

SKRIPSI

PENGEMBANGAN APLIKASI MANAJEMEN UJIAN DI LAB KOMPUTASI



Gunawan Christianto

NPM: 2016730011

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2021

UNDERGRADUATE THESIS

**DEVELOPMENT OF MANAGEMENT APPLICATION FOR
EXAMINATION IN COMPUTATIONAL LAB**



Gunawan Christianto

NPM: 2016730011

**DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
2021**

ABSTRAK

Ujian menjadi salah satu syarat yang mutlak untuk memenuhi komponen penilaian suatu mata kuliah. Ujian pada Lab Komputasi dilakukan dengan bantuan aplikasi Oxam. Aplikasi tersebut akan bertugas untuk melakukan pembuatan slot ujian, halaman pengunggahan jawaban, pengacakan daftar tempat duduk, dan *script* untuk distribusi soal. Namun fitur yang terbatas membuat Oxam menjadi tidak efektif lagi untuk memanajemen kebutuhan ujian komputasi lab.

Masalah yang terdapat pada aplikasi Oxam yang saat ini ada diantaranya dukungan terhadap nomor pokok mahasiswa (NPM) dan kode mata kuliah yang baru tidak ada. Pemindahan peserta yang masih harus dilakukan secara manual, beberapa *bug* yang sering muncul pada halaman pengumpulan jawaban peserta.

Pada penelitian ini, sebuah perangkat lunak yang mampu menyelesaikan masalah-masalah tersebut dibangun ulang. Penelitian akan melakukan survei untuk mengetahui masalah-masalah yang terjadi pada setiap pengguna. Kemudian analisis fitur dilakukan dari hasil survei tersebut. Implementasi dilakukan dalam *Framework* FatFree Framework dan *library* Reactjs.

Hasil dari pengujian yang dilakukan menyimpulkan bahwa aplikasi Oxam yang dibangun dapat menyelesaikan masalah-masalah tersebut dengan baik. Hasil pengujian fungsional juga memperlihatkan bahwa logika dan *query* yang diimplementasi dapat berjalan sesuai dengan ekspektasi.

Kata-kata kunci: Ujian, Lab Komputasi, Oxam, Reactjs, FatFree Framework, REST API

ABSTRACT

Examination is one of the absolute requirements to fulfill the assessment components of a course. Examination on computational lab are held with the help of the Oxam App. The application will be responsible of making exam slots, serving answer submission page, scrambling seat lists, and generating scripts for problem distribution. However, the limited features make Oxam no longer effective in managing the needs of examination on computational lab.

Issues with the current Oxam App consisted of the lack of support for the new student ID (NPM) and new course codes, the manual relocation of participant, and several bugs that often appear on the answer submission page.

In this research, a software that are capable of solving the problems is rebuilt. A survey will be conducted to find out any issues that occurs for each user. Then a feature analysis is concluded based on survey results. Solution is implemented in the FatFree Framework Framework and the Reactjs library.

Results from the performed tests concludes that the rebuilt Oxam are capable to solve those issues well. Functional test also performed to ensure that the logic and query is within expectancy.

Keywords: Examination, Computational Lab, Reactjs, FatFree Framework, REST API

DAFTAR ISI

DAFTAR ISI	ix
DAFTAR GAMBAR	xi
DAFTAR TABEL	xv
1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	3
1.3 Tujuan	3
1.4 Batasan Masalah	4
1.5 Metodologi	4
1.6 Sistematika Pembahasan	5
2 LANDASAN TEORI	7
2.1 Pelaksanaan Ujian	7
2.1.1 Pedoman Pelaksanaan Ujian untuk Peserta Ujian	7
2.1.2 Pedoman Pelaksanaan Ujian untuk Pengawas Ujian	7
2.1.3 Pedoman Pelaksanaan Ujian untuk Admin Lab	8
2.1.4 Panduan Ujian untuk Dosen Koordinator	9
2.2 Aplikasi Berbasis Web	10
2.2.1 Back-end	10
2.2.2 Front-end	11
2.3 <i>Library</i> dan <i>Framework</i>	11
2.3.1 Fat-free Framework	11
2.3.2 React.js	13
2.4 API	18
2.4.1 REST API	18
2.5 <i>Continuous Integration/Continuous Delivery/Deployment (CI/CD)</i>	20
2.6 Docker	21
3 ANALISIS	23
3.1 Analisis Sistem Masa Kini	23
3.1.1 Praujian	23
3.1.2 Ujian	25
3.1.3 Ujian dengan shift	28
3.1.4 Pascaujian	28
3.1.5 Aplikasi Manajemen Masa Kini	28
3.2 Analisis Kebutuhan	29
3.2.1 Dosen	29
3.2.2 Peserta	31
3.2.3 Tim Admin	33
3.2.4 Analisis Fitur Aplikasi	36

3.2.5	Timer Tidak Terintegrasi	36
3.2.6	Notifikasi	36
3.2.7	Pemindahan Tempat Duduk Peserta	36
3.2.8	Pengiriman berkas otomatis	36
3.2.9	<i>NPM Converter</i>	37
3.2.10	Admin Panel	37
3.2.11	Bug fix	37
3.3	Analisis Pemilihan Framework dan Library	37
3.3.1	FatFree Framework	37
3.3.2	React.js	38
3.3.3	CI/CD	38
3.4	Analisis Pengguna	38
3.4.1	Skenario Penggunaan	40
4	PERANCANGAN	57
4.1	Rancangan Antarmuka	57
4.1.1	Rancangan Antarmuka untuk Peserta	57
4.1.2	Rancangan Antarmuka untuk Admin	62
4.1.3	Rancangan Antarmuka untuk Dosen Pengawas	81
4.1.4	Rancangan Antarmuka Tambahan	84
4.2	Perancangan Sistem Backend	85
4.2.1	Racangan Basis Data	86
4.2.2	Rancangan REST API	95
4.2.3	Desain Kelas	101
4.2.4	<i>Namespace Service, Cronjob dan View</i>	131
4.3	Perancangan Sistem CI/CD dan Unit Testing	133
5	IMPLEMENTASI DAN PENGUJIAN	135
5.1	Implementasi	135
5.1.1	Lingkungan Implementasi Perangkat Lunak	135
5.1.2	Hasil Implementasi	138
5.1.3	Halaman untuk Peserta	138
5.1.4	Halaman untuk Tim Admin	141
5.1.5	Halaman untuk Dosen Pengawas / Layar Proyektor	151
5.1.6	Halaman Tambahan untuk Dosen Koordinator	151
5.1.7	Bentuk Struktur Direktori Penyimpanan Ujian	155
5.2	Pengujian Experimental	155
5.3	Pengujian Fungsional	157
5.3.1	Analisis Hasil Pengujian	160
6	KESIMPULAN DAN SARAN	161
6.1	Kesimpulan	161
6.2	Saran	161
DAFTAR REFERENSI	163	
A	Checklist PERSIAPAN UJIAN	165
B	KODE PROGRAM	167

DAFTAR GAMBAR

1.1	Tampilan cuplikan layar dari Oxam, aplikasi manajemen ujian di Lab Komputasi.	2
1.2	Daftar peserta yang <i>digenerate</i> oleh Oxam dalam bentuk berkas.	3
1.3	Struktur folder jawaban pada sistem Oxam.	4
2.1	Ilustrasi sistem <i>backend</i> dan <i>frontend</i> pada pengembangan aplikasi berbasis web. .	10
20figure.caption.16		
21figure.caption.17		
2.4	Ilustrasi perbandingan lapisan sistem pada Docker (kiri) dan <i>Virtual Machine</i> (kanan).	22
3.1	Diagram alur pelaksanaan ujian secara garis besar.	23
3.2	Diagram alur detil persiapan ujian.	24
3.3	Diagram alur ujian dengan tanpa <i>shift</i>	26
3.4	Diagram alur ujian dengan <i>shift</i>	27
3.5	Kuisisioner Dosen.	30
3.6	Respon kuisisioner untuk masalah yang telah diketahui pada saat survei.	32
3.7	Pendapat peserta ujian pada sistem ujian yang berjalan saat ini.	33
3.8	Hasil <i>benchmark</i> terhadap beberapa <i>framework</i> pada bahasa pemrograman PHP[1]. (Lebih tinggi lebih baik)	38
3.9	Diagram <i>Use Case</i> untuk aplikasi Oxam yang baru.	39
4.1	Diagram alur antarmuka secara keseluruhan.	58
4.2	Rancangan antarmuka untuk peserta, saat ujian sedang tidak berjalan.	59
4.3	Rancangan antarmuka untuk peserta, saat ujian akan berjalan.	59
4.4	Rancangan antarmuka untuk peserta, saat ujian sedang berjalan.	60
4.5	Rancangan antarmuka keseluruhan untuk notifikasi peserta. (A) Daftar notifikasi; (B) Notifikasi terbuka.	61
4.6	Rancangan tampilan untuk pengumpulan jawaban dengan nama berkas yang tidak sesuai.	62
4.7	Rancangan antarmuka untuk halaman otentikasi untuk Admin.	63
4.8	Rancangan antarmuka untuk daftar ujian pada halaman Admin Panel.	64
4.9	Rancangan antarmuka untuk modal konfirmasi penghapusan ujian.	64
4.10	Rancangan antarmuka untuk membuat ujian baru, langkah pertama dari empat. .	65
4.11	Rancangan antarmuka untuk membuat ujian baru, langkah kedua dari empat. .	67
4.12	Rancangan antarmuka untuk membuat ujian baru, langkah ketiga dari empat. .	67
4.13	Rancangan antarmuka untuk membuat ujian baru, langkah keempat dari empat. .	68
4.14	Rancangan antarmuka untuk tampilan detil ujian.	69
4.15	Rancangan antarmuka untuk modal slot jawaban baru.	70
4.16	Rancangan antarmuka untuk modal pelaporan otomatis. (A) Buat baru. (B) Hapus.	70
4.17	Rancangan antarmuka untuk beberapa tampilan modal jenis notifikasi. (A) Modal jenis notifikasi; (B) Notifikasi Kata Sandi; (C) Notifikasi Lainnya.	73
4.18	Rancangan antarmuka untuk modal konfirmasi penghapusan notifikasi.	74
4.19	Rancangan tampilan untuk daftar hadir peserta ujian.	75
4.20	Rancangan antarmuka untuk tampilan pada layar proyektor.	76

4.21 Rancangan tampilan untuk pemindahan peserta. (A) Daftar pemindahan; (B) Pencari peserta target; (C) Pencari komputer target; (D) Konfirmasi dan pengunduhan <i>script</i> pemindahan.	77
4.22 Rancangan antarmuka untuk Minipanel.	78
4.23 Rancangan antarmuka untuk daftar entri entitas.	79
4.24 Rancangan antarmuka untuk <i>editor</i> entri. (A) Buat baru; (B) Ubah yang sudah ada.	80
4.25 Rancangan antarmuka untuk menghapus entri.	80
4.26 Rancangan antarmuka untuk otentikasi, dengan menekankan bagian tertentu.	81
4.27 Rancangan antarmuka untuk halaman Peta Ruangan Ujian.	82
4.28 Rancangan antarmuka untuk halaman Timer.	83
4.29 Rancangan antarmuka untuk <i>Overtime</i> Ujian.	83
4.30 Rancangan antarmuka untuk email laporan ujian.	85
4.31 Rancangan antarmuka untuk halaman pengunduhan berkas jawaban ujian.	86
4.32 Diagram <i>ERD</i> untuk sistem aplikasi yang baru.	87
4.33 Potongan diagram entitas untuk <i>ACL</i> dan <i>ACLItem</i>	88
4.34 Potongan diagram entitas untuk <i>IPLLogin</i>	90
4.35 Potongan diagram entitas untuk <i>Location</i> dan <i>Computer</i>	91
4.36 Potongan diagram entitas untuk <i>Lecture</i> dan <i>LecturePeriod</i>	92
4.37 Potongan diagram entitas untuk <i>Participant</i>	93
4.38 Potongan diagram entitas untuk <i>Exam</i>	94
4.39 Potongan diagram relasi entitas untuk <i>Submission</i>	95
4.40 Gambaran besar dari perancangan kelas untuk sistem <i>back-end</i>	101
4.41 Diagram kelas untuk <i>namespace Controller</i>	102
4.42 Potongan diagram kelas untuk <i>namespace Controller/Api</i>	104
4.43 Potongan diagram kelas untuk <i>namespace Controller/Api/Manage</i>	108
4.44 Potongan diagram kelas untuk <i>namespace Model</i>	113
4.45 Potongan diagram kelas untuk <i>namespace Model/ujian</i>	118
4.46 Diagram kelas untuk <i>namespace Helper</i>	128
4.47 Diagram kelas untuk <i>namespace Output</i>	130
4.48 Diagram kelas untuk <i>namespace Service, Cronjob</i> dan <i>View</i>	134
5.1 Tangkapan layar dari halaman ujian, dengan ujian yang tidak aktif.	139
5.2 Tangkapan layar dari halaman ujian, dengan ujian yang akan aktif.	139
5.3 Tangkapan layar dari halaman ujian, dengan ujian yang sedang aktif.	139
5.4 Tangkapan layar dari halaman ujian untuk peringatan nama berkas yang tidak sesuai.	140
5.5 Tangkapan layar dari halaman ujian, bagian notifikasi.	140
5.6 Tangkapan layar halaman otentikasi	141
5.7 Tangkapan layar daftar ujian untuk admin.	142
5.8 Tangkapan layar konfirmasi penghapusan ujian.	142
5.9 Tangkapan layar untuk membuat ujian, langkah pertama.	143
5.10 Tangkapan layar untuk membuat ujian, langkah kedua.	144
5.11 Tangkapan layar untuk membuat ujian, langkah ketiga.	144
5.12 Tangkapan layar untuk membuat ujian, langkah keempat.	145
5.13 Tangkapan layar untuk detil ujian	145
5.14 Tangkapan layar absensi untuk ditempel pada pintu.	146
5.15 Tangkapan layar absensi untuk tanda tangan peserta.	146
5.16 Tangkapan layar panel perangkat bergerak.	147
5.17 Tangkapan layar untuk pemindah peserta.	147
5.18 Tangkapan layar untuk pemindah peserta, langkah memilih peserta.	148
5.19 Tangkapan layar untuk pemindah peserta, langkah memilih komputer.	148
5.20 Tangkapan layar untuk pemindah peserta, langkah mengunduh <i>script</i>	149
5.21 Tangkapan layar untuk layar proyektor Admin.	149

5.22 Tangkapan layar untuk slot jawaban. (A) untuk menambahkan slot jawaban, (B) untuk menghapus slot jawaban.	150
5.23 Tangkapan layar untuk fitur notifikasi. (A) modal pemilihan jenis. (B) modal notifikasi kata sandi. (C) modal untuk notifikasi lainnya.	150
5.24 Tangkapan layar untuk konfirmasi penghapusan notifikasi.	151
5.25 Tangkapan layar untuk daftar entri pada entitas.	151
5.26 Tangkapan layar untuk <i>editor</i> entri	152
5.27 Tangkapan layar untuk konfirmasi penghapusan entri.	152
5.28 Tangkapan layar untuk tampilan denah tempat duduk ruangan ujian.	153
5.29 Tangkapan layar untuk tampilan timer untuk ditampilkan pada proyektor.	153
5.30 Tangkapan layar untuk tampilan untuk modal penambahan waktu lebih.	154
5.31 Tangkapan layar untuk tampilan email yang akan diterima oleh dosen.	154
5.32 Tangkapan layar untuk tampilan halaman pengunduhan	155
5.33 Tampilan daftar direktori pada sistem backend aplikasi Oxam.	156
5.34 Hasil kuisioner pasca demo pada pertanyaan berbentuk skala.	157

DAFTAR TABEL

1.1	NPM lama dan NPM baru yang sistem informasi gunakan.	1
2.1	Tabel contoh representasi <i>method</i> pada protokol HTTP.	18
3.1	NPM lama, baru dan username yang mahasiswa gunakan untuk login.	34
3.2	Tabel deskripsi lengkap <i>use case</i> untuk setiap peran.	40
4.1	Tabel representasi nilai biner pada kolom <code>permission</code> di entitas <code>ACLItem</code>	88
4.2	Definisi tipe semester dengan nilainya untuk kolom <code>period_code</code> pada entitas <code>LecturePeriod</code>	92
5.4	Rata-rata nilai kepuasan berdasarkan kuisioner yang telah dilakukan.	160

¹

BAB 1

²

PENDAHULUAN

³ 1.1 Latar Belakang

⁴ Ujian menjadi salah satu syarat yang mutlak untuk memenuhi komponen penilaian suatu mata
⁵ kuliah. Salah satu bentuk ujian tersebut dilakukan secara praktik. Ujian praktik ini dilaksanakan
⁶ pada lab komputasi dengan bantuan aplikasi. Pihak yang bertanggung jawab untuk mempersiapkan
⁷ ruangan dan sistem adalah *System Administrator* atau Admin. Peserta akan diberi soal ujian
⁸ melalui sistem yang berjalan di lab sesuai prosedur dan aturan yang berlaku.

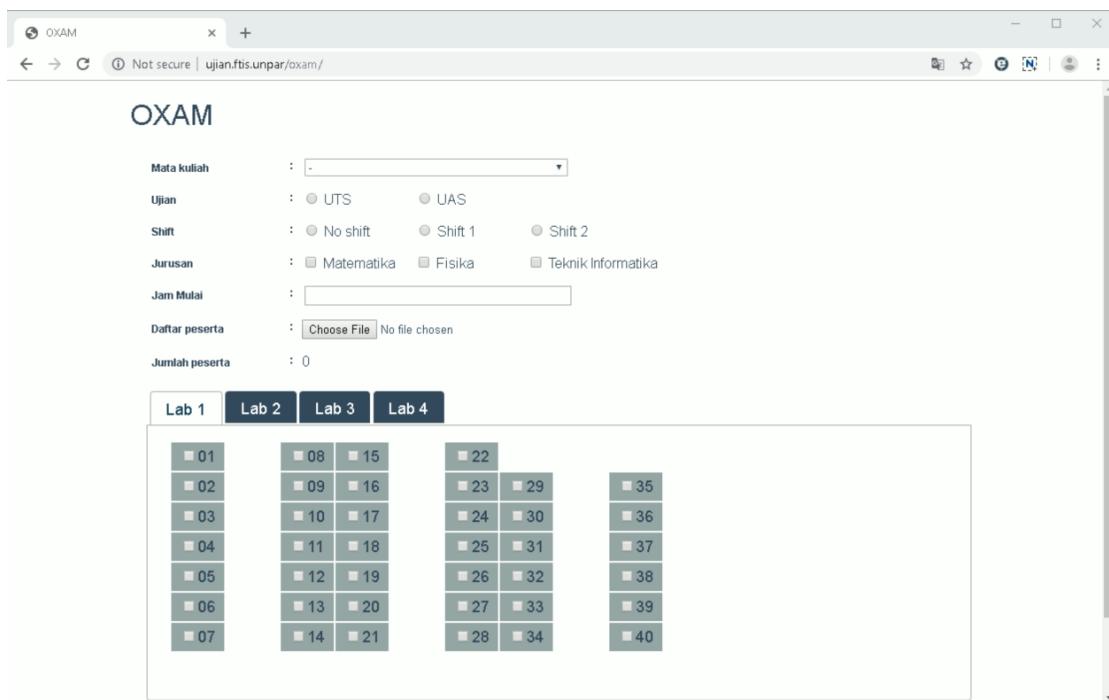
⁹ Ujian pada Lab Komputasi dilakukan dengan bantuan aplikasi. Aplikasi tersebut membantu
¹⁰ mengatur berbagai kebutuhan seperti pengumpulan jawaban, pengacakan daftar peserta, serta
¹¹ pengarsipan berkas jawaban. Aplikasi yang saat ini digunakan bernama Oxam (Gambar 1.1).
¹² Oxam bekerja dengan meminta parameter berupa kode matakuliah, tipe ujian, jurusan, jam mulai
¹³ ujian, daftar peserta, *slot* tempat duduk yang dapat digunakan, dan daftar nama berkas yang akan
¹⁴ dikumpulkan. Aplikasi Oxam akan secara otomatis membuatkan daftar tempat duduk peserta yang
¹⁵ sudah teracak, dan membuatkan *script* untuk menyalin berkas ujian ke komputer peserta.

¹⁶ Namun fitur yang terbatas membuat Oxam menjadi tidak efektif untuk menyelesaikan insiden-
¹⁷ insiden khusus. Salah satu masalah yang sering dihadapi adalah pemindahan posisi peserta ke
¹⁸ meja lain saat masalah terjadi. Admin harus mengubah secara manual entri pada basis data yang
¹⁹ bersangkutan, lalu memindahkan berkas ujian tersebut secara manual ke posisi yang baru. Selain
²⁰ itu dengan berubahnya NPM (Nomor Pokok Mahasiswa) untuk angkatan 2018 dan selanjutnya
²¹ membuat sistem Oxam yang lama tidak dapat digunakan tanpa harus mengubah NPM tersebut ke
²² bentuk yang lama. Perubahan tersebut dapat dilihat pada Tabel 1.1.

NPM Lama	NPM Baru
2016730011	6181601011

Tabel 1.1: NPM lama dan NPM baru yang sistem informasi gunakan.

²³ Karena sistem Oxam tertegragi dengan layanan server lain, maka NPM harus distandarisasi
²⁴ dengan memetakan NPM ke username. Pemetaan NPM menjadi username ini menjadi bermasalah
²⁵ karena perbedaan struktur NPM yang berbeda. Perbedaan ini meliputi seperti, nomor kode jurusan
²⁶ (Informatika adalah 73, saat ini menjadi 618), lalu posisi tahun yang berpindah dan adanya kode
²⁷ reguler (01) dan non-reguler pada depan nomor urut. Perbedaan ini membuat sistem lama tidak



Gambar 1.1: Tampilan cuplikan layar dari Oxam, aplikasi manajemen ujian di Lab Komputasi.

- 1 dapat memetakan NPM baru ke username yang biasanya digunakan oleh sistem yang sudah ada di lab komputasi saat ini.
- 2 Selain itu runtutan kegiatan yang dilakukan pada saat persiapan ujian pada Lab Komputasi dengan aplikasi ini terlalu banyak. Berdasarkan pengalaman, hal ini menimbulkan beberapa *human error* sebagai berikut:
 - Berkas daftar duduk peserta yang tertimpa oleh sesi ujian berikutnya. Berkas daftar tempat duduk dibuat menjadi berkas HTML yang harus dicetak. Daftar berkas tersebut dapat dilihat pada Gambar 1.2.
 - Jika Admin lupa mencetak atau menyalin berkas tersebut ke komputer lokal, Admin tersebut diharuskan untuk menghapus entri ujian tersebut, lalu mendaftarkan ulang sesi ujian tersebut beserta dengan daftar peserta dan daftar tempat duduk yang digunakan.
 - Jika Admin melakukan *copy* dengan urutan yang salah, *folder* untuk ujian tidak akan terbuat, atau bahkan tidak dapat diakses oleh peserta.
 - Salah memasukkan daftar peserta ujian.
 - Menghapus folder berkas ujian yang lama pada server. Jika petugas tersebut lupa, maka konsekuensinya adalah pada saat pengumpulan, Admin yang bertugas harus memisahkan berkas ujian lama dan yang baru secara manual.
- 3 Masalah berikutnya muncul pada saat proses ujian tersebut berjalan. Pertama, terdapat *bug* waktu ujian telah habis, pada kenyataannya waktu ujian belum habis. Kedua, *timer* yang digunakan untuk menunjukkan sisa waktu ujian tidak tersinkronisasi dengan Oxam. Sehingga pada saat timer berbunyi, tempat pengumpulan tidak langsung tertutup. Ketiga, entri ujian yang sudah dihapus

```
hayashi@leo: /var/www/ujian-global/oxam
File Edit View Search Terminal Help
hayashi@leo:/var/www/ujian-global/oxam$ tree result/
result/
├── cp.bat
├── lab01(tempel).html
├── lab01(ttd).html
├── lab02(tempel).html
├── lab02(ttd).html
├── lab03(tempel).html
├── lab03(ttd).html
├── lab04(tempel).html
└── lab04(ttd).html
    └── mkdir.bat
        └── takeown.bat
0 directories, 11 files
```

Gambar 1.2: Daftar peserta yang digenerate oleh Oxam dalam bentuk berkas.

1 masih muncul pada tempat pengumpulan. Hal ini biasanya diatasi oleh tim admin dengan cara
2 mengubah tanggal sesinya ke tahun lalu.

3 Pada fase pengumpulan berkas jawaban ujian ke dosen koordinator, sistem tidak secara otomatis
4 mengumpulkan berkas tersebut. Sehingga seringkali Admin yang bertugas lupa untuk mengirimkan
5 berkas tersebut. Pengumpulan berkas tersebut seharusnya dikirimkan sesegera mungkin saat ujian
6 sudah selesai. Hal ini dimaksudkan agar jawaban tidak diubah di kemudian hari tanpa izin.

7 Selain masalah-masalah pada tiap fase tersebut, masalah lain ada pada sistem itu sendiri. Oxam
8 menyimpan berkas tanpa mengacak lokasi atau nama berkas jawaban tersebut, diperlihatkan pada
9 Gambar 1.3. Hal ini dapat mempermudah penyerang sistem untuk mengubah berkas jawaban
10 tersebut.

11 Pada penelitian ini, akan dibangun ulang aplikasi baru untuk menyelesaikan masalah-masalah
12 yang muncul pada aplikasi lama dengan menggunakan *framework Fat-free* dan *React*.

13 1.2 Rumusan Masalah

14 Pada skripsi ini, aplikasi akan membantu memecahkan masalah:

- 15 • Apa saja kebutuhan aplikasi untuk ujian untuk sistem manajemen ujian di lab komputasi?
16 • Bagaimana implementasi pemenuhan kebutuhan aplikasi sistem manajemen ujian di lab
17 komputasi?

18 1.3 Tujuan

19 Tujuan dari skripsi ini adalah sebagai berikut:

- 20 • Melakukan survei dan analisis untuk mendapatkan daftar kebutuhan aplikasi ujian untuk
21 aplikasi pendukung manajemen ujian di lab komputasi.

```
hayashi@leo:/var/www/ujian-global$ tree jawaban/
jawaban/
|-- uas
    |-- AIF182204
    |   |-- i12078
    |       |-- uas12078.zip
    |-- i13056
    |   |-- uas13056.zip
    |-- i13063
    |   |-- uas13063.zip
    |-- i14042
    |   |-- uas14042.zip
    |-- i14065
    |   |-- uas14065.zip
    |-- i15015
    |   |-- uas15015.zip
    |-- i15055
    |   |-- uas15055.zip
    |-- i15065
    |   |-- uas15065.zip
    |-- i16016
    |   |-- uas16016.zip
    |-- i16021
    |   |-- uas16021.zip
    |-- i16027
    |   |-- uas16027.zip
    |-- i16035
    |   |-- uas16035.zip
    |-- i16040
    |   |-- uas16040.zip
    |-- i16042
    |   |-- uas16042.zip
    |-- i16044
    |   |-- uas16044.zip
    |-- i16049
    |   |-- uas16049.zip
```

Gambar 1.3: Struktur folder jawaban pada sistem Oxam.

