON.parse(data);
244| +
245| +         if (data.content.trim() === "") {
246| +             canItBeDisabled();
247| +             return;
248| +         }
249| +
250| +         befRecording = await JSON.parse(data.content);
251| +         canItBeDisabled();
252| +     }
253| +
254| +     if (problem_id == 0) {
255| +         disableEditor(true);
256| +         editor.setValue("");
257| +     } else {
258| +         disableEditor(true);
259| +         await $.ajax({
260| +             url: shj.site_url + "submit/load_rec/" + problem_id,
261| +             cache: false,
262| +             success: function (data) {
263| +                 funcLoad(data);
264| +             },
265| +             error: function (error) {
266| +                 console.error(error);
267| +                 canItBeDisabled();
268| +             },
269| +         });
270| +     }
271| + }
272| +
273| + $("select#problems").change(function () {
274| +     var v = $(this).val();
275| +     loadCode(v);
276| +     recordStart();
277| +     loadBeforeRec(v);
278| +     $("select#languages").empty();
279| +     $(
280| +         '<option value="0" selected="selected">-- Select Language --</option>'
281| +     ).appendTo("select#languages");
282| +     for (var i = 0; i < shj.p[v].length; i++) {
283| +         $(
284| +             '<option value="' + shj.p[v][i] + '">' + shj.p[v][i] + "</option>"
285| +         ).appendTo("select#languages");
286| +
287| +     });
288| +
289| + $("select#languages").change(function () {
290| +     if (this.value.toLowerCase().includes("java")) {
291| +         editor.session.setMode("ace/mode/java");
292| +     } else if (this.value.toLowerCase().includes("python")) {
293| +         editor.session.setMode("ace/mode/python");
294| +     } else if (this.value.toLowerCase().includes("c")) {
295| +         editor.session.setMode("ace/mode/c_cpp");
296| +     } else if (this.value.toLowerCase().includes("txt")) {
297| +         editor.session.setMode("ace/mode/plain_text");
298| +         $("#editor_execute").prop("disabled", true);
299| +         $("#editor_input").prop("disabled", true);
300| +     } else {
301| +         editor.session.setMode("ace/mode/plain_text");
302| +     }
303| +
304| +     canItBeDisabled();
305| + });
306| +
307| + $("#editor_save").click(function () {
308| +     disableEditor(true);
309| +     handlers.save();
310| +
311| +     $.ajax({
312| +         type: "POST",
313| +         url: shj.site_url + "submit/save",
314| +         data: {
315| +             shj_csrf_token: shj.csrf_token,

```