- Pemenuhan kebutuhan diimplementasi dengan membuat ulang perangkat lunak dengan menggunakan *framework*, dengan harapan dapat terus dipelihara oleh tim admin di kemudian hari.

1.4 Batasan Masalah

- Batasan masalah pada penelitian ini adalah sebagai berikut:
1. Aplikasi pendukung ujian akan berjalan pada server berbasis Linux, sehingga dibutuhkannya bantuan untuk mengeksekusi *script batch* pada sistem operasi Windows.
 2. *Script* yang dihasilkan hanya mendukung untuk sistem operasi Windows.

1.5 Metodologi

- Metodologi yang dilakukan pada penelitian ini adalah sebagai berikut:
1. Studi literatur bahasa dan *framework* Fat-free dan *library* React.js.
 2. Melakukan survei sistem dan menyebarkan kuisioner mengenai sistem ujian yang berjalan pada lab.

- 1 3. Melakukan perancangan ERD basis data aplikasi Oxam.
- 2 4. Melakukan perancangan tampilan antarmuka aplikasi Oxam.
- 3 5. Mengimplementasi modul/entitas berikut:
 - 4 • Mata Kuliah
 - 5 • Peserta Ujian
 - 6 • Ujian (Sesi Ujian)
 - 7 • Print Daftar Peserta
 - 8 • Pengumuman untuk Peserta
- 9 6. Implementasi Tampilan untuk Peserta dengan detil:
 - 10 • Tempat Pengumpulan
 - 11 • *Sumary* Ujian
 - 12 • Bagian informasi/notifikasi
- 13 7. Implementasi Admin Panel untuk Admin.
- 14 8. Melakukan *deployment* dan pengujian pada fungsionalitas aplikasi.
- 15 9. Implementasi pengiriman berkas jawaban ujian secara otomatis ke dosen bersangkutan.
- 16 10. Menarik kesimpulan dan saran berdasarkan proses penelitian dan pengujian.

17 **1.6 Sistematika Pembahasan**

- 18 Pembahasan penelitian akan dilakukan secara sistematis dengan detail sebagai berikut:
- 19 • Bab 1 Pendahuluan
20 Berisi latar belakang dibuatnya penelitian aplikasi manajemen ujian di lab komputasi, rumusan
21 masalah, tujuan, batasan masalah, metodologi serta sistematika pembahasan penelitian ini.
 - 22 • Bab 2 Landasan Teori
23 Bab ini berisi Pedoman Pelaksanaan Ujian di Lab Komputasi, landasan teori dari Aplikasi
24 Berbasis Web, *Framework* dan *Library*, REST API, CI/CD serta Docker yang akan menjadi
25 landasaan untuk membantu analisis penelitian aplikasi manajemen ujian di lab komputasi.
 - 26 • Bab 3 Analisis
27 Berisi pembahasan analisa sistem ujian masa kini pada lab komputasi, pelaksanaan ujian,
28 analisa kebutuhan dan fitur aplikasi Oxam berdasarkan kuisioner, pemilihan *framework* dan
29 *library*, serta Analisis pengguna.
 - 30 • Bab 4 Perancangan
31 Pada bab ini akan dijabarkan tentang perancangan aplikasi yang akan diimplementasi untuk
32 membantu memanajemen ujian di lab komputasi. Perancangan tersebut akan terdiri dari
33 perancangan tampilan antar muka untuk peserta, admin, layar proyektor dan lembar jawab.
34 Lalu perancangan dilakukan juga untuk basis data, API, juga sistem CI/CD.

1 • Bab 5 Implementasi dan Pengujian

2 Berisi pembahasan implementasi aplikasi yang telah dirancang dan pengujian aplikasi tersebut.

3 Pengujian akan terdiri dari pengujian eksperimental dan fungsional.

4 • Bab 6 Kesimpulan dan Saran

5 Berisi kesimpulan dan saran dari penelitian aplikasi manajemen ujian di lab komputasi
6 berdasarkan perancangan, implementasi dan pengujian yang telah dilakukan.

1

BAB 2

2

LANDASAN TEORI

3 2.1 Pelaksanaan Ujian

4 Pelaksanaan ujian dimulai dengan memperhatikan *Standard Operating Procedure* (SOP) ujian di
5 lab[2]. Standar prosedur ini dibuat oleh Kepala Lab yang kemudian diikuti oleh beberapa pihak.
6 Pembahasan pedoman pelaksanaan ujian ini akan dimulai dengan membahas pedoman untuk peserta
7 ujian, Pengawas, Admin Lab, dan dosen koordinator.

8 2.1.1 Pedoman Pelaksanaan Ujian untuk Peserta Ujian

9 Pedoman pelaksanaan ujian untuk peserta dimulai beberapa hari sebelum ujian dimulai. Peserta
10 diharuskan memperhatikan pengumuman dan jadwal *shift* yang diterbitkan oleh Kepala Lab
11 Komputasi (KLK). Peserta diharuskan untuk segera melaporkan jika jadwal yang diterbitkan
12 memiliki masalah dengan jadwal ujian yang dimiliki oleh peserta, paling lambat satu hari sebelum
13 masa ujian dimulai. Jika peserta gagal melaporkan masalah yang ada sebelum masa ujian dimulai,
14 maka konsekuensinya peserta akan mengikuti jadwal ujian yang telah diterbitkan sebelumnya.

15 Menjelang ujian, peserta diharuskan mempersiapkan perlengkapan ujian, dan datang dua puluh
16 menit sebelum ujian dimulai. Posisi tempat duduk mahasiswa akan mengikuti posisi tempat duduk
17 yang diterbitkan oleh Tim Admin Lab. Peserta yang akan memasuki wilayah lab diharuskan
18 menunjukkan kartu tanda mahasiswa (KTM) pada petugas admin dan lab. Setelah masuk ke dalam
19 lab, peserta diharuskan menunggu aba-aba dari Pengawas untuk dapat memasuki ruangan ujian.
20 Peserta yang akan masuk hanya diperbolehkan membawa peralatan ujian yang izinkan saja.

21 Setelah peserta memasuki ruang ujian, peserta dipersilahkan untuk mempersiapkan tempat
22 mengerjakan ujian. Persiapan tersebut meliputi melakukan login pada komputer yang ditugaskan,
23 membuka aplikasi pendukung dan melakukan tes cepat terhadap aplikasi tersebut. Peserta
24 diharapkan untuk melaporkan masalah yang terjadi. Peserta diharuskan melakukan pengumpulan
25 jawaban pada sistem Oxam atau *judge* ujian. Lembar jawaban berupa fisik akan dikumpulkan
26 ke Pengawas ujian. Peserta yang sudah selesai mengerjakan ujian diperbolehkan meninggalkan
27 ruangan.

28 2.1.2 Pedoman Pelaksanaan Ujian untuk Pengawas Ujian

29 Pengawas ujian akan memulai persiapan ujian beberapa saat sebelum jadwal ujian dimulai. Pengawas
30 diharapkan datang 15 menit sebelum jadwal ujian. Selain itu, Pengawas ujian diharuskan untuk
31 mengambil berkas soal ujian di ruang tata usaha, jika terdapat soal ujian dalam bentuk fisik.

1 Sebelum ujian dimulai, Pengawas melakukan koordinasi dengan Koordinator untuk memastikan
2 kebutuhan ujian seperti pembagian soal, kertas buram, kata sandi soal, ralat soal, dan sebagainya.
3 Selain itu, Pengawas diharuskan untuk memastikan seluruh peserta memiliki berkas ujian. Setiap
4 masalah yang ada akan dibantu bersama Tim Admin. Jika dirasa perlu, pengawas dapat memberikan
5 penerangan pada peserta atas aturan ujian, *folder* penggerjaan peserta, tempat pengumpulan jawaban
6 dan kebutuhan lainnya. Pengawas dapat memulai ujian dengan memulai timer yang terdapat pada
7 layar proyektor.

8 Saat ujian berlangsung, pengawas diharuskan untuk mengedarkan daftar hadir untuk ditandata-
9 ngani mahasiswa dan memeriksa KTM. Setiap kendala teknis yang terjadi dapat dilaporkan pada
10 Tim Admin. Tim admin akan berada pada ruangan pada lima menit pertama. Setelah lima menit
11 tersebut Tim Admin akan berada pada ruang admin.

12 Ujian akan berakhir pada saat timer di layar telah habis. Pengawas berwenang untuk memu-
13 tusk an solusi yang diambil jika terjadi masalah pada pengumpulan jawaban. Jika ujian hanya
14 memiliki satu *shift*, maka Pengawas dapat mengizinkan peserta untuk meninggalkan ruangan saat
15 peserta selesai mengerjakan atau waktu ujian telah habis. Jika ujian terdiri lebih dari satu *shift*,
16 maka Pengawas dilarang untuk mengizinkan peserta keluar dari ruangan ujian tanpa koordinasi
17 dengan Tim Admin.

18 Setelah ujian selesai Pengawas diharuskan untuk mengisi Berita Acara, daftar hadir peserta,
19 berkas tertulis (jika ada) lalu dikumpulkan pada ruangan Tata Usaha.

20 2.1.3 Pedoman Pelaksanaan Ujian untuk Admin Lab

21 Pedoman pelaksanaan dimulai dengan persiapan pelaksanaan. Tim Admin yang bertugas diharuskan
22 mengikuti jadwal khusus. Jadwal khusus tersebut terdiri dari:

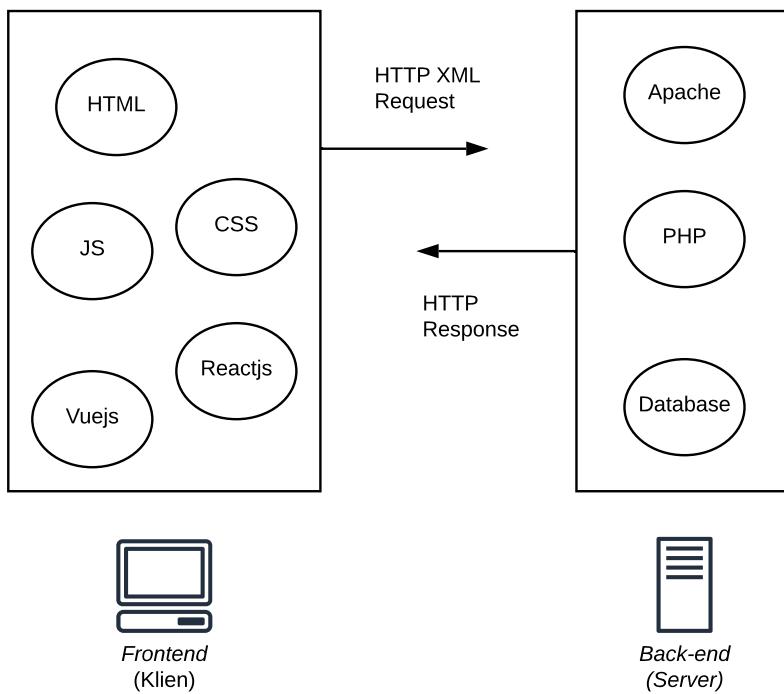
- 23 • 45 Menit sebelum ujian Tim Admin diharuskan sudah berada di lab.
- 24 • Maksimal 30 menit sebelum ujian dimulai, berkas ujian yang dibutuhkan sudah harus disiapkan.
25 Berkas tersebut termasuk *script*, absensi dan posisi tempat duduk ujian.
- 26 • 30 Menit sebelum ujian, posisi tempat duduk ditempelkan pada pintu utama Lab untuk dapat
27 dilihat oleh peserta.
- 28 • 20 Menit sebelum ujian dimulai, Admin membuka pintu utama. Admin kemudian akan
29 melakukan pengecekan KTM. Peserta yang tidak memiliki KTM atau surat izin sejenis tidak
30 diperbolehkan untuk memasuki lobi.
31 Admin kemudian memindahkan posisi tempat duduk ke dalam lobi.
- 32 • 5 Menit sebelum ujian, Tim Admin diharuskan mengunci pintu masuk utama lab. Peserta
33 yang terlambat tidak dapat diberi izin untuk masuk oleh Tim Admin.
- 34 • Admin menunggu aba-aba dari pengawas untuk mobilisasi peserta masuk ke ruang ujian.
- 35 • Admin menunggu aba-aba dari pengawas untuk memulai timer.
- 36 • 5 Menit setelah timer dimulai, Tim Admin diharuskan *stand-by* di dalam ruangan ujian untuk
37 menangani masalah teknis yang mungkin muncul.

- 1 • Setelah ujian berakhir, Admin memeriksa jawaban yang sudah terkumpul. Jika terjadi
2 masalah, Admin diharuskan untuk melaporkan ke pengawas ujian.
3 Lalu jawaban ujian dikirimkan pada dosen mata kuliah yang bersangkutan.
- 4 • Untuk ujian dengan lebih dari satu *shift*, Admin diharuskan untuk memastikan bahwa pintu
5 belakang telah terbuka sebelum *shift* pertama berakhir.

6 2.1.4 Panduan Ujian untuk Dosen Koordinator

7 Panduan Ujian yang harus diikuti oleh Dosen Koordinator dimulai dengan melaporkan kebutuhan
8 khusus ujian paling lambat satu minggu sebelum ujian tersebut. Kemudian dosen Koordinator
9 diharuskan menentukan beberapa hal, yaitu

- 10 • Jenis ujian, yang terdiri dari:
 - 11 – *Softcopy*, soal akan didistribusikan ke masing-masing komputer.
 - 12 – *Hardcopy*, soal akan dicetak oleh Tata Usaha seperti pada ujian kelas.
 - 13 – Kombinasi *Softcopy* dan *Hardcopy*.
 - 14 – Dengan layanan lain, seperti *Judge*.
- 15 • Durasi ujian. Disarankan maksimal 110 Menit.
- 16 • Tipe ujian, yang terdiri dari:
 - 17 – *Close Book* (Tidak menggunakan berkas bantuan apapun.)
 - 18 – *Open Book* (Hanya diperkenankan dalam bentuk *hard copy*.)
 - 19 – *Open File* (Menggunakan berkas yang didistribusi pada *folder* ujian masing-masing
20 peserta.)
 - 21 – Lainnya (dengan layanan seperti *Judge*, basis data, dan sebagainya.)
 - 22 – Kombinasi.
- 23 • Nama dan ekstensi berkas pengumpulan. Penamaan berkas mengikuti format
24 <prefix>xxyy<postfix> dengan detil:
 - 25 – xx adalah dua digit tahun angkatan, dan
 - 26 – yyy adalah NPM dari peserta.
- 27 Pengumpulan berkas lebih dari lima disarankan menggunakan zip.
- 28 • Kata sandi soal jika diperlukan.
- 29 Soal dan informasi tersebut kemudian dilaporkan pada email-email tertentu dan Kepala Lab
30 Komputasi paling lambat satu hari sebelum ujian dimulai.
31 Pada saat ujian dimulai, Dosen Koordinator menjadi koordinator pengawas ujian. Koordina-
32 tor pengawas berwenang memberikan instruksi saat ujian berlangsung, termasuk aba-aba mulai,
33 pemberian kata sandi, waktu habis dan sebagainya.



Gambar 2.1: Ilustrasi sistem *backend* dan *frontend* pada pengembangan aplikasi berbasis web.

2.2 Aplikasi Berbasis Web

Pada pengembangan aplikasi berbasis web, terdapat banyak cara untuk mengaplikasikan solusi yang dirancang berdasarkan kebutuhan dan batasan yang ada. Karena aplikasi berjalan pada *server* dan *browser*, maka secara umum pengembang memecah sistem menjadi dua subsistem besar.

Dua subsistem ini mengikuti pola pemrograman *Server-client*, dengan aplikasi klien biasanya ditangani oleh *browser* atau peramban. Namun dengan berkembangnya teknologi, sistem pada peramban juga mulai berevolusi. Oleh karena itu sistem-sistem yang ada harus dirancang dengan baik sesuai dengan disiplin sistemnya sendiri.

Seperti pada Gambar 2.1 secara umum, *front-end* akan membantu sistem menampilkan data dengan baik dengan memanipulasi peramban pengguna. Sedangkan *back-end* akan membantu mengolah data yang sistem *front-end* berikan. Komunikasi tersebut akan mengikuti protokol HTTP.

2.2.1 Back-end

Back-end adalah sebuah subsistem yang melayani program namun tidak diakses secara langsung oleh penggunanya, melainkan dengan melalui Front-end[3]. Pada umumnya Back-end memiliki tugas untuk menyimpan data, memproses data, dan memvalidasi data.

Sesuai dengan namanya, sistem back-end hanya akan berjalan pada server dan sumber kode yang terdapat pada sistem ini tidak akan ditampilkan pada pengguna.

¹ 2.2.2 Front-end

² *Front-end* adalah sebuah subsistem yang melayani pengguna secara langsung^[4]. *Front-end* ini
³ memiliki tugas untuk menyajikan data pada pengguna, serta melakukan pemanggilan perintah ke
⁴ *back-end*.

⁵ Sesuai dengan namanya, subsistem front-end akan berjalan pada sisi klien dari sistem, oleh
⁶ karena itu kode subsistem ini harus dapat ditransmisikan ke peramban pengguna agar sistem dapat
⁷ berjalan sesuai ekspektasi. Karena kode pada subsistem ini dapat dilihat oleh pengguna, maka
⁸ sistem back-end penting untuk melakukan sanitasi data untuk memastikan data yang diberikan
⁹ tidak mengancam ketebalan siklus hidup sistem keseluruhan.

¹⁰ 2.3 *Library* dan *Framework*

¹¹ *Library* adalah kumpulan fungsi yang dikompilasi bersama dan biasanya dibagikan untuk aplikasi
¹² gunakan dengan cara pemanggilan tertentu. *Library* biasanya digunakan oleh pengembang atau
¹³ *developer* untuk mempermudah manajemen dan melakukan pemanggilan fungsi tertentu pada
¹⁴ sistem. Pada pengembangan sistem web, *Library* biasanya mengacu pada kumpulan fungsi atau
¹⁵ modul-modul yang membungkus sekuens tertentu dan fungsi khusus yang disediakan oleh peramban
¹⁶ (*browser*) untuk pengembang. Maka dari itu istilah *Library* sering juga disebut *Package* atau *Module*
¹⁷ tergantung dari konteks letak dan peranan fungsi tersebut^{[5][6]}.

¹⁸ *Framework* adalah salah satu bentuk abstraksi yang dibuat untuk menyediakan layanan yang
¹⁹ generik pada pengembang. Selain itu *framework* membantu pengembangan dan penerbitan aplikasi
²⁰ menjadi lebih mudah dan cepat. Karena bentuk abstraksi tersebut, struktur kode yang dibuat
²¹ menjadi berpola. Dengan adanya bantuan dokumentasi, pengembang aplikasi dapat dengan mudah
²² mencari dan memanfaatkan kode *framework* tersebut.

²³ Pada pengembangan sistem web, sebuah *framework* biasanya memiliki beberapa *package* yang
²⁴ terdiri dari beberapa *library* yang telah dibuat generik berdasarkan memorandum atau RFC (*Request*
²⁵ *For Comments*) yang disepakati oleh tim pengembang tersebut^[7]. Beberapa *library* bahkan dibuat
²⁶ terpisah untuk mempertegas fungsi khusus. RFC biasanya memiliki beberapa detil yang menjelaskan
²⁷ informasi tentang bagaimana sebuah *library* berfungsi dan digunakan pada kasus tertentu. Sehingga
²⁸ beberapa *package* dibuat terhubung sedemikian rupa, membentuk *framework* dan dapat digunakan
²⁹ oleh para-pengembang.

³⁰ Beberapa *library* dapat digunakan-ulang untuk mempermudah manajemen kode sumber pada
³¹ pengembangan. Kode yang dapat digunakan-ulang menyebabkan implementasi fitur yang dibuat
³² konsisten dengan satu dengan yang lain. Dengan stabilitas fitur sistem ini, maka para pengembang
³³ dapat dengan langsung menggunakan fitur yang terdapat pada sistem, menambah fungsionalitas
³⁴ pada kode tersebut untuk menyesuaikan dengan fitur aplikasi, dan melakukan perilisan fitur dengan
³⁵ lebih cepat.

³⁶ 2.3.1 Fat-free Framework

³⁷ Fat-free Framework¹ (translasi literal: Framework bebas lemak), atau disingkat F3, adalah *Fra-*
³⁸ *mework* PHP yang dikembangkan oleh bcosca. Kode dasar *framework* ini memiliki ukuran yang

¹Lihat <https://github.com/bcosca/fatfree>.

1 relatif kecil dan memiliki performa yang cukup baik berdasarkan *benchmark* yang dilakukan oleh
 2 **kenjis** pada tahun 2017[1]. Dengan kecilnya kode dasar tersebut, maka F3 ini memungkinkan
 3 penggunanya untuk mengeksekusi aplikasi dengan lebih efisien dan lebih cepat.

4 F3 memiliki struktur *framework* yang relatif fleksibel sehingga dapat ubah-sesuaikan menurut
 5 kebutuhan pengembang. Salah satu masalah yang biasanya terdapat pada *framework* adalah bentuk
 6 kelas dan objek yang kompleks. F3 memiliki kelas yang relatif sederhana[8], oleh karena itu waktu
 7 yang dibutuhkan untuk mempelajari dokumentasi dari *framework* ini relatif lebih cepat.

8 Composer² adalah salah satu peralatan untuk mengelola *package* untuk aplikasi pada pem-
 9 rograman berbahasa PHP. Dengan Composer yang terintegrasi dengan F3, *framework* FatFree
 10 dapat dengan mudah diintegrasikan dan di perbarui di berbagai macam proyek PHP. Composer
 11 ini kemudian akan secara otomatis melakukan *resolve package* yang akan diperlukan oleh sebuah
 12 proyek, mengunduh paket tersebut lalu melakukan pembuatan *autoloader* untuk *package* tersebut
 13 agar dapat digunakan pada lingkungan pengembangan aplikasi.

14 **Memulai dengan FatFree**

15 Penggunaan FatFree dimulai dengan melakukan *require* pada berkas yang dimiliki oleh FatFree.
 16 Dengan bantuan Composer, aplikasi cukup melakukan *require* pada berkas pemuat otomatis
 17 (*Autoloader*) milik Composer. Seluruh berkas yang dibutuhkan oleh sistem F3 dan aplikasi akan
 18 dimuat otomatis oleh Composer.

Listing 2.1 Implementasi F3 minimal

```
19
20
21 1  <?php
22 2  require 'vendor/autoload.php';
23 3  $f3 = Base::instance();
24 4  $f3->route('GET /',
25 5   function() {
26 6     echo 'Hello, world!';
27 7   }
28 8 );
29 9  $f3->run();
```

32 Pada potongan kode listing 2.1 memperlihatkan implementasi minimal dengan menggunakan
 33 FatFree Framework dengan *package manager* Composer. Sistem *routing* adalah sistem yang
 34 memetakan alamat URL pada kelas tertentu. Kelas tersebut akan melayani *request* dari klien dan
 35 bertugas untuk memberikan respon kembali. Sistem *routing* pada FatFree Framework diimplementasi
 36 seperti pada baris 4 hingga 8. FatFree akan meminta pengembang untuk memberikan informasi
 37 pola *routing* dan fungsi penanganannya. Pada baris 9, sistem FatFree akan dipanggil untuk
 38 memberitahukan bahwa inisialisasi sesi telah selesai, dan mulai untuk memproses *request*.

Listing 2.2 Implementasi F3 dengan *routing* via *Class Path*

```
39
40
41 1  class ReplicaBot {
42 2   function greetings() {
43 3     echo "Nice to meet you. My name is Replica, I am Yuma's guardian.";
44 4   }
45 }
```

²Lihat <https://getcomposer.org/>.

```

1   5      }
2
3   7      $f3->route('GET /about', 'ReplicaBot->greetings');
4

```

6 Berdasarkan APInya, sistem routing dapat dilakukan juga dengan hanya memberikan *Class*
 7 *Path* menuju fungsi *handler*nya (Listing 2.2). Dengan begitu, kita dapat mengimplementasi
 8 penanganannya pada kelas dan berkas yang berbeda.

Listing 2.3 Routing dengan parameter pada F3

```

9
10 1      $f3->route('GET /brew/@count',
11 2          function($f3) {
12 3              echo $f3->get('PARAMS.count').' bottles of beer on the wall.';
13 4          }
14 5      );
15
16 7      $f3->route('GET /replica/@method', 'ReplicaBot->@method');
17
18

```

20 Penggunaan parameter pada sistem *routing* pada FatFree framework dapat diimplementasi dengan
 21 memberikan pola variabel pada definisi *routing* tersebut. Pada listing 2.3 baris 1, variabel diimple-
 22 mentasi dengan `@count`. Variabel tersebut akan dapat diakses via `Base::get`, dapat dilihat pada
 23 baris ketiga.

24 Selain itu, penggunaan variabel juga dapat diimplementasikan pada pendefinisian fungsi *handler*
 25 *request* tersebut. Pada listing 2.3 baris 7, kita dapat memberikan kebebasan pengguna untuk
 26 mengakses *method* apapun yang bersifat publik pada kelas `ReplicaBot`.

Listing 2.4 Berkas konfigurasi untuk F3

```

27
28 1      [routes]
29 2      GET /=home
30 3      GET /404=App->page404
31 4      GET /page/@num=Page->controller
32 5      ; Cache the route for 10 minutes
33 6      GET /contact=App->contact, 600
34 7      ; named route
35 8      GET @about: /about=Page->about
36
37
38

```

39 Konfigurasi *routing* pada F3 dapat dilakukan dengan membuat berkas berekstensi `ini` pada
 40 *folder* konfigurasi, dengan format yang telah didokumentasikan pada web FatFree. Berkas ter-
 41 sebut kemudian dimuat pada sistem dengan memanggil `Base::instance()->config($iniFile)`.
 42 Sebagai contoh pada potongan kode listing 2.4, *routing* didefinisikan dengan format *HTTP Me-*
*43 thod, path URL, dan handler*nya. Berkas konfigurasi tersebut dapat *diload* dengan memanggil
 44 `Base::instance()->config('konfigurasi/routings.ini')`.

2.3.2 React.js

45 React.js³ atau Reactjs adalah salah satu *library* front-end yang digunakan untuk membuat antarmuka
 46 dalam bahasa Javascript. Reactjs memiliki keunggulan seperti bahasa yang deklaratif, berbasis

³Lihat <https://reactjs.org/>

komponen, dan "Pelajari sekali, tulis dimanapun" atau yang disebut juga *Learn Once, Write Anywhere*, (LOWA)[9]. Reactjs ini digunakan oleh banyak pengembang aplikasi web karena keunggulannya, terutama dengan alasan LOWA. Pengembang dapat menggunakan React, bahkan pada bahasa berbasis Js apapun. Normalnya pengembang menggunakan ES6, atau EcmaScript 6 untuk *library* ini namun mereka dapat menggunakan bahasa lain seperti Typescript[10].

Kode pada Reactjs ini bersifat *open-source* atau sumber-terbuka, sehingga banyak kontributor yang ikut membantu pengembangan *library* ini. Dengan banyaknya kontribusi yang diberikan maka hasil kode yang diberikan akan lebih cepat *mature* dan memiliki *bug* yang lebih sedikit.