```

316| +         code_editor: editor.getValue(),
317| +         problem_id: $("select#problems").val(),
318| +         language: $("select#languages").val(),
319| +         rec_data: JSON.stringify({
320| +             ...befRecording,
321| +             [recording.startTime]: recording.events
322| +         }),
323| +     },
324| +     cache: false,
325| +     success: function (data) {
326| +         data = JSON.parse(data);
327| +
328| +         $("#ajax_status").html(data.message);
329| +         disableEditor(false);
330| +
331| +     },
332| +     error: function (error) {
333| +         console.error(error);
334| +         disableEditor(false);
335| +     },
336| + });
337| +
338| + $("#editor_submit").click(function () {
339| +     disableEditor(true);
340| +     handlers.submit();
341| +
342| +     $.ajax({
343| +         type: "POST",
344| +         url: shj.site_url + "submit/save/submit",
345| +         data: {
346| +             shj.csrf_token: shj.csrf_token,
347| +             code_editor: editor.getValue(),
348| +             problem_id: $("select#problems").val(),
349| +             language: $("select#languages").val(),
350| +             rec_data: JSON.stringify({
351| +                 ...befRecording,
352| +                 [recording.startTime]: recording.events
353| +             }),
354| +
355| +         },
356| +         cache: false,
357| +         success: function (data) {
358| +             data = JSON.parse(data);
359| +             $("#ajax_status").html(data.message);
360| +             disableEditor(false);
361| +             if (data.status) {
362| +                 window.location.href = shj.site_url + "submissions/all";
363| +             }
364| +
365| +         },
366| +         error: function (error) {
367| +             console.error(error);
368| +             disableEditor(false);
369| +
370| +         },
371| +         recordStop();
372| +
373| +     });
374| +     $("#editor_execute").click(function () {
375| +         disableEditor(true);
376| +         handlers.execute();
377| +
378| +         $.ajax({
379| +             type: "POST",
380| +             url: shj.site_url + "submit/save/execute",
381| +             data: {
382| +                 shj.csrf_token: shj.csrf_token,
383| +                 code_editor: editor.getValue(),
384| +                 editor_input: $("textarea#editor_input").val(),
385| +                 problem_id: $("select#problems").val(),
386| +                 language: $("select#languages").val(),
387| +                 rec_data: JSON.stringify({
388| +                     ...befRecording,
389| +                     [recording.startTime]: recording.events
390| +                 }),
391| +
392| +             },
393| +             cache: false,
394| +             success: function (data) {
395| +                 data = JSON.parse(data);
396| +                 $("#ajax_status").html(data.message);
397| +                 if (data.status) {
398| +                     (function update() {
399| +                         $.ajax({
400| +                             url:
401| +                             shj.site_url +
402| +                             "submit/get_output/" +
403| +                             $("select#problems").val(),
404| +                             cache: false,
405| +                             success: function (data) {
406| +                                 data = JSON.parse(data);
407| +                                 $("textarea#editor_output").val(data.content);
408| +                                 // ---
409| +                                 $("textarea#editor_output").trigger(
410| +                                     "output_change",
411| +                                     data.content
412| +                                 );
413| +                                 // ---
414| +                                 if (!data.status) {
415| +                                     setTimeout(update, 1000);
416| +                                 } else {
417| +                         }
418| +                     });
419| +                 });
420| +             }
421| +         });
422| +
423| +     });
424| +
425| +     $("#editor_stop").click(function () {
426| +         disableEditor(true);
427| +         handlers.stop();
428| +
429| +     });
430| +
431| +     $("#editor_start").click(function () {
432| +         disableEditor(false);
433| +         handlers.start();
434| +
435| +     });
436| +
437| +     $("#editor_update").click(function () {
438| +         disableEditor(true);
439| +         handlers.update();
440| +
441| +     });
442| +
443| +     $("#editor_clear").click(function () {
444| +         disableEditor(true);
445| +         handlers.clear();
446| +
447| +     });
448| +
449| +     $("#editor_execute").click(function () {
450| +         disableEditor(true);
451| +         handlers.execute();
452| +
453| +     });
454| +
455| +     $("#editor_stop").click(function () {
456| +         disableEditor(true);
457| +         handlers.stop();
458| +
459| +     });
460| +
461| +     $("#editor_start").click(function () {
462| +         disableEditor(false);
463| +         handlers.start();
464| +
465| +     });
466| +
467| +     $("#editor_update").click(function () {
468| +         disableEditor(true);
469| +         handlers.update();
470| +
471| +     });
472| +
473| +     $("#editor_clear").click(function () {
474| +         disableEditor(true);
475| +         handlers.clear();
476| +
477| +     });
478| +
479| +     $("#editor_submit").click(function () {
480| +         disableEditor(true);
481| +         handlers.submit();
482| +
483| +     });
484| +
485| +     $("#editor_execute").click(function () {
486| +         disableEditor(true);
487| +         handlers.execute();
488| +
489| +     });
490| +
491| +     $("#editor_stop").click(function () {
492| +         disableEditor(true);
493| +         handlers.stop();
494| +
495| +     });
496| +
497| +     $("#editor_start").click(function () {
498| +         disableEditor(false);
499| +         handlers.start();
500| +
501| +     });
502| +
503| +     $("#editor_update").click(function () {
504| +         disableEditor(true);
505| +         handlers.update();
506| +
507| +     });
508| +
509| +     $("#editor_clear").click(function () {
510| +         disableEditor(true);
511| +         handlers.clear();
512| +
513| +     });
514| +
515| +     $("#editor_submit").click(function () {
516| +         disableEditor(true);
517| +         handlers.submit();
518| +
519| +     });
520| +
521| +     $("#editor_execute").click(function () {
522| +         disableEditor(true);
523| +         handlers.execute();
524| +
525| +     });
526| +
527| +     $("#editor_stop").click(function () {
528| +         disableEditor(true);
529| +         handlers.stop();
530| +
531| +     });
532| +
533| +     $("#editor_start").click(function () {
534| +         disableEditor(false);
535| +         handlers.start();
536| +
537| +     });
538| +
539| +     $("#editor_update").click(function () {
540| +         disableEditor(true);
541| +         handlers.update();
542| +
543| +     });
544| +
545| +     $("#editor_clear").click(function () {
546| +         disableEditor(true);
547| +         handlers.clear();
548| +
549| +     });
550| +
551| +     $("#editor_submit").click(function () {
552| +         disableEditor(true);
553| +         handlers.submit();
554| +
555| +     });
556| +
557| +     $("#editor_execute").click(function () {
558| +         disableEditor(true);
559| +         handlers.execute();
560| +
561| +     });
562| +
563| +     $("#editor_stop").click(function () {
564| +         disableEditor(true);
565| +         handlers.stop();
566| +
567| +     });
568| +
569| +     $("#editor_start").click(function () {
570| +         disableEditor(false);
571| +         handlers.start();
572| +
573| +     });
574| +
575| +     $("#editor_update").click(function () {
576| +         disableEditor(true);
577| +         handlers.update();
578| +
579| +     });
580| +
581| +     $("#editor_clear").click(function () {
582| +         disableEditor(true);
583| +         handlers.clear();
584| +
585| +     });
586| +
587| +     $("#editor_submit").click(function () {
588| +         disableEditor(true);
589| +         handlers.submit();
590| +
591| +     });
592| +
593| +     $("#editor_execute").click(function () {
594| +         disableEditor(true);
595| +         handlers.execute();
596| +
597| +     });
598| +
599| +     $("#editor_stop").click(function () {
600| +         disableEditor(true);
601| +         handlers.stop();
602| +
603| +     });
604| +
605| +     $("#editor_start").click(function () {
606| +         disableEditor(false);
607| +         handlers.start();
608| +
609| +     });
610| +
611| +     $("#editor_update").click(function () {
612| +         disableEditor(true);
613| +         handlers.update();
614| +
615| +     });
616| +
617| +     $("#editor_clear").click(function () {
618| +         disableEditor(true);
619| +         handlers.clear();
620| +
621| +     });
622| +
623| +     $("#editor_submit").click(function () {
624| +         disableEditor(true);
625| +         handlers.submit();
626| +
627| +     });
628| +
629| +     $("#editor_execute").click(function () {
630| +         disableEditor(true);
631| +         handlers.execute();
632| +
633| +     });
634| +
635| +     $("#editor_stop").click(function () {
636| +         disableEditor(true);
637| +         handlers.stop();
638| +
639| +     });
640| +
641| +     $("#editor_start").click(function () {
642| +         disableEditor(false);
643| +         handlers.start();
644| +
645| +     });
646| +
647| +     $("#editor_update").click(function () {
648| +         disableEditor(true);
649| +         handlers.update();
650| +
651| +     });
652| +
653| +     $("#editor_clear").click(function () {
654| +         disableEditor(true);
655| +         handlers.clear();
656| +
657| +     });
658| +
659| +     $("#editor_submit").click(function () {
660| +         disableEditor(true);
661| +         handlers.submit();
662| +
663| +     });
664| +
665| +     $("#editor_execute").click(function () {
666| +         disableEditor(true);
667| +         handlers.execute();
668| +
669| +     });
670| +
671| +     $("#editor_stop").click(function () {
672| +         disableEditor(true);
673| +         handlers.stop();
674| +
675| +     });
676| +
677| +     $("#editor_start").click(function () {
678| +         disableEditor(false);
679| +         handlers.start();
680| +
681| +     });
682| +
683| +     $("#editor_update").click(function () {
684| +         disableEditor(true);
685| +         handlers.update();
686| +
687| +     });
688| +
689| +     $("#editor_clear").click(function () {
690| +         disableEditor(true);
691| +         handlers.clear();
692| +
693| +     });
694| +
695| +     $("#editor_submit").click(function () {
696| +         disableEditor(true);
697| +         handlers.submit();
698| +
699| +     });
700| +
701| +     $("#editor_execute").click(function () {
702| +         disableEditor(true);
703| +         handlers.execute();
704| +
705| +     });
706| +
707| +     $("#editor_stop").click(function () {
708| +         disableEditor(true);
709| +         handlers.stop();
710| +
711| +     });
712| +
713| +     $("#editor_start").click(function () {
714| +         disableEditor(false);
715| +         handlers.start();
716| +
717| +     });
718| +
719| +     $("#editor_update").click(function () {
720| +         disableEditor(true);
721| +         handlers.update();
722| +
723| +     });
724| +
725| +     $("#editor_clear").click(function () {
726| +         disableEditor(true);
727| +         handlers.clear();
728| +
729| +     });
730| +
731| +     $("#editor_submit").click(function () {
732| +         disableEditor(true);
733| +         handlers.submit();
734| +
735| +     });
736| +
737| +     $("#editor_execute").click(function () {
738| +         disableEditor(true);
739| +         handlers.execute();
740| +
741| +     });
742| +
743| +     $("#editor_stop").click(function () {
744| +         disableEditor(true);
745| +         handlers.stop();
746| +
747| +     });
748| +
749| +     $("#editor_start").click(function () {
750| +         disableEditor(false);
751| +         handlers.start();
752| +
753| +     });
754| +
755| +     $("#editor_update").click(function () {
756| +         disableEditor(true);
757| +         handlers.update();
758| +
759| +     });
760| +
761| +     $("#editor_clear").click(function () {
762| +         disableEditor(true);
763| +         handlers.clear();
764| +
765| +     });
766| +
767| +     $("#editor_submit").click(function () {
768| +         disableEditor(true);
769| +         handlers.submit();
770| +
771| +     });
772| +
773| +     $("#editor_execute").click(function () {
774| +         disableEditor(true);
775| +         handlers.execute();
776| +
777| +     });
778| +
779| +     $("#editor_stop").click(function () {
780| +         disableEditor(true);
781| +         handlers.stop();
782| +
783| +     });
784| +
785| +     $("#editor_start").click(function () {
786| +         disableEditor(false);
787| +         handlers.start();
788| +
789| +     });
790| +
791| +     $("#editor_update").click(function () {
792| +         disableEditor(true);
793| +         handlers.update();
794| +
795| +     });
796| +
797| +     $("#editor_clear").click(function () {
798| +         disableEditor(true);
799| +         handlers.clear();
800| +
801| +     });
802| +
803| +     $("#editor_submit").click(function () {
804| +         disableEditor(true);
805| +         handlers.submit();
806| +
807| +     });
808| +
809| +     $("#editor_execute").click(function () {
810| +         disableEditor(true);
811| +         handlers.execute();
812| +
813| +     });
814| +
815| +     $("#editor_stop").click(function () {
816| +         disableEditor(true);
817| +         handlers.stop();
818| +
819| +     });
820| +
821| +     $("#editor_start").click(function () {
822| +         disableEditor(false);
823| +         handlers.start();
824| +
825| +     });
826| +
827| +     $("#editor_update").click(function () {
828| +         disableEditor(true);
829| +         handlers.update();
830| +
831| +     });
832| +
833| +     $("#editor_clear").click(function () {
834| +         disableEditor(true);
835| +         handlers.clear();
836| +
837| +     });
838| +
839| +     $("#editor_submit").click(function () {
840| +         disableEditor(true);
841| +         handlers.submit();
842| +
843| +     });
844| +
845| +     $("#editor_execute").click(function () {
846| +         disableEditor(true);
847| +         handlers.execute();
848| +
849| +     });
850| +
851| +     $("#editor_stop").click(function () {
852| +         disableEditor(true);
853| +         handlers.stop();
854| +
855| +     });
856| +
857| +     $("#editor_start").click(function () {
858| +         disableEditor(false);
859| +         handlers.start();
860| +
861| +     });
862| +
863| +     $("#editor_update").click(function () {
864| +         disableEditor(true);
865| +         handlers.update();
866| +
867| +     });
868| +
869| +     $("#editor_clear").click(function () {
870| +         disableEditor(true);
871| +         handlers.clear();
872| +
873| +     });
874| +
875| +     $("#editor_submit").click(function () {
876| +         disableEditor(true);
877| +         handlers.submit();
878| +
879| +     });
880| +
881| +     $("#editor_execute").click(function () {
882| +         disableEditor(true);
883| +         handlers.execute();
884| +
885| +     });
886| +
887| +     $("#editor_stop").click(function () {
888| +         disableEditor(true);
889| +         handlers.stop();
890| +
891| +     });
892| +
893| +     $("#editor_start").click(function () {
894| +         disableEditor(false);
895| +         handlers.start();
896| +
897| +     });
898| +
899| +     $("#editor_update").click(function () {
900| +         disableEditor(true);
901| +         handlers.update();
902| +
903| +     });
904| +
905| +     $("#editor_clear").click(function () {
906| +         disableEditor(true);
907| +         handlers.clear();
908| +
909| +     });
910| +
911| +     $("#editor_submit").click(function () {
912| +         disableEditor(true);
913| +         handlers.submit();
914| +
915| +     });
916| +
917| +     $("#editor_execute").click(function () {
918| +         disableEditor(true);
919| +         handlers.execute();
920| +
921| +     });
922| +
923| +     $("#editor_stop").click(function () {
924| +         disableEditor(true);
925| +         handlers.stop();
926| +
927| +     });
928| +
929| +     $("#editor_start").click(function () {
930| +         disableEditor(false);
931| +         handlers.start();
932| +
933| +     });
934| +
935| +     $("#editor_update").click(function () {
936| +         disableEditor(true);
937| +         handlers.update();
938| +
939| +     });
940| +
941| +     $("#editor_clear").click(function () {
942| +         disableEditor(true);
943| +         handlers.clear();
944| +
945| +     });
946| +
947| +     $("#editor_submit").click(function () {
948| +         disableEditor(true);
949| +         handlers.submit();
950| +
951| +     });
952| +
953| +     $("#editor_execute").click(function () {
954| +         disableEditor(true);
955| +         handlers.execute();
956| +
957| +     });
958| +
959| +     $("#editor_stop").click(function () {
960| +         disableEditor(true);
961| +         handlers.stop();
962| +
963| +     });
964| +
965| +     $("#editor_start").click(function () {
966| +         disableEditor(false);
967| +         handlers.start();
968| +
969| +     });
970| +
971| +     $("#editor_update").click(function () {
972| +         disableEditor(true);
973| +         handlers.update();
974| +
975| +     });
976| +
977| +     $("#editor_clear").click(function () {
978| +         disableEditor(true);
979| +         handlers.clear();
980| +
981| +     });
982| +
983| +     $("#editor_submit").click(function () {
984| +         disableEditor(true);
985| +         handlers.submit();
986| +
987| +     });
988| +
989| +     $("#editor_execute").click(function () {
990| +         disableEditor(true);
991| +         handlers.execute();
992| +
993| +     });
994| +
995| +     $("#editor_stop").click(function () {
996| +         disableEditor(true);
997| +         handlers.stop();
998| +
999| +     });
999| +
999| +     $("#editor_start").click(function () {
1000| +         disableEditor(false);
1001| +         handlers.start();
1002| +
1003| +     });
1004| +
1005| +     $("#editor_update").click(function () {
1006| +         disableEditor(true);
1007| +         handlers.update();
1008| +
1009| +     });
1010| +
1011| +     $("#editor_clear").click(function () {
1012| +         disableEditor(true);
1013| +         handlers.clear();
1014| +
1015| +     });
1016| +
1017| +     $("#editor_submit").click(function () {
1018| +         disableEditor(true);
1019| +         handlers.submit();
1020| +
1021| +     });
1022| +
1023| +     $("#editor_execute").click(function () {
1024| +         disableEditor(true);
1025| +         handlers.execute();
1026| +
1027| +     });
1028| +
1029| +     $("#editor_stop").click(function () {
1030| +         disableEditor(true);
1031| +         handlers.stop();
1032| +
1033| +     });
1034| +
1035| +     $("#editor_start").click(function () {
1036| +         disableEditor(false);
1037| +         handlers.start();
1038| +
1039| +     });
1040| +
1041| +     $("#editor_update").click(function () {
1042| +         disableEditor(true);
1043| +         handlers.update();
1044| +
1045| +     });
1046| +
1047| +     $("#editor_clear").click(function () {
1048| +         disableEditor(true);
1049| +         handlers.clear();
1050| +
1051| +     });
1052| +
1053| +     $("#editor_submit").click(function () {
1054| +         disableEditor(true);
1055| +         handlers.submit();
1056| +
1057| +     });
1058| +
1059| +     $("#editor_execute").click(function () {
1060| +         disableEditor(true);
1061| +         handlers.execute();
1062| +
1063| +     });
1064| +
1065| +     $("#editor_stop").click(function () {
1066| +         disableEditor(true);
1067| +         handlers.stop();
1068| +
1069| +     });
1070| +
1071| +     $("#editor_start").click(function () {
1072| +         disableEditor(false);
1073| +         handlers.start();
1074| +
1075| +     });
1076| +
1077| +     $("#editor_update").click(function () {
1078| +         disableEditor(true);
1079| +         handlers.update();
1080| +
1081| +     });
1082| +
1083| +     $("#editor_clear").click(function () {
1084| +         disableEditor(true);
1085| +         handlers.clear();
1086| +
1087| +     });
1088| +
1089| +     $("#editor_submit").click(function () {
1090| +         disableEditor(true);
1091| +         handlers.submit();
1092| +
1093| +     });
1094| +
1095| +     $("#editor_execute").click(function () {
1096| +         disableEditor(true);
1097| +         handlers.execute();
1098| +
1099| +     });
1100| +
1101| +     $("#editor_stop").click(function () {
1102| +         disableEditor(true);
1103| +         handlers.stop();
1104| +
1105| +     });
1106| +
1107| +     $("#editor_start").click(function () {
1108| +         disableEditor(false);
1109| +         handlers.start();
1110| +
1111| +     });
1112| +
1113| +     $("#editor_update").click(function () {
1114| +         disableEditor(true);
1115| +         handlers.update();
1116| +
1117| +     });
1118| +
1119| +     $("#editor_clear").click(function () {
1120| +         disableEditor(true);
1121| +         handlers.clear();
1122| +
1123| +     });
1124| +
1125| +     $("#editor_submit").click(function () {
1126| +         disableEditor(true);
1127| +         handlers.submit();
1128| +
1129| +     });
1130| +
1131| +     $("#editor_execute").click(function () {
1132| +         disableEditor(true);
1133| +         handlers.execute();
1134| +
1135| +     });
1136| +
1137| +     $("#editor_stop").click(function () {
1138| +         disableEditor(true);
1139| +         handlers.stop();
1140| +
1141| +     });
1142| +
1143| +     $("#editor_start").click(function () {
1144| +         disableEditor(false);
1145| +         handlers.start();
1146| +
1147| +     });
1148| +
1149| +     $("#editor_update").click(function () {
1150| +         disableEditor(true);
1151| +         handlers.update();
1152| +
1153| +     });
1154| +
1155| +     $("#editor_clear").click(function () {
1156| +         disableEditor(true);
1157| +         handlers.clear();
1158| +
1159| +     });
1160| +
1161| +     $("#editor_submit").click(function () {
1162| +         disableEditor(true);
1163| +         handlers.submit();
1164| +
1165| +     });
1166| +
1167| +     $("#editor_execute").click(function () {
1168| +         disableEditor(true);
1169| +         handlers.execute();
1170| +
1171| +     });
1172| +
1173| +     $("#editor_stop").click(function () {
1174| +         disableEditor(true);
1175| +         handlers.stop();
1176| +
1177| +     });
1178| +
1179| +     $("#editor_start").click(function () {
1180| +         disableEditor(false);
1181| +         handlers.start();
1182| +
1183| +     });
1184| +
1185| +     $("#editor_update").click(function () {
1186| +         disableEditor(true);
1187| +         handlers.update();
1188| +
1189| +     });
1190| +
1191| +     $("#editor_clear").click(function () {
1192| +         disableEditor(true);
1193| +         handlers.clear();
1194| +
1195| +     });
1196| +
1197| +     $("#editor_submit").click(function () {
1198| +         disableEditor(true);
1199| +         handlers.submit();
1200| +
1201| +     });
1202| +
1203| +     $("#editor_execute").click(function () {
1204| +         disableEditor(true);
1205| +         handlers.execute();
1206| +
1207| +     });
1208| +
1209| +     $("#editor_stop").click(function () {
1210| +         disableEditor(true);
1211| +         handlers.stop();
1212| +
1213| +     });
1214| +
1215| +     $("#editor_start").click(function () {
1216| +         disableEditor(false);
1217| +         handlers.start();
1218| +
1219| +     });
1220| +
1221| +     $("#editor_update").click(function () {
1222| +         disableEditor(true);
1223| +         handlers.update();
1224| +
1225| +     });
1226| +
1227| +     $("#editor_clear").click(function () {
1228| +         disableEditor(true);
1229| +         handlers.clear();
1230| +
1231| +     });
1232| +
1233| +     $("#editor_submit").click(function () {
1234| +         disableEditor(true);
1235| +         handlers.submit();
1236| +
1237| +     });
1238| +
1239| +     $("#editor_execute").click(function () {
1240| +         disableEditor(true);
1241| +         handlers.execute();
1242| +
1243| +     });
1244| +
1245| +     $("#editor_stop").click(function () {
1246| +         disableEditor(true);
1247| +         handlers.stop();
1248| +
1249| +     });
1250| +
1251| +     $("#editor_start").click(function () {
1252| +         disableEditor(false);
1253| +         handlers.start();
1254| +
1255| +     });
1256| +
1257| +     $("#editor_update").click(function () {
1258| +         disableEditor(true);
1259| +         handlers.update();
1260| +
1261| +     });
1262| +
1263| +     $("#editor_clear").click(function () {
1264| +         disableEditor(true);
1265| +         handlers.clear();
1266| +
1267| +     });
1268| +
1269| +     $("#editor_submit").click(function () {
1270| +         disableEditor(true);
1271| +         handlers.submit();
1272| +
1273| +     });
1274| +
1275| +     $("#editor_execute").click(function () {
1276| +         disableEditor(true);
1277| +         handlers.execute();
1278| +
1279| +     });
1280| +
1281| +     $("#editor_stop").click(function () {
1282| +         disableEditor(true);
1283| +         handlers.stop();
1284| +
1285| +     });
1286| +
1287| +     $("#editor_start").click(function () {
1288| +         disableEditor(false);
1289| +         handlers.start();
1290| +
1291| +     });
1292| +
1293| +     $("#editor_update").click(function () {
1294| +         disableEditor(true);
1295| +         handlers.update();
1296| +
1297| +     });
1298| +
1299| +     $("#editor_clear").click(function () {
1300| +         disableEditor(true);
1301| +         handlers.clear();
1302| +
1303| +     });
1304| +
1305| +     $("#editor_submit").click(function () {
1306| +         disableEditor(true);
1307| +         handlers.submit();
1308| +
1309| +     });
1310| +
1311| +     $("#editor_execute").click(function () {
1312| +         disableEditor(true);
1313| +         handlers.execute();
1314| +
1315| +     });
1316| +
1317| +     $("#editor_stop").click(function () {
1318| +         disableEditor(true);
1319| +         handlers.stop();
1320| +
1321| +     });
1322| +
1323| +     $("#editor_start").click(function () {
1324| +         disableEditor(false);
1325| +         handlers.start();
1326| +
1327| +     });
1328| +
1329| +     $("#editor_update").click(function () {
1330| +         disableEditor(true);
1331| +         handlers.update();
1332| +
1333| +     });
1334| +
1335| +     $("#editor_clear").click(function () {
1336| +         disableEditor(true);
1337| +         handlers.clear();
1338| +
1339| +     });
1340| +
1341| +     $("#editor_submit").click(function () {
1342| +         disableEditor(true);
1343| +         handlers.submit();
1344| +
1345| +     });
1346| +
1347| +     $("#editor_execute").click(function () {
1348| +         disableEditor(true);
1349| +         handlers.execute();
1350| +
1351| +     });
1352| +
1353| +     $("#editor_stop").click(function () {
1354| +         disableEditor(true);
1355| +         handlers.stop();
1356| +
1357| +     });
1358| +
1359| +     $("#editor_start").click(function () {
1360| +         disableEditor(false);
1361| +         handlers.start();
1362| +
1363| +     });
1364| +
1365| +     $("#editor_update").click(function () {
1366| +         disableEditor(true);
1367| +         handlers.update();
1368| +
1369| +     });
1370| +
1371| +     $("#editor_clear").click(function () {
1372| +         disableEditor(true);
1373| +         handlers.clear();
1374| +
1375| +     });
1376| +
1377| +     $("#editor_submit").click(function () {
1378| +         disableEditor(true);
1379| +         handlers.submit();
1380| +
1381| +     });
1382| +
1383| +     $("#editor_execute").click(function () {
1384| +         disableEditor(true);
1385| +         handlers.execute();
1386| +
1387| +     });
1388| +
1389| +     $("#editor_stop").click(function () {
1390| +         disableEditor(true);
1391| +         handlers.stop();
1392| +
1393| +     });
1394| +
1395| +     $("#editor_start").click(function () {
1396| +         disableEditor(false);
1397| +         handlers.start();
1398| +
1399| +     });
1400| +
1401| +     $("#editor_update").click(function () {
1402| +         disableEditor(true);
1403| +         handlers.update();
1404| +
1405| +     });
1406| +
1407| +     $("#editor_clear").click(function () {
1408| +         disableEditor(true);
1409| +         handlers.clear();
1410| +
1411| +     });
1412| +
1413| +     $("#editor_submit").click(function () {
1414| +         disableEditor(true);
1415| +         handlers.submit();
1416| +
1417| +     });
1418| +
1419| +     $("#editor_execute").click(function () {
1420| +         disableEditor(true);
1421| +         handlers.execute();
1422| +
1423| +     });
1424| +
1425| +     $("#editor_stop").click(function () {
1426| +         disableEditor(true);
1427| +         handlers.stop();
1428| +
1429| +     });
1430| +
1431| +     $("#editor_start").click(function () {
1432| +         disableEditor(false);
1433| +         handlers.start();
1434| +
1435| +     });
1436| +
1437| +     $("#editor_update").click(function () {
1438| +         disableEditor(true);
1439| +         handlers.update();
1440| +
1441| +     });
1442| +
1443| +     $("#editor_clear").click(function () {
1444| +         disableEditor(true);
1445| +         handlers.clear();
1446| +
1447| +     });
1448| +
1449| +     $("#editor_submit").click(function () {
1450| +         disableEditor(true);
1451| +         handlers.submit();
1452| +
1453| +     });
1454| +
1455| +     $("#editor_execute").click(function () {
1456| +         disableEditor(true);
1457| +         handlers.execute();
1458| +
1459| +     });
1460| +
1461| +     $("#editor_stop").click(function () {
1462| +         disableEditor(true);
1463| +         handlers.stop();
1464| +
1465| +     });
1466| +
1467| +     $("#editor_start").click(function () {
1468| +         disableEditor(false);
1469| +         handlers.start();
1470| +
1471| +     });
1472| +
1473| +     $("#editor_update").click(function () {
1474| +         disableEditor(true);
1475| +         handlers.update();
1476| +
1477| +     });
1478| +
1479| +     $("#editor_clear").click(function () {
1480| +         disableEditor(true);
1481| +         handlers.clear();
1482| +
1483| +     });
1484| +
1485| +     $("#editor_submit").click(function () {
1486| +         disableEditor(true);
1487| +         handlers.submit();
1488| +
1489| +     });
1490| +
1491| +     $("#editor_execute").click(function () {
1492| +         disableEditor(true);
1493| +         handlers.execute();
1494| +
1495| +     });
1496| +
1497| +     $("#editor_stop").click(function () {
1498| +         disableEditor(true);
1499| +         handlers.stop();
1500| +
1501| +     });
1502| +
1503| +     $("#editor_start").click(function () {
1504| +         disableEditor(false);
1505| +         handlers.start();
1506| +
1507| +     });
1508| +
1509| +     $("#editor_update").click(function () {
1510| +         disableEditor(true);
1511| +         handlers.update();
1512| +
1513| +     });
1514| +
1515| +     $("#editor_clear").click(function () {
1516| +         disableEditor(true);
1517| +         handlers.clear();
1518| +
1519| +     });
1520| +
1521| +     $("#editor_submit").click(function () {
1522| +         disableEditor(true);
1523| +         handlers.submit();
1524| +
1525| +     });
1526| +
1527| +     $("#editor_execute").click(function () {
1528| +         disableEditor(true);
1529| +         handlers.execute();
1530| +
1531| +     });
1532| +
1533| +     $("#editor_stop").click(function () {
1534| +         disableEditor(true);
1535| +         handlers.stop();
1536| +
1537| +     });
1538| +
1539| +     $("#editor_start").click(function () {
1540| +         disableEditor(false);
1541| +         handlers.start();
1542| +
1543| +     });
1544| +
1545| +     $("#editor_update").click(function () {
1546| +         disableEditor(true);
1547| +         handlers.update();
1548| +
1549| +     });
1550| +
1551
```