Virtual DOM adalah salah satu konsep pemrograman yang simulasikan DOM secara ideal pada memory[11]. Pada peramban, setiap elemen disebut dengan DOM. Dalam memanipulasi DOM, Javascript memiliki akses yang lebih lambat dibandingkan dengan mencoba mensimulasikannya pada memori. Konsep *Virtual DOM* ini memanfaatkan kelas elemen pada peramban untuk mensimulasikan manipulasi dan komputasi DOM pada memori, lalu melakukan perubahan tersebut pada DOM yang sesungguhnya. Reactjs akan menyimpan representasi *state* dari komponen-komponen yang telah diinstansiasi pada memori, melakukan komputasi, dan membandingkannya dengan DOM asli yang berada pada peramban. Setiap perubahan *state* yang terjadi, Reactjs akan membandingkan DOM Virtual tersebut pada memori dan melakukan perubahan jika diperlukan.

18 JSX

Reactjs menggunakan sintaks JSX. Sintaks tersebut dibentuk dari turunan Javascript dengan tambahan sintaks mirip elemen pada HTML. Untuk mendeklarasikan sebuah komponen React, pada sintaks JSX, kita dapat membuat sebuah elemen yang diapit oleh kurung siku dan kemudian ditutup dengan nama elemen yang sama, namun diawali dengan garis miring. Sintaks tersebut akan mendeklarasikan sebuah badan komponen Reactjs. Satu komponen pada Reactjs dapat memiliki satu atau lebih dari satu badan komponen.

Listing 2.5 Sintaks JSX

```
26
27   1      const name = 'Shinji';
28   2      const element = <h1>Hello, {name}</h1>;
29
30   4      ReactDOM.render(
31   5          element,
32   6          document.getElementById('root')
33   7      );
34
```

Contoh potongan kode pada listing 2.5 menampilkan sebuah elemen *Heading 1* dengan kalimat *Hello, Shinji*. Element tersebut akan dirender pada elemen HTML dengan id elemen *root*.

Penambahan atribut untuk memberikan konteks data lebih lanjut juga dapat dilakukan melalui JSX. Sebagai contoh, potongan kode pada listing 2.6 menambahkan atribut *src* pada komponen *img* atau HTML *Image*.

Listing 2.6 Sintaks JSX dengan atribut

```
41
42   1      const element = <img src={user.avatarUrl}></img>;
43
```

- 1 Selain atribut, JSX juga dapat memiliki anak element. Sebagai contoh, potongan kode 2.7
 2 menambahkan beberapa anak pada sebuah elemen `div`.

Listing 2.7 JSX dengan beberapa anak di dalamnya

```

3
4   1       const element = (
5
6     2         <div>
7       3           <h1>Equivalent Exchange</h1>
8         4             In order to obtain or create something,
9             something of equal value must be lost or destroyed.
10
11        7             </h2>
12       8         </div>
13
14      9     );
15

```

- 16 Berdasarkan implementasinya, penggunaan JSX akan mencegah celah kemanan *code injection*.
 17 Sintaks pada JSX akan disanitasi terlebih dahulu oleh React sebelum disisipkan pada element
 18 HTML pada dokumen.

19 **Komponen pada React**

- 20 Komponen pada Reactjs dapat diimplementasi dengan dua cara. Dengan melakukan ekstensi
 21 kelas `React.Component` atau yang paling mudah dengan membuat fungsi (Listing 2.8).

Listing 2.8 JSX dengan beberapa anak di dalamnya

```

22
23
24   1       // Dengan membuat fungsi
25     2         function Welcome(props) {
26       3           return <h1>Hello, {props.name}</h1>;
27     4     }
28
29
30   6       // Dengan membuat kelas.
31     7         class Welcome extends React.Component {
32       8           render() {
33         9             return <h1>Hello, {this.props.name}</h1>;
34       10          }
35     11       }
36

```

- 37 Komponen-komponen tersebut dapat digabungkan dengan komponen lain (Listing 2.9). komponen-
 38 komponen tersebut dapat memanfaatkan properti yang diberikan sebagai parameter pada fungsi,
 39 ataupun sebagai atribut `props` pada kelas ekstensi `Component`. Pada potongan kode Listing 2.9,
 40 komponen `Welcome` digunakan sebanyak empat kali dengan atribut yang berbeda-beda. Atribut
 41 yang diberikan memiliki sifat hanya-baca (*Read-only*), sehingga mutasi yang dilakukan pada atribut
 42 oleh komponen tidak akan mengubah `value` atribut.

Listing 2.9 JSX dengan beberapa anak di dalamnya

```

43
44
45   1       function Welcome(props) {
46     2         return <h1>Hello, {props.name}</h1>;
47   3       }
48
49   5       function App() {

```

```

1   6           return (
2   7             <div>
3   8               <Welcome name="Raymond" />
4   9               <Welcome name="Chris" />
5  10              <Welcome name="Yehezkiel" />
6  11              <Welcome name="Cahyadi" />
7  12              <Welcome name="Patrick" />
8  13              <Welcome name="Michael" />
9  14           </div>
10 15         );
11 16     }
13

```

14 State pada React

15 Berdasarkan implementasinya, React tidak akan selalu melakukan pembaharuan pada elemen
 16 HTML. Selain itu, perubahan variabel juga tidak akan diperhatikan oleh React, oleh karena itu
 17 *State* menjadi salah satu cara untuk React mengetahui perubahan pada suatu variabel. Perubahan
 18 *state* pada React akan memanggil ulang fungsi *render* pada komponen, dan melakukan pembaharuan
 19 HTML pada saat perubahan dideteksi dengan memanfaatkan *Virtual DOM*.

Listing 2.10 Komponen dengan *state*

```

20
21
22  1   class Clock extends React.Component {
23  2     constructor(props) {
24  3       super(props);
25  4       this.state = {date: new Date()};
26  5     }
27
28  7     componentDidMount() {
29  8       this.timerID = setInterval(
30  9         () => this.tick(),
31 10         1000
32 11       );
33 12     }
34
35 14     componentWillUnmount() {
36 15       clearInterval(this.timerID);
37 16     }
38
39 18     tick() {
40 19       this.setState({
41 20         date: new Date()
42 21       });
43 22     }
44
45 24     render() {
46 25       return (
47 26         <div>
48 27           <h1>Hello, world!</h1>
49 28           <h2>It is {this.state.date.toLocaleTimeString()}.</h2>
50 29         </div>
51 30       );
52 31     }
53 32   }

```

1 Komponen pada listing 2.10 memiliki sebuah *state* dengan nama `date` (baris 4). *State* tersebut
 2 akan diperbaharui oleh fungsi `tick` (baris 18-22). Fungsi `tick` itu sendiri akan dipanggil oleh *routine*
 3 *interval* dari baris 8. Siklus hidup pada komponen React, `componentDidMount` dieksekusi setelah
 4 komponen berhasil dimasukkan pada elemen dokumen. Sedangkan fungsi `componentWillUnmount`
 5 akan dieksekusi sesaat sebelum komponen dihapus dari element dokumen.

6 Setiap *routine interval* dijalankan, fungsi `tick` akan melakukan pembaharuan *state date* dengan
 7 nilai tanggal hari ini. Perubahan *state* ini akan memaksa React melakukan pemanggilan fungsi
 8 `render` dan memperbaharui elemen HTML yang ada.

9 Penanganan *Event*

Listing 2.11 Penanganan *Event* pada HTML vanila

```
10
11
12 1 <button onclick="diveToAbyss()">
13 2   Start the Journey
14 3 </button>
```

Listing 2.12 Penanganan *Event* pada React

```
17
18
19 1 <button onClick={diveToAbyss}>
20 2   Start the Journey
21 3 </button>
```

24 Penanganan *event* dari HTML pada React tidak jauh berbeda dengan JavaScript. Pada potongan
 25 kode listing 2.11, terlihat implementasi penanganan *event* dilakukan dengan memasukkan nama
 26 fungsi pada atribut `onclick`. Pada potongan kode listing 2.12 terlihat hal yang mirip, dengan huruf
 27 `c` pada penulisan *event onClick* menjadi huruf kapital dan dibanding menggunakan kutip dua,
 28 komponen React tersebut menggunakan kurung kurawal untuk memberikan fungsi.

29 Dengan implementasi seperti itu, secara tidak langsung tiap komponen dapat bertanggung jawab
 30 untuk dirinya sendiri. Maka dari itu komponen yang dibuat akan menjadi lebih modular dan kode
 31 penanganan yang dibuat akan berada dekat pada komponen tersebut.

32 Berpikir Dalam Konsep React

33 Filosofi yang dimiliki pada React membuat pendekatan pada pembuatan aplikasi berbasis *library*
 34 tersebut berbeda dengan pembuatan aplikasi web konvensional. Pendekatan tersebut dapat ditempuh
 35 dengan langkah-langkah yang diberikan oleh Facebook sebagai pembuat React[12] seperti berikut:

- 36 • Pecah bagian *UI* menjadi beberapa bagian hirarki komponen.
- 37 • Buat versi statisnya pada React.
- 38 • Identifikasi representasi minimal *state* pada *UI*.
- 39 • Identifikasi tempat *state* tersebut harus diletakkan.
- 40 • Tambahkan alur datanya.

<i>Method</i>	Deskripsi
GET	Berikan representasi dari target yang dimaksud.
POST	Lakukan proses spesifik pada target yang dimaksud, dengan data yang diberikan pada badan <i>request</i> .
PUT	Ganti representasi target dengan data pada badan <i>request</i> .
DELETE	Hapus target.

Tabel 2.1: Tabel contoh representasi *method* pada protokol HTTP.

1 2.4 API

2 API atau *Application Programming Interface* adalah sebuah antarmuka atau protokol komunikasi
 3 antara klien dan server dengan intensi untuk menyederhanakan pembuatan aplikasi klien-server.
 4 API seringkali dianggap sebagai "kesepakatan" antara klien dan server. Format-format tertentu
 5 akan disepakati untuk komunikasi antar klien-server[13].

6 2.4.1 REST API

7 REST atau *Representational State Transfer* adalah salah satu jenis protokol komunikasi API untuk
 8 layanan berbasis web yang representatif[14]. REST akan memanfaatkan HTTP Method sebagai
 9 informasi intensi permintaan klien untuk server. Komunikasi yang dilakukan pada aplikasi berbasis
 10 web normalnya menggunakan protokol HTTP. Pada protokol tersebut, pesan HTTP memiliki dua
 11 buah bagian: HTTP Header; dan HTTP Body[15]. Protokol ini bekerja dengan membentuk pesan
 12 permintaan pada HTTP Header, dan informasi tentang form pada badan HTTP Request tersebut
 13 (jika ada). Pada HTTP Header terdapat kolom *Method* yang menandakan keinginan klien untuk
 14 server lakukan, seperti pada tabel 2.1[15, P. 21].

15 Sistem RESTful didefinisikan sebagai sistem yang mengikuti beberapa aturan penting tentang
 16 REST[16]. Aturan-aturan tersebut membatasi dan mengatur cara server memproses dan memberikan
 17 respon pada klien.

18 1. *Starting with the Null Style*

19 Setiap request dimulai dari *state* yang kosong, bersih. Solusi yang dibutuhkan dibangun
 20 sesuai dengan kebutuhan aplikasi tersebut secara perlahan. Selain itu pengembang ingin
 21 membuat aplikasinya secara penuh, tanpa halangan dan aturan yang kemudian secara perlahan
 22 mengimplementasikan aturan-aturan dan identitas pada setiap elemen. *Null style* ini kemudian
 23 dikenal sebagai titik awalnya REST.

24 2. *Client-server architecture*

25 Arsitektur klien-server memiliki pemahaman dasar tentang pemisahan antara pengolah data
 26 dan tempat penyimpanan data. Proses pemisahan ini akan menyebabkan antarmuka (*interface*)
 27 program yang lebih sederhana sehingga meningkatkan portabilitas komponen server.

28 Pada kasus penelitian ini, pemisahan antara klien dan server ini menambahkan fungsionalitas

1 seperti tersedianya API. API ini nantinya akan menyediakan kesempatan untuk aplikasi dapat
2 dikembangkan dan diintegrasikan dengan aplikasi lain.

3 **3. Statelessness**

4 Aturan *Stateless* memiliki pemahaman bahwa setiap permintaan yang dilakukan ke server
5 memiliki informasi yang cukup untuk server menyelesaikan tugasnya. Perintah permintaan
6 tidak boleh bergantung dari informasi yang server set sebelumnya. Seluruh konteks dan
7 informasi yang dibutuhkan harus diberikan seluruhnya oleh klien.

8 Pada pemrograman web, implementasi untuk aturan ini pada umumnya adalah dengan tidak
9 mengimplementasikan *session*. Session menyimpan informasi pada server, sehingga permintaan
10 yang diberikan oleh klien akan bergantung pada informasi yang diatur oleh server.

11 **4. Cacheability**

12 *Cacheability* adalah kemampuan untuk menentukan sebuah respon dapat di-*cache* atau tidak.
13 Menentukan sebuah respon dari server dapat di-*cache* dapat memberikan informasi untuk
14 klien menyimpannya pada memori atau tidak perlu. Dengan ditaatinya aturan ini, klien dapat
15 mencegah menampilkan data yang sudah usang dari server.

16 Peramban web pada umumnya akan mengimplementasi manajemen respon dari server secara
17 otomatis. Permintaan yang pernah dikirim ke server dapat dengan mudah dicek oleh peramban.
18 Jika informasi pada server tidak berubah, maka server tidak perlu mengirim respon yang
19 sama kembali. Oleh karena itu penggunaan *cache* dapat memperkecil jumlah interaksi yang
20 dibutuhkan, menyebabkan komunikasi antar komponen lebih efisien.

21 **5. Layered system**

22 *Load balancer* adalah aplikasi yang bertugas untuk membagi pekerjaan (load) pada instansi
23 aplikasi yang berjalan pada server. Penambahan lapisan untuk sistem seperti *load balancer*
24 dapat meningkatkan skalabilitas sebuah sistem. Lapisan ini tidak akan mempengaruhi
25 komunikasi antara server dan klien, dan aplikasi pada klien tidak perlu berubah.

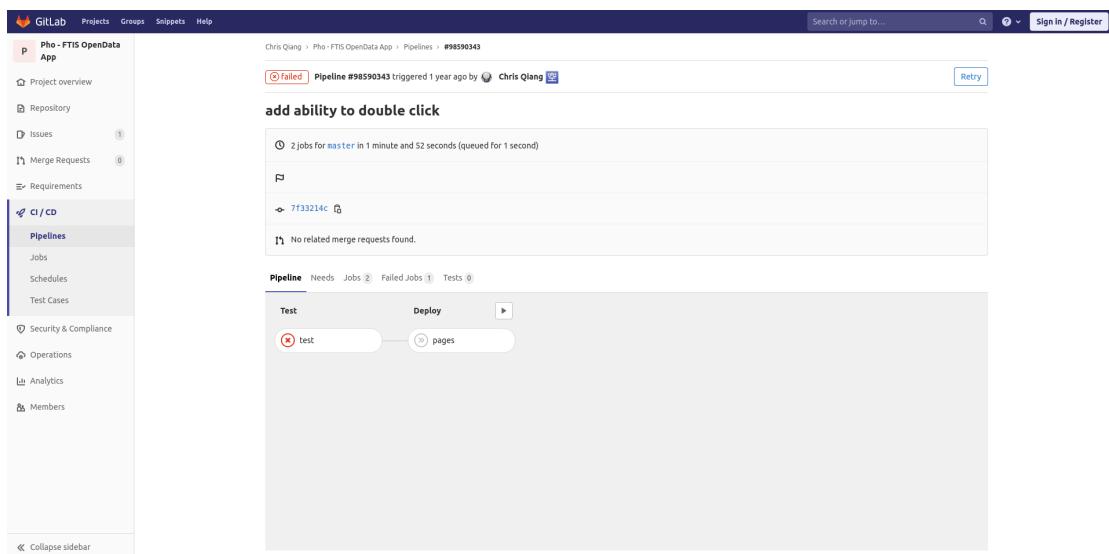
26 **6. Code on demand (opsional)**

27 Server dapat dengan sementara menambah atau menyesuaikan fungsionalitas aplikasi klien
28 dengan memberikan kode yang dapat dieksekusi oleh klien.

29 **7. Uniform interface**

30 Antarmuka yang seragam sangat penting dalam sistem RESTful. Dengan mengaplikasikan
31 generalisasi antar komponen, kita dapat menyederhanakan antarmuka antar komponen. De-
32 ngan generalisasi tersebut, terdapat standar yang sama pada setiap komponen, membuat
33 aplikasi klien dapat melakukan permintaan pada server lain tanpa harus membuat *preprocessor*
34 terlebih dahulu.

35 Karena implementasi dapat dipisahkan, maka antar komponen server dapat berkembang
36 secara mandiri memenuhi kebutuhan dari aplikasi tersebut.



Gambar 2.2: Eksekusi *job* yang gagal pada GitLab CI/CD. Repotori dari Pho - FTIS OpenData App^a

^aLihat <https://gitlab.com/chez14/ftis-opendata-app>

1 2.5 *Continuous Integration/Continuous Delivery/Deployment (CI- 2 /CD)*

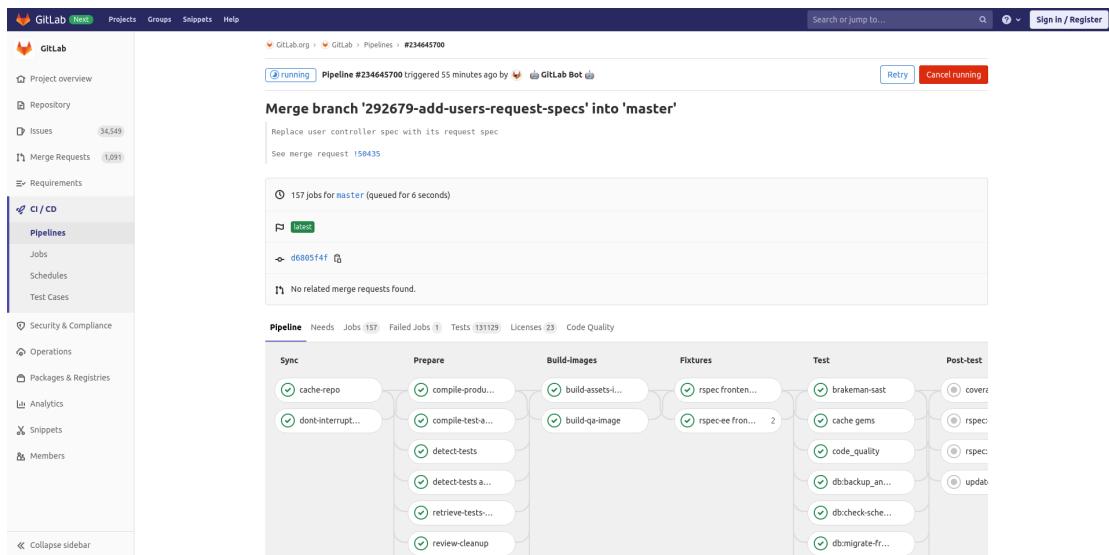
3 *Continuous Integration/Continuous Delivery/Deployment* adalah salah satu metodologi pada pe-
4 ngembangan aplikasi dengan metodologi berkelanjutan. Metodologi tersebut dilakukan dengan
5 mengotomatisasi pengujian, kompilasi dan validasi terhadap kode yang diberikan oleh developer[17].
6 Metodologi CI/CD ini memiliki tiga bagian utama[17].

- 7 • *Continuous Integration*, dilakukan dengan melakukan *testing* terhadap setiap kode yang di
8 *push* oleh developer ke server Git.
- 9 • *Continuous Delivery*, dengan melakukan *deployment* secara berkelanjutan setelah *Continuous
10 Integration* berhasil dilakukan. Metode ini dilakukan secara manual.
- 11 • *Continuous Deployment*, memiliki memiripan dengan *Continuous Delivery*, namun metode
12 tersebut dilakukan secara otomatis.

13 Penggunaan metodologi ini diimplementasi oleh GitLab sejak versi 9[18]. Penggunaan fitur
14 CI/CD ini dapat dimulai dengan membuat berkas konfigurasi `.gitlab-ci.yml` pada *root* dokumen
15 proyek. Berkas tersebut akan berisi sejumlah kumpulan perintah yang dapat dieksekusi berdasarkan
16 kondisi tertentu. Pemilik proyek dapat menambahkan perintah-perintah tersebut dalam bentuk
17 langkah-langkah bertahap. Setelah berkas tersebut berhasil ditambahkan, GitLab akan mendeteksi
18 berkas tersebut dan melakukan eksekusi.

19 *Script* yang ada akan dikelompokkan sebagai *jobs*. Sekumpulan *job* akan membentuk *pipeline*.
20 Sebagai contoh, berkas `.gitlab-ci.yml` memiliki format minimal sebagai berikut:

21 `before_script:`



Gambar 2.3: Contoh *Pipeline* pada GitLab CI/CD. Repositori dari GitLab.^a

^aLihat <https://gitlab.com/gitlab-org/gitlab>

```

1   - apt-get install rubygems ruby-dev -y
2
3   run-test:
4     script:
5       - ruby --version
6

```

7 Pada berkas contoh, format `.gitlab-ci.yml` memiliki format YAML. Oleh karena itu indentasi
8 dengan spasi dan tabulasi akan sangat berpengaruh. Bagian pertama pada berkas tersebut adalah
9 bagian `before_script`. Pada bagian ini, perintah yang dilakukan adalah melakukan instalasi
10 penjalan `script` ruby, dan lingkungan pengembangannya.

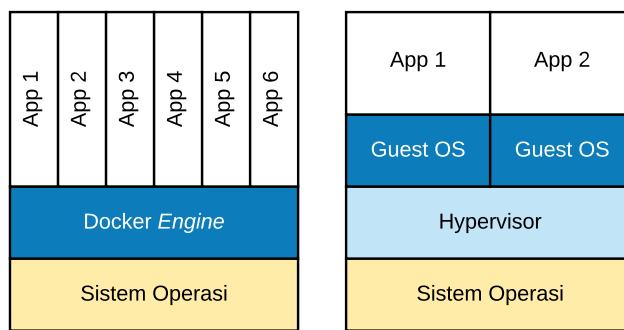
11 Berikutnya, sebuah `job` dengan nama `run-test` akan ditambahkan pada sebuah `pipeline` pada
12 dasbor CI/CD pada GitLab. Lingkungan pada `before_script` akan digunakan pada `job` ini untuk
13 melakukan eksekusi pada `script` yang diberikan. Seperti pada contoh Gambar 2.2. Jika perintah
14 tersebut gagal dieksekusi, maka `job` pada tahap berikutnya tidak akan dieksekusi, dan pemilik kode
15 akan diberitahu bahwa kompilasi gagal.

16 Dengan melakukan konfigurasi tertentu, pemilik proyek dapat membuat rencana CI/CD yang
17 lebih kompleks dan bertahap. *Pipeline* yang sudah dibuat akan ditampilkan pada dasbor secara
18 diagram (Gambar 2.3).

19 2.6 Docker

20 *Containerization* adalah salah satu bentuk abstraksi pada level aplikasi yang membungkus kode dan
21 kebutuhan *dependency* masing-masing aplikasi tersebut. Sehingga aplikasi berjalan secara konsisten
22 pada berbagai sistem operasi[19]. Kontainer ini akan menjalankan *image* yang memiliki kernel yang
23 sama ataupun berbeda antar satu dan yang lain. Selain itu kontainer juga mungkin dapat berbagi
24 *kernel* antara satu *image* dengan yang lain.

1 Lingkungan produksi adalah lingkungan yang digunakan oleh server yang melayani permintaan
 2 kustomer. Karena lingkungan tersebut tempat aplikasi dijalankan, maka lingkungan pengembangan
 3 sebaik mungkin dibuat semirip mungkin dengan lingkungan produksi. Jika lingkungan produksi
 4 dapat dibuat kontainer, maka kita dapat menyalin kontainer tersebut dan membuatnya menjadi
 5 lingkungan pengembangan. Jika lingkungan pengembangan membutuhkan lingkungan produksi
 6 untuk diperbarui, maka kontainer tersebut hanya perlu disalin ulang dan dijalankan kembali.



Gambar 2.4: Ilustrasi perbandingan lapisan sistem pada Docker (kiri) dan *Virtual Machine* (kanan).

7 Kontainer akan menjalankan *image* diatas *platform* mereka setelah infrastruktur sistem operasi
 8 itu sendiri (Gambar 2.4 kiri). Berbeda dengan *Virtual Machine* (atau VM), aplikasi pada kontainer
 9 tidak berjalan di atas sistem operasi yang dijalankan oleh VM (Gambar 2.4 kanan). *Image* pada
 10 kontainer itu sendiri biasanya berukuran cukup kecil (puluhan hingga ratusan MB), sehingga
 11 menjalankan kontainer jauh lebih ringan dibanding menjalankan VM.

12 *Hypervisor* adalah salah satu aplikasi yang memungkinkan menjalankan VM diatas sistem
 13 operasi. (Gambar 2.4 kanan). *Virtual Machine* yang dibuat biasanya akan menjalankan kernel
 14 sistem operasi dan *dependencies*-nya, sehingga memori yang dibutuhkan sangat banyak.

15 Dengan karakteristik seperti itu, aplikasi yang dijalankan pada Docker memiliki beberapa
 16 keunggulan.

17 • Cepat dan *Portable*

18 Docker dapat dikonfigurasi dengan membuat berkas konfigurasi khusus. Berkas konfigurasi
 19 tersebut akan memberikan informasi *image* yang akan digunakan. Sehingga waktu yang
 20 diperlukan untuk melakukan konfigurasi pada server baru relatif cepat. Waktu konfigurasi
 21 secara mayoritas akan habis pada saat pendefinisian berkas konfigurasi Docker tersebut.

22 • Konsisten

23 Dengan memanfaatkan *image* dan *engine* Docker, aplikasi akan berjalan pada *platform* yang
 24 sama. Dengan karakteristik seperti itu, aplikasi yang didirikan secara teori akan dapat berjalan
 25 dengan konsisten pada berbagai sistem operasi.

26 • Aman

27 Karakteristik Docker yang berjalan secara terisolasi dapat meningkatkan keamanan hingga
 28 standar industri[19] secara *default*.

1

BAB 3

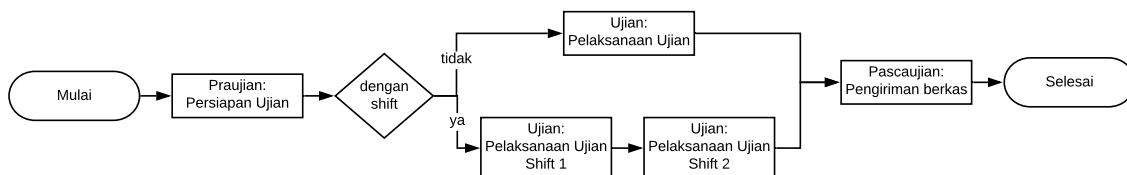
2

ANALISIS

3 3.1 Analisis Sistem Masa Kini

4 Analisis sistem yang digunakan pada saat ini di lab komputasi akan dimulai dengan menganalisis
5 alur pelaksanaan ujian terlebih dahulu. Kemudian dilanjutkan dengan melakukan penyebaran
6 kuisioner untuk mendapatkan informasi lebih lanjut tentang masalah yang tidak teridentifikasi pada
7 saat survei pelaksanaan ujian.