```

415| +                     $("#ajax_status").html("Completed");
416| +                     disableEditor(false);
417| +
418| +                 },
419| +             error: function (error) {
420| +                 console.error(error);
421| +                 disableEditor(false);
422| +
423| +             });
424| +         }());
425| +     } else {
426| +         disableEditor(false);
427| +
428| +     },
429| +     error: function (error) {
430| +         console.error(error);
431| +         disableEditor(false);
432| +
433| +     });
434| + });
435| +
436| + loadCode($("#select#problems").val());
437| +
438| + // ##### Recording #####
439| + // ##### Listener #####
440| + // #####
441| +
442| + // Local Variable
443| + const Range = ace.Range;
444| +
445| + let hidden = "hidden";
446| + let visibilityChange = "visibilitychange";
447| + let isRetrieved = false;
448| +
449| + // Saved Recording from before...
450| + let befRecording = {};
451| +
452| + // Saved Event
453| + const recording = {
454| +     events: [],
455| +     startTime: -1,
456| +
457| +     init: () => {
458| +         recording.events = [];
459| +         recording.startTime = Date.now();
460| +
461| +     },
462| +
463| + // What's recording does the system will record
464| + const include = {
465| +     editor: true,
466| +     web: true,
467| +     pdf: true,
468| +     input: true,
469| +     output: true,
470| +     // action: true, // always true
471| +     others: false, // default value for other recording
472| +
473| +
474| + // ##### Editor Event #####
475| + // Detected Every Command that executed in editor
476| + // #####
477| +
478| + const handlers = {
479| +     // ##### Editor Event #####
480| +     // Detected Every Command that executed in editor
481| +     editor_change: (e) =>
482| +         recordEvent(e.action, {
483| +             data: e.lines,
484| +             start: e.start,
485| +             end: e.end,
486| +         }),
487| +     // Detected on cursor change
488| +     editor_cursor: () => recordEvent("cursor_selection", getSelection(editor)),
489| +     // Detected on selection
490| +     editor_selection: () => recordEvent("sel_selection", getSelection(editor)),
491| +
492| +     // ##### Windows Event #####
493| +     // Detected Leaving Focus in almost all browser (still active page, but on different windows or on iFrame PDF viewer)
494| +     focus: () => {
495| +         removeListener.focus();
496| +         addListener.blur();
497| +
498| +         recordEvent("focus");
499| +
500| +     },
501| +     // Detected on Focus in almost all browser (active page)
502| +     blur: () => {
503| +         removeListener.blur();
504| +         addListener.focus();
505| +
506| +         recordEvent("blur");
507| +
508| +     },
509| +     // Detected Leaving Page in almost all browser (page not visible anymore)
510| +     visibility: (evt) => {
511| +         let v = true; // page is visible
512| +         let h = false; // page is hidden
513| +
514| +         let evtMap = {
515| +             focus: v,
516| +
517| +         };
518| +
519| +         if (v) {
520| +             recordEvent("focus");
521| +
522| +         } else {
523| +             recordEvent("blur");
524| +
525| +         }
526| +
527| +     }
528| +
529| +     // Detecting when the user scrolls
530| +     scroll: () => {
531| +         recordEvent("scroll");
532| +
533| +     }
534| +
535| +     // Detecting when the user types
536| +     type: () => {
537| +         recordEvent("type");
538| +
539| +     }
540| +
541| +     // Detecting when the user pastes
542| +     paste: () => {
543| +         recordEvent("paste");
544| +
545| +     }
546| +
547| +     // Detecting when the user deletes
548| +     delete: () => {
549| +         recordEvent("delete");
550| +
551| +     }
552| +
553| +     // Detecting when the user copies
554| +     copy: () => {
555| +         recordEvent("copy");
556| +
557| +     }
558| +
559| +     // Detecting when the user cuts
560| +     cut: () => {
561| +         recordEvent("cut");
562| +
563| +     }
564| +
565| +     // Detecting when the user pastes
566| +     paste: () => {
567| +         recordEvent("paste");
568| +
569| +     }
570| +
571| +     // Detecting when the user types
572| +     type: () => {
573| +         recordEvent("type");
574| +
575| +     }
576| +
577| +     // Detecting when the user deletes
578| +     delete: () => {
579| +         recordEvent("delete");
580| +
581| +     }
582| +
583| +     // Detecting when the user copies
584| +     copy: () => {
585| +         recordEvent("copy");
586| +
587| +     }
588| +
589| +     // Detecting when the user cuts
590| +     cut: () => {
591| +         recordEvent("cut");
592| +
593| +     }
594| +
595| +     // Detecting when the user scrolls
596| +     scroll: () => {
597| +         recordEvent("scroll");
598| +
599| +     }
599| +
600| +     // Detecting when the user types
601| +     type: () => {
602| +         recordEvent("type");
603| +
604| +     }
605| +
606| +     // Detecting when the user deletes
607| +     delete: () => {
608| +         recordEvent("delete");
609| +
610| +     }
611| +
612| +     // Detecting when the user copies
613| +     copy: () => {
614| +         recordEvent("copy");
615| +
616| +     }
617| +
618| +     // Detecting when the user cuts
619| +     cut: () => {
620| +         recordEvent("cut");
621| +
622| +     }
623| +
624| +     // Detecting when the user scrolls
625| +     scroll: () => {
626| +         recordEvent("scroll");
627| +
628| +     }
629| +
630| +     // Detecting when the user types
631| +     type: () => {
632| +         recordEvent("type");
633| +
634| +     }
635| +
636| +     // Detecting when the user deletes
637| +     delete: () => {
638| +         recordEvent("delete");
639| +
640| +     }
641| +
642| +     // Detecting when the user copies
643| +     copy: () => {
644| +         recordEvent("copy");
645| +
646| +     }
647| +
648| +     // Detecting when the user cuts
649| +     cut: () => {
650| +         recordEvent("cut");
651| +
652| +     }
653| +
654| +     // Detecting when the user scrolls
655| +     scroll: () => {
656| +         recordEvent("scroll");
657| +
658| +     }
659| +
660| +     // Detecting when the user types
661| +     type: () => {
662| +         recordEvent("type");
663| +
664| +     }
665| +
666| +     // Detecting when the user deletes
667| +     delete: () => {
668| +         recordEvent("delete");
669| +
670| +     }
671| +
672| +     // Detecting when the user copies
673| +     copy: () => {
674| +         recordEvent("copy");
675| +
676| +     }
677| +
678| +     // Detecting when the user cuts
679| +     cut: () => {
680| +         recordEvent("cut");
681| +
682| +     }
683| +
684| +     // Detecting when the user scrolls
685| +     scroll: () => {
686| +         recordEvent("scroll");
687| +
688| +     }
689| +
690| +     // Detecting when the user types
691| +     type: () => {
692| +         recordEvent("type");
693| +
694| +     }
695| +
696| +     // Detecting when the user deletes
697| +     delete: () => {
698| +         recordEvent("delete");
699| +
700| +     }
701| +
702| +     // Detecting when the user copies
703| +     copy: () => {
704| +         recordEvent("copy");
705| +
706| +     }
707| +
708| +     // Detecting when the user cuts
709| +     cut: () => {
710| +         recordEvent("cut");
711| +
712| +     }
713| +
714| +     // Detecting when the user scrolls
715| +     scroll: () => {
716| +         recordEvent("scroll");
717| +
718| +     }
719| +
720| +     // Detecting when the user types
721| +     type: () => {
722| +         recordEvent("type");
723| +
724| +     }
725| +
726| +     // Detecting when the user deletes
727| +     delete: () => {
728| +         recordEvent("delete");
729| +
730| +     }
731| +
732| +     // Detecting when the user copies
733| +     copy: () => {
734| +         recordEvent("copy");
735| +
736| +     }
737| +
738| +     // Detecting when the user cuts
739| +     cut: () => {
740| +         recordEvent("cut");
741| +
742| +     }
743| +
744| +     // Detecting when the user scrolls
745| +     scroll: () => {
746| +         recordEvent("scroll");
747| +
748| +     }
749| +
750| +     // Detecting when the user types
751| +     type: () => {
752| +         recordEvent("type");
753| +
754| +     }
755| +
756| +     // Detecting when the user deletes
757| +     delete: () => {
758| +         recordEvent("delete");
759| +
760| +     }
761| +
762| +     // Detecting when the user copies
763| +     copy: () => {
764| +         recordEvent("copy");
765| +
766| +     }
767| +
768| +     // Detecting when the user cuts
769| +     cut: () => {
770| +         recordEvent("cut");
771| +
772| +     }
773| +
774| +     // Detecting when the user scrolls
775| +     scroll: () => {
776| +         recordEvent("scroll");
777| +
778| +     }
779| +
780| +     // Detecting when the user types
781| +     type: () => {
782| +         recordEvent("type");
783| +
784| +     }
785| +
786| +     // Detecting when the user deletes
787| +     delete: () => {
788| +         recordEvent("delete");
789| +
790| +     }
791| +
792| +     // Detecting when the user copies
793| +     copy: () => {
794| +         recordEvent("copy");
795| +
796| +     }
797| +
798| +     // Detecting when the user cuts
799| +     cut: () => {
800| +         recordEvent("cut");
801| +
802| +     }
803| +
804| +     // Detecting when the user scrolls
805| +     scroll: () => {
806| +         recordEvent("scroll");
807| +
808| +     }
809| +
810| +     // Detecting when the user types
811| +     type: () => {
812| +         recordEvent("type");
813| +
814| +     }
815| +
816| +     // Detecting when the user deletes
817| +     delete: () => {
818| +         recordEvent("delete");
819| +
820| +     }
821| +
822| +     // Detecting when the user copies
823| +     copy: () => {
824| +         recordEvent("copy");
825| +
826| +     }
827| +
828| +     // Detecting when the user cuts
829| +     cut: () => {
830| +         recordEvent("cut");
831| +
832| +     }
833| +
834| +     // Detecting when the user scrolls
835| +     scroll: () => {
836| +         recordEvent("scroll");
837| +
838| +     }
839| +
840| +     // Detecting when the user types
841| +     type: () => {
842| +         recordEvent("type");
843| +
844| +     }
845| +
846| +     // Detecting when the user deletes
847| +     delete: () => {
848| +         recordEvent("delete");
849| +
850| +     }
851| +
852| +     // Detecting when the user copies
853| +     copy: () => {
854| +         recordEvent("copy");
855| +
856| +     }
857| +
858| +     // Detecting when the user cuts
859| +     cut: () => {
860| +         recordEvent("cut");
861| +
862| +     }
863| +
864| +     // Detecting when the user scrolls
865| +     scroll: () => {
866| +         recordEvent("scroll");
867| +
868| +     }
869| +
870| +     // Detecting when the user types
871| +     type: () => {
872| +         recordEvent("type");
873| +
874| +     }
875| +
876| +     // Detecting when the user deletes
877| +     delete: () => {
878| +         recordEvent("delete");
879| +
880| +     }
881| +
882| +     // Detecting when the user copies
883| +     copy: () => {
884| +         recordEvent("copy");
885| +
886| +     }
887| +
888| +     // Detecting when the user cuts
889| +     cut: () => {
890| +         recordEvent("cut");
891| +
892| +     }
893| +
894| +     // Detecting when the user scrolls
895| +     scroll: () => {
896| +         recordEvent("scroll");
897| +
898| +     }
899| +
900| +     // Detecting when the user types
901| +     type: () => {
902| +         recordEvent("type");
903| +
904| +     }
905| +
906| +     // Detecting when the user deletes
907| +     delete: () => {
908| +         recordEvent("delete");
909| +
910| +     }
911| +
912| +     // Detecting when the user copies
913| +     copy: () => {
914| +         recordEvent("copy");
915| +
916| +     }
917| +
918| +     // Detecting when the user cuts
919| +     cut: () => {
920| +         recordEvent("cut");
921| +
922| +     }
923| +
924| +     // Detecting when the user scrolls
925| +     scroll: () => {
926| +         recordEvent("scroll");
927| +
928| +     }
929| +
930| +     // Detecting when the user types
931| +     type: () => {
932| +         recordEvent("type");
933| +
934| +     }
935| +
936| +     // Detecting when the user deletes
937| +     delete: () => {
938| +         recordEvent("delete");
939| +
940| +     }
941| +
942| +     // Detecting when the user copies
943| +     copy: () => {
944| +         recordEvent("copy");
945| +
946| +     }
947| +
948| +     // Detecting when the user cuts
949| +     cut: () => {
950| +         recordEvent("cut");
951| +
952| +     }
953| +
954| +     // Detecting when the user scrolls
955| +     scroll: () => {
956| +         recordEvent("scroll");
957| +
958| +     }
959| +
960| +     // Detecting when the user types
961| +     type: () => {
962| +         recordEvent("type");
963| +
964| +     }
965| +
966| +     // Detecting when the user deletes
967| +     delete: () => {
968| +         recordEvent("delete");
969| +
970| +     }
971| +
972| +     // Detecting when the user copies
973| +     copy: () => {
974| +         recordEvent("copy");
975| +
976| +     }
977| +
978| +     // Detecting when the user cuts
979| +     cut: () => {
980| +         recordEvent("cut");
981| +
982| +     }
983| +
984| +     // Detecting when the user scrolls
985| +     scroll: () => {
986| +         recordEvent("scroll");
987| +
988| +     }
989| +
990| +     // Detecting when the user types
991| +     type: () => {
992| +         recordEvent("type");
993| +
994| +     }
995| +
996| +     // Detecting when the user deletes
997| +     delete: () => {
998| +         recordEvent("delete");
999| +
1000| +     }
1001| +
1002| +     // Detecting when the user copies
1003| +     copy: () => {
1004| +         recordEvent("copy");
1005| +
1006| +     }
1007| +
1008| +     // Detecting when the user cuts
1009| +     cut: () => {
1010| +         recordEvent("cut");
1011| +
1012| +     }
1013| +
1014| +     // Detecting when the user scrolls
1015| +     scroll: () => {
1016| +         recordEvent("scroll");
1017| +
1018| +     }
1019| +
1020| +     // Detecting when the user types
1021| +     type: () => {
1022| +         recordEvent("type");
1023| +
1024| +     }
1025| +
1026| +     // Detecting when the user deletes
1027| +     delete: () => {
1028| +         recordEvent("delete");
1029| +
1030| +     }
1031| +
1032| +     // Detecting when the user copies
1033| +     copy: () => {
1034| +         recordEvent("copy");
1035| +
1036| +     }
1037| +
1038| +     // Detecting when the user cuts
1039| +     cut: () => {
1040| +         recordEvent("cut");
1041| +
1042| +     }
1043| +
1044| +     // Detecting when the user scrolls
1045| +     scroll: () => {
1046| +         recordEvent("scroll");
1047| +
1048| +     }
1049| +
1050| +     // Detecting when the user types
1051| +     type: () => {
1052| +         recordEvent("type");
1053| +
1054| +     }
1055| +
1056| +     // Detecting when the user deletes
1057| +     delete: () => {
1058| +         recordEvent("delete");
1059| +
1060| +     }
1061| +
1062| +     // Detecting when the user copies
1063| +     copy: () => {
1064| +         recordEvent("copy");
1065| +
1066| +     }
1067| +
1068| +     // Detecting when the user cuts
1069| +     cut: () => {
1070| +         recordEvent("cut");
1071| +
1072| +     }
1073| +
1074| +     // Detecting when the user scrolls
1075| +     scroll: () => {
1076| +         recordEvent("scroll");
1077| +
1078| +     }
1079| +
1080| +     // Detecting when the user types
1081| +     type: () => {
1082| +         recordEvent("type");
1083| +
1084| +     }
1085| +
1086| +     // Detecting when the user deletes
1087| +     delete: () => {
1088| +         recordEvent("delete");
1089| +
1090| +     }
1091| +
1092| +     // Detecting when the user copies
1093| +     copy: () => {
1094| +         recordEvent("copy");
1095| +
1096| +     }
1097| +
1098| +     // Detecting when the user cuts
1099| +     cut: () => {
1100| +         recordEvent("cut");
1101| +
1102| +     }
1103| +
1104| +     // Detecting when the user scrolls
1105| +     scroll: () => {
1106| +         recordEvent("scroll");
1107| +
1108| +     }
1109| +
1110| +     // Detecting when the user types
1111| +     type: () => {
1112| +         recordEvent("type");
1113| +
1114| +     }
1115| +
1116| +     // Detecting when the user deletes
1117| +     delete: () => {
1118| +         recordEvent("delete");
1119| +
1120| +     }
1121| +
1122| +     // Detecting when the user copies
1123| +     copy: () => {
1124| +         recordEvent("copy");
1125| +
1126| +     }
1127| +
1128| +     // Detecting when the user cuts
1129| +     cut: () => {
1130| +         recordEvent("cut");
1131| +
1132| +     }
1133| +
1134| +     // Detecting when the user scrolls
1135| +     scroll: () => {
1136| +         recordEvent("scroll");
1137| +
1138| +     }
1139| +
1140| +     // Detecting when the user types
1141| +     type: () => {
1142| +         recordEvent("type");
1143| +
1144| +     }
1145| +
1146| +     // Detecting when the user deletes
1147| +     delete: () => {
1148| +         recordEvent("delete");
1149| +
1150| +     }
1151| +
1152| +     // Detecting when the user copies
1153| +     copy: () => {
1154| +         recordEvent("copy");
1155| +
1156| +     }
1157| +
1158| +     // Detecting when the user cuts
1159| +     cut: () => {
1160| +         recordEvent("cut");
1161| +
1162| +     }
1163| +
1164| +     // Detecting when the user scrolls
1165| +     scroll: () => {
1166| +         recordEvent("scroll");
1167| +
1168| +     }
1169| +
1170| +     // Detecting when the user types
1171| +     type: () => {
1172| +         recordEvent("type");
1173| +
1174| +     }
1175| +
1176| +     // Detecting when the user deletes
1177| +     delete: () => {
1178| +         recordEvent("delete");
1179| +
1180| +     }
1181| +
1182| +     // Detecting when the user copies
1183| +     copy: () => {
1184| +         recordEvent("copy");
1185| +
1186| +     }
1187| +
1188| +     // Detecting when the user cuts
1189| +     cut: () => {
1190| +         recordEvent("cut");
1191| +
1192| +     }
1193| +
1194| +     // Detecting when the user scrolls
1195| +     scroll: () => {
1196| +         recordEvent("scroll");
1197| +
1198| +     }
1199| +
1200| +     // Detecting when the user types
1201| +     type: () => {
1202| +         recordEvent("type");
1203| +
1204| +     }
1205| +
1206| +     // Detecting when the user deletes
1207| +     delete: () => {
1208| +         recordEvent("delete");
1209| +
1210| +     }
1211| +
1212| +     // Detecting when the user copies
1213| +     copy: () => {
1214| +         recordEvent("copy");
1215| +
1216| +     }
1217| +
1218| +     // Detecting when the user cuts
1219| +     cut: () => {
1220| +         recordEvent("cut");
1221| +
1222| +     }
1223| +
1224| +     // Detecting when the user scrolls
1225| +     scroll: () => {
1226| +         recordEvent("scroll");
1227| +
1228| +     }
1229| +
1230| +     // Detecting when the user types
1231| +     type: () => {
1232| +         recordEvent("type");
1233| +
1234| +     }
1235| +
1236| +     // Detecting when the user deletes
1237| +     delete: () => {
1238| +         recordEvent("delete");
1239| +
1240| +     }
1241| +
1242| +     // Detecting when the user copies
1243| +     copy: () => {
1244| +         recordEvent("copy");
1245| +
1246| +     }
1247| +
1248| +     // Detecting when the user cuts
1249| +     cut: () => {
1250| +         recordEvent("cut");
1251| +
1252| +     }
1253| +
1254| +     // Detecting when the user scrolls
1255| +     scroll: () => {
1256| +         recordEvent("scroll");
1257| +
1258| +     }
1259| +
1260| +     // Detecting when the user types
1261| +     type: () => {
1262| +         recordEvent("type");
1263| +
1264| +     }
1265| +
1266| +     // Detecting when the user deletes
1267| +     delete: () => {
1268| +         recordEvent("delete");
1269| +
1270| +     }
1271| +
1272| +     // Detecting when the user copies
1273| +     copy: () => {
1274| +         recordEvent("copy");
1275| +
1276| +     }
1277| +
1278| +     // Detecting when the user cuts
1279| +     cut: () => {
1280| +         recordEvent("cut");
1281| +
1282| +     }
1283| +
1284| +     // Detecting when the user scrolls
1285| +     scroll: () => {
1286| +         recordEvent("scroll");
1287| +
1288| +     }
1289| +
1290| +     // Detecting when the user types
1291| +     type: () => {
1292| +         recordEvent("type");
1293| +
1294| +     }
1295| +
1296| +     // Detecting when the user deletes
1297| +     delete: () => {
1298| +         recordEvent("delete");
1299| +
1300| +     }
1301| +
1302| +     // Detecting when the user copies
1303| +     copy: () => {
1304| +         recordEvent("copy");
1305| +
1306| +     }
1307| +
1308| +     // Detecting when the user cuts
1309| +     cut: () => {
1310| +         recordEvent("cut");
1311| +
1312| +     }
1313| +
1314| +     // Detecting when the user scrolls
1315| +     scroll: () => {
1316| +         recordEvent("scroll");
1317| +
1318| +     }
1319| +
1320| +     // Detecting when the user types
1321| +     type: () => {
1322| +         recordEvent("type");
1323| +
1324| +     }
1325| +
1326| +     // Detecting when the user deletes
1327| +     delete: () => {
1328| +         recordEvent("delete");
1329| +
1330| +     }
1331| +
1332| +     // Detecting when the user copies
1333| +     copy: () => {
1334| +         recordEvent("copy");
1335| +
1336| +     }
1337| +
1338| +     // Detecting when the user cuts
1339| +     cut: () => {
1340| +         recordEvent("cut");
1341| +
1342| +     }
1343| +
1344| +     // Detecting when the user scrolls
1345| +     scroll: () => {
1346| +         recordEvent("scroll");
1347| +
1348| +     }
1349| +
1350| +     // Detecting when the user types
1351| +     type: () => {
1352| +         recordEvent("type");
1353| +
1354| +     }
1355| +
1356| +     // Detecting when the user deletes
1357| +     delete: () => {
1358| +         recordEvent("delete");
1359| +
1360| +     }
1361| +
1362| +     // Detecting when the user copies
1363| +     copy: () => {
1364| +         recordEvent("copy");
1365| +
1366| +     }
1367| +
1368| +     // Detecting when the user cuts
1369| +     cut: () => {
1370| +         recordEvent("cut");
1371| +
1372| +     }
1373| +
1374| +     // Detecting when the user scrolls
1375| +     scroll: () => {
1376| +         recordEvent("scroll");
1377| +
1378| +     }
1379| +
1380| +     // Detecting when the user types
1381| +     type: () => {
1382| +         recordEvent("type");
1383| +
1384| +     }
1385| +
1386| +     // Detecting when the user deletes
1387| +     delete: () => {
1388| +         recordEvent("delete");
1389| +
1390| +     }
1391| +
1392| +     // Detecting when the user copies
1393| +     copy: () => {
1394| +         recordEvent("copy");
1395| +
1396| +     }
1397| +
1398| +     // Detecting when the user cuts
1399| +     cut: () => {
1400| +         recordEvent("cut");
1401| +
1402| +     }
1403| +
1404| +     // Detecting when the user scrolls
1405| +     scroll: () => {
1406| +         recordEvent("scroll");
1407| +
1408| +     }
1409| +
1410| +     // Detecting when the user types
1411| +     type: () => {
1412| +         recordEvent("type");
1413| +
1414| +     }
1415| +
1416| +     // Detecting when the user deletes
1417| +     delete: () => {
1418| +         recordEvent("delete");
1419| +
1420| +     }
1421| +
1422| +     // Detecting when the user copies
1423| +     copy: () => {
1424| +         recordEvent("copy");
1425| +
1426| +     }
1427| +
1428| +     // Detecting when the user cuts
1429| +     cut: () => {
1430| +         recordEvent("cut");
1431| +
1432| +     }
1433| +
1434| +     // Detecting when the user scrolls
1435| +     scroll: () => {
1436| +         recordEvent("scroll");
1437| +
1438| +     }
1439| +
1440| +     // Detecting when the user types
1441| +     type: () => {
1442| +         recordEvent("type");
1443| +
1444| +     }
1445| +
1446| +     // Detecting when the user deletes
1447| +     delete: () => {
1448| +         recordEvent("delete");
1449| +
1450| +     }
1451| +
1452| +     // Detecting when the user copies
1453| +     copy: () => {
1454| +         recordEvent("copy");
1455| +
1456| +     }
1457| +
1458| +     // Detecting when the user cuts
1459| +     cut: () => {
1460| +         recordEvent("cut");
1461| +
1462| +     }
1463| +
1464| +     // Detecting when the user scrolls
1465| +     scroll: () => {
1466| +         recordEvent("scroll");
1467| +
1468| +     }
1469| +
1470| +     // Detecting when the user types
1471| +     type: () => {
1472| +         recordEvent("type");
1473| +
1474| +     }
1475| +
1476| +     // Detecting when the user deletes
1477| +     delete: () => {
1478| +         recordEvent("delete");
1479| +
1480| +     }
1481| +
1482| +     // Detecting when the user copies
1483| +     copy: () => {
1484| +         recordEvent("copy");
1485| +
1486| +     }
1487| +
1488| +     // Detecting when the user cuts
1489| +     cut: () => {
1490| +         recordEvent("cut");
1491| +
1492| +     }
1493| +
1494| +     // Detecting when the user scrolls
1495| +     scroll: () => {
1496| +         recordEvent("scroll");
1497| +
1498| +     }
1499| +
1500| +     // Detecting when the user types
1501| +     type: () => {
1502| +         recordEvent("type");
1503| +
1504| +     }
1505| +
1506| +     // Detecting when the user deletes
1507| +     delete: () => {
1508| +         recordEvent("delete");
1509| +
1510| +     }
1511| +
1512| +     // Detecting when the user copies
1513| +     copy: () => {
1514| +         recordEvent("copy");
1515| +
1516| +     }
1517| +
1518| +     // Detecting when the user cuts
1519| +     cut: () => {
1520| +         recordEvent("cut");
1521| +
1522| +     }
1523| +
1524| +     // Detecting when the user scrolls
1525| +     scroll: () => {
1526| +         recordEvent("scroll");
1527| +
1528| +     }
1529| +
1530| +     // Detecting when the user types
1531| +     type: () => {
1532| +         recordEvent("type");
1533| +
1534| +     }
1535| +
1536| +     // Detecting when the user deletes
1537| +     delete: () => {
1538| +         recordEvent("delete");
1539| +
1540| +     }
1541| +
1542| +     // Detecting when the user copies
1543| +     copy: () => {
1544| +         recordEvent("copy");
1545| +
1546| +     }
1547| +
1548| +     // Detecting when the user cuts
1549| +     cut: () => {
1550| +         recordEvent("cut");
1551| +
1552| +     }
1553| +
1554| +     // Detecting when the user scrolls
1555| +     scroll: () => {
1556| +         recordEvent("scroll");
1557| +
155
```