8 Kedua sumber informasi tersebut kemudian akan dianalisis untuk membuat usulan fitur baru
9 yang akan diimplementasi pada sistem yang baru.



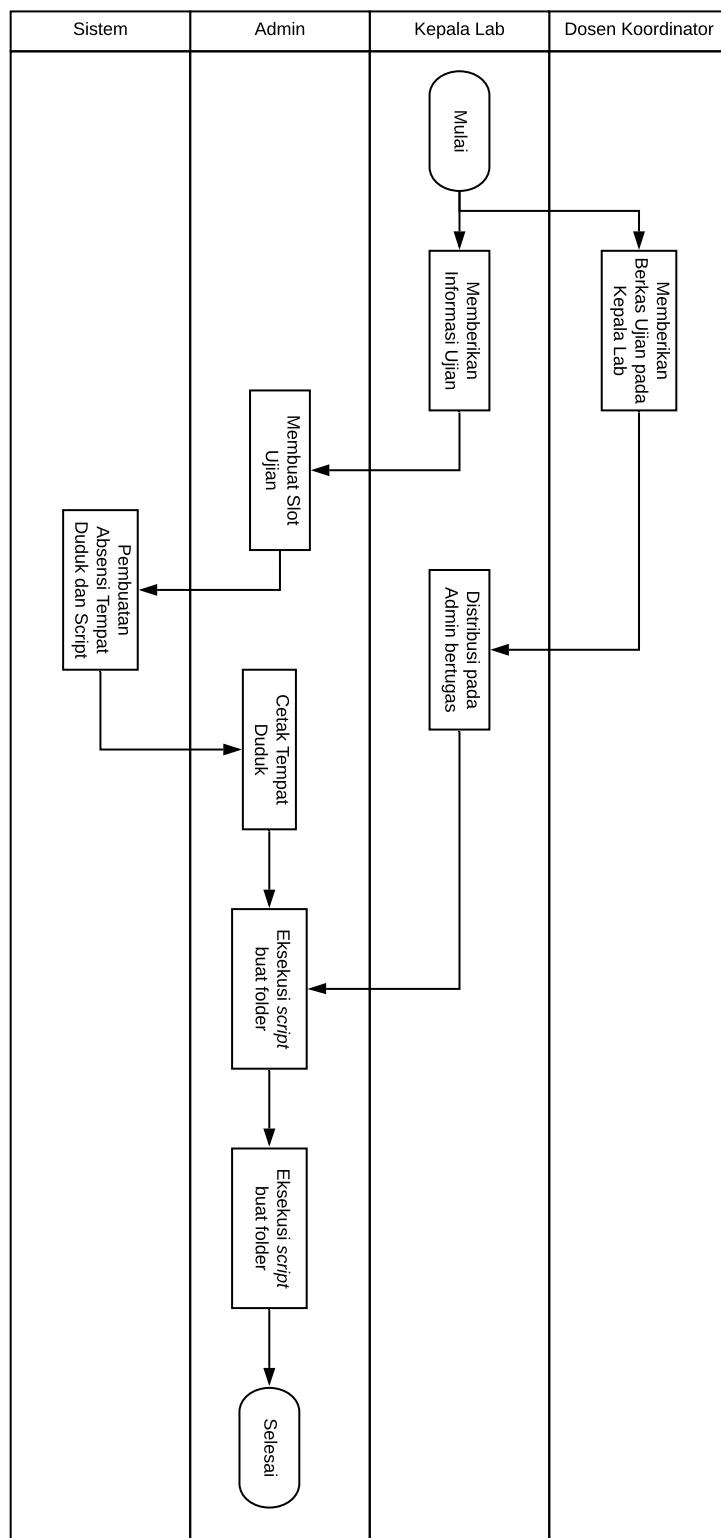
Gambar 3.1: Diagram alur pelaksanaan ujian secara garis besar.

10 Pada pelaksanaan ujian, terdapat tiga tahap penting yang akan dilalui oleh tiap peran. Pada
11 Gambar 3.1, proses ujian dimulai dengan tahap persiapan ujian. Kemudian pada tahap berikutnya
12 terdapat pelaksanaan ujian bergantung pada ada atau tidaknya *shift* pada ujian tersebut. Kemudian
13 ditutup dengan pengiriman berkas ujian pada tahap pascaujian.

14 3.1.1 Praujian

15 Berdasarkan survei lapangan yang dilakukan, persiapan yang dilakukan untuk ujian dilakukan
16 beberapa hari sebelum ujian dilaksanakan. Pada tahap ini pihak yang terlibat dalam pembuatan
17 slot ujian adalah Admin dan Dosen. Alur praujian pada Gambar 3.1 diperjelas pada Gambar 3.2
18 dengan mendetilkan masing-masing peran yang terlibat.

19 Admin akan membuatkan slot ujian pada sistem Oxam, dan mengatur posisi tempat duduk
20 sesuai dengan jumlah dan informasi ujian yang diberikan dari kepala lab. Sistem kemudian akan
21 menghasilkan daftar tempat duduk yang telah diacak oleh sistem. Admin kemudian mencetak
22 daftar tempat duduk tersebut untuk nantinya didistribusikan sesuai ruangannya. Dosen koordinator
23 yang bertugas untuk membuat soal ujian akan memberikan berkas tersebut ke kepala lab. Berkas
24 tersebut kemudian didistribusikan pada Admin yang bertugas untuk menjaga ujian.



Gambar 3.2: Diagram alur detil persiapan ujian.

- 1 Sistem juga menghasilkan beberapa *script* dengan fungsi sebagai berikut:
- 2 • Membuat folder lembar kerja untuk peserta pada masing-masing komputer yang telah dima-
3 sukkan pada sistem.
- 4 • Mendistribusikan berkas soal ujian dan berkas bantuan pada masing-masing komputer peserta.
- 5 • Mengambil alih pemilik berkas pada folder lembar kerja peserta pada masing-masing komputer.
- 6 Persiapan ujian lalu berlanjut pada eksekusi *script* yang diberikan oleh aplikasi Oxam. Admin
7 yang bertugas kemudian memasukan berkas-berkas ujian pada lokasi khusus di server *deployment*.
8 Kemudian script pembuatan folder dan distribusi berkas dieksekusi. *Script* yang dieksekusi akan
9 menghasilkan log yang menginformasikan keberhasilan pendistribusian berkas tersebut. Admin
10 yang bertugas akan memperhatikan log tersebut dan melakukan hal-hal yang perlu dilakukan jika
11 terdapat masalah pada log tersebut.
- 12 Daftar tempat duduk akan didistribusikan pada pintu lobi dan pintu ruang ujian sesaat sebelum
13 ujian dimulai.

14 3.1.2 Ujian

15 Proses Ujian kemudian berlanjut pada hari pelaksanaan ujian tersebut diadakan. Alur pelaksanaan
16 ujian pada diagram 3.1, dijelaskan dengan lebih mendetail pada diagram 3.3. Peserta yang telah
17 diminta hadir 30 menit sebelumnya akan diarahkan untuk masuk dan menunggu pada lobi lab.
18 Admin yang bertugas akan memastikan setiap peserta yang akan memasuki ruangan ujian telah
19 membawa kartu mahasiswa. Dosen ditugaskan untuk menjaga akan melakukan persiapan ujian
20 seperti membagikan kertas buram dan sterilisasi ruangan.

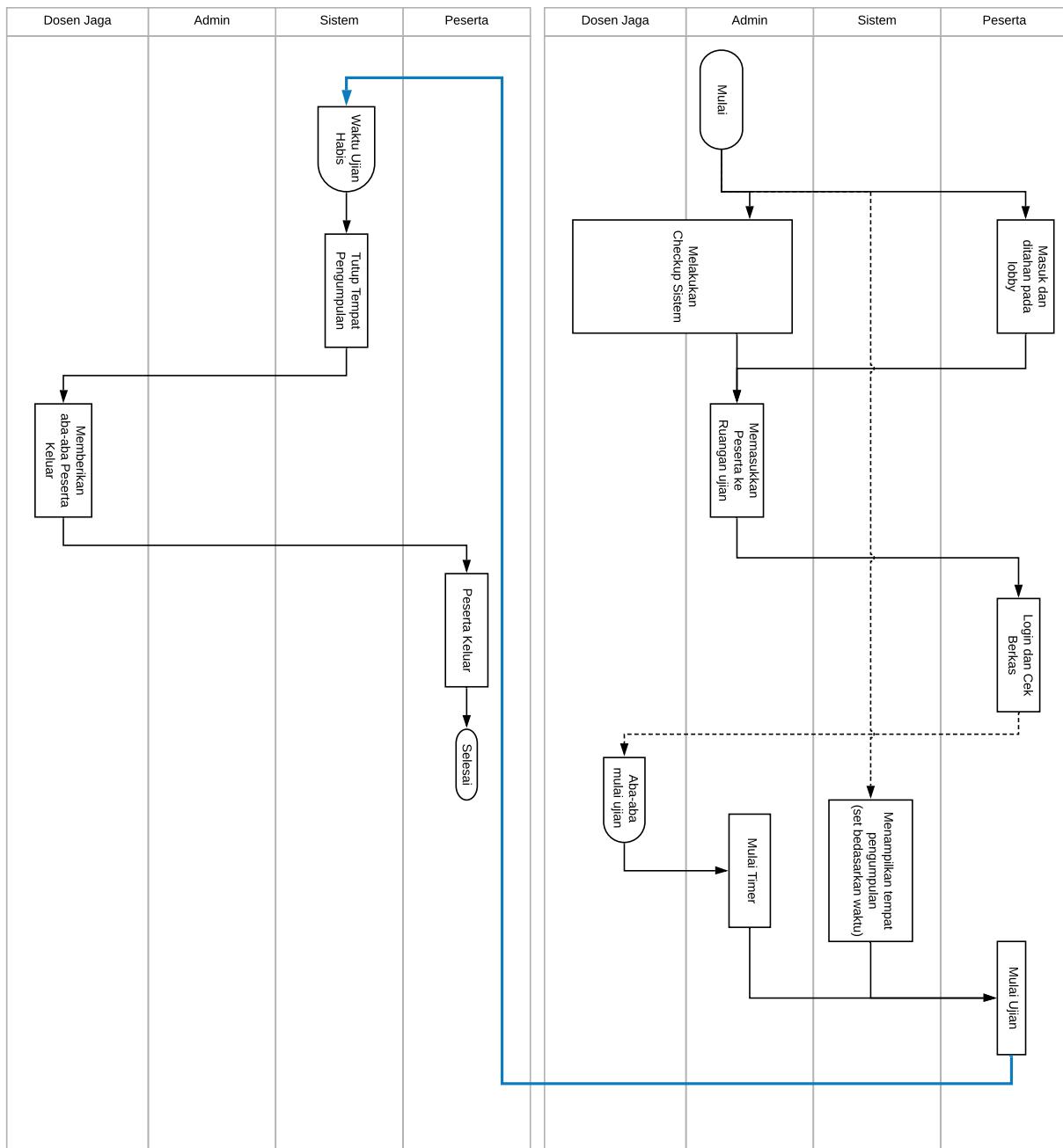
21 Sementara itu, Admin beserta dengan dosen yang bertugas untuk menjaga selama ujian akan
22 melakukan *check up* terakhir untuk memastikan bahwa distribusi soal tidak bermasalah. *Check up*
23 dilakukan dengan bantuan daftar check yang telah dibuat Kepala Lab (Lampiran A). Pemasangan
24 timer dilakukan pada tahap ini.

25 Setelah *check up* selesai dan tidak terdapat masalah, admin akan membuka sekat pemisah ruang
26 lobi dan ruang ujian. Peserta kemudian masuk ke ruangan ujian, dan dipersilahkan duduk sesuai
27 dengan daftar tempat duduk yang telah didistribusikan.

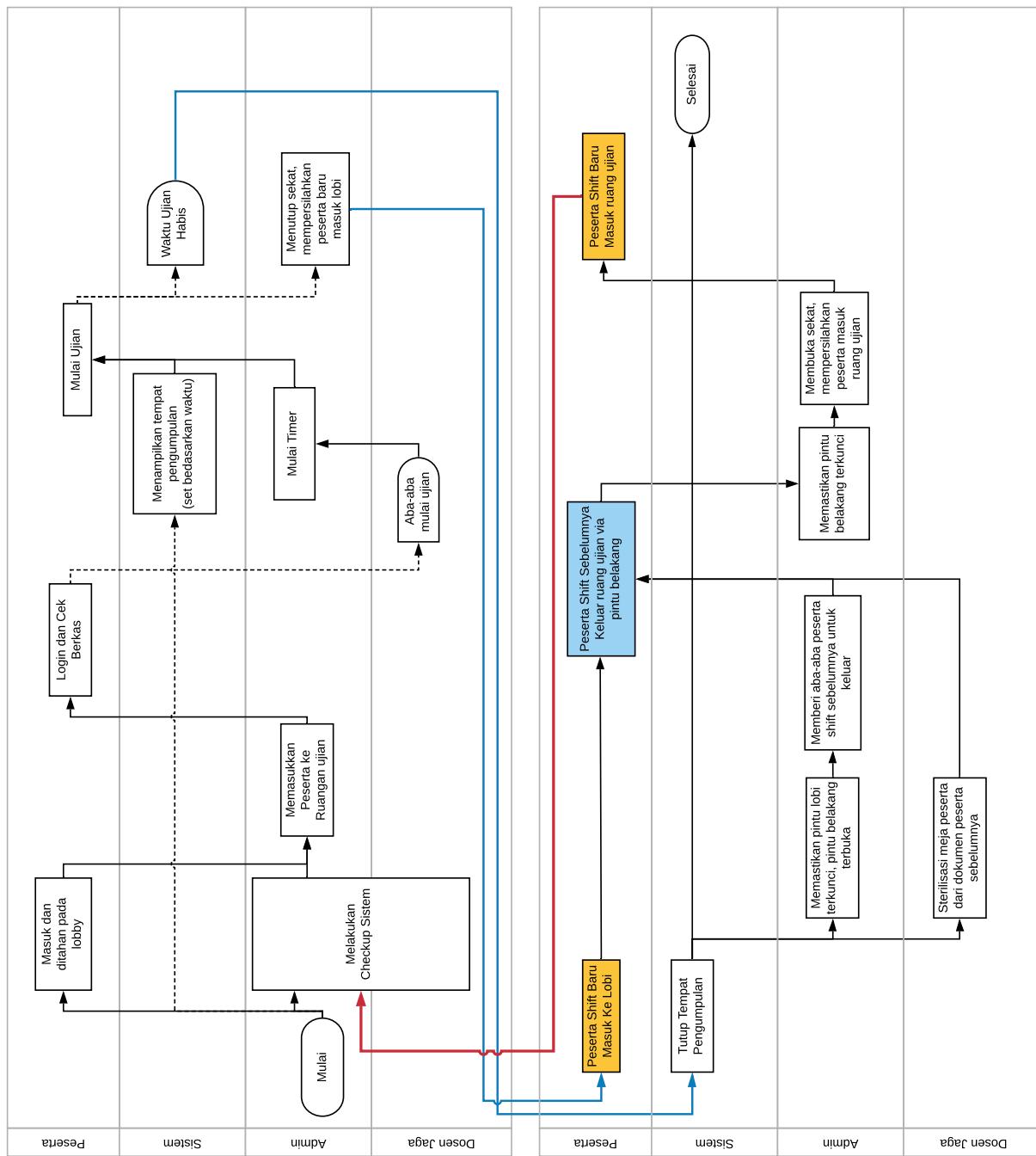
28 Peserta yang sudah duduk kemudian dipersilahkan untuk segera login dan melakukan cek pada
29 berkas ujian tersebut. Peserta yang sudah siap akan menunggu aba-aba dari dosen penjaga untuk
30 memulai ujian. Admin yang bertugas akan bersiap pada komputer proyektor untuk memulai timer.

31 Pada saat dosen penjaga memberikan aba-aba untuk memulai ujian, admin yang bertugas akan
32 memulai timer dan mahasiswa dapat memulai ujian. Slot untuk mengumpulkan tempat jawaban
33 akan muncul pada waktu yang ditentukan. Peserta ujian dapat melakukan pengunggahan berkas
34 jawaban ke portal web dari sistem yang telah dibuka.

35 Pada saat waktu ujian telah habis, dosen yang berjaga akan memberikan aba-aba untuk peserta
36 menghentikan pengeraaan ujian. Lalu dilanjutkan dengan memberikan aba-aba untuk mengeluarkan
37 peserta dari ruangan.



Gambar 3.3: Diagram alur ujian dengan tanpa *shift*.



Gambar 3.4: Diagram alur ujian dengan shift.

1 3.1.3 Ujian dengan shift

2 Ujian dengan shift memiliki sedikit perbedaan dalam pelaksanaan ujian. Dapat dilihat pada
3 diagram 3.4, perbedaan pelaksanaan ini terdapat pada proses memasukan dan mengeluarkan peserta
4 antar shift). Pelaksanaan ujian dengan shift ini akan mengutamakan isolasi peserta dari peserta
5 yang sudah mengerjakan soal ujian. Isolasi tersebut berguna untuk mencegah kecurangan dengan
6 membagikan jawaban atau informasi tentang soal pada peserta yang akan ujian. Isolasi tersebut
7 dimulai sesaat sebelum waktu ujian pada peserta shift sebelumnya telah habis. Admin akan datang
8 pada ruangan untuk menahan peserta yang sedang ujian untuk tidak keluar terlebih dahulu. Admin
9 lainnya akan memasukan peserta baru ke dalam lobi yang terisolasi. Sekat yang memisahkan
10 ruangan lobi dan peserta akan dipasang kembali, menahan peserta shift baru tetap di lobi.

11 Saat timer tanda waktu ujian telah habis berbunyi, Admin akan berkoordinasi memastikan
12 bahwa pintu lobi telah dikunci dan pintu belakang telah dibuka. Admin kemudian akan memberikan
13 aba-aba untuk mengeluarkan peserta shift sebelumnya melalui pintu belakang. Dosen yang berjaga
14 kemudian melakukan persiapan ujian kembali dengan mensterilkan meja peserta dari berkas lama
15 peserta sebelumnya, dan membagikan soal.

16 Setelah peserta ujian pada shift sebelumnya telah keluar sepenuhnya dari ruang ujian, Admin
17 akan mengunci pintu belakang, mengisolasi seluruh peserta shift baru. Sekat kemudian dibuka
18 dan peserta ujian pada shift baru dipersilahkan masuk ke dalam ruangan ujian. Kemudian ujian
19 dilakukan sesuai dengan prosedur pelaksanaan ujian seperti biasa.

20 3.1.4 Pascaujian

21 Alur ujian berikutnya yang dilalui adalah proses manajemen berkas jawaban ujian. Lembar kerja
22 peserta pada komputer yang digunakan pertama-tama akan diganti pemilik berkasnya. Berkas yang
23 dimiliki oleh peserta ujian tersebut, sekarang menjadi milik akun administrator fakultas. Dengan
24 pergantian kepemilikan tersebut, akun peserta secara efektif tidak akan dapat mengakses,
25 menulis atau pun mengubah berkas jawaban tersebut.

26 Tim Admin yang bertugas kemudian mengirimkan berkas jawaban yang telah diunggah kepada
27 dosen koordinator. Berkas jawaban tersebut pertama-tama harus diunduh dari sistem Oxam yang
28 ada secara manual. Kemudian, berkas jawaban tersebut dikirim dengan format *archive* seperti zip
29 atau rar.

30 3.1.5 Aplikasi Manajemen Masa Kini

31 Aplikasi yang digunakan untuk ujian masa kini memiliki beberapa fitur yang mendukung sistem
32 ujian masa kini. Berdasarkan survei lapangan yang dilakukan, aplikasi tersebut memiliki fitur
33 sebagai berikut:

- 34 • Membuat ujian.
35 • Membuat slot jawaban.
36 • Membuat daftar hadir.
37 • Membuat *script*.

- 1 • Mengunduh jawaban.
- 2 • Mengunggah jawaban.

3 3.2 Analisis Kebutuhan

4 Analisis kebutuhan kemudian dilanjut dengan melakukan kuisioner untuk setiap peran untuk
5 mengetahui masalah yang terdapat pada tiap pihak. Kemudian dari kuisioner tersebut, akan
6 dilakukan analisis untuk mendapatkan poin-poin masalah yang ada.

7 Kuisioner dibuat secara virtual di Google Forms dan didistribusikan melalui sosial media.
8 Kuisioner ini ditujukan pada dua subjek. Subjek pertama adalah dosen Koordinator jurusan
9 Informatika yang pernah mengadakan ujian praktik di lab. Subjek berikutnya adalah peserta ujian
10 dari jurusan Informatika yang pernah mengikuti ujian di lab.

11 3.2.1 Dosen

12 Untuk menganalisis kebutuhan untuk peran dosen, pertama-tama akan dilakukan kuisioner terlebih
13 dahulu. Kemudian kuisioner tersebut akan dianalisis untuk mendapatkan detil masalah dan solusi
14 yang akan diberikan.

15 Kuisioner

16 Pada subjek dosen, pertanyaan yang diajukan adalah sebagai berikut:

- 17 • Apakah Bapak/Ibu puas dengan aplikasi pengumpulan ujian?

18 Jawaban diberikan dalam bentuk skala 1 (tidak puas) hingga 5 (sangat puas). Pertanyaan ini
19 bertujuan untuk mengetahui apakah alur pengumpulan ujian sudah nyaman atau belum.

- 20 • Apakah Bapak/Ibu puas dengan format berkas ujian?

21 Jawaban diberikan dalam bentuk skala 1 (tidak puas) hingga 5 (sangat puas). Pertanyaan ini
22 bertujuan untuk mengetahui apakah format berkas ujian yang diberikan sudah nyaman
23 digunakan atau belum.

- 24 • Apakah Bapak/Ibu puas dengan pengiriman berkas ujian?

25 Jawaban diberikan dalam bentuk skala 1 (tidak puas) hingga 5 (sangat puas). Pertanyaan ini
26 bertujuan untuk mengetahui apakah alur pengiriman berkas ujian via email sudah nyaman
27 atau belum.

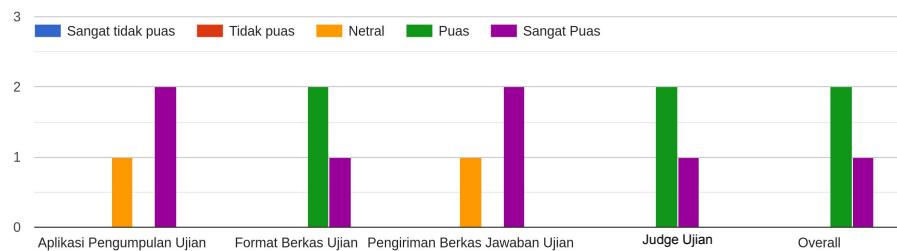
- 28 • Apakah Bapak/Ibu puas dengan *judge* ujian (*random password*, dsb)?

29 Jawaban diberikan dalam bentuk skala 1 (tidak puas) hingga 5 (sangat puas). Pertanyaan ini
30 bertujuan untuk mengetahui apakah alur pemberian informasi kredensial sudah nyaman atau
31 belum.

- 32 • Apakah Bapak/Ibu puas dengan keseluruhan pengalaman berujian di lab?

33 Jawaban diberikan dalam bentuk skala 1 (tidak puas) hingga 5 (sangat puas). Pertanyaan ini
34 bertujuan untuk mengetahui apakah alur pelaksanaan ujian di lab sudah nyaman atau belum.

Bagaimana pengalaman Bapak/Ibu saat melaksanakan ujian di lab komputasi?



Gambar 3.5: Kuisioner Dosen.

- Masalah apa saja yang biasanya bapak/ibu alami?

Jawaban diberikan dalam bentuk kotak teks yang dapat diisi dengan teks yang cukup banyak. Pertanyaan ini bertujuan untuk mengetahui masalah yang belum diketahui selama pelaksanaan survei dari sudut padang dosen pengawas.

Pada kuisioner tersebut, terdapat tiga responden yang memberikan respon. Tabel grafik dapat dilihat pada gambar 3.5. Berdasarkan respon-respon yang diberikan, dosen-dosen memiliki *feedback* yang positif pada sistem ujian yang saat ini telah berjalan. Sehingga sistem ujian yang nantinya akan berjalan pada lab dibuat memiliki perubahan seminimal mungkin dari sudut padang dosen.

Pada pertanyaan terakhir, masalah yang dikeluhkan diantaranya:

- Komputer peserta ujian terkadang mengalami *hang*.
- Hanya ada satu admin yang datang tepat waktu
- Pertukaran shift kadang membingungkan pengawas, karena pernah suatu kali, daftar hadir belum diperbarui
- Pemberian password kadang masih manual melalui kertas.
- Kebingungan karena terkadang mahasiswa tidak bisa *submit* karena dikatakan "waktu telah habis", tetapi bisa lagi jika di-*refresh*.

Kemudian dari hasil kuisioner tersebut didapatkan beberapa masalah yang

18 Bug Waktu Telah Habis

Laporan masalah pertama berasal dari sisi Dosen yang mengawasi ujian di lab komputasi. Masalah yang sering kali dikeluhkan adalah *bug* waktu telah habis. *Bug* ini didapatkan pada saat peserta sudah membuka halaman pengumpulan sudah cukup lama tanpa melakukan penyegaran-ulang (*refresh*) ataupun pengumpulan (*submission*). Untuk mengatasi masalah ini, biasanya Tim Admin akan meminta peserta ujian untuk melakukan penyegaran-ulang beberapa kali hingga pesan kesalahan tersebut hilang. Setelah diselidiki, bug tersebut disebabkan oleh *cache* yang memiliki umur yang pendek. Umur pendek ini menyebabkan *cache* kadaluarsa dan dianggap tidak *valid* oleh PHP, sehingga memancing aplikasi untuk memunculkan pesan kesalahan tersebut, mencegah peserta untuk mengumpulkan jawaban pada web.

¹ **Pembagian *Password* yang Masih Manual**

² Masalah berikutnya yang Dosen berikan pada kuisioner adalah pembagian *password* yang masih
³ manual. Pengujian dengan aplikasi khusus adalah salah satu hal yang sering dilakukan di Lab
⁴ Komputasi. Program tersebut biasanya menggunakan otentikasi khusus yang tidak dapat diintegra-
⁵ sikan dengan aplikasi Oxam utama untuk mencegah kecurangan. Hal ini saat ini diatasi dengan
⁶ membagikan kertas pada tiap meja, atau membagikannya lewat berkas teks pada folder ujian.
⁷ Sistem ini dilakukan dengan cara melakukan penciptaan kata sandi acak yang nantinya dimasukkan
⁸ pada *script* tertentu. Selanjutnya Tim Admin akan melakukan pembuatan akun tersebut dengan
⁹ bantuan *script* tersebut, lalu kredensial tersebut nantinya akan diolah untuk nantinya dicetak atau
¹⁰ pun dipecah menjadi beberapa berkas teks sebelum nantinya diedarkan.

¹¹ **Daftar Hadir yang Membingungkan Pengawas**

¹² Berikutnya, daftar hadir yang membingungkan pengawas. Pada hal ini, pengawas kebingungan
¹³ karena kesalahan Tim Admin melakukan pendaftaran peserta. Masalah ini nantinya akan didalami
¹⁴ pada saat pembahasan survei dari Tim Admin.

¹⁵ **3.2.2 Peserta**

¹⁶ **Kuisioner**

¹⁷ Formulir kuisioner yang dibagikan untuk peserta ujian memiliki pertanyaan sebagai berikut:

- ¹⁸ • Masalah apa saja yang pernah anda alami?

¹⁹ Tujuan pertanyaan ini adalah untuk mengidentifikasi masalah-masalah yang sering dialami
²⁰ oleh peserta ujian. Bentuk jawaban adalah daftar kotak yang dapat diceklis.

- ²¹ – "Waktu ujian telah habis"
²² – Waktu ujian tidak terlihat dengan baik/jelas
²³ – Daftar tempat duduk yang membingungkan
²⁴ – *Credential*¹ untuk *Judge* bermasalah
²⁵ – Tidak memiliki masalah

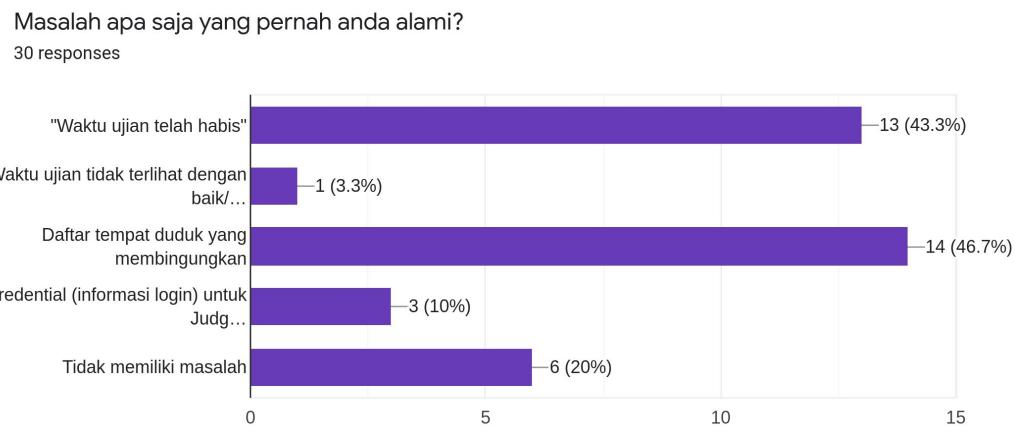
- ²⁶ • Selain dari masalah tersebut, apakah anda memiliki masalah lainnya?

²⁷ Pertanyaan ini ditujukan untuk mengetahui masalah yang tidak diketahui oleh Tim Admin
²⁸ atau tidak terlihat secara langsung pada saat survei lapangan. Bentuk jawaban adalah kotak
²⁹ teks panjang.

- ³⁰ • Bagaimana pendapat anda tentang ujian di Lab Komputasi?

³¹ Pertanyaan ini ditujukan untuk mengetahui pendapat peserta ujian tentang pengalaman ujian
³² di lab. Jawaban berupa skala satu hingga lima, dengan satu paling buruk, dan lima paling
³³ baik.

¹informasi *login*



Gambar 3.6: Respon kuisioner untuk masalah yang telah diketahui pada saat survei.

1 Kuisioner ditujukan pada peserta ujian yang pernah mengikuti ujian di lab pada saat sistem ini
 2 masih digunakan. Kuisioner ini direspon oleh 30 responden yang terdiri dari beberapa angkatan.
 3 Berdasarkan respon yang diberikan oleh peserta untuk pertanyaan pertama, secara mayoritas
 4 masalah yang dialami adalah daftar tempat duduk yang membingungkan, diikuti dengan masalah
 5 bug waktu ujian telah habis (Gambar 3.6).

6 Pada pertanyaan kedua, peserta ujian mengeluhkan beberapa masalah seperti:

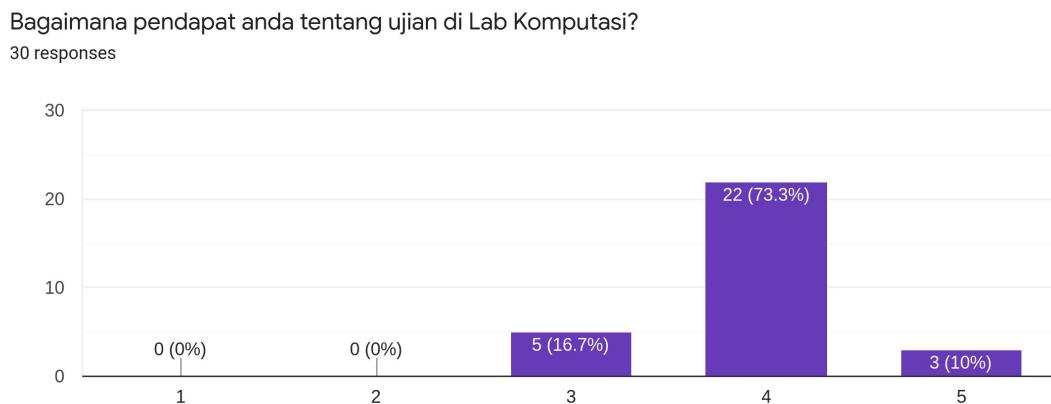
- 7 • Kadang-kadang sesuatu yang ditulis di papan tulis tidak terlihat jelas pada posisi duduk tertentu.
- 8
- 9 • Beberapa komputer memiliki masalah jaringan
- 10 • Daftar tempat duduk yang miring dan tanpa garis
- 11 • Waktu ujian yang kurang
- 12 • Komputer *hang*
- 13 • Perangkat lunak yang tidak dapat digunakan ketika ujian

14 Dari respon-respon tersebut, tidak semua dapat diselesaikan dengan mengimplementasikan solusi
 15 pada sistem, sehingga masalah-masalah tersebut masuk pada batasan masalah.

16 Selain itu, secara keseluruhan (lihat Gambar 3.7), peserta ujian memberikan respon yang positif
 17 terhadap sistem yang saat ini berjalan pada lab. Sehingga pada sisi peserta, perubahan sistem akan
 18 dibuat seminimal mungkin.

19 **Komputer Hang**

20 Kuisioner kemudian dilakukan juga untuk peserta ujian yang pernah mengikuti ujian di lab
 21 komputasi. Dari sisi peserta, Masalah yang muncul pada saat ujian adalah komputer yang *hang*.
 22 Masalah ini muncul secara acak pada saat ujian berlangsung. Peserta biasanya akan diminta untuk
 23 direlokasi untuk mendapatkan komputer yang tidak bermasalah. Peserta akan diberikan waktu



Gambar 3.7: Pendapat peserta ujian pada sistem ujian yang berjalan saat ini.

- ¹ tambahan sesuai dengan durasi lamanya komputer tidak dapat digunakan hingga komputer peserta
- ² yang baru diberikan dapat digunakan kembali. Pada kasus ini, masalah tersebut masuk dalam
- ³ batasan masalah yang tidak dapat diatasi.

⁴ Tulisan Pada Papan Tulis yang Tidak Terlihat

- ⁵ Selain itu, peserta mengeluhkan bahwa tulisan yang ada di papan tulis seringkali tidak terlihat.
- ⁶ Pada ujian, biasanya informasi seperti kata sandi untuk membuka berkas soal, informasi *URL* untuk
- ⁷ masuk ke dalam sistem informasi, atau revisi soal biasanya ditulis pada papan tulis yang ada di
- ⁸ depan. Masalah yang timbul adalah peserta yang mendapatkan tempat yang terjauh dari papan
- ⁹ tulis mengaku kesulitan dalam melihat informasi tersebut. Hal ini dinilai tidak adil untuk peserta
- ¹⁰ yang duduk dekat dengan papan tulis, sehingga dibutuhkannya fitur untuk memberi tahu informasi
- ¹¹ ini.

¹² 3.2.3 Tim Admin

¹³ Menghapus Berkas yang Gagal

- ¹⁴ Dari sisi Tim Admin, survei yang dilakukan menghasilkan banyak keluhan dan masukan. Salah
- ¹⁵ satunya, Tim Admin mempermasalahkan bahwa mereka tidak dapat menghapus berkas-berkas yang
- ¹⁶ gagal di *deploy*. Masalah tersebut biasanya muncul pada saat Tim Admin melakukan kesalahan
- ¹⁷ memasukan data pada saat pembuatan ujian pada aplikasi Oxam. Tim Admin yang tidak mengetahui
- ¹⁸ kesalahan tersebut dapat saja melakukan *deployment* terlebih dahulu sebelum akhirnya menyadari
- ¹⁹ bahwa ia telah melakukan kesalahan. Masalah yang dikeluhkan berakar dari folder yang tidak
- ²⁰ digunakan tersebut membingungkan bagi peserta dan berpotensi menjadi celah keamanan untuk
- ²¹ mencontek.

²² Selain folder, Tim Administrator juga harus menghapus daftar yang ada pada database. Setelah

²³ diselidiki, masalah timbul dari *record* yang ditandai sebagai di hapus dengan mengisi kolom

²⁴ *isDeleted* menjadi 1, namun tidak membuat sistem pengumpulan berubah. Masalah yang timbul

1 adalah sebagian dari peserta mendapatkan tempat pengumpulan yang berbeda dari seharusnya.

2 **Pindah Tempat Duduk Peserta**

3 Selain itu, beberapa masalah lainnya adalah pindah tempat duduk peserta. Bersinggungan dengan
4 masalah yang dimiliki peserta tentang komputer yang *hang* pada [3.2.2](#), Tim Administrator memiliki
5 kesulitan untuk melakukan pemindahan komputer karena Aplikasi yang saat ini ada memiliki
6 masalah tentang mengubah posisi tempat duduk. Masalah yang muncul adalah dibutuhkannya
7 orang untuk melakukan:

- 8 • Pemindahan peserta
- 9 • Pemindahan berkas ujian peserta
- 10 • Pemindahan *record* pada database Oxam

11 Dengan begitu jika terdapat lebih dari seorang peserta yang mengalami masalah ini maka
12 dibutuhkan waktu yang sangat lama untuk menyelesaikan masalah tersebut.

13 **Pengiriman Berkas Jawaban Otomatis**

14 Normalnya, setelah ujian selesai maka Tim Admin akan mengunduh berkas jawaban tersebut untuk
15 dikirimkan ke Dosen Koordinator. Tim Admin akan mengakses aplikasi Oxam, lalu mengunduh
16 berkas jawaban ujian dan mengirimnya secara manual ke Dosen Koordinator mata kuliah tersebut.
17 Masalah yang muncul adalah tidak setiap kali ujian Tim Admin ingat untuk mengirimkan berkas.

18 **NPM Jenis Baru**

19 Peserta yang menjalani ujian praktik pada lab komputasi adalah mahasiswa. Bergantinya sistem
20 informasi akademik kampus berdampak pada pergantian format NPM yang dimiliki oleh mahasiswa.
21 NPM pada mahasiswa memiliki format tertentu yang dapat menginformasikan asal dari mahasiswa
22 tersebut.

NPM Lama	NPM Baru
2016730011	6181601011
username	
i16011	

Tabel 3.1: NPM lama, baru dan username yang mahasiswa gunakan untuk login.

23 NPM lama memiliki empat komponen besar yang digunakan untuk mengidentifikasi jenis
24 mahasiswa (Tabel [1.1](#), kolom NPM Lama).

- 25 • Empat karakter pertama pada NPM adalah informasi tahun mahasiswa tersebut memulai kulih.
26 Pada contoh tampak angka 2016, yang berarti mahasiswa tersebut memulai perkuliahan pada
27 tahun 2016 (angkatan 2016).

- 1 • Dua karakter berikutnya menandakan jurusan (dan fakultas) mahasiswa tersebut belajar.
2 Pada contoh tampak angka 73 yang menandakan bahwa mahasiswa tersebut belajar pada
3 Fakultas Teknologi Informasi dan Sains (7), dan berada pada jurusan Informatika (73).
- 4 • Satu karakter berikutnya menandakan program yang diambil oleh mahasiswa tersebut. Angka
5 0 menunjukkan bahwa mahasiswa tersebut mengikuti program Sarjana, seperti pada tabel
6 contoh. Angka 1 menandakan mahasiswa tersebut mengambil program pascasarjana dan
7 angka 2 menunjukkan program doktoral.
- 8 • Tiga angka berikutnya yang menjadi komponen terakhir dari NPM lama ini adalah nomor urut
9 mahasiswa tersebut. Dengan spesifikasi berikut, berdasarkan tabel contoh, mahasiswa dengan
10 NPM 2016730011 adalah mahasiswa angkatan 2016, mengikuti program sarjana informatika
11 pada Fakultas Teknologi Informasi dan Sains dengan nomor urut 11.

12 NPM baru ini dibuat pada tahun 2018, dan memiliki struktur yang jauh berbeda dari NPM
13 lama. Format NPM tersebut didefinisikan sebagai berikut:

- 14 • Angka pertama menandakan program jenjang yang diambil oleh mahasiswa tersebut. Angka
15 5 untuk diploma, 6 untuk sarjana, 8 pascasarjana dan 9 untuk doktoral.
- 16 • Dua digit angka berikutnya menunjukkan program studi yang diambil oleh mahasiswa tersebut,
17 pada contoh 18 berarti Informatika.
- 18 • Dua digit berikutnya menunjukkan angka tahun mulai perkuliahan (angkatan) dalam format
19 representasi tahun dengan dua digit. Pada contoh tampak angka 16 yang berarti mahasiswa
20 tersebut adalah angkatan 2016.
- 21 • Dua digit berikutnya menginformasikan jenis mahasiswa. Pada contoh tampak angka 01 yang
22 menandakan bahwa mahasiswa tersebut berjenis reguler.
- 23 • Tiga digit terakhir menginformasikan nomor urut mahasiswa tersebut.

24 Pada tabel 3.1 terdapat kolom username yang digunakan untuk menstandarisasi NPM tersebut.
25 Informasi username ini nantinya dimanfaatkan untuk mengintegrasikan berbagai macam sistem yang
26 membutuhkan informasi mahasiswa. Digit pertama pada username menginformasikan jurusan
27 mahasiswa tersebut. Huruf **i** menginformasikan mahasiswa tersebut adalah mahasiswa jurusan
28 Informatika, huruf **m** untuk matematika, dan huruf **f** untuk fisika. Dua digit berikutnya mengin-
29 formasikan tahun angkatan mahasiswa tersebut dalam bentuk representasi tahun dalam dua digit.
30 Pada contoh tampak angka 16, yang menandakan bahwa username ini adalah milik mahasiswa
31 angkatan 2016. Tiga digit terakhir berikutnya menginformasikan nomor urut mahasiswa tersebut.

32 **Kode Matakuliah Baru**

33 Kurikulum yang saat ini ada memiliki kode matakuliah yang baru. Masalah yang muncul adalah
34 sistem aplikasi yang lama tidak memiliki admin panel untuk melakukan manajemen mata kuliah
35 yang baru.

1 **Timer yang tidak terintegrasi**

2 Pada saat pelaksanaan ujian, Tim Admin menggunakan timer yang disediakan oleh sistem operasi
3 Windows. Masalah yang timbul adalah buka dan tutupnya tempat pengumpulan ujian tidak berjalan
4 bersamaan dengan timer. Sebagai contoh, pada saat timer menunjukkan waktu sudah habis, peserta
5 masih dapat mengumpulkan berkas ujian.

6 **3.2.4 Analisis Fitur Aplikasi**

7 Berdasarkan survei yang dilakukan dan kuisioner yang disebar, didapatkan beberapa masalah yang
8 menandakan bahwa aplikasi pendukung ujian di lab komputasi belum efektif membantu pelaksanaan
9 ujian. Masalah-masalah tersebut akan diselesaikan dengan merekayasa ulang aplikasi yang telah
10 ada pada sistem yang lama. Rekayasa ulang tersebut dilakukan dengan bantuan *framework* agar
11 sumber kode yang dibuat dapat dirawat oleh tim Admin.

12 **3.2.5 Timer Tidak Terintegrasi**

13 Sesuai dengan kebutuhan yang dijelaskan pada 3.2.3, timer akan diimplementasi secara terintegrasi.
14 Timer nantinya akan menutup tempat jawaban pada saat waktu telah habis. Timer harus dapat
15 mengatur buka tutupnya tempat pengumpulan jawaban peserta.

16 **3.2.6 Notifikasi**

17 Fitur pertama yang ingin ditambahkan adalah sistem notifikasi. Fitur ini nantinya akan menye-
18 lesaikan masalah pada keluhan kata sandi yang disebarluaskan manual (3.2.1), dan informasi yang
19 dituliskan di papantulis tidak terlihat (3.2.2). Dosen pengawas nantinya dapat membuat buah
20 notifikasi baru untuk peserta dan peserta dapat langsung melihat notifikasi tersebut pada tempat
21 pengumpulan.

22 **3.2.7 Pemindahan Tempat Duduk Peserta**

23 Fitur kedua yang ingin ditambahkan adalah fitur pemindahan tempat duduk peserta. Fitur ini
24 nantinya akan menyelesaikan masalah peserta yang mengalami *hang* pada saat ujian berlangsung
25 (3.2.2) dan Tim Admin yang ingin memindahkan tempat duduk peserta(3.2.3). Fitur ini nantinya
26 hanya akan dapat digunakan oleh Tim Admin. Dosen pengawas diharapkan untuk mengontak
27 Tim Admin untuk melakukan perpindahan untuk memastikan komputer yang baru akan memiliki
28 kemungkinan bermasalah yang lebih kecil.

29 **3.2.8 Pengiriman berkas otomatis**

30 Fitur lainnya adalah pengiriman berkas otomatis. Fitur ini untuk menyelesaikan masalah pada
31 pengiriman berkas jawaban ujian oleh Tim Admin (3.2.3). Pada dasarnya fitur ini dapat implementasi
32 dengan cara membuat *cronjob* setiap menit yang nantinya mengecek apakah terdapat berkas ujian
33 yang dapat dikirim. Pengiriman dapat berlangsung secara otomatis lewat email.

¹ **3.2.9 NPM Converter**

- ² NPM *Converter* nantinya akan mengabstraksi seluruh NPM yang saat ini aktif di sistem UNPAR.
³ Abstraksi tersebut nantinya diharapkan dapat menstandarisasi dua format yang saat ini berjalan.

⁴ **3.2.10 Admin Panel**

- ⁵ Admin panel ditambahkan dengan tujuan agar dapat memanajemen seluruh entitas database
⁶ pendukung ujian. Sebagai contoh, penambahan atau penghapusan mata kuliah. Saat ini, pada
⁷ sistem yang aktif tidak terdapat tempat untuk menambahkan atau menghapus mata kuliah.

⁸ **3.2.11 Bug fix**

- ⁹ Implementasi sisanya, terdapat beberapa masukan yang dapat dilakukan, yaitu untuk menyelesaikan
¹⁰ masalah tentang *bug* waktu ujian telah habis (3.2.1) dapat diimplementasi dengan melakukan
¹¹ authentikasi berdasarkan IP yang tidak disimpan menggunakan *cache* atau *session*. Hal ini
¹² memperkecil masalah *cache* atau *session* yang kadaluarsa karena waktu yang diberikan terlalu kecil.
¹³ Sehingga setiap kali koneksi dijalankan dari klien ke *server*, server akan langsung mengecek IP
¹⁴ tersebut dengan yang ada di database melakukan apapun.

¹⁵ **3.3 Analisis Pemilihan Framework dan Library**

- ¹⁶ Penggunaan *framework* diharapkan dapat membuat aplikasi dikembangkan lebih lanjut oleh penerus
¹⁷ Tim Admin berikutnya. Sifat *framework* yang dapat digunakan-ulang dan modular membuat
¹⁸ *framework* memiliki pola yang dapat dipelajari dengan cepat. Memungkinkan Tim Admin untuk
¹⁹ dapat menambahkan fitur baru dengan lebih cepat.

²⁰ Berdasarkan survei yang dilakukan oleh StackOverflow pada tahun 2018[20] dan 2019[21], PHP
²¹ adalah salah satu bahasa yang mayoritas digunakan oleh pengembang aplikasi web. Dengan ba-
²² nyaknya pengguna bahasa tersebut, maka komunitas, *library* dan firum akan lebih besar. Pilihan
²³ tersebut dinilai lebih menguntungkan karena pengembang hanya perlu mempelajari cara menggu-
²⁴ nakan *library* tersebut tanpa harus mengimplementasi algoritmanya dari dasar. Dengan begitu
²⁵ banyaknya pilihan yang ada, developer yang ingin menambahkan fitur tersebut dapat lebih berfokus
²⁶ untuk mengimplementasi fitur tersebut.

²⁷ **3.3.1 FatFree Framework**

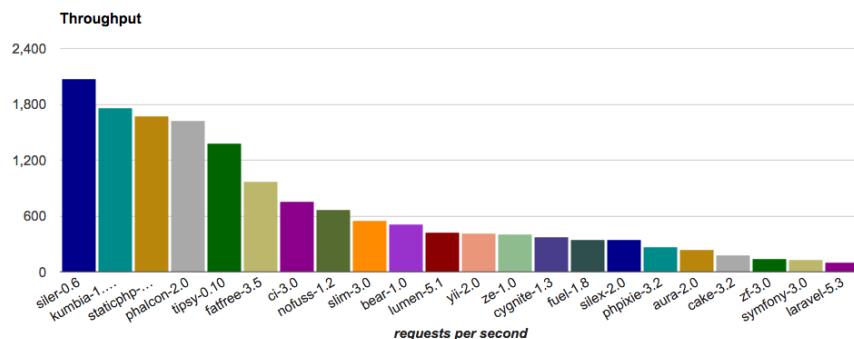
- ²⁸ Beberapa *framework* memiliki sifat yang fleksibel, sehingga memungkinkan developer mengadopsi
²⁹ *framework* tersebut dengan gaya mereka masing-masing. Fat-Free *framework* tidak memiliki pola
³⁰ pemrograman yang pasti, namun sistem mereka dirancang agar dapat digunakan pada tipe pola
³¹ pemrograman apapun. Kontras dengan *framework* besar seperti Laravel, atau Code Igniter. Fat-free
³² sendiri dengan jelas menuliskan untuk menggunakan gaya apapun yang pengembang butuhkan².

³³ Dengan sifat analisis kebutuhan yang cepat, maka penggunaan *framework* yang kecil dan ringan
³⁴ dibutuhkan. Selain itu dengan kebutuhan yang mengharuskan implementasi cukup fleksibel untuk

²<https://fatfreeframework.com/3.6/getting-started/#EnoughSaid-SeeForYourself>

PHP Framework Benchmark

Hello World Benchmark



Gambar 3.8: Hasil *benchmark* terhadap beberapa *framework* pada bahasa pemrograman PHP^[1]. (Lebih tinggi lebih baik)

- ¹ kemudian hari dikembangkan lebih lanjut, maka FatFree Framework atau F3 akan digunakan untuk
² penelitian ini.

³ **3.3.2 React.js**

- ⁴ Menurut survei yang dilakukan oleh StackOverflow pada tahun 2019, Reactjs adalah salah satu
⁵ *framework* javascript yang paling banyak digunakan pada web³. Selain itu menurut survei tersebut,
⁶ Reactjs adalah salah satu *framework* yang paling disukai oleh pengembang. Dengan dua alasan
⁷ tersebut penelitian ini menggunakan Reactjs sebagai *framework* Frontend.

⁸ **3.3.3 CI/CD**

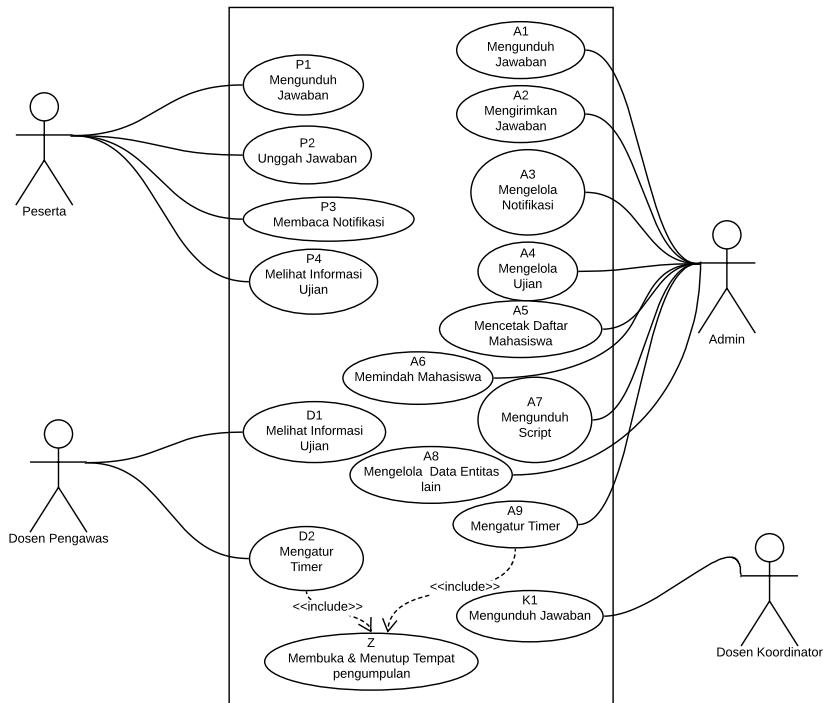
- ⁹ Penggunaan *library* React pada front-end membutuhkan tahap tambahan untuk melakukan *build*
¹⁰ sebelum subsistem dapat dijalankan di peramban pengguna. Adanya tahap tambahan ini membuat
¹¹ penelitian ini membutuhkan sebuah sistem *build* terpusat untuk mempermudah mengelola kode
¹² sumber yang akan di-deploy pada server produksi. Sistem build ini akan diimplementasi dengan
¹³ menggunakan teknologi CI/CD yang disediakan oleh penyedia penyimpanan kode sumber, GitLab.

¹⁴ Sistem CI/CD nantinya harus dapat berjalan hampir otomatis seluruhnya untuk menyederha-
¹⁵ nakan alur pengembangan aplikasi. Sistem CI/CD ini diharapkan untuk mempermudah pekerjaan
¹⁶ pengembang untuk melakukan *build* dengan standar produksi dengan package dan modules untuk
¹⁷ development tidak diikutkan pada hasil *build*. Harapan lainnya adalah dengan adanya CI/CD ini,
¹⁸ pengembang dapat menambahkan *routine* apapun yang dibutuhkan dikemudian hari.

¹⁹ **3.4 Analisis Pengguna**

- ²⁰ Pengguna aplikasi sistem Oxam yang ada saat ini terdapat tiga pihak utama. Pihak pertama adalah
²¹ Tim Admin yang bertugas untuk menyiapkan ruangan untuk ujian. Pihak kedua adalah peserta.
²² Peserta adalah subjek yang akan menggunakan ruangan ujian. Pihak ketiga adalah dosen. Dosen

³https://insights.stackoverflow.com/survey/2019/#technology-__web-frameworks



Gambar 3.9: Diagram *Use Case* untuk aplikasi Oxam yang baru.

terdiri dari dua tugas, dosen pengawas, dan dosen koordinator. Dosen pengawas akan melakukan pengawasan terhadap pelaksanaan ujian, sedangkan dosen koordinator adalah dosen yang akan memberikan ujian, dan mengkoordinasikan dosen pengawas.