```

514| +         focusin: v,
515| +         pageshow: v,
516| +         blur: h,
517| +         focusout: h,
518| +         pagehide: h,
519| +     );
520| +
521| +     let isVisible = true;
522| +
523| +     evt = evt || window.event;
524| +
525| +     if (evt.type in evtMap) isVisible = evtMap[evt.type];
526| +     else isVisible = document[hidden] ? h : v;
527| +
528| +     if (isVisible) {
529| +         // detect focus or blur if visible again
530| +         addListener.focus();
531| +         addListener.pdf_focus();
532| +     } else {
533| +         // no need to detect focus or blur if not visible
534| +         removeListener.focus();
535| +         removeListener.blur();
536| +         removeListener.pdf_focus();
537| +         removeListener.pdf.blur();
538| +     }
539| +
540| +     recordEvent("visibility", isVisible);
541| +
542| +
543| + // ##### PDF Viewer #####
544| + // Detected when user click/focus on the pdf viewer IDE
545| + pdf_focus: () => {
546| +     removeListener.pdf_focus();
547| +     addListener.pdf.blur();
548| +
549| +     recordEvent("pdf_focus");
550| },
551| + // Detected when user click outside/blur of the pdf viewer IDE
552| + pdf.blur: () => {
553| +     removeListener.pdf.blur();
554| +     addListener.pdf_focus();
555| +
556| +     recordEvent("pdf.blur");
557| },
558| +
559| + // ##### Input Event #####
560| + input_change: (e) => recordEvent("input_change", e.currentTarget.value),
561| +
562| + // ##### Output Event #####
563| + output_change: (_, data) => recordEvent("output_change", data),
564| +
565| + // ##### Action Event #####
566| + save: () => recordEvent("save"),
567| + submit: () => recordEvent("submit"),
568| + execute: () => recordEvent("execute"),
569| );
570| +
571| + const addListener = {
572| +     editor_change: () => editor.session.on("change", handlers.editor_change),
573| +     editor_cursor: () =>
574| +         editor.session.selection.on("changeCursor", handlers.editor_cursor),
575| +     editor_selection: () =>
576| +         editor.session.selection.on("changeSelection", handlers.editor_selection),
577| +     focus: () => addEvent(window, "focus", handlers.focus),
578| +     blur: () => addEvent(window, "blur", handlers.blur),
579| +     visibility: () => addEvent(document, visibilityChange, handlers.visibility),
580| +     pdf_focus: () =>
581| +         addEvent(
582| +             $("#pdf_viewer")[0].contentWindow,
583| +             "focusin",
584| +             handlers.pdf_focus
585| +         ),
586| +     pdf.blur: () =>
587| +         addEvent(
588| +             $("#pdf_viewer")[0].contentWindow,
589| +             "focusout",
590| +             handlers.pdf.blur
591| +         ),
592| +     input_change: () => $("#editor_input").on("input", handlers.input_change),
593| +     output_change: () =>
594| +         $("textarea#editor_output").on("output_change", handlers.output_change),
595| + };
596| +
597| + const removeListener = {
598| +     editor_change: () => editor.commands.off("afterExec", handlers.editor_change),
599| +     editor_cursor: () =>
600| +         editor.selection.off("changeCursor", handlers.editor_cursor),
601| +     editor_selection: () =>
602| +         editor.selection.off("changeSelection", handlers.editor_selection),
603| +     focus: () => removeEvent(window, "focus", handlers.focus),
604| +     blur: () => removeEvent(window, "blur", handlers.blur),
605| +     visibility: () =>
606| +         removeEvent(document, visibilityChange, handlers.visibility),
607| +     pdf_focus: () =>
608| +         removeEvent(
609| +             $("#pdf_viewer")[0].contentWindow,
610| +             "focusin",
611| +             handlers.pdf_focus
612| +         ),

```