Diagram *use case* dari aplikasi ini dapat dilihat pada Gambar 3.9. Pada diagram terdapat empat kasus penggunaan berdasarkan perannya. Untuk peran dosen, peran dibagi menjadi dua berdasarkan tugasnya. Informasi lengkap tentang penggunaan tiap peran dapat dilihat pada Tabel 3.2.

Pengguna sistem ini nantinya akan terdiri dari empat pihak. Pihak pertama, Pihak Administrator. Administrator adalah salah satu pengguna yang memiliki hak akses penuh terhadap sistem manajemen ujian. Pihak ini nantinya akan digunakan oleh tim Administrator Laboratorium. Tim ini nantinya akan membantu menyiapkan ujian, ruangan ujian, dan berkas-berkas administrasi yang nantinya digunakan selama ujian.

Pihak berikutnya, yaitu pihak kedua, nantinya digunakan oleh peserta ujian. Nantinya peserta ini akan menggunakan aplikasi ini untuk mengumpulkan berkas jawaban ujian. Peserta ujian hanya akan dapat mengakses halaman pengumpulan. Pada halaman ini nantinya peserta ujian hanya dapat melihat informasi ujian, informasi ruangan, format nama berkas ujian, dan waktu sisa ujian. Waktu ini disinkronisasi dengan waktu yang terdapat pada server dan layar proyektor.

Nomor	Peran	Deskripsi
P1	Peserta	Peserta dapat mengunduh berkas jawaban yang telah mereka unggah.
P2	Peserta	Peserta dapat mengunggah berkas jawaban pada slot jawaban yang tersedia.
P3	Peserta	Peserta dapat membaca notifikasi yang diberikan oleh sistem.
P4	Peserta	Peserta dapat melihat informasi ujian.
A1	Admin	Tim Admin akan dapat mengunduh seluruh jawaban yang telah peserta unggah.
A2	Admin	Tim Admin dapat mengirimkan jawaban yang telah peserta unggah, secara manual ataupun terjadwal.
A3	Admin	Tim Admin dapat membuat, menghapus, mengubah notifikasi pada ujian tertentu.
A4	Admin	Tim Admin dapat membuat, menghapus, mengubah ujian.
A5	Admin	Tim Admin dapat mencetak daftar tempat duduk peserta ujian.
A6	Admin	Tim Admin dapat memindahkan posisi peserta.
A7	Admin	Tim Admin dapat mengunduh <i>script</i> yang diperlukan.
A8	Admin	Tim Admin dapat membuat, menghapus dan mengubah data entri entitas yang tersedia pada sistem.
A9	Admin	Tim Admin dapat menghentikan, memulai, dan menyetel ulang timer ujian.
Z	Admin dan Dosen Pengawas	Tim Admin dan Dosen pengawas dapat membuka menutup tempat pengumpulan ujian dengan memulai atau menutup timer ujian.
D1	Dosen Pengawas	Dosen pengawas dapat melihat informasi ujian.
D2	Dosen Pengawas	Dosen dapat menghentikan, memulai, dan menyetel ulang timer ujian.
K1	Dosen Koordinator	Dosen koordinator dapat mengunduh berkas jawaban peserta.

Tabel 3.2: Tabel deskripsi lengkap *use case* untuk setiap peran.

3.4.1 Skenario Penggunaan

2 Login untuk Admin

<i>Use case</i>	Login
Aktor	Admin
Tujuan	Mendapatkan akses ke halaman Admin.
Kondisi Awal	Admin mengakses halaman admin
Aksi Aktor	Respon Aplikasi
Memasukan kredensial pada formulir login.	
Menekan tombol Login	Sistem menampilkan halaman admin.
Kondisi Akhir	Admin berhasil masuk ke halaman Admin.
Alternatif:	Perangkat lunak menampilkan pesan kesalahan.

1 Login untuk Dosen Pengawas

<i>Use case</i>	Login dengan IPLogin	
Aktor	Dosen Pengawas	
Tujuan	Mendapatkan akses ke halaman proyektor.	
Kondisi Awal	Pengawas mengakses halaman proyektor	
	Aksi Aktor	Respon Aplikasi
²	Menekan tombol login dengan IPLogin	
Langkah		Sistem menampilkan halaman proyektor.
Kondisi Akhir	Dosen Pengawas berhasil masuk ke halaman proyektor.	
Alternatif:	Perangkat lunak menampilkan pesan kesalahan.	

3 Buat Ujian Baru

<i>Use case</i>	Membuat ujian baru	
Aktor	Admin	
Tujuan	Membuat ujian.	
Kondisi Awal	Admin sudah login dan mengakses halaman daftar ujian	
	Aksi Aktor	Respon Aplikasi
⁴	Menekan tombol buat ujian baru.	Sistem menampilkan formulir detil ujian.
Langkah	Mengisi formulir detil ujian. Menekan tombol <i>seat plotting</i> .	Sistem menampilkan formulir detil ujian.
		Sistem menampilkan peta tempat duduk untuk <i>plotting</i> .
	Memilih tempat duduk yang akan digunakan untuk ujian.	
	Menekan tombol lanjut.	Sistem menampilkan halaman konfirmasi.
	Menekan tombol buat ujian.	Sistem membuat ujian.
		Sistem menampilkan halaman selesai dan tombol lihat ujian.
	Menekan tombol lihat ujian.	Sistem menampilkan halaman detil ujian.
Kondisi Akhir	Ujian terbuat dan admin sedang berada pada halaman detil ujian.	
Alternatif:	Perangkat lunak menampilkan pesan kesalahan.	

1 Mencetak Daftar Hadir

<i>Use case</i>	Mencetak Daftar Hadir
Aktor	Admin
Tujuan	Mencetak daftar hadir untuk ditempelkan di pintu dan absensi peserta.
Kondisi Awal	Admin sudah login dan mengakses halaman detil ujian
Aksi Aktor	Respon Aplikasi
Menekan tombol daftar hadir untuk pintu.	Sistem menampilkan daftar hadir untuk pintu.
Menekan tombol cetak pada peramban.	
Menekan tombol kembali.	
² Langkah	Menampilkan halaman detil ujian.
Menekan tombol daftar hadir untuk absensi peserta.	Menampilkan daftar hadir untuk absensi.
Menekan tombol cetak pada peramban.	
Menekan tombol kembali.	
Kondisi Akhir	Menampilkan halaman detil ujian.
Alternatif:	Daftar hadir tercetak dan admin sedang berada pada halaman detil ujian.

3 Memulai Timer Ujian (Admin)

<i>Use case</i>	Memulai Timer Ujian untuk Admin
Aktor	Admin
Tujuan	Memulai timer ujian.
Kondisi Awal	Admin sudah login dan mengakses halaman detil ujian
Aksi Aktor	Respon Aplikasi
Menekan tombol layar proyektor.	
⁴ Langkah	Menampilkan halaman untuk layar proyektor.
Menekan tombol mulai.	Memulai timer.
Kondisi Akhir	Timer dalam keadaan mulai dan admin sedang berada pada halaman proyektor.
Alternatif:	Perangkat lunak menampilkan pesan kesalahan.

1 Menghentikan Timer Ujian (Admin)

<i>Use case</i>	Menghentikan Timer Ujian untuk Admin	
Aktor	Admin	
Tujuan	Menghentikan timer ujian.	
Kondisi Awal	Admin sudah login dan mengakses halaman detil ujian	
	Aksi Aktor	Respon Aplikasi
	Menekan tombol layar proyektor.	
		Menampilkan halaman untuk layar proyektor.
2 Langkah		
	Menekan tombol stop.	Menghentikan timer.
	Menekan tombol reset.	Mengatur ulang keadaan timer.
Kondisi Akhir	Timer dalam keadaan terhenti dan admin sedang berada pada halaman proyektor.	
Alternatif:	Perangkat lunak menampilkan pesan kesalahan.	

3 Menyetel Ulang Timer Ujian (Admin)

<i>Use case</i>	Menyetel Ulang Timer Ujian untuk Admin	
Aktor	Admin	
Tujuan	Menyetel ulang timer ujian.	
Kondisi Awal	Admin sudah login dan mengakses halaman detil ujian	
	Aksi Aktor	Respon Aplikasi
	Menekan tombol layar proyektor.	
4 Langkah		Menampilkan halaman untuk layar proyektor.
	Menekan tombol reset.	Mengatur ulang keadaan timer.
Kondisi Akhir	Timer dalam keadaan akan mulai dan admin sedang berada pada halaman proyektor.	
Alternatif:	Perangkat lunak menampilkan pesan kesalahan.	

1 Mengunduh Script

<i>Use case</i>	Mengunduh <i>Script</i> .
Aktor	Admin
Tujuan	Mengunduh <i>Script</i> .
Kondisi Awal	Admin sudah login dan mengakses halaman detil ujian
	<hr/>
	Aksi Aktor
	Respon Aplikasi
² Langkah	Menekan tombol unduh <i>script</i> .
	Menyediakan berkas <i>script</i> yang baru saja dibuat dengan data terbaru.
Kondisi Akhir	<i>Script</i> terunduh dan admin sedang berada pada halaman proyektor.
Alternatif:	Perangkat lunak menampilkan pesan kesalahan.

3 Memindahkan Peserta

<i>Use case</i>	Memindahkan peserta.
Aktor	Admin
Tujuan	Memindahkan peserta ke tempat baru.
Kondisi Awal	Admin sudah login dan mengakses halaman detil ujian
	<hr/>
	Aksi Aktor
	Respon Aplikasi
	Menekan tambah pindah peserta.
	menampilkan daftar kosong peserta yang akan dipindah.
	Menekan tombol tambah peserta.
	menampilkan daftar peserta pada ujian yang sedang dipilih.
	Memilih peserta.
	menampilkan peta tempat duduk.
⁴ Langkah	Memilih tempat duduk baru.
	Menambah peserta pada daftar tersebut.
	Memilih peserta dan tempat duduk baru lainnya (jika ada).
	Menambah peserta pada daftar tersebut. (jika ada)
	Menekan tombol pindah.
	Memindahkan peserta tersebut lalu membuatkan <i>script</i> baru untuk migrasi.
Kondisi Akhir	<i>Script</i> terunduh, peserta sudah dimigrasi dan admin sedang berada pada halaman detil ujian.
Alternatif:	Admin melakukan pembatalan.

1 Menambahkan Slot Jawaban

<i>Use case</i>	Menambahkan slot jawaban untuk peserta.	
Aktor	Admin	
Tujuan	Menambahkan slot jawaban untuk peserta	
Kondisi Awal	Admin sudah login dan mengakses halaman detil ujian	
	Aksi Aktor	Respon Aplikasi
	Menekan tombol tambah pada bagian slot jawaban.	
2 Langkah		Menampilkan formulir slot jawaban.
	Mengisi formulir tersebut.	
	Menekan tombol tambah.	
		Menutup formulir dan menambahkan slot jawaban.
Kondisi Akhir	Slot jawaban tertambah dan admin sedang berada pada halaman detil ujian.	
Alternatif:	Admin melakukan pembatalan.	

3 Menambahkan Notifikasi Kata Sandi

<i>Use case</i>	Menambahkan notifikasi kata sandi untuk peserta	
Aktor	Admin	
Tujuan	Mendistribusikan kredensial untuk peserta.	
Kondisi Awal	Admin sudah login dan mengakses halaman detil ujian	
	Aksi Aktor	Respon Aplikasi
	Menekan tombol tambah pada Notifikasi.	
		Menanyakan jenis notifikasi.
4 Langkah	Memilih jenis <i>Password</i> atau kata sandi.	
		Menampilkan formulir notifikasi.
	Mengisi formulir tersebut.	
	Menekan tombol tambah.	
		Menutup formulir dan menyebar notifikasi.
Kondisi Akhir	Notifikasi tertambah dan admin sedang berada pada halaman detil ujian.	
Alternatif:	Admin melakukan pembatalan.	

1 Menambahkan Notifikasi Lainnya

<i>Use case</i>	Menambahkan notifikasi generik untuk peserta
Aktor	Admin
Tujuan	Mendistribusikan informasi untuk peserta.
Kondisi Awal	Admin sudah login dan mengakses halaman detil ujian
Aksi Aktor	Respon Aplikasi
Menekan tombol tambah pada Notifikasi.	Menanyakan jenis notifikasi.
² Langkah	Memilih jenis lainnya.
	Menampilkan formulir notifikasi.
	Mengisi formulir tersebut.
	Menekan tombol tambah.
Kondisi Akhir	Menutup formulir dan menyebar notifikasi.
Alternatif:	Notifikasi tertambah dan admin sedang berada pada halaman detil ujian.
	Admin melakukan pembatalan.

1 Mengirimkan Laporan Jawaban

<i>Use case</i>	Mengirimkan laporan ujian	
Aktor	Admin	
Tujuan	Mengirikan jawaban peserta ujian pada dosen.	
Kondisi Awal	Admin sudah login dan mengakses halaman detil ujian	
	Aksi Aktor	Respon Aplikasi
	Menekan tombol tambah pada bagian <i>Autoreport</i>	Menanyakan tujuan email
Langkah	Mengisi tujuan email	
		Merekan permintaan
		Menutup formulir
		Pada saat ujian selesai, sistem mengirimkan email untuk dosen dapat mengakses link pengunduhan.
² Kondisi Akhir	Jawaban dijadwalkan untuk dikirim dan admin sedang berada pada halaman detil ujian.	
Alternatif:	Admin melakukan pengiriman paksa.	
	Aksi Aktor	Respon Aplikasi
	Menekan tombol kirim pada salah satu entri di bagian <i>Autoreport</i>	Mengirim link pengunduhan pada email yang dituju
Alternatif:	Admin melakukan pengunduhan jawaban saja.	
	Aksi Aktor	Respon Aplikasi
	Menekan tombol unduh pada bagian slot jawaban	Menyediakan berkas jawaban ujian pada admin

1 Penghapusan Ujian

<i>Use case</i>	Menghapus Ujian
Aktor	Admin
Tujuan	Menghapus ujian
Kondisi Awal	Admin sudah login dan mengakses halaman daftar ujian
	Aksi Aktor
	Respon Aplikasi
	Menekan tombol hapus pada entri ujian yang ingin dihapus.
2 Langkah	Menampilkan modal konfirmasi penghapusan
	Melakukan konfirmasi.
	Menghapus ujian.
Kondisi Akhir	Ujian terhapus dan admin sedang berada pada halaman detail ujian.
Alternatif: Admin melakukan pembatalan.	

3 Manipulasi Entitas: Menambah Entri baru

<i>Use case</i>	Menambah Entri baru pada entitas tertentu
Aktor	Admin
Tujuan	Memanipulasi entitas dengan menambah entri baru
Kondisi Awal	Admin sudah login dan mengakses halaman daftar ujian
	Aksi Aktor
	Respon Aplikasi
	Menekan tombol entitas (Exam Param atau Exam Param) pada menu.
4 Langkah	Menampilkan daftar entri pada entitas tersebut
	Menekan tombol tambah.
	Menampilkan formulir yang berisi data yang dibutuhkan entitas tersebut.
	Mengisi formulir.
	Menekan tombol simpan.
Kondisi Akhir	Sistem merekam permintaan tersebut dan membuat entri baru
Alternatif: Admin melakukan pembatalan.	Sistem mengarahkan pada halaman pengubah entitas
	Entri baru dibuat admin sedang berada pada halaman pengubah entri.

1 Manipulasi Entitas: Mengubah Entri baru

	<i>Use case</i>	Mengubah Entri baru pada entitas tertentu
	Aktor	Admin
	Tujuan	Memanipulasi entitas dengan mengubah entri baru
	Kondisi Awal	Admin sudah login dan mengakses halaman daftar ujian
	Aksi Aktor	Respon Aplikasi
		Menekan tombol entitas (Exam Param atau Exam Param) pada menu.
		Menampilkan daftar entri pada entitas tersebut
2 Langkah		Menekan tombol <i>edit</i> pada entri yang ingin diubah.
		Menampilkan formulir kolom entri
		Ubah data pada kolom tersebut
		Sistem mengubah entri tersebut.
	Kondisi Akhir	Detail entri diubah admin sedang berada pada halaman pengubah entri.
	Alternatif: Admin melakukan pembatalan.	

1 Manipulasi Entitas: Menghapus Entri baru

	<i>Use case</i>	Menghapus Entri baru pada entitas tertentu
	Aktor	Admin
	Tujuan	Memanipulasi entitas dengan menghapus entri baru
	Kondisi Awal	Admin sudah login dan mengakses halaman daftar ujian
	Aksi Aktor	Respon Aplikasi
	Menekan tombol entitas (Exam Param atau Exam Param) pada menu.	Menampilkan daftar entri pada entitas tersebut
	Menekan tombol hapus pada entri yang ingin dihapus.	Menampilkan modal konfirmasi penghapusan.
	Mengkonfirmasi dengan menekan tombol hapus.	Sistem menghapus entri tersebut dan mengarahkan pada daftar entri.
	Kondisi Akhir	Entri terhapus admin sedang berada pada halaman daftar entri.
	Alternatif: Admin melakukan pembatalan.	

3 Melihat Ujian akan Diadakan

	<i>Use case</i>	Melihat informasi ujian
	Aktor	Peserta
	Tujuan	Melihat informasi ujian
	Kondisi Awal	-
	Aksi Aktor	Respon Aplikasi
	Mengakses aplikasi ujian pada komputer peserta	Menampilkan halaman informasi ujian yang akan aktif
	Langkah	
	Kondisi Akhir	Peserta sedang berada pada halaman informasi ujian.
	Alternatif: Peserta tidak berada pada komputer yang tepat.	

1 Melihat Notifikasi

<i>Use case</i>	Melihat informasi notifikasi	
Aktor	Peserta	
Tujuan	Melihat informasi notifikasi	
Kondisi Awal	Peserta sedang mengakses aplikasi ujian	
	Aksi Aktor	Respon Aplikasi
	Mengakses menu notifikasi	
		Menampilkan daftar notifikasi yang aktif.
2 Langkah	Memilih salah satu notifikasi	Menampilkan notifikasi.
	Menekan tombol silang.	Menutup notifikasi.
Kondisi Akhir	Peserta sudah melihat notifikasi, dan peserta sedang berada pada halaman informasi ujian.	
Alternatif:	-	

3 Mengunggah Berkas Jawaban

<i>Use case</i>	Mengunggah berkas jawaban	
Aktor	Peserta	
Tujuan	Menunggah berkas jawaban	
Kondisi Awal	Peserta sedang mengakses aplikasi ujian, timer sedang berjalan	
	Aksi Aktor	Respon Aplikasi
	Menekan tombol unggah pada salah satu slot jawaban yang tersedia.	
Langkah		Menampilkan <i>file picker</i> untuk memilih berkas.
	Memilih berkas	
		Mengunggah berkas.
4		Menyimpan berkas yang diunggah.
Kondisi Akhir	Berkas jawaban terunggah dan peserta sedang berada pada halaman informasi ujian.	
	Aksi Aktor	Respon Aplikasi
	Memilih berkas yang salah	
Alternatif: Peserta mengunggah berkas jawaban yang namanya tidak sesuai	Melakukan konfirmasi.	Mengkonfirmasi apakah nama berkas ingin diganti otomatis.
		Mengunggah berkas.
		Menyimpan berkas yang diunggah.

1 Mengunduh Berkas Jawaban

<i>Use case</i>	Mengunduh berkas jawaban
Aktor	Peserta
Tujuan	Mengunduh berkas jawaban
Kondisi Awal	Peserta sedang mengakses aplikasi ujian, timer sedang berjalan
	Aksi Aktor
	Respon Aplikasi
² Langkah	Menekan tombol unduh pada slot jawaban yang sudah terisi
	Sistem menyediakan berkas tersebut untuk pengunduhan.
Kondisi Akhir	Berkas jawaban terunduh dan peserta sedang berada pada halaman informasi ujian.
Alternatif: -	

3 Mengunduh Berkas Jawaban

<i>Use case</i>	Mengunduh berkas jawaban
Aktor	Peserta
Tujuan	Melihat informasi notifikasi
Kondisi Awal	Peserta sedang mengakses aplikasi ujian, timer sedang berjalan
	Aksi Aktor
	Respon Aplikasi
⁴ Langkah	Menekan tombol unduh pada slot jawaban yang sudah terisi
	Sistem menyediakan berkas tersebut untuk pengunduhan.
Kondisi Akhir	Berkas jawaban terunduh dan peserta sedang berada pada halaman informasi ujian.
Alternatif: -	

1 Memulai Timer Ujian (Dosen Pengawas)

<i>Use case</i>	Memulai Timer Ujian untuk Dosen Pengawas
Aktor	Dosen Pengawas
Tujuan	Memulai timer ujian.
Kondisi Awal	Dosen pengawas sudah login dan sedang berada pada halaman layar proyektor
	Aksi Aktor Respon Aplikasi
2	Menekan tab timer.
Langkah	Menampilkan halaman untuk timer.
	Menekan tombol mulai.
	Memulai timer.
Kondisi Akhir	Timer dalam keadaan mulai dan Dosen Pengawas sedang berada pada halaman proyektor.
Alternatif:	Perangkat lunak menampilkan pesan kesalahan.

3 Menghentikan Timer Ujian (Dosen Pengawas)

<i>Use case</i>	Menghentikan Timer Ujian untuk Dosen Pengawas
Aktor	Dosen Pengawas
Tujuan	Menghentikan timer ujian.
Kondisi Awal	Dosen pengawas sudah login dan sedang berada pada halaman layar proyektor
	Aksi Aktor Respon Aplikasi
4	Menekan tab timer.
Langkah	Menampilkan halaman untuk timer.
	Menekan tombol stop.
	Menghentikan timer.
Kondisi Akhir	Timer dalam keadaan terhenti dan Dosen Pengawas sedang berada pada halaman proyektor.
Alternatif:	Perangkat lunak menampilkan pesan kesalahan.

1 Menyetel Ulang Timer Ujian (Dosen Pengawas)

<i>Use case</i>	Menyetel Ulang Timer Ujian untuk Dosen Pengawas	
Aktor	Dosen Pengawas	
Tujuan	Menyetel ulang timer ujian.	
Kondisi Awal	Dosen pengawas sudah login dan sedang berada pada halaman layar proyektor	
	Aksi Aktor	Respon Aplikasi
²	Menekan tab timer.	
Langkah		Menampilkan halaman untuk timer.
	Menekan tombol reset.	Mengatur ulang keadaan timer.
Kondisi Akhir	Timer dalam keadaan akan mulai dan Dosen Pengawas sedang berada pada halaman proyektor.	
Alternatif:	Perangkat lunak menampilkan pesan kesalahan.	

3 Menambah Waktu Ujian (*Overtime*)

<i>Use case</i>	Menambahkan waktu ujian	
Aktor	Dosen Pengawas	
Tujuan	Menambah waktu ujian.	
Kondisi Awal	Dosen pengawas sudah login dan sedang berada pada halaman layar proyektor	
	Aksi Aktor	Respon Aplikasi
⁴	Menekan tab timer.	
Langkah		Menampilkan halaman untuk timer.
	Menekan tombol tambah waktu.	Menampilkan formulir untuk menambahkan waktu.
	Mengisi formulir tambahan waktu dalam satuan menit.	Menambahkan waktu timer.
Kondisi Akhir	Waktu untuk timer sudah ditambahkan dan Dosen Pengawas sedang berada pada halaman proyektor.	
Alternatif:	Perangkat lunak menampilkan pesan kesalahan.	

1 Mengunduh Jawaban (Dosen Koordinator)

<i>Use case</i>	Mengunduh Jawaban
Aktor	Dosen Koordinator
Tujuan	Mengunduh berkas jawaban yang telah diunggah oleh peserta ujian.
Kondisi Awal	Dosen Koordinator sudah mendapatkan email ujian
	<hr/>
Aksi Aktor	Respon Aplikasi
	Menekan tautan pada email.
<hr/>	
2 Langkah	Menampilkan halaman informasi ujian dan tombol unduh.
	Menekan tombol unduh.
	Sistem menyediakan berkas untuk diunduh.
Kondisi Akhir	Berkas terunduh, Dosen Koordinator masih pada halaman pengunduhan.
Alternatif:	Tautan sudah kadaluarsa

¹

BAB 4

²

PERANCANGAN

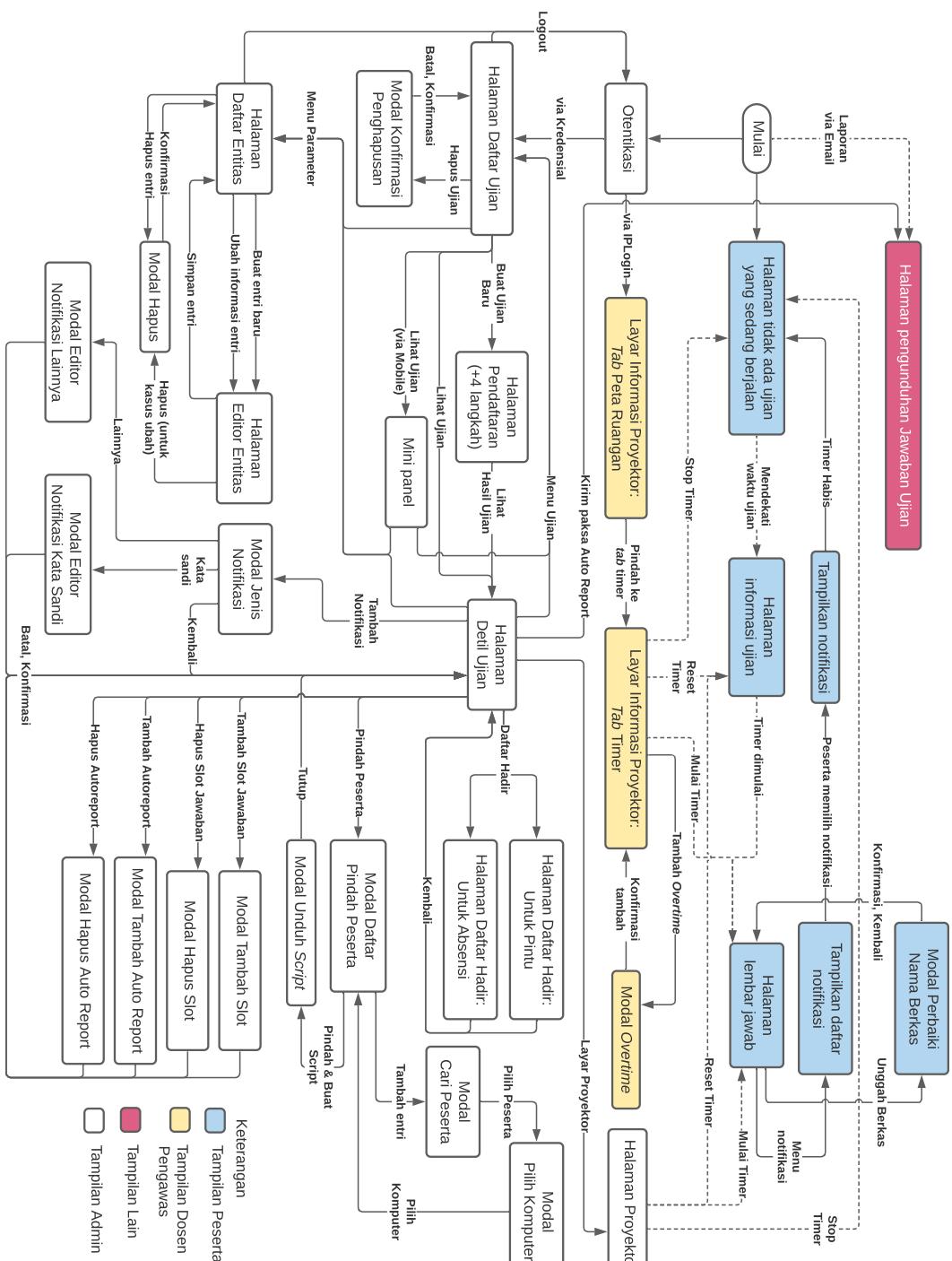
³ 4.1 Rancangan Antarmuka

⁴ Peracangan antarmuka dilakukan dengan membuat *mockup* aplikasi. Desain *mockup* dibagi menjadi
⁵ tiga bagian besar berdasarkan peranan penggunanya. Setiap pengguna akan memiliki antarmukanya
⁶ tersendiri. Secara garis besar, diagram alur antarmuka untuk aplikasi ini dapat dilihat pada Gambar
⁷ 4.1.

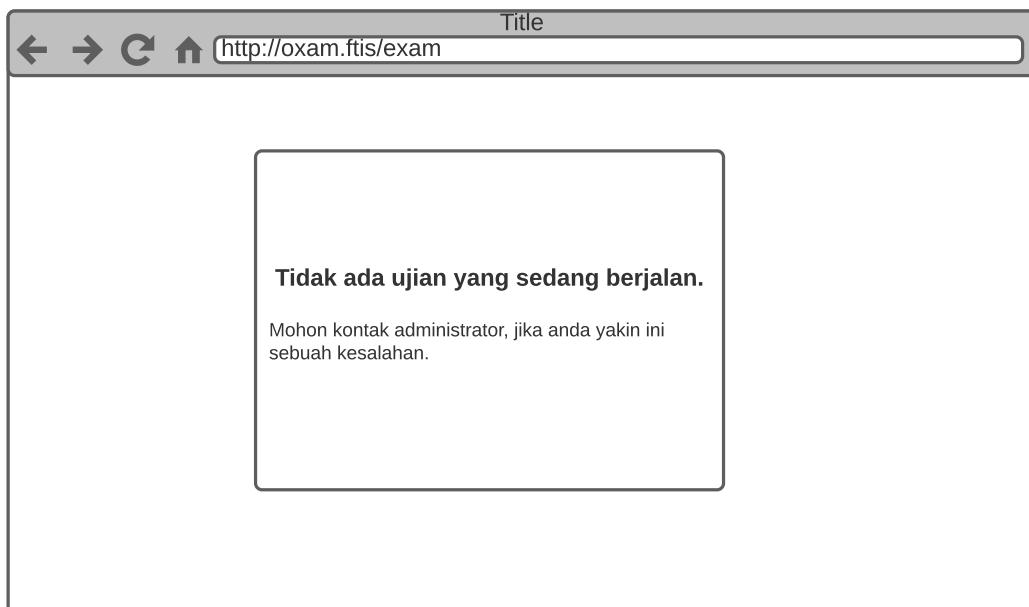
⁸ Pembahasan akan dimulai dengan rancangan antarmuka untuk Peserta, lalu perancangan
⁹ antarmuka untuk Admin, Dosen Pengawas, dan kemudian rancangan antarmuka tambahan.

¹⁰ 4.1.1 Rancangan Antarmuka untuk Peserta

¹¹ Rancangan antarmuka untuk peserta didasari dari kasus penggunaan dari aplikasi. Berdasarkan
¹² hasil analisis pada survei, peserta sudah merasa nyaman dengan antarmuka pada sistem yang lama.
¹³ Sehingga penulis mengadopsi antarmuka yang lama dan diperbarui dengan kebutuhan yang baru.
¹⁴ Beberapa kebutuhan yang baru seperti penutupan dan pembukaan lembar jawaban, informasi ujian
¹⁵ dan waktu, serta fitur notifikasi untuk kredensial login dan pengumuman lainnya.

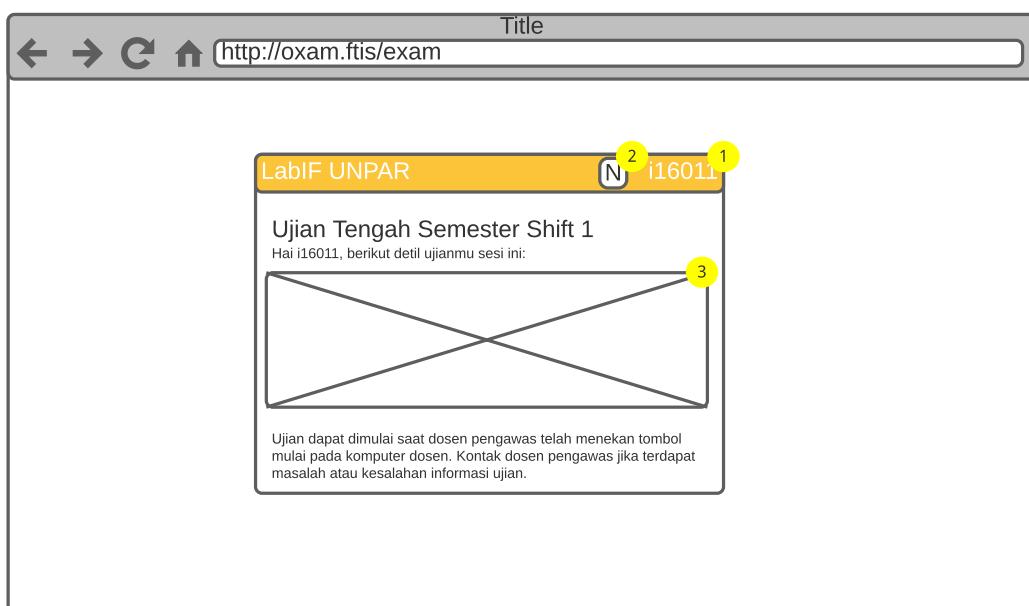


Gambar 4.1: Diagram alur antarmuka secara keseluruhan.



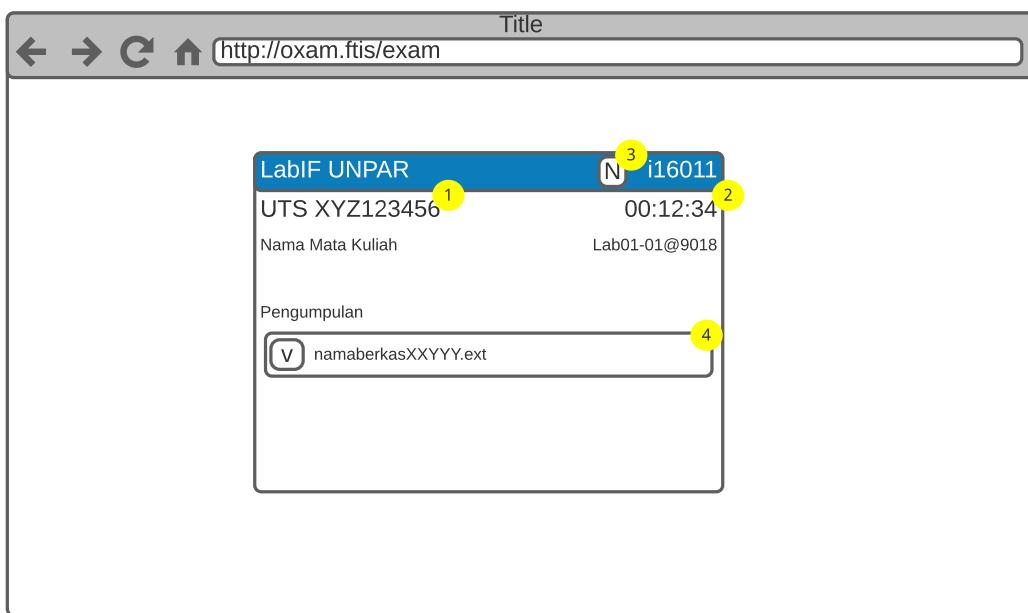
Gambar 4.2: Rancangan antarmuka untuk peserta, saat ujian sedang tidak berjalan.

- 1 Tampilan antarmuka pertama yang akan dibahas adalah tampilan untuk menunjukkan informasi bahwa tidak ada ujian yang sedang berjalan, dapat dilihat pada Gambar 4.2. Hal ini diperuntukkan untuk membantu tim admin melakukan pengecekan lebih cepat karena tampilan layar yang beda secara signifikan. Selain itu, tampilan pesan yang cukup ramah untuk pengguna menampilkan informasi bahwa mungkin peserta mengambil tempat duduk yang salah.



Gambar 4.3: Rancangan antarmuka untuk peserta, saat ujian akan berjalan.

- 6 Rancangan berikutnya adalah tampilan antarmuka untuk memulai ujian, dapat dilihat pada



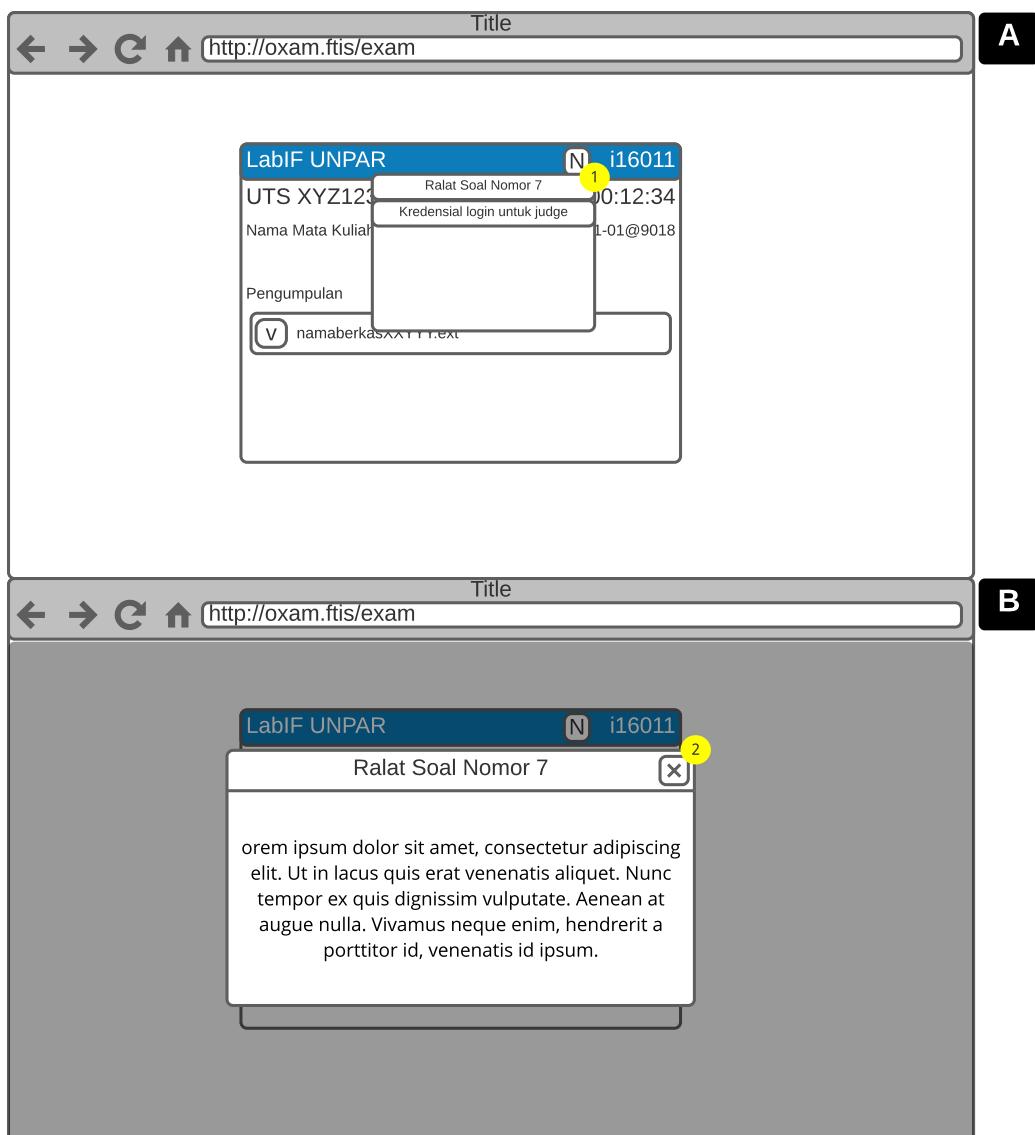
Gambar 4.4: Rancangan antarmuka untuk peserta, saat ujian sedang berjalan.

1 4.3. Tampilan ini akan ditampilkan pada saat ujian akan dimulai di ruangan. Tampilan ini dibuat
2 untuk menahan peserta untuk langsung mengakses lembar jawaban ujian. Pada bagian ini terdapat
3 beberapa informasi tentang peserta (poin 1), fitur notifikasi untuk mengecek kredensial login lainnya
4 (poin 2), dan informasi lengkap ujian dalam bentuk tabel (poin 3). Selain itu terdapat deskripsi
5 kecil untuk membantu peserta mengerti konteks dari tampilan layar ini.

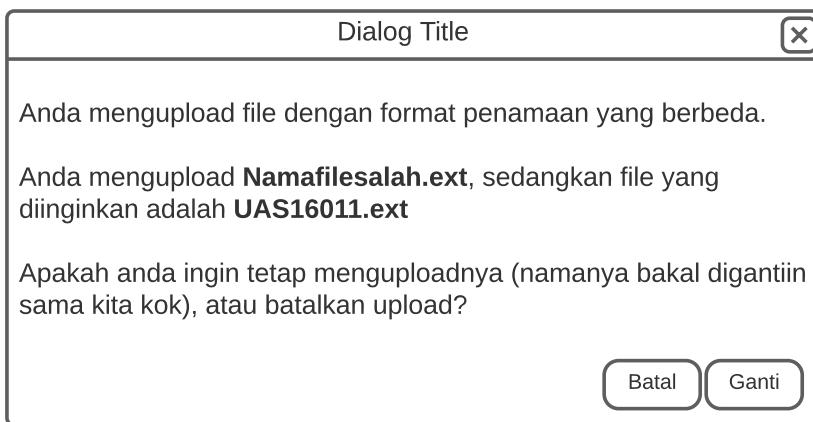
6 Antarmuka berikutnya adalah antarmuka saat ujian sedang berjalan, dapat dilihat pada Gambar
7 4.4. Tampilan rancangan antarmuka ini memiliki beberapa bagian yang mirip dengan tampilan
8 sebelumnya seperti bagian kepala (poin 3), namun dilengkapi dengan informasi singkat tentang
9 ujian (poin 1) dan timer ujian serta lokasi tempat duduk (poin 2). Selain itu, untuk mempercepat
10 admin melihat status pada layar, bagian kepala diwarnai berbeda dengan tampilan sebelumnya
11 (dari kuning menjadi biru).

12 Bagian yang ditunjukkan oleh poin 4 adalah bagian pengunggahan. Bagian ini memiliki informasi
13 tentang nama berkas yang harus dikumpulkan dan tombol unggah dan unduh. Tombol unduh akan
14 dapat diklik pada saat peserta berhasil mengunggah. Tombol unduh ditambahkan untuk alasan
15 pengecekan. Berdasarkan survei lapangan, peserta biasanya akan diarahkan untuk melakukan
16 pengecekan ulang. Pengencekan tersebut melibatkan peserta untuk mengunduh berkas ujian
17 untuk kemudian dipastikan apakah berkas jawaban tersebut sudah sesuai dengan apa yang mereka
18 kirimkan.

19 Rancangan berikutnya adalah fitur notifikasi. Fitur ini akan dimunculkan sejak tahap ujian
20 sedang akan dimulai. Pada Gambar 4.3 dan 4.4 dapat dilihat bahwa terdapat tombol dengan
21 lambang 'N' dekat dengan informasi singkat peserta. tombol tersebut adalah tombol notifikasi.
22 Jika tombol tersebut diklik, maka akan tampil sebuah daftar notifikasi dalam bentuk *dropdown*
23 (Perhatikan Gambar 4.5, bagian A). Pada rancangan tampilan, peserta dapat melihat beberapa
24 notifikasi yang telah diberikan untuk peserta tersebut (perhatikan Poin 1). Jika Poin 1 diklik,



Gambar 4.5: Rancangan antarmuka keseluruhan untuk notifikasi peserta.
(A) Daftar notifikasi; (B) Notifikasi terbuka.



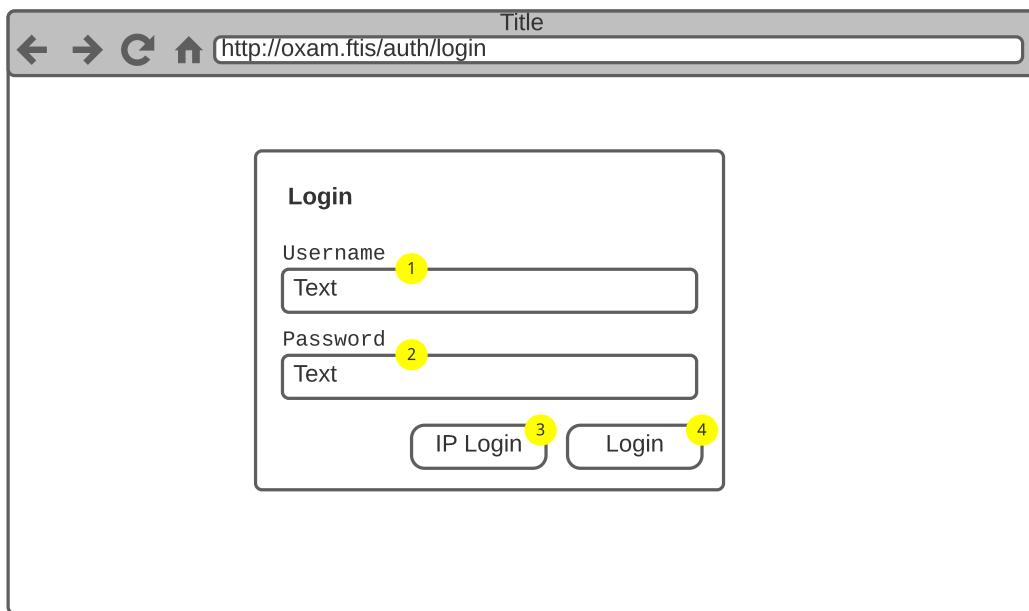
Gambar 4.6: Rancangan tampilan untuk pengumpulan jawaban dengan nama berkas yang tidak sesuai.

- 1 maka notifikasi akan ditampilkan secara modal (Perhatikan Gambar 4.5, bagian B). Tampilan
2 interaktif dengan modal dipilih untuk menunjukkan bahwa peserta dapat menutup modal tersebut
3 tanpa kehilangan formulir lembar pengumpulan jawaban ujian. Selain itu dengan menggunakan
4 modal, tataletak yang sudah ada tidak berubah, dan memperkecil *learning curve* untuk peserta
5 membiasakan diri dengan aplikasi ini.
- 6 Rancangan antarmuka lainnya adalah tampilan pesan peringatan pengubahan nama berkas
7 untuk peserta pada saat peserta mengirimkan berkas dengan nama yang tidak sesuai. Sistem
8 akan mengkonfirmasi peserta untuk mengganti nama berkas sebelum menungguh berkas ke server.
9 Rancangan tersebut dapat dilihat pada Gambar 4.6.

10 **4.1.2 Rancangan Antarmuka untuk Admin**

11 **Otentikasi**

- 12 Untuk dapat mengakses halaman admin panel, pengguna diharuskan untuk melakukan otentikasi
13 terlebih dahulu. Rancangan tampilan admin panel tersebut dapat dilihat pada Gambar 4.7. Pada
14 rancangan tampilan, terdapat beberapa bagian yang dapat pengguna isi untuk login. Bagian yang
15 ditunjukan oleh poin 1 adalah bidang username dari admin. Bidang ini rancang untuk dapat
16 menerima namapengguna dan email. Lalu bagian berikutnya adalah bidang yang ditunjukkan
17 oleh poin 2. Bidang tersebut adalah bidang kata sandi. Lalu pada bagian yang ditunjukan oleh
18 poin 3 dan 4 adalah tombol aksi untuk melakukan login. Tombol yang ditunjukkan oleh poin
19 3 tidak memanfaatkan bidang namapengguna dan kata sandi, namun akan langsung melakukan
20 pemanggilan API ke backend untuk melakukan otentikasi secara nir-kata sandi dengan bantuan IP.
21 Proses ini akan dijelaskan lebih lanjut pada bagian rancangan antarmuka untuk dosen pengawas.
22 Bagian yang ditunjukkan pada poin 4 adalah tombol login sesungguhnya. Pengguna yang telah
23 dipastikan sebagai admin akan dapat melakukan login dan diarahkan menuju daftar ujian. Namun
24 jika pengguna memasukkan kredensial login yang salah sebuah pesan kesalahan akan dimunculkan
25 dalam bentuk *Alert*.



Gambar 4.7: Rancangan antarmuka untuk halaman otentikasi untuk Admin.

¹ Daftar Ujian

² Rancangan antarmuka untuk manajemen ujian dimulai dengan halaman daftar ujian, dapat dilihat ³ pada Gambar 4.8. Pada rancangan tersebut secara garis besar menampilkan informasi tentang ⁴ ujian-ujian yang sudah terdaftar pada sistem berserta informasi singkatnya. Bagian-bagian yang ⁵ ditunjukkan dengan poin dijelaskan sebagai berikut:

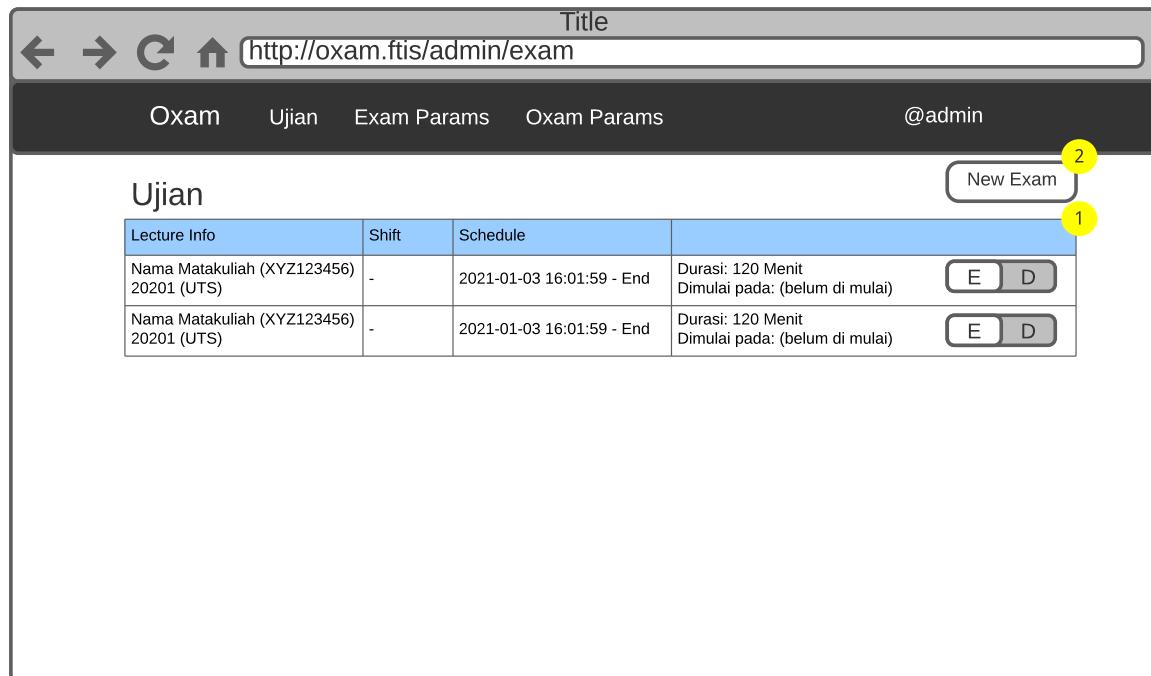
- ⁶ • Bagian yang ditunjukkan pada Poin 1 adalah daftar ujian yang sudah teregistrasi pada sistem. ⁷ Daftar tersebut disajikan dalam bentuk tabel yang kemudian dapat diklik untuk menuju ⁸ halaman detail ujian. Tabel akan menunjukkan setidaknya informasi mata kuliah, periode ujian, ⁹ UTS atau UAS, *shift*, serta tanggal dan jam mulai serta berakhirnya ujian jika ujian sudah ¹⁰ dimulai.

¹¹ Selain itu setiap tabel akan memiliki tombol lihat dan tombol hapus. Tombol hapus akan mem-¹² buka modal konfirmasi hapus yang dapat dilihat pada gambar 4.9. Modal akan menampilkan ¹³ informasi singkat ujian yang akan dihapus untuk konfirmasi.

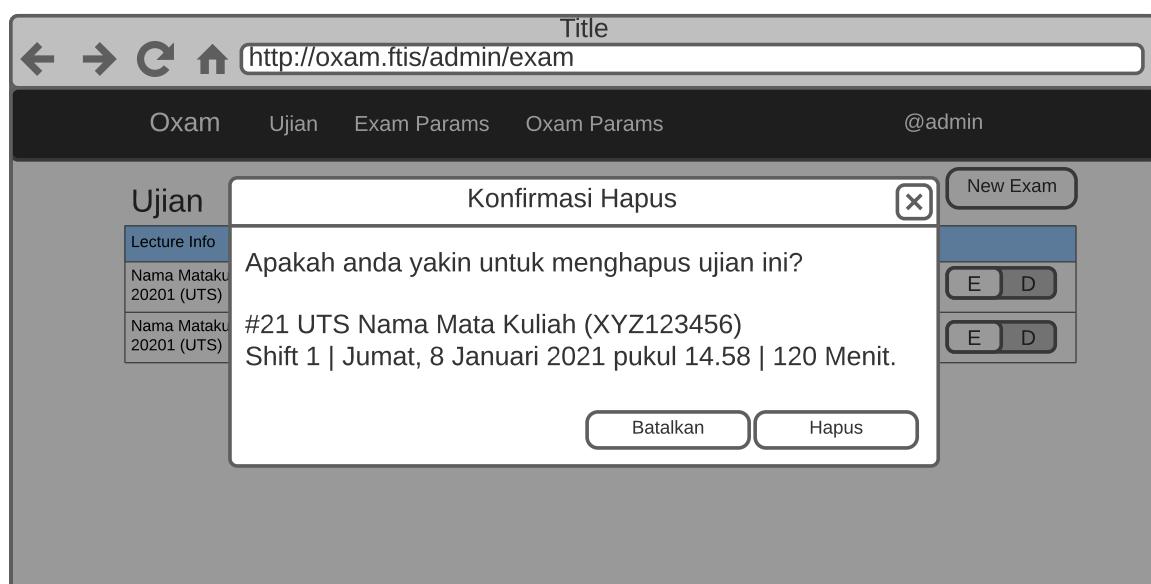
- ¹⁴ • Bagian yang ditunjukkan pada Poin 2 adalah tombol aksi untuk membuat ujian yang baru.