```

613 +     pdf.blur: () =>
614 +       removeEvent(
615 +         $("#pdf_viewer")[0].contentWindow,
616 +         "focusout",
617 +         handlers.pdf.blur
618 +       ),
619 +     input_change: () => $("#editor_input").off("input", handlers.input_change),
620 +     output_change: () =>
621 +       $("textarea#editor_output").off("output_change", handlers.output_change),
622 +   };
623 +
624 + // ##### Methods #####
625 + // ##### Methods #####
626 + // #####
627 +
628 + const record = {
629 +   // Code Editor
630 +   editor: () => {
631 +     // ##### Editor #####
632 +     recording.startValue = editor.getValue();
633 +     recording.startSelection = getSelection(editor);
634 +
635 +     // Exec command
636 +     addListener.editor_change();
637 +
638 +     // For Cursor
639 +     addListener.editor_cursor();
640 +     addListener.editor_selection();
641 +
642 +   },
643 +   // Overall page/tabs
644 +   web: () => {
645 +     // ##### Web Page #####
646 +
647 +     // for every type of browser.
648 +     if (hidden in document) {
649 +       visibilityChange = "visibilitychange";
650 +     } else if ((hidden = "mozHidden") in document) {
651 +       visibilityChange = "mozvisibilitychange";
652 +     } else if ((hidden = "webkitHidden") in document) {
653 +       visibilityChange = "webkitvisibilitychange";
654 +     } else if ((hidden = "msHidden") in document) {
655 +       visibilityChange = "msvisibilitychange";
656 +
657 +     if (visibilityChange != null) {
658 +       // addlistener.focus();
659 +       addListener.blur();
660 +       addListener.visibility();
661 +     } else if ("onfocusin" in document) {
662 +       // IE 9 and lower:
663 +       document.onfocusin = document.onfocusout = handlers.visibility;
664 +     } else {
665 +       // All others:
666 +       window.onpageshow =
667 +         window.onpagehide =
668 +         window.onfocus =
669 +         window.onblur =
670 +           handlers.visibility;
671 +
672 +
673 +     },
674 +     // PDF Viewer
675 +     pdf: () => {
676 +       let el = document.getElementById("pdf_viewer");
677 +
678 +       var observer = new IntersectionObserver(function () {
679 +         if (el.src != "") {
680 +           // addlistener.pdf.blur();
681 +           addListener.pdf_focus();
682 +         }
683 +
684 +         observer.observe(el, { attributes: true, childList: true });
685 +
686 +       },
687 +       // Input field
688 +       input: () => {
689 +         addListener.input_change();
690 +
691 +       },
692 +       // Output field
693 +       output: () => {
694 +         addListener.output_change();
695 +
696 +       },
697 +       // Action Button added in the onclick event listener.
698 +
699 +     };
700 +
701 +     // Methods to start recording.
702 +     const recordStart = () => {
703 +       recording.init();
704 +
705 +       Object.keys(record).forEach((evtName) => {
706 +         const inInclude = evtName in include;
707 +         if (inInclude[evtName] || (!inInclude && include["others"])) {
708 +           record[evtName]();
709 +         }
710 +       });
711 +     };
712 +
713 +     const recordStop = () => {
714 +       Object.values(removeListener).forEach((func) => {
715 +         func();
716 +       });
717 +     };
718 +
719 +     const recordStop = () => {
720 +       Object.values(removeListener).forEach((func) => {
721 +         func();
722 +       });
723 +
724 +     };
725 +
726 +     const recordStop = () => {
727 +       Object.values(removeListener).forEach((func) => {
728 +         func();
729 +       });
730 +
731 +     };
732 +
733 +     const recordStop = () => {
734 +       Object.values(removeListener).forEach((func) => {
735 +         func();
736 +       });
737 +
738 +     };
739 +
740 +     const recordStop = () => {
741 +       Object.values(removeListener).forEach((func) => {
742 +         func();
743 +       });
744 +
745 +     };
746 +
747 +     const recordStop = () => {
748 +       Object.values(removeListener).forEach((func) => {
749 +         func();
750 +       });
751 +
752 +     };
753 +
754 +     const recordStop = () => {
755 +       Object.values(removeListener).forEach((func) => {
756 +         func();
757 +       });
758 +
759 +     };
760 +
761 +     const recordStop = () => {
762 +       Object.values(removeListener).forEach((func) => {
763 +         func();
764 +       });
765 +
766 +     };
767 +
768 +     const recordStop = () => {
769 +       Object.values(removeListener).forEach((func) => {
770 +         func();
771 +       });
772 +
773 +     };
774 +
775 +     const recordStop = () => {
776 +       Object.values(removeListener).forEach((func) => {
777 +         func();
778 +       });
779 +
780 +     };
781 +
782 +     const recordStop = () => {
783 +       Object.values(removeListener).forEach((func) => {
784 +         func();
785 +       });
786 +
787 +     };
788 +
789 +     const recordStop = () => {
790 +       Object.values(removeListener).forEach((func) => {
791 +         func();
792 +       });
793 +
794 +     };
795 +
796 +     const recordStop = () => {
797 +       Object.values(removeListener).forEach((func) => {
798 +         func();
799 +       });
800 +
801 +     };
802 +
803 +     const recordStop = () => {
804 +       Object.values(removeListener).forEach((func) => {
805 +         func();
806 +       });
807 +
808 +     };
809 +
810 +     const recordStop = () => {
811 +       Object.values(removeListener).forEach((func) => {
812 +         func();
813 +       });
814 +
815 +     };
816 +
817 +     const recordStop = () => {
818 +       Object.values(removeListener).forEach((func) => {
819 +         func();
820 +       });
821 +
822 +     };
823 +
824 +     const recordStop = () => {
825 +       Object.values(removeListener).forEach((func) => {
826 +         func();
827 +       });
828 +
829 +     };
830 +
831 +     const recordStop = () => {
832 +       Object.values(removeListener).forEach((func) => {
833 +         func();
834 +       });
835 +
836 +     };
837 +
838 +     const recordStop = () => {
839 +       Object.values(removeListener).forEach((func) => {
840 +         func();
841 +       });
842 +
843 +     };
844 +
845 +     const recordStop = () => {
846 +       Object.values(removeListener).forEach((func) => {
847 +         func();
848 +       });
849 +
850 +     };
851 +
852 +     const recordStop = () => {
853 +       Object.values(removeListener).forEach((func) => {
854 +         func();
855 +       });
856 +
857 +     };
858 +
859 +     const recordStop = () => {
860 +       Object.values(removeListener).forEach((func) => {
861 +         func();
862 +       });
863 +
864 +     };
865 +
866 +     const recordStop = () => {
867 +       Object.values(removeListener).forEach((func) => {
868 +         func();
869 +       });
870 +
871 +     };
872 +
873 +     const recordStop = () => {
874 +       Object.values(removeListener).forEach((func) => {
875 +         func();
876 +       });
877 +
878 +     };
879 +
880 +     const recordStop = () => {
881 +       Object.values(removeListener).forEach((func) => {
882 +         func();
883 +       });
884 +
885 +     };
886 +
887 +     const recordStop = () => {
888 +       Object.values(removeListener).forEach((func) => {
889 +         func();
890 +       });
891 +
892 +     };
893 +
894 +     const recordStop = () => {
895 +       Object.values(removeListener).forEach((func) => {
896 +         func();
897 +       });
898 +
899 +     };
900 +
901 +     const recordStop = () => {
902 +       Object.values(removeListener).forEach((func) => {
903 +         func();
904 +       });
905 +
906 +     };
907 +
908 +     const recordStop = () => {
909 +       Object.values(removeListener).forEach((func) => {
910 +         func();
911 +       });
912 +
913 +     };
914 +
915 +     const recordStop = () => {
916 +       Object.values(removeListener).forEach((func) => {
917 +         func();
918 +       });
919 +
920 +     };
921 +
922 +     const recordStop = () => {
923 +       Object.values(removeListener).forEach((func) => {
924 +         func();
925 +       });
926 +
927 +     };
928 +
929 +     const recordStop = () => {
930 +       Object.values(removeListener).forEach((func) => {
931 +         func();
932 +       });
933 +
934 +     };
935 +
936 +     const recordStop = () => {
937 +       Object.values(removeListener).forEach((func) => {
938 +         func();
939 +       });
940 +
941 +     };
942 +
943 +     const recordStop = () => {
944 +       Object.values(removeListener).forEach((func) => {
945 +         func();
946 +       });
947 +
948 +     };
949 +
950 +     const recordStop = () => {
951 +       Object.values(removeListener).forEach((func) => {
952 +         func();
953 +       });
954 +
955 +     };
956 +
957 +     const recordStop = () => {
958 +       Object.values(removeListener).forEach((func) => {
959 +         func();
960 +       });
961 +
962 +     };
963 +
964 +     const recordStop = () => {
965 +       Object.values(removeListener).forEach((func) => {
966 +         func();
967 +       });
968 +
969 +     };
970 +
971 +     const recordStop = () => {
972 +       Object.values(removeListener).forEach((func) => {
973 +         func();
974 +       });
975 +
976 +     };
977 +
978 +     const recordStop = () => {
979 +       Object.values(removeListener).forEach((func) => {
980 +         func();
981 +       });
982 +
983 +     };
984 +
985 +     const recordStop = () => {
986 +       Object.values(removeListener).forEach((func) => {
987 +         func();
988 +       });
989 +
990 +     };
991 +
992 +     const recordStop = () => {
993 +       Object.values(removeListener).forEach((func) => {
994 +         func();
995 +       });
996 +
997 +     };
998 +
999 +     const recordStop = () => {
1000 +       Object.values(removeListener).forEach((func) => {
1001 +         func();
1002 +       });
1003 +
1004 +     };
1005 +
1006 +     const recordStop = () => {
1007 +       Object.values(removeListener).forEach((func) => {
1008 +         func();
1009 +       });
1010 +
1011 +     };
1012 +
1013 +     const recordStop = () => {
1014 +       Object.values(removeListener).forEach((func) => {
1015 +         func();
1016 +       });
1017 +
1018 +     };
1019 +
1020 +     const recordStop = () => {
1021 +       Object.values(removeListener).forEach((func) => {
1022 +         func();
1023 +       });
1024 +
1025 +     };
1026 +
1027 +     const recordStop = () => {
1028 +       Object.values(removeListener).forEach((func) => {
1029 +         func();
1030 +       });
1031 +
1032 +     };
1033 +
1034 +     const recordStop = () => {
1035 +       Object.values(removeListener).forEach((func) => {
1036 +         func();
1037 +       });
1038 +
1039 +     };
1040 +
1041 +     const recordStop = () => {
1042 +       Object.values(removeListener).forEach((func) => {
1043 +         func();
1044 +       });
1045 +
1046 +     };
1047 +
1048 +     const recordStop = () => {
1049 +       Object.values(removeListener).forEach((func) => {
1050 +         func();
1051 +       });
1052 +
1053 +     };
1054 +
1055 +     const recordStop = () => {
1056 +       Object.values(removeListener).forEach((func) => {
1057 +         func();
1058 +       });
1059 +
1060 +     };
1061 +
1062 +     const recordStop = () => {
1063 +       Object.values(removeListener).forEach((func) => {
1064 +         func();
1065 +       });
1066 +
1067 +     };
1068 +
1069 +     const recordStop = () => {
1070 +       Object.values(removeListener).forEach((func) => {
1071 +         func();
1072 +       });
1073 +
1074 +     };
1075 +
1076 +     const recordStop = () => {
1077 +       Object.values(removeListener).forEach((func) => {
1078 +         func();
1079 +       });
1080 +
1081 +     };
1082 +
1083 +     const recordStop = () => {
1084 +       Object.values(removeListener).forEach((func) => {
1085 +         func();
1086 +       });
1087 +
1088 +     };
1089 +
1090 +     const recordStop = () => {
1091 +       Object.values(removeListener).forEach((func) => {
1092 +         func();
1093 +       });
1094 +
1095 +     };
1096 +
1097 +     const recordStop = () => {
1098 +       Object.values(removeListener).forEach((func) => {
1099 +         func();
1100 +       });
1101 +
1102 +     };
1103 +
1104 +     const recordStop = () => {
1105 +       Object.values(removeListener).forEach((func) => {
1106 +         func();
1107 +       });
1108 +
1109 +     };
1110 +
1111 +     const recordStop = () => {
1112 +       Object.values(removeListener).forEach((func) => {
1113 +         func();
1114 +       });
1115 +
1116 +     };
1117 +
1118 +     const recordStop = () => {
1119 +       Object.values(removeListener).forEach((func) => {
1120 +         func();
1121 +       });
1122 +
1123 +     };
1124 +
1125 +     const recordStop = () => {
1126 +       Object.values(removeListener).forEach((func) => {
1127 +         func();
1128 +       });
1129 +
1130 +     };
1131 +
1132 +     const recordStop = () => {
1133 +       Object.values(removeListener).forEach((func) => {
1134 +         func();
1135 +       });
1136 +
1137 +     };
1138 +
1139 +     const recordStop = () => {
1140 +       Object.values(removeListener).forEach((func) => {
1141 +         func();
1142 +       });
1143 +
1144 +     };
1145 +
1146 +     const recordStop = () => {
1147 +       Object.values(removeListener).forEach((func) => {
1148 +         func();
1149 +       });
1150 +
1151 +     };
1152 +
1153 +     const recordStop = () => {
1154 +       Object.values(removeListener).forEach((func) => {
1155 +         func();
1156 +       });
1157 +
1158 +     };
1159 +
1160 +     const recordStop = () => {
1161 +       Object.values(removeListener).forEach((func) => {
1162 +         func();
1163 +       });
1164 +
1165 +     };
1166 +
1167 +     const recordStop = () => {
1168 +       Object.values(removeListener).forEach((func) => {
1169 +         func();
1170 +       });
1171 +
1172 +     };
1173 +
1174 +     const recordStop = () => {
1175 +       Object.values(removeListener).forEach((func) => {
1176 +         func();
1177 +       });
1178 +
1179 +     };
1180 +
1181 +     const recordStop = () => {
1182 +       Object.values(removeListener).forEach((func) => {
1183 +         func();
1184 +       });
1185 +
1186 +     };
1187 +
1188 +     const recordStop = () => {
1189 +       Object.values(removeListener).forEach((func) => {
1190 +         func();
1191 +       });
1192 +
1193 +     };
1194 +
1195 +     const recordStop = () => {
1196 +       Object.values(removeListener).forEach((func) => {
1197 +         func();
1198 +       });
1199 +
1200 +     };
1201 +
1202 +     const recordStop = () => {
1203 +       Object.values(removeListener).forEach((func) => {
1204 +         func();
1205 +       });
1206 +
1207 +     };
1208 +
1209 +     const recordStop = () => {
1210 +       Object.values(removeListener).forEach((func) => {
1211 +         func();
1212 +       });
1213 +
1214 +     };
1215 +
1216 +     const recordStop = () => {
1217 +       Object.values(removeListener).forEach((func) => {
1218 +         func();
1219 +       });
1220 +
1221 +     };
1222 +
1223 +     const recordStop = () => {
1224 +       Object.values(removeListener).forEach((func) => {
1225 +         func();
1226 +       });
1227 +
1228 +     };
1229 +
1230 +     const recordStop = () => {
1231 +       Object.values(removeListener).forEach((func) => {
1232 +         func();
1233 +       });
1234 +
1235 +     };
1236 +
1237 +     const recordStop = () => {
1238 +       Object.values(removeListener).forEach((func) => {
1239 +         func();
1240 +       });
1241 +
1242 +     };
1243 +
1244 +     const recordStop = () => {
1245 +       Object.values(removeListener).forEach((func) => {
1246 +         func();
1247 +       });
1248 +
1249 +     };
1250 +
1251 +     const recordStop = () => {
1252 +       Object.values(removeListener).forEach((func) => {
1253 +         func();
1254 +       });
1255 +
1256 +     };
1257 +
1258 +     const recordStop = () => {
1259 +       Object.values(removeListener).forEach((func) => {
1260 +         func();
1261 +       });
1262 +
1263 +     };
1264 +
1265 +     const recordStop = () => {
1266 +       Object.values(removeListener).forEach((func) => {
1267 +         func();
1268 +       });
1269 +
1270 +     };
1271 +
1272 +     const recordStop = () => {
1273 +       Object.values(removeListener).forEach((func) => {
1274 +         func();
1275 +       });
1276 +
1277 +     };
1278 +
1279 +     const recordStop = () => {
1280 +       Object.values(removeListener).forEach((func) => {
1281 +         func();
1282 +       });
1283 +
1284 +     };
1285 +
1286 +     const recordStop = () => {
1287 +       Object.values(removeListener).forEach((func) => {
1288 +         func();
1289 +       });
1290 +
1291 +     };
1292 +
1293 +     const recordStop = () => {
1294 +       Object.values(removeListener).forEach((func) => {
1295 +         func();
1296 +       });
1297 +
1298 +     };
1299 +
1300 +     const recordStop = () => {
1301 +       Object.values(removeListener).forEach((func) => {
1302 +         func();
1303 +       });
1304 +
1305 +     };
1306 +
1307 +     const recordStop = () => {
1308 +       Object.values(removeListener).forEach((func) => {
1309 +         func();
1310 +       });
1311 +
1312 +     };
1313 +
1314 +     const recordStop = () => {
1315 +       Object.values(removeListener).forEach((func) => {
1316 +         func();
1317 +       });
1318 +
1319 +     };
1320 +
1321 +     const recordStop = () => {
1322 +       Object.values(removeListener).forEach((func) => {
1323 +         func();
1324 +       });
1325 +
1326 +     };
1327 +
1328 +     const recordStop = () => {
1329 +       Object.values(removeListener).forEach((func) => {
1330 +         func();
1331 +       });
1332 +
1333 +     };
1334 +
1335 +     const recordStop = () => {
1336 +       Object.values(removeListener).forEach((func) => {
1337 +         func();
1338 +       });
1339 +
1340 +     };
1341 +
1342 +     const recordStop = () => {
1343 +       Object.values(removeListener).forEach((func) => {
1344 +         func();
1345 +       });
1346 +
1347 +     };
1348 +
1349 +     const recordStop = () => {
1350 +       Object.values(removeListener).forEach((func) => {
1351 +         func();
1352 +       });
1353 +
1354 +     };
1355 +
1356 +     const recordStop = () => {
1357 +       Object.values(removeListener).forEach((func) => {
1358 +         func();
1359 +       });
1360 +
1361 +     };
1362 +
1363 +     const recordStop = () => {
1364 +       Object.values(removeListener).forEach((func) => {
1365 +         func();
1366 +       });
1367 +
1368 +     };
1369 +
1370 +     const recordStop = () => {
1371 +       Object.values(removeListener).forEach((func) => {
1372 +         func();
1373 +       });
1374 +
1375 +     };
1376 +
1377 +     const recordStop = () => {
1378 +       Object.values(removeListener).forEach((func) => {
1379 +         func();
1380 +       });
1381 +
1382 +     };
1383 +
1384 +     const recordStop = () => {
1385 +       Object.values(removeListener).forEach((func) => {
1386 +         func();
1387 +       });
1388 +
1389 +     };
1390 +
1391 +     const recordStop = () => {
1392 +       Object.values(removeListener).forEach((func) => {
1393 +         func();
1394 +       });
1395 +
1396 +     };
1397 +
1398 +     const recordStop = () => {
1399 +       Object.values(removeListener).forEach((func) => {
1400 +         func();
1401 +       });
1402 +
1403 +     };
1404 +
1405 +     const recordStop = () => {
1406 +       Object.values(removeListener).forEach((func) => {
1407 +         func();
1408 +       });
1409 +
1410 +     };
1411 +
1412 +     const recordStop = () => {
1413 +       Object.values(removeListener).forEach((func) => {
1414 +         func();
1415 +       });
1416 +
1417 +     };
1418 +
1419 +     const recordStop = () => {
1420 +       Object.values(removeListener).forEach((func) => {
1421 +         func();
1422 +       });
1423 +
1424 +     };
1425 +
1426 +     const recordStop = () => {
1427 +       Object.values(removeListener).forEach((func) => {
1428 +         func();
1429 +       });
1430 +
1431 +     };
1432 +
1433 +     const recordStop = () => {
1434 +       Object.values(removeListener).forEach((func) => {
1435 +         func();
1436 +       });
1437 +
1438 +     };
1439 +
1440 +     const recordStop = () => {
1441 +       Object.values(removeListener).forEach((func) => {
1442 +         func();
1443 +       });
1444 +
1445 +     };
1446 +
1447 +     const recordStop = () => {
1448 +       Object.values(removeListener).forEach((func) => {
1449 +         func();
1450 +       });
1451 +
1452 +     };
1453 +
1454 +     const recordStop = () => {
1455 +       Object.values(removeListener).forEach((func) => {
1456 +         func();
1457 +       });
1458 +
1459 +     };
1460 +
1461 +     const recordStop = () => {
1462 +       Object.values(removeListener).forEach((func) => {
1463 +         func();
1464 +       });
1465 +
1466 +     };
1467 +
1468 +     const recordStop = () => {
1469 +       Object.values(removeListener).forEach((func) => {
1470 +         func();
1471 +       });
1472 +
1473 +     };
1474 +
1475 +     const recordStop = () => {
1476 +       Object.values(removeListener).forEach((func) => {
1477 +         func();
1478 +       });
1479 +
1480 +     };
1481 +
1482 +     const recordStop = () => {
1483 +       Object.values(removeListener).forEach((func) => {
1484 +         func();
1485 +       });
1486 +
1487 +     };
1488 +
1489 +     const recordStop = () => {
1490 +       Object.values(removeListener).forEach((func) => {
1491 +         func();
1492 +       });
1493 +
1494 +     };
1495 +
1496 +     const recordStop = () => {
1497 +       Object.values(removeListener).forEach((func) => {
1498 +         func();
1499 +       });
1500 +
1501 +     };
1502 +
1503 +     const recordStop = () => {
1504 +       Object.values(removeListener).forEach((func) => {
1505 +         func();
1506 +       });
1507 +
1508 +     };
1509 +
1510 +     const recordStop = () => {
1511 +       Object.values(removeListener).forEach((func) => {
1512 +         func();
1513 +       });
1514 +
1515 +     };
1516 +
1517 +     const recordStop = () => {
1518 +       Object.values(removeListener).forEach((func) => {
1519 +         func();
1520 +       });
1521 +
1522 +     };
1523 +
1524 +     const recordStop = () => {
1525 +       Object.values(removeListener).forEach((func) => {
1526 +         func();
1527 +       });
1528 +
1529 +     };
1530 +
1531 +     const recordStop = () => {
1532 +       Object.values(removeListener).forEach((func) => {
1533 +         func();
1534 +       });
1535 +
1536 +     };
1537 +
1538 +     const recordStop = () => {
1539 +       Object.values(removeListener).forEach((func) => {
1540 +         func();
1541 +       });
1542 +
1543 +     };
1544 +
1545 +     const recordStop = () => {
1546 +       Object.values(removeListener).forEach((func) => {
1547 +         func();
1548 +       });
1549 +
1550 +     };
1551 +
1552 +     const recordStop = () => {
1553 +       Object.values(removeListener).forEach((func) => {
1554 +         func();
1555 +       });
1556 +
1557 +     };
1558 +
1559 +     const recordStop = () => {
1560 +       Object.values(removeListener).forEach((func) => {
1561 +         func();
1562 +       });
1563 +
1564 +     };
1565 +
1566 +     const recordStop = () => {
1567 +       Object.values(removeListener).forEach((func) => {
1568 +         func();
1569 +       });
1570 +
1571 +     };
1572 +
1573 +     const recordStop = () => {
1574 +       Object.values(removeListener).forEach((func) => {
1575 +         func();
1576 +       });
1577 +
1578 +     };
1579 +
1580 +     const recordStop = () => {
1581 +       Object.values(removeListener).forEach((func) => {
1582 +         func();
1583 +       });
1584 +
1585 +     };
1586 +
1587 +     const recordStop = () => {
1588 +       Object.values(removeListener).forEach((func) => {
1589 +         func();
1590 +       });
1591 +
1592 +     };
1593 +
1594 +     const recordStop = () => {
1595 +       Object.values(removeListener).forEach((func) => {
1596 +         func();
1597 +       });
1598 +
1599 +     };
1600 +
1601 +     const recordStop = () => {
1602 +       Object.values(removeListener).forEach((func) => {
1603 +         func();
1604 +       });
1605 +
1606 +     };
1607 +
1608 +     const recordStop = () => {
1609 +       Object.values(removeListener).forEach((func) => {
1610 +         func();
1611 +       });
1612 +
1613 +     };
1614 +
1615 +     const recordStop = () => {
1616 +       Object.values(removeListener).forEach((func) => {
1617 +         func();
1618 +       });
1619 +
1620 +     };
1621 +
1622 +     const recordStop = () => {
1623 +       Object.values(removeListener).forEach((func) => {
1624 +         func();
1625 +       });
1626 +
1627 +     };
1628 +
1629 +     const recordStop = () => {
1630 +       Object.values(removeListener).forEach((func) => {
1631 +         func();
1632 +       });
1633 +
1634 +     };
1635 +
1636 +     const recordStop = () => {
1637 +       Object.values(removeListener).forEach((func) => {
1638 +         func();
1639 +       });
1640 +
1641 +     };
1642 +
1643 +     const recordStop = () => {
1644 +       Object.values(removeListener).forEach((func) => {
1645 +         func();
1646 +       });
1647 +
1648 +     };
1649 +
1650 +     const recordStop = () => {
1651 +       Object.values(removeListener).forEach((func) => {
1652 +         func();
1653 +       });
1654 +
1655 +     };
1656 +
1657 +     const recordStop = () => {
1658 +       Object.values(removeListener).forEach((func) => {
1659 +         func();
1660 +       });
1661 +
1662 +     };
1663 +
1664 +     const recordStop = () => {
1665 +       Object.values(removeListener).forEach((func) => {
1666 +         func();
1667 +       });
1668 +
1669 +     };
1670 +
1671 +     const recordStop = () => {
1672 +       Object.values(removeListener).forEach((func) => {
1673 +         func();
1674 +       });
1675 +
1676 +     };
1677 +
1678 +     const recordStop = () => {
1679 +       Object.values(removeListener).forEach((func) => {
1680 +         func();
1681 +       });
1682 +
1683 +     };
1684 +
1685 +     const recordStop = () => {
1686 +       Object.values(removeListener).forEach((func) => {
1687 +         func();
1688 +       });
1689 +
1690 +     };
1691 +
1692 +     const recordStop = () => {
1693 +       Object.values(removeListener).forEach((func) => {
1694 +         func();
1695 +       });
1696 +
1697 +     };
1698 +
1699 +     const recordStop = () => {
1700 +       Object.values(removeListener).forEach((func) => {
1701 +         func();
1702 +       });
1703 +
1704 +     };
1705 +
1706 +     const recordStop = () => {
1707 +       Object.values(removeListener).forEach((func) => {
1708 +         func();
1709 +       });
1710 +
1711 +     };
1712 +
1713 +     const recordStop = () => {
1714 +       Object.values(removeListener).forEach((func) => {
1715 +         func();
1716 +       });
1717 +
1718 +     };
1719 +
1720 +     const recordStop = () => {
1721 +       Object.values(removeListener).forEach((func) => {
1722 +         func();
1723 +       });
1724 +
1725 +     };
1726 +
1727 +     const recordStop = () => {
1728 +       Object.values(removeListener).forEach((func) => {
1729 +         func();
1730 +       });
1731 +
1732 +     };
1733 +
1734 +     const recordStop = () => {
1735 +       Object.values(removeListener).forEach((func) => {
1736 +         func();
1737 +       });
1738 +
1739 +     };
1740 +
1741 +     const recordStop = () => {
1742 +       Object.values(removeListener).forEach((func) => {
1743 +         func();
1744 +       });
1745 +
1746 +     };
1747 +
1748 +     const recordStop = () => {
1749 +       Object.values(removeListener).forEach((func) => {
1750 +         func();
1751 +       });
1752 +
1753 +     };
1754 +
1755 +     const recordStop = () => {
1756 +       Object.values(removeListener).forEach((func) => {
1757 +         func();
1758 +       });
1759 +
1760 +     };
1761 +
1762 +     const recordStop = () => {
1763 +       Object.values(removeListener).forEach((func) => {
1764 +         func();
1765 +       });
1766 +
1767 +     };
1768 +
1769 +     const recordStop = () => {
1770 +       Object.values(removeListener).forEach((func) => {
1771 +         func();
1772 +       });
1773 +
1774 +     };
1775 +
1776 +     const recordStop = () => {
1777 +       Object.values(removeListener).forEach((func) => {
1778 +         func();
1779 +       });
1780 +
1781 +     };
1782 +
1783 +     const recordStop = () => {
1784 +       Object.values(removeListener).forEach((func) => {
1785 +         func();
1786 +       });
1787 +
1788 +     };
1789 +
1790 +     const recordStop = () => {
1791 +       Object.values(removeListener).forEach((func) => {
1792 +         func();
1793 +       });
1794 +
1795 +     };
1796 +
1797 +     const recordStop = () => {
1798 +       Object.values(removeListener).forEach((func) => {
1799 +         func();
1800 +       });
1801 +
1802 +     };
1803 +
1804 +     const recordStop = () => {
1805 +       Object.values(removeListener).forEach((func) => {
1806 +         func();
1807 +       });
1808 +
1809 +     };
1810 +
1811 +     const recordStop = () => {
1812 +       Object.values(removeListener).forEach((func) => {
1813 +         func();
1814 +       });
1815 +
1816 +     };
1817 +
18
```

```

712| +      });
713| +  });
714| +
715| + // ##### Misc #####
716| + // ##### Misc #####
717| + // ##### Misc #####
718| +
719| + // Method to record listener.
720| + const recordEvent = (event, args) => {
721| +   let curTime = getCurrentTime();
722| +
723| +   recording.events.push({
724| +     time: curTime,
725| +     event,
726| +     args,
727| +   });
728| +
729| +
730| + const addEvent = (obj, evType, fn, isCapturing) => {
731| +   if (isCapturing == null) isCapturing = false;
732| +   if (obj.addEventListener) {
733| +     // Firefox
734| +     obj.addEventListener(evType, fn, isCapturing);
735| +     return true;
736| +   } else if (obj.attachEvent) {
737| +     // MSIE
738| +     var r = obj.attachEvent("on" + evType, fn);
739| +     return r;
740| +   } else {
741| +     return false;
742| +   }
743| +
744| +
745| + const removeEvent = (obj, evType, fn, isCapturing) => {
746| +   if (isCapturing == null) isCapturing = false;
747| +   if (obj.removeEventListener) {
748| +     // Firefox
749| +     obj.removeEventListener(evType, fn, isCapturing);
750| +     return true;
751| +   } else if (obj.detachEvent) {
752| +     // MSIE
753| +     var r = obj.detachEvent("on" + evType, fn);
754| +     return r;
755| +   } else {
756| +     return false;
757| +   }
758| +
759| +
760| + const getSelection = (editor) => {
761| +   var data = editor.multiSelect.toJSON();
762| +   if (!data.length) data = [data];
763| +   data = data.map(function (x) {
764| +     var a, c;
765| +     if (x.isBackwards) {
766| +       a = x.end;
767| +       c = x.start;
768| +     } else {
769| +       c = x.end;
770| +       a = x.start;
771| +     }
772| +     return Range.comparePoints(a, c)
773| +       ? [a.row, a.column, c.row, c.column]
774| +       : [a.row, a.column];
775| +   });
776| +   return data.length > 1 ? data : data[0];
777| +
778| +
779| + const getCurrentTime = () => {
780| +   return Date.now() - recording.startTime;
781| +

```

Kode A.11: side_bar.twig

```

1 diff --git a/application/views/templates/side_bar.twig b/application/views/templates/side_bar.twig
2 index ce5a0e2..857c6d3 100644
3 --- a/application/views/templates/side_bar.twig
4 +++ b/application/views/templates/side_bar.twig
5 @@ -12,18 +12,18 @@
6   </a>
7   </li>
8   {% if user.level == 3 %}
9   <li class="color-settings{{ selected=='settings' ? ' selected' }}">
10  <a href="{{ site_url('settings') }}" aria-labelledby="settings-label">
11    <i class="fa fa-gear fa-lg"></i>
12    <span class="sidebar_text" id="settings-label">Settings</span>
13  </a>
14  </li>
15  <li class="color-users{{ selected=='users' ? ' selected' }}">
16  <a href="{{ site_url('users') }}" aria-labelledby="users-label">
17    <i class="fa fa-users fa-lg"></i>
18    <span class="sidebar_text" id="users-label">Users</span>
19  </a>
20  </li>
21  <li class="color-settings{{ selected=='settings' ? ' selected' }}">
22  <a href="{{ site_url('settings') }}" aria-labelledby="settings-label">
23    <i class="fa fa-gear fa-lg"></i>
24    <span class="sidebar_text" id="settings-label">Settings</span>
25  </a>

```

```

26| +
27|     </li>
28|     <li class="color-users{{ selected=='users' ? ' selected' }}">
29|         <a href="{{ site_url('users') }}" aria-labelledby="users-label">
30|             <i class="fa fa-users fa-lg"></i>
31|             <span class="sidebar_text" id="users-label">Users</span>
32|     </li>
33|     {% endif %}
34|     <li class="color-notifications{{ selected=='notifications' ? ' selected' }}">
35|         <a href="{{ site_url('notifications') }}" aria-labelledby="notifications-label">
36|             <span class="sidebar_text" id="submit-label">Submit</span>
37|         </a>
38|     </li>
39|     {% if user.level > 1 %}
40|         <li class="color-recording{{ selected=='recording' ? ' selected' }}">
41|             <a href="{{ site_url('recording/all') }}" aria-labelledby="recording-label">
42|                 <i class="fa fa-video-camera fa-lg"></i>
43|                 <span class="sidebar_text" id="recording-label">Recording</span>
44|             </a>
45|         </li>
46|         {% endif %}
47|         <li class="color-final_submissions{{ selected=='final_submissions' ? ' selected' }}">
48|             <a href="{{ site.url('submissions/final') }}" aria-labelledby="final-submission-label">
49|                 <i class="fa fa-map-marker fa-lg"></i>
50|             <span class="sidebar_text" id="scoreboard-label">Scoreboard</span>
51|             </a>
52|         </li>
53|     {% if user.level == 3 %}
54|         <li class="color-halloffame{{ selected=='halloffame' ? ' selected' }}">
55|             <li class="color-halloffame{{ selected=='halloffame' ? ' selected' }}">
56|                 <a href="{{ site.url('halloffame') }}" aria-labelledby="hall-of-fame-label">
57|                     <i class="fa fa-list-alt fa-lg"></i>
58|                     <span class="sidebar_text" id="hall-of-fame-label">Hall of Fame</span>
59|                 </a>
60|             </li>
61|         </li>
62|     {% if user.level == 3 %}
63|         <li class="color-logs{{ selected=='logs' ? ' selected' }}">
64|             <a href="{{ site.url('logs') }}" aria-labelledby="24-hour-log-label">
65|                 <i class="fa fa-book fa-lg"></i>
66|                 <span class="sidebar_text" id="24-hour-log-label">24-hour Log</span>
67|             </a>
68|         </li>
69|     {% endif %}
70|     {% if user.level == 3 %}
71|         <li class="color-logs{{ selected=='logs' ? ' selected' }}">
72|             <a href="{{ site.url('logs') }}" aria-labelledby="24-hour-log-label">
73|                 <i class="fa fa-book fa-lg"></i>
74|                 <span class="sidebar_text" id="24-hour-log-label">24-hour Log</span>
75|             </a>
76|         </li>
77|     {% endif %}
78|     </ul>
79|     <div id="sidebar_bottom">
80|         <p>
81|             <a href="https://github.com/ifunpar/Sharif-Judge" target="_blank">&copy; SharIF Judge {{ SHJ_VERSION }}</a>
82|             <a href="https://github.com/ifunpar/Sharif-Judge" target="_blank">&copy; SharIF Judge
83|             {{ SHJ_VERSION }}</a>
84|             <a href="https://github.com/ifunpar/Sharif-Judge/tree/docs" target="_blank">Docs</a>
85|         </p>
86|         <p class="timer"></p>
87|         <div id="shj Collapse" class="pointer">
88|             <i id="collapse" class="fa fa-caret-square-o-left fa-lg"></i><span class="sidebar_text">Collapse Menu</span>
89|             <i id="collapse" class="fa fa-caret-square-o-left fa-lg"></i>
90|             <span class="sidebar_text">Collapse Menu</span>
91|         </div>
92|     </div>
93| </div>

```

Kode A.12: Submit.php

```

1 diff --git a/application/controllers/Submit.php b/application/controllers/Submit.php
2 index 8517271..d40750a 100644
3 --- a/application/controllers/Submit.php
4 +++ b/application/controllers/Submit.php
5 @@ -270,16 +270,42 @@ class Submit extends CI_Controller
6     if(!write_file($input_path, '')){
7         if(!write_file($output_path, '')){
8             if (!file_exists($file_path)){
9                 $response = json_encode(array(content=>'', message=>'No saved file'));
10            $response = json_encode(array('content'=>'', 'message'=>'No saved file'));
11        }
12    }else{
13        $file_content = file_get_contents($file_path);
14        if ($file_content === FALSE){
15            $response = json_encode(array(content=>'', message=>'Unable to load'));
16            $response = json_encode(array('content'=>'', 'message'=>'Unable to load'));
17        }
18    }else{
19        addslashes($file_content);
20        $response = json_encode(array(content=>$file_content, message=>'Loaded'));
21        $response = json_encode(array('content'=>$file_content, 'message'=>'Loaded'));
22    }
23}
24echo $response;
25}
26
27// -----

```

```

28+
29+ /**
30+ * Load recording in files from recording file
31+ */
32+ public function load_rec($problem_id) {
33+     $user_dir = rtrim($this->assignment_root, '/').'/assignment_.'.$this->user->selected_assignment['id'].'/p'.$problem_id.'/';
34+     $file_path = $user_dir.'/.RECORD_FILE_NAME.'.RECORD_FILE_EXT;
35+
36+     $this->load->helper('file');
37+     if (!file_exists($file_path)){
38+         $response = json_encode(array('content'=>'', 'message'=>'No recording file'));
39+     }
40+     else{
41+         $file_content = file_get_contents($file_path);
42+         if ($file_content === FALSE){
43+             $response = json_encode(array('content'=>'', 'message'=>'Unable to load'));
44+         }
45+         else{
46+             addslashes($file_content);
47+             $response = json_encode(array('content'=>$file_content, 'message'=>'Loaded'));
48+         }
49+     }
50+     echo $response;
51+ @@@ -295,6 +321,10 @@ class Submit extends CI_Controller
52+     $data = $_POST['code_editor'];
53+     $problem_id = $_POST['problem_id'];
54+     $language = $_POST['language'];
55+     $rec = $_POST['rec_data'];
56+
57+     $user_dir = rtrim($this->assignment_root, '/').'/assignment_.'.$this->user->selected_assignment['id'].'/p'.$problem_id.'/';
58+     $this->user->username;
59+     if (!file_exists($user_dir)){
59+ @@@ -302,47 +332,76 @@ class Submit extends CI_Controller
60+     }
61+     $file_path = $user_dir.'/.EDITOR_FILE_NAME.'.EDITOR_FILE_EXT;
62+     $input_path = $user_dir.'/.EDITOR_IN_NAME.'.EDITOR_FILE_EXT;
63+
64+     $rec_path = $user_dir.'/.RECORD_FILE_NAME.'.RECORD_FILE_EXT;
65+
66+     $this->load->helper('file');
67+     if (!write_file($file_path, $data)){
68+         $response = json_encode(array(status=>FALSE, message=>'Unable to save'));
69+     }
70+     if (!(write_file($file_path, $data) && write_file($rec_path, $rec))){
71+         $response = json_encode(array('status'=>FALSE, 'message'=>'Unable to save'));
72+         echo $response;
73+     }
74+     else{
75+         $response = json_encode(array(status=>TRUE, message=>'Saved'));
76+
77+         $response = json_encode(array('status'=>TRUE, 'message'=>'Saved'));
78+
79+         $this->load->model('recording_model');
80+         $this->recording_model->add_recording(array(
81+             'rec_id'      => 0,
82+             'username'   => $this->user->username,
83+             'assignment' => $this->user->selected_assignment['id'],
84+             'problem'    => $problem_id,
85+             'upload_at'  => shj_now_str(),
86+         ));
87+
88+         if($type === FALSE){ // If only saved
89+             echo $response;
90+         }
91+         else{
92+             else{ // If want to execute/submit
93+                 $now = shj_now();
94+                 if ( $this->queue_model->in_queue($this->user->username,$this->user->selected_assignment['id'], $this->problem['id']
95+                     '])){
96+                     $response = json_encode(array(status=>FALSE, message=>'You have already submitted for this problem. Your last
96+                     submission is still in queue.'));
97+                     $response = json_encode(array('status'=>FALSE, 'message'=>'You have already submitted for this problem. Your
97+                     last submission is still in queue.'));
98+                     echo $response;
99+                 }
100-                $response = json_encode(array(status=>FALSE, message=>'Selected assignment has been closed.'));
101+                $response = json_encode(array('status'=>FALSE, 'message'=>'Selected assignment has been closed.'));
102+                echo $response;
103+
104-                else if ($now < strtotime($this->user->selected_assignment['start_time'])){
105-                    $response = json_encode(array(status=>FALSE, message=>'Selected assignment has not started.'));
106+                    $response = json_encode(array('status'=>FALSE, 'message'=>'Selected assignment has not started.'));
107+                    echo $response;
108+
109-                else if ($now > strtotime($this->user->selected_assignment['finish_time'])+$this->user->selected_assignment['
109-                    extra_time']){
110-                    $response = json_encode(array(status=>FALSE, message=>'Selected assignment has finished.'));
111+                    $response = json_encode(array('status'=>FALSE, 'message'=>'Selected assignment has finished.'));
112+                    echo $response;
113+
114-                else if ( ! $this->assignment_model->is_participant($this->user->selected_assignment['participants'],$this->user->
114-                    username)){
115-                    $response = json_encode(array(status=>FALSE, message=>'You are not registered for submitting.'));
116+                    $response = json_encode(array('status'=>FALSE, 'message'=>'You are not registered for submitting.'));
117+                    echo $response;
118+
119+                }
119+

```

```

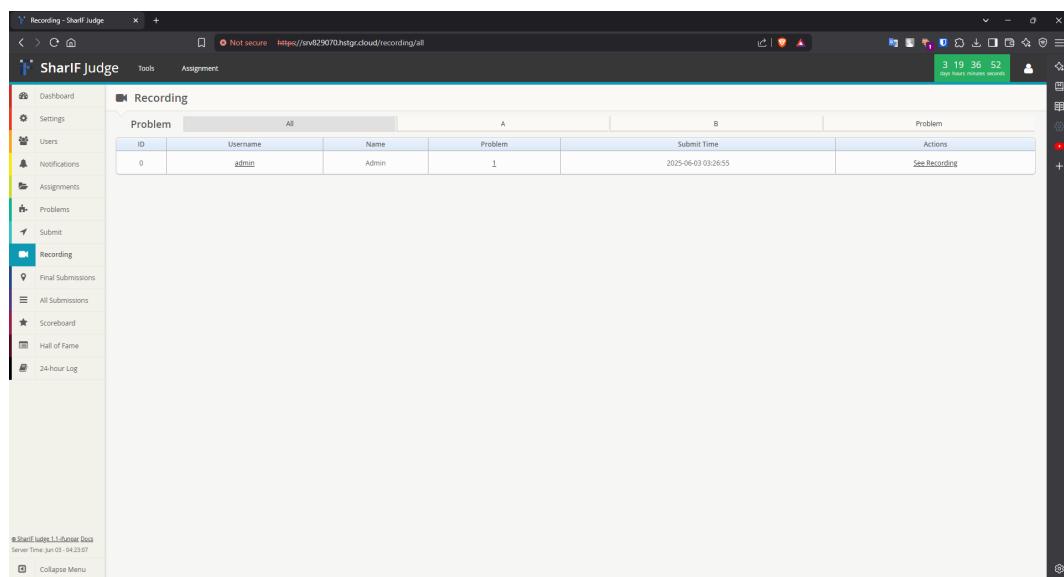
120         if($type === 'submit'){
121 -             $this->submit($data, $problem_id, $language, $user_dir);
122 +             $this->_submit($data, $problem_id, $language, $user_dir, $rec);
123     }
124     else if($type === 'execute'){
125 -         $editor_input = $_POST['editor_input'];
126 -         if (!write_file($input_path, $editor_input)){
127 -             $response = json_encode(array('status'=>FALSE, 'message'=>'Unable to write input file'));
128 +             $response = json_encode(array('status'=>FALSE, 'message'=>'Unable to write input file'));
129         }
130     }
131     else{
132 @@ -362,13 +421,19 @@ class Submit extends CI_Controller
133     /**
134      * Add code to queue for judging
135     */
136 -    private function _submit($data, $problem_id, $language, $user_dir){
137 +    private function _submit($data, $problem_id, $language, $user_dir, $rec){
138 -        $file_type = $this->language_to_type(strtolower(trim($language)));
139 -        $file_ext = $this->language_to_ext(strtolower(trim($language)));
140 -        $file_name = EDITOR_FILE_NAME;
141 -        $file_fname = $file_name.'-'.$this->user->selected_assignment['total_submits']+1;
142 -        $file_path = $user_dir.'/'.$file_fname.'.'.$file_ext;
143 +
144 +        $rec_file_name = RECORD_FILE_NAME;
145 +        $rec_file_fname = $rec_file_name.'-'.$this->user->selected_assignment['total_submits']+1;
146 +        $rec_file_path = $user_dir.'/'.$rec_file_fname.'.RECORD_FILE_EXT';
147 +
148 +        $old_file_path = $user_dir.'/'.$RECORD_FILE_NAME.'.RECORD_FILE_EXT';
149 +
150         foreach($this->problems as $item)
151             if ($item['id'] == $problem_id)
152             {
153 @@ -376,8 +441,8 @@ class Submit extends CI_Controller
154             break;
155         }
156
157         if (!write_file($file_path, $data)){
158 -             $response = json_encode(array('status'=>FALSE, 'message'=>'Unable to submit'));
159 +             if (!write_file($file_path, $data) && rename($old_file_path, $rec_file_path)){
160 +                 $response = json_encode(array('status'=>FALSE, 'message'=>'Unable to submit'));
161             }
162         else{
163             $this->load->model('submit_model');
164 @@ -394,6 +459,17 @@ class Submit extends CI_Controller
165             'pre_score' => 0,
166             'time' => shj_now_str(),
167         );
168 +
169 +            $this->load->model('recording_model');
170 +            $this->recording_model->add_recording(array(
171 +                'rec_id'      => $submit_info['submit_id'],
172 +                'username'    => $submit_info['username'],
173 +                'assignment'  => $submit_info['assignment'],
174 +                'problem'     => $submit_info['problem'],
175 +                'upload_at'   => shj_now_str(),
176 +            ));
177 +            $this->recording_model->remove_saveonly_recording($submit_info['assignment'], $submit_info['problem'], $submit_info['username']);
178 +
179             if ($this->problem['is_upload_only'] == 0)
180             {
181                 $this->queue_model->add_to_queue($submit_info);
182 @@ -404,12 +480,11 @@ class Submit extends CI_Controller
183             $this->submit_model->add_upload_only($submit_info);
184         }
185
186         $response = json_encode(array('status'=>TRUE, 'message'=>"Submitted"));
187 +         $response = json_encode(array('status'=>TRUE, 'message'=>"Submitted"));
188     }
189     echo $response;
190 }
191
192 // -----
193 /**
194 @@ -424,7 +499,7 @@ class Submit extends CI_Controller
195     $output_path = $user_dir.'/'.$EDITOR_OUT_NAME.'.EDITOR_FILE_EXT';
196
197     if (!write_file($file_path, $data)){
198 -         $response = json_encode(array('status'=>FALSE, 'message'=>'Unable to execute', 'debug'=>$file_path));
199 +         $response = json_encode(array('status'=>FALSE, 'message'=>'Unable to execute', 'debug'=>$file_path));
200     }
201     else{
202         $submit_info = array(
203 @@ -442,15 +517,15 @@ class Submit extends CI_Controller
204         if($this->queue_model->add_to_queue_exec($submit_info)){
205             if (!write_file($output_path, 'Queueing...')){
206 -                 $response = json_encode(array('status'=>FALSE, 'message'=>'Unable to write output file'));
207 +                 $response = json_encode(array('status'=>FALSE, 'message'=>'Unable to write output file'));
208             }
209             else{
210                 process_the_queue();
211                 $response = json_encode(array('status'=>TRUE, 'message'=>'Executing'));
212 +                 $response = json_encode(array('status'=>TRUE, 'message'=>'Executing'));
213             }
214         }
215     }
216 }
217

```

```
218     else{
219 -         $response = json_encode(array(status=>FALSE, message=>'Still in queue'));
220 +         $response = json_encode(array('status'=>FALSE, 'message'=>'Still in queue'));
221     }
222 }
223 echo $response;
224 @@ -467,21 +542,21 @@ class Submit extends CI_Controller
225     $file_path = $user_dir.'/'.$EDITOR_OUT_NAME.'.'.$EDITOR_FILE_EXT;
226
227     if (!file_exists($file_path)){
228 -         $response = json_encode(array(status=>FALSE, content=>''));
229 +         $response = json_encode(array('status'=>FALSE, 'content'=>''));
230     }
231     else{
232         $this->load->helper('file');
233         $file_content = file_get_contents($file_path);
234         if ($file_content === FALSE){
235 -             $response = json_encode(array(status=>FALSE, content=>''));
236 +             $response = json_encode(array('status'=>FALSE, 'content'=>''));
237         }
238     else{
239         $complete_status = strpos($file_content, 'Total Execution Time');
240         if($complete_status === FALSE){
241 -             $response = json_encode(array(status=>FALSE, content=>$file_content));
242 +             $response = json_encode(array('status'=>FALSE, 'content'=>$file_content));
243         }
244     else{
245 -         $response = json_encode(array(status=>TRUE, content=>$file_content));
246 +         $response = json_encode(array('status'=>TRUE, 'content'=>$file_content));
247     }
248 }
249 }
```

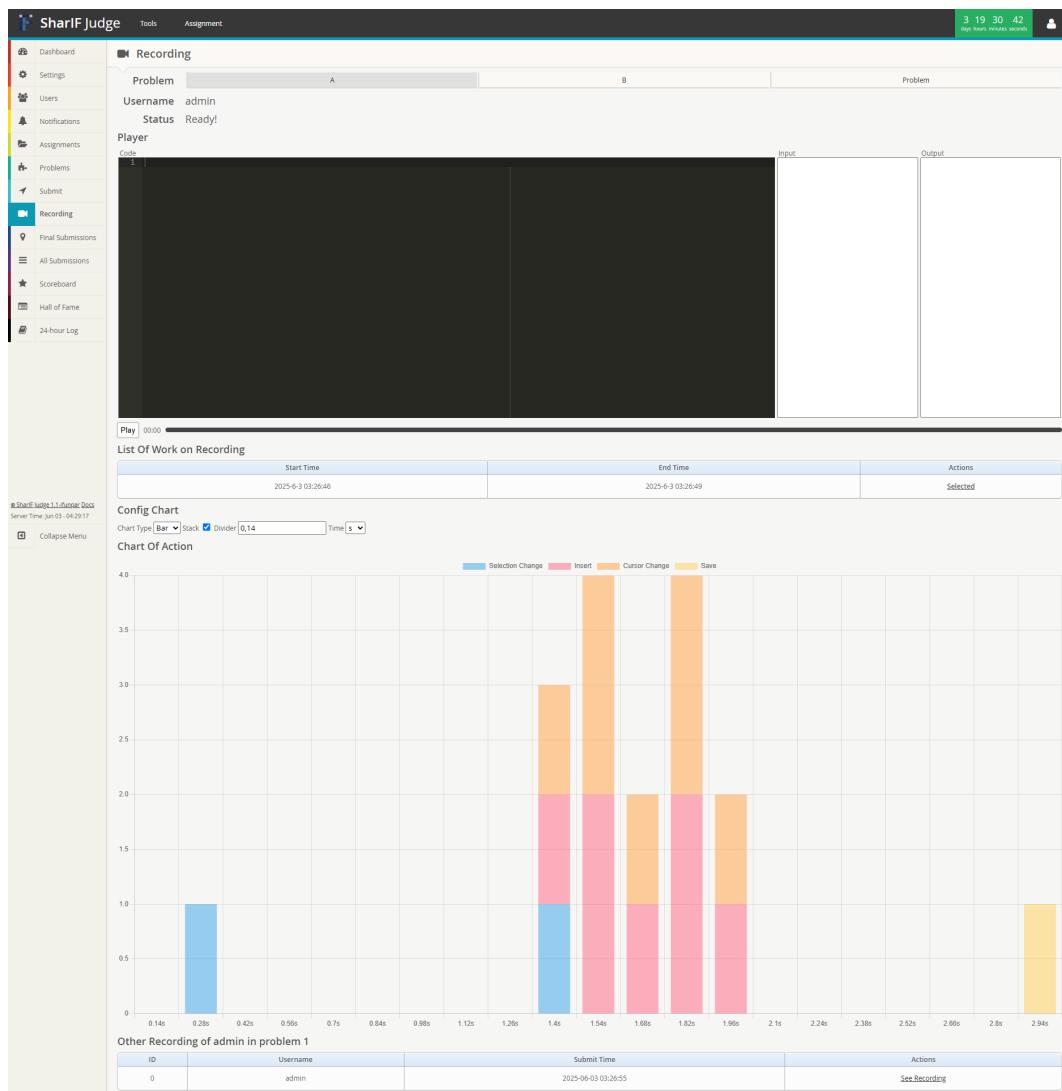
LAMPIRAN B

HASIL HALAMAN



The screenshot shows the Sharf Judge software interface. The title bar reads "Recording - Sharf Judge". The address bar shows the URL "https://svd839070.ngrok.io/recording/all" with a "Not secure" warning. The top menu includes "Tools" and "Assignment". A green status bar at the top right indicates "3 days 19 hours 56 minutes 52 seconds". The left sidebar has a navigation menu with items: Dashboard, Settings, Users, Notifications, Problems, Submit, Recording (which is selected), Final Submissions, All Submissions, Scoreboard, Hall of Fame, and 24-hour Log. The main content area is titled "Recording" and displays a table with one row of data. The table columns are: Problem (with dropdown options "All", "A", and "B"), ID, Username, Name, Problem, Submit Time, and Actions. The single entry in the table is: ID 0, Username admin, Name Admin, Problem 1, Submit Time 2025-06-03 03:26:55, and Actions "See Recording". At the bottom left, there is a note: "© Sharf Judge 1.1.5 (User) Doc Server Time: Jun 03 04:23:07".

Gambar B.1: Halaman Daftar Rekaman



Gambar B.2: Halaman Rekaman

LAMPIRAN C

FILE DOCKER EKSPERIMENT

Kode C.1: File *docker-compose* yang digunakan untuk Experiment

```
1 version: "3"
2 services:
3   codeigniter-3:
4     build: .
5     ports:
6       - "81:80"
7     volumes:
8       - ./var/www/html
9     depends_on:
10      - db
11
12   db:
13     image: mysql:5.7
14     ports:
15       - "3306:3306"
16     environment:
17       MYSQL_TABLE: judge
18       MYSQL_USER: sharif
19       MYSQL_PASSWORD: judge
20       MYSQL_ROOT_PASSWORD: root
21     command: --sql_mode=STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION
22     volumes:
23       - ./mysql:/var/lib/mysql
24
25   phpmyadmin:
26     image: phpmyadmin/phpmyadmin
27     ports:
28       - "8080:80"
29     environment:
30       PMA_HOST: db
31       MYSQL_ROOT_PASSWORD: freehost
32     depends_on:
33       - db
```

Kode C.2: File *Dockerfile* yang digunakan untuk Experiment

```
1 # Menggunakan image PHP 7.3 sebagai base image
2 FROM php:7.3-apache
3
4 # Install dependensi dan ekstensi PHP yang dibutuhkan untuk CodeIgniter
5 RUN apt-get update && apt-get install -y \
6     libpng-dev \
7     libjpeg-dev \
8     libldap2-dev \
9     libcurl4 \
10    libcurl4-openssl-dev \
11    libzip-dev \
12    libfreetype6-dev \
13    zip \
14    unzip \
15    default-jdk \
16    g++ \
17    python2 \
18    python3
19
20 # Install ekstensi GD dan mysqli
21 RUN docker-php-ext-configure gd --with-freetype-dir=/usr/include/ --with-jpeg-dir=/usr/include/ \
22     && docker-php-ext-install gd mysqli
23
24 RUN docker-php-ext-install curl
25
26 RUN docker-php-ext-configure ldap --with-libdir=lib/x86_64-linux-gnu/ \
27     && docker-php-ext-install ldap
28
29 RUN docker-php-ext-install fileinfo
30 RUN docker-php-ext-install mbstring
31 RUN docker-php-ext-install zip
32
33 RUN cp /usr/local/etc/php/php.ini-production /usr/local/etc/php/php.ini && \
34     sed -i -e "s/^memory_limit.*memory_limit=4G/g" /usr/local/etc/php/php.ini && \
35     sed -i -e "s/^max_input_vars.*max_input_vars=3000000/g" /usr/local/etc/php/php.ini && \
36     sed -i -e "s/^post_max_size.*post_max_size=50M/g" /usr/local/etc/php/php.ini && \
37     sed -i -e "s/^upload_max_filesize.*upload_max_filesize=50M/g" /usr/local/etc/php/php.ini
```

```
38
39 # Aktifkan mod_rewrite untuk Apache
40 RUN a2enmod rewrite
41
42 # Copy kode CodeIgniter ke dalam container
43 COPY . /var/www/html/
44
45 # Set direktori kerja
46 WORKDIR /var/www/html/
47
48 # Make Folder tester writeable by PHP
49 RUN chmod 777 /var/www/html/restricted/tester
50 RUN chmod 777 /var/www/html/application/cache/Twig
51
52 # Expose port 80
53 EXPOSE 80
54
55 # Jalankan Apache server
56 CMD ["apache2-foreground"]
```