¹⁵ Pembuatan Ujian Baru

¹⁶ Untuk mendaftarkan ujian Admin harus mengisi formulir yang disediakan oleh sistem. Formulir ¹⁷ akan dipisah menjadi beberapa langkah untuk mempermudah pengecekan. Formulir pada langkah ¹⁸ pertama dapat dilihat pada Gambar 4.10. Halaman ini dapat diakses pada saat Admin menekan ¹⁹ tombol yang ditunjukkan oleh poin 2 pada gambar 4.8. Poin-poin yang terdapat pada gambar ²⁰ dijelaskan sebagai berikut:



Gambar 4.8: Rancangan antarmuka untuk daftar ujian pada halaman Admin Panel.



Gambar 4.9: Rancangan antarmuka untuk modal konfirmasi penghapusan ujian.

Title
http://oxam.ftis/admin/exam/new

Oxam Ujian Exam Params Oxam Params @admin

Exam Plotter

1 <Stepper>

Tipe Ujian Mata Kuliah
2 UTS XYZ123456

Shift Mulai Pada Selama
Tidak Ada Shift 3 1/1/10 4 120 Menit

Peserta
Text
5 atau, unggah berkas daftar peserta?
Upload

Seat Plotting >

Gambar 4.10: Rancangan antarmuka untuk membuat ujian baru, langkah pertama dari empat.

- 1 • Poin 1 adalah tampilan *stepper*. *Stepper* menunjukkan langkah yang harus ditempuh oleh tim
2 admin untuk membuat mengisi formulir ini. Untuk saat ini terdapat empat langkah besar
3 yang terdiri dari:

- 4 1. Pengisian detil ujian (*Exam Details*).
5 2. Pangalokasian tempat duduk ujian (*Seat Plotting*).
6 3. Konfirmasi (*Confirmation*).
7 4. Penyelesaian (*Finish*).

8 Seperti yang telah dijelaskan, halaman formulir ini berada pada tahap pengisian detil ujian.

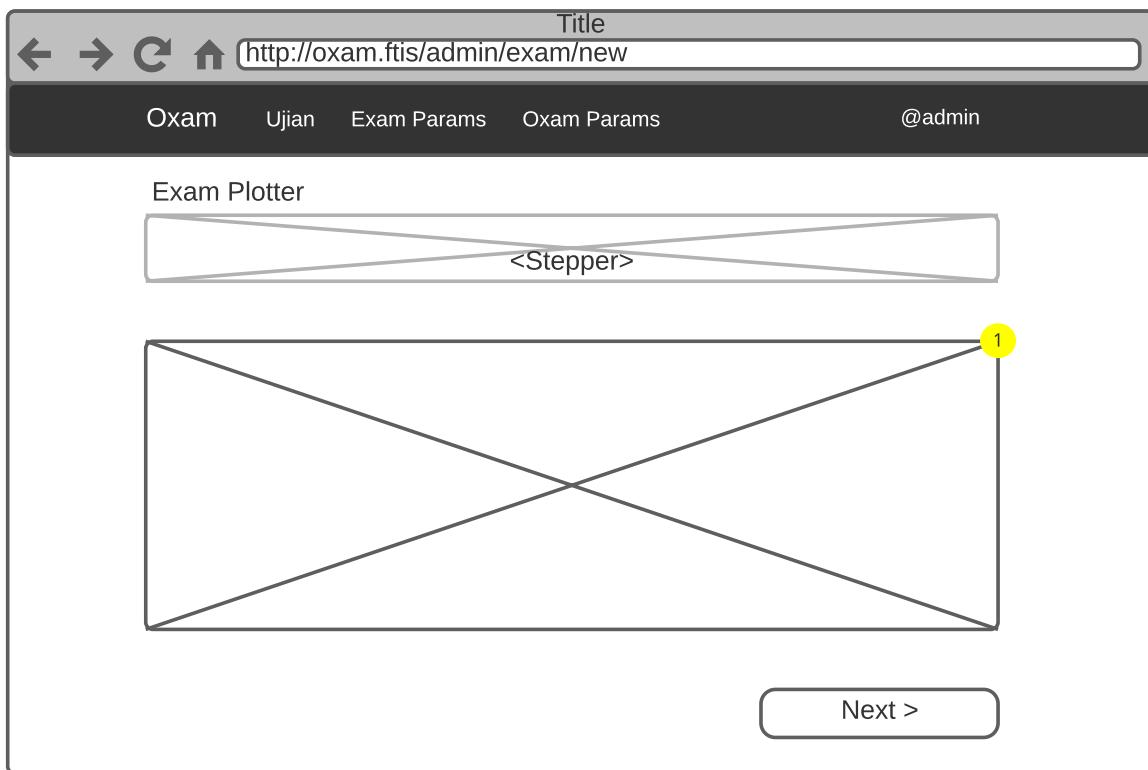
- 9 • Poin 2 menunjukkan bagian detil ujian seperti tipe ujian yang dapat diisi sebagai UTS atau
10 pun UAS, mata kuliah datu ujian ini, *shift*, mulai pada tanggal jam berapa, hingga durasi waktu
11 ujian. Informasi yang ditampilkan pada bagian ini berhubungan langsung dengan entitas **Exam**.
12 Sesuai dengan jenis bidangnya, jika sebuah bidang memiliki pilihan spesifik, maka bidang
13 tersebut akan diimplementasi dengan *dropdown*. Sebagai contoh bidang tipe ujian hanya
14 akan memiliki pilihan UTS atau pun UAS, maka bidang diimplementasi dengan *dropdown*
15 dengan nilai pilihan yang sudah disebutkan. Untuk bidang tanggal dan jam, implementasi
16 akan dilakukan dengan bantuan dari *date picker*.

- 17 • Poin 3 menunjukkan daftar peserta yang akan mengikuti ujian ini. Daftar peserta berisi daftar
18 NPM yang mengikuti ujian, dipisah dengan sebuah enter.
19 • Poin 4 adalah sebuah tombol unggah yang akan diproses langsung pada browser untuk
20 memasukkan daftar peserta. Tombol ini ditambahkan dengan alasan kompatibilitas UX
21 dengan sistem yang lama.
22 • Poin 5 adalah sebuah tombol aksi untuk menuju langkah berikutnya, pengalokasian tempat
23 duduk ujian.

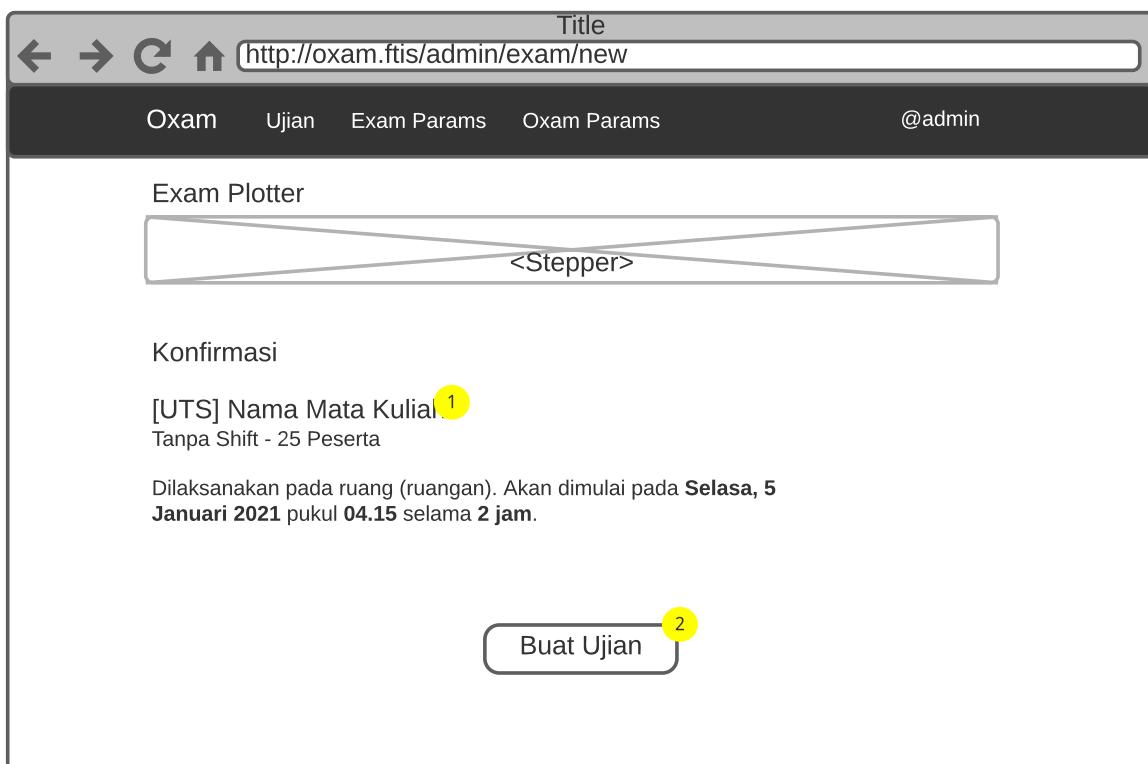
24 Setelah tim admin selesai mengisi informasi detil ujian, maka langkah selanjutnya yang akan
25 dilakukan adalah mengalokasi tempat duduk ujian. Alokasi ini dilakukan dengan bantuan peta
26 ruangan yang telah didefinisikan sebelumnya oleh sistem. Rancangan antarmuka tersebut dapat
27 dilihat pada Gambar 4.11. Secara garis besar tampilan memiliki tataletak yang mirip dengan
28 langkah sebelumnya: Judul langkah diatas, diikuti dengan *stepper* di bawahnya. Namun bagian
29 yang ditunjukkan pada poin 1 akan berisi peta tempat duduk ruangan. Peta tempat duduk ini
30 nantinya akan memiki tampilan yang sama dengan yang ditampilkan pada proyektor.

31 Setelah tempat duduk dialokasi, langkah berikutnya adalah konfirmasi. Tim admin diharapkan
32 untuk mengkonfirmasi detil ujian yang akan dibuat sebelum akhirnya difinialisasi oleh sistem.
33 Rancangan tampilan layar tersebut dapat diperhatikan pada 4.12. Pada rancangan tampilan
34 tersebut terdapat informasi singkat dari ujian yang akan dibuat. Bagian yang ditunjukkan dengan
35 poin-poin pada gambar akan dijelaskan sebagai berikut:

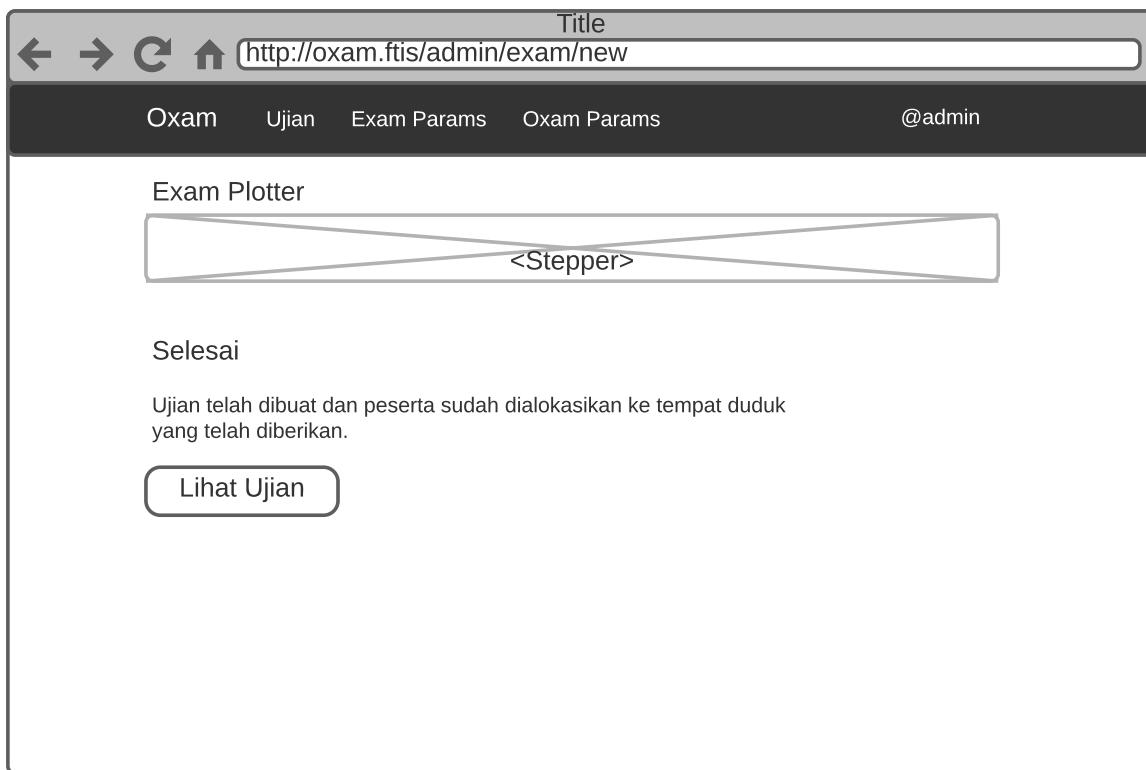
- 36 • Poin 1 menunjukkan informasi tentang nama matakuliah, banyak peserta, informasi shift
37 serta informasi jadwal ujian tersebut.



Gambar 4.11: Rancangan antarmuka untuk membuat ujian baru, langkah kedua dari empat.



Gambar 4.12: Rancangan antarmuka untuk membuat ujian baru, langkah ketiga dari empat.



Gambar 4.13: Rancangan antarmuka untuk membuat ujian baru, langkah keempat dari empat.

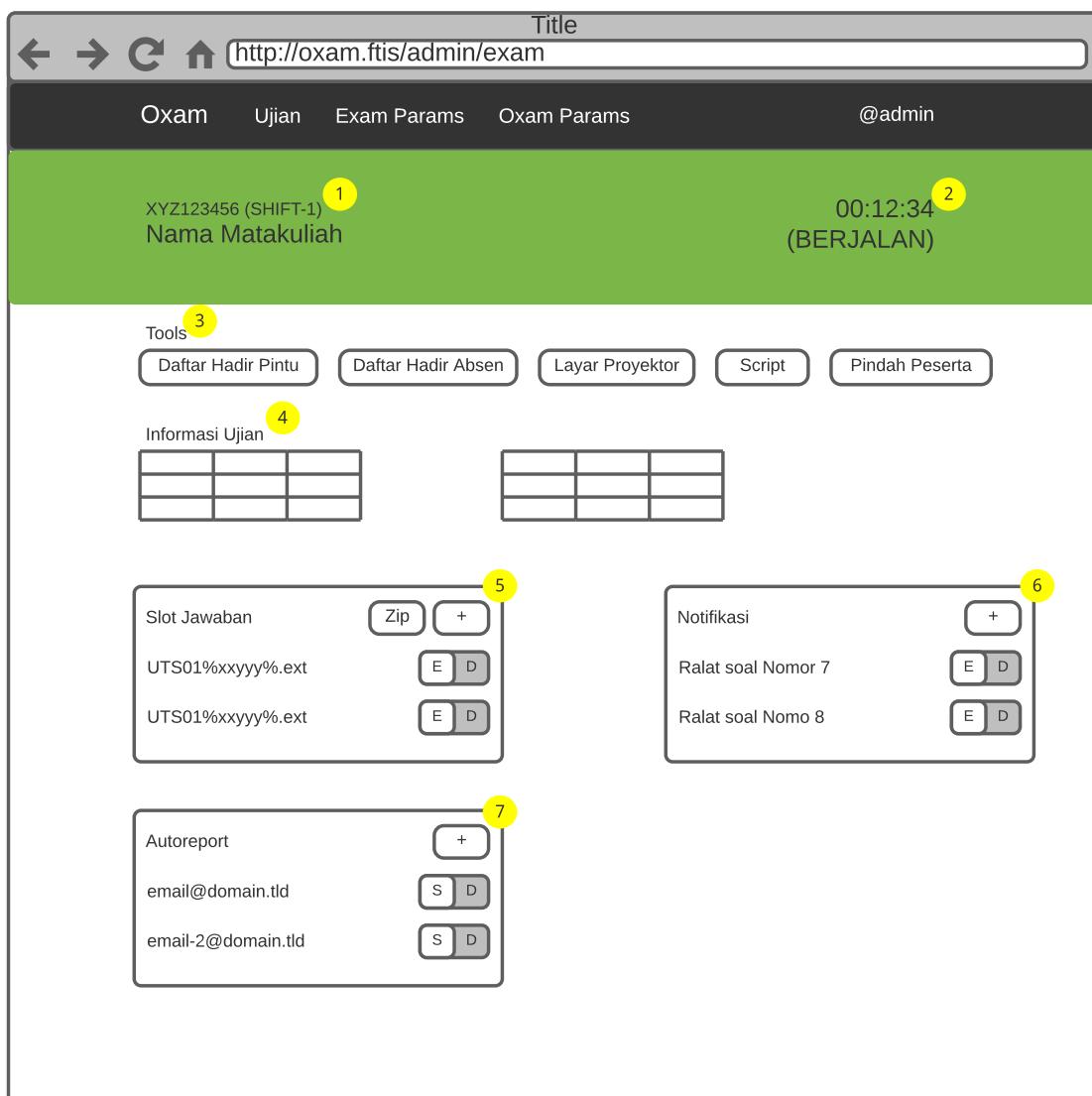
- 1 • Poin 2 adalah tombol aksi untuk melakukan pembuatan ujian. Tombol ini akan melakukan
2 beberapa pemanggilan ke sistem backend sekaligus sebelum akhirnya memindahkan tim admin
3 ke tahap berikutnya.

4 Antarmuka terakhir untuk tahap pembuatan ujian adalah tahap penyelesaian. Tampilan
5 antarmuka ini dapat dilihat pada gambar 4.13. Tampilan ini berfungsi untuk menampilkan respon
6 dari sistem bahwa ujian telah berhasil dibuat dengan informasi yang telah diberikan. Antarmuka
7 pada tahap ini hanya akan berisi pesan respon singkat disertai dengan tombol "Lihat Ujian" tombol
8 ini akan mengarahkan tim admin ke halaman ujian.

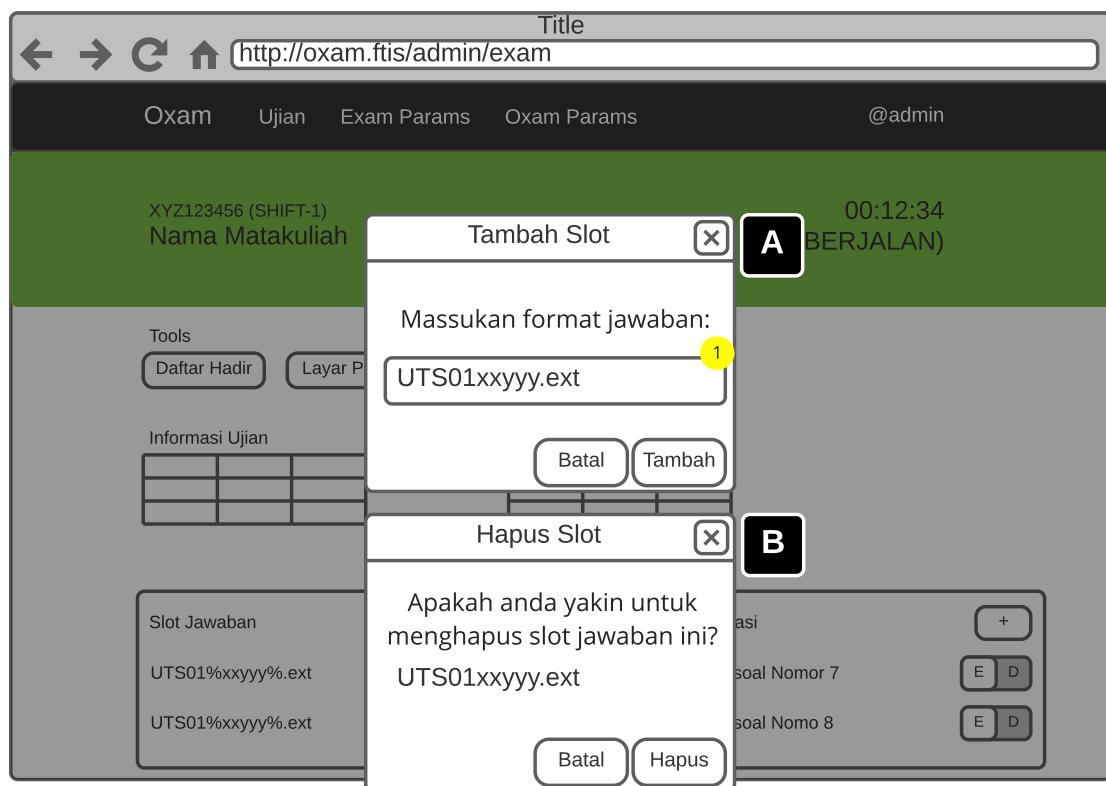
9 **Detil Ujian**

10 Halaman detil ujian akan memuat seluruh informasi ujian yang ditampung pada entitas **Exam** dan
11 beberapa entitas lainnya yang terhubung dengan entitas ini. Rancangan tampilan detil ujian dapat
12 dilihat pada gambar 4.14. Secara garis besar, tampilan memiliki beberapa bagian yang spesifik
13 dengan perannya. Hal ini dibuat demikian untuk mempermudah tim admin memindai informasi
14 pada saat membuka beberapa *tab* pada peramban. Bagian-bagian yang ditunjukkan dengan poin
15 berwarna kuning akan dijelaskan sebagai berikut:

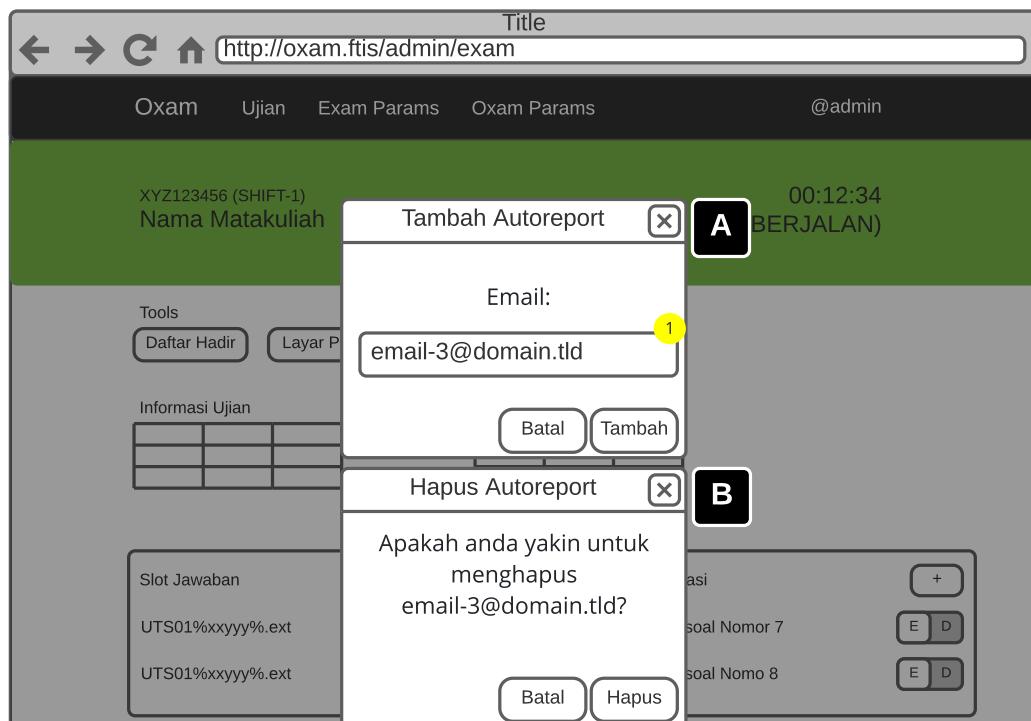
- 16 • Poin 1 adalah bagian informasi singkat tentang ujian. Informasi akan berisi minimal kode
17 mata kuliah, *shift* (jika ada), dan nama matakuliah.



Gambar 4.14: Rancangan antarmuka untuk tampilan detil ujian.



Gambar 4.15: Rancangan antarmuka untuk modal slot jawaban baru.



Gambar 4.16: Rancangan antarmuka untuk modal pelaporan otomatis.

(A) Buat baru. (B) Hapus.

- 1 • Poin 2 akan berisi status dari ujian tersebut, serta sisa waktu yang ada jika ujian tersebut
2 sedang berjalan.
- 3 • Poin 3 adalah kotak alat untuk ujian ini. Kotak alat tersebut terdiri dari beberapa tombol
4 yaitu
 - 5 – Tombol cetak daftar hadir, untuk pintu, maupun untuk absensi.
 - 6 – Tombol layar proyektor.
 - 7 – Tombol unduh *script* untuk membuat folder dan berkas ujian.
 - 8 – Tombol pemindahan peserta.
- 9 Kotak alat diimplementasikan untuk memenuhi kebutuhan yang muncul dari admin sebelum-
10 nya.
- 11 • Poin 4 menunjukkan informasi ujian dengan lebih detil, disajikan dalam bentuk tabel.
- 12 • Poin 5 adalah bagian pengelolaan Slot Jawaban. Bagian ini terdiri dari beberapa tombol.
 - 13 – Pada bagian atas, terdapat tombol Zip yang digunakan untuk melakukan pengumpulan
14 berkas jawaban menjadi sebuah *archive zip*. Peramban akan kemudian mengunduh
15 berkas tersebut dan tim admin dapat mengirimkan berkas jawaban tersebut secara
16 manual ke dosen.
 - 17 – Pada sebelah kanan tombol Zip, terdapat tombol tambah yang akan membuka sebuah
18 modal yang akan menanyakan format slot jawaban yang ada. Modal tersebut dapat
19 dilihat pada Gambar 4.15 Bagian A. Pada poin 1 dari Gambar 4.15, format yang diterima
20 dapat berupa format *xxyy* seperti pada sistem sebelumnya.
 - 21 – Untuk setiap slot jawaban yang ada, akan terdapat tombol ubah untuk mengubah lembar
22 jawab, dan tombol hapus yang dapat digunakan untuk menghapus entri tersebut. Tombol
23 hapus tersebut akan membuka modal konfirmasi seperti pada Gambar 4.15. Modal akan
24 menampilkan informasi slot ujian dan tombol aksi.
- 25 • Poin 6 menunjukkan bagian fitur notifikasi yang akan ditampilkan pada peserta ujian. Pada
26 tampilan ini, terdapat beberapa tombol yang dapat digunakan untuk memanipulasi ujian
27 tersebut. Tombol-tombol tersebut terdiri dari
 - 28 – Tombol Tambah untuk membuat notifikasi baru. Pada saat diklik, sistem akan me-
29 nampilkan sebuah modal yang menanyakan informasi jenis notifikasi. Karena alur yang
30 cukup panjang, alur kerja untuk pembuatan notifikasi akan dibahas setelah bagian ini.
 - 31 – Untuk setiap entri notifikasi akan memiliki dua buah tombol yang dapat digunakan
32 untuk mengubah dan menghapus entri tersebut.
- 33 • Poin 7 menunjukkan bagian fitur pelaporan otomatis. Fitur pelaporan otomatis ini memiliki
34 beberapa tombol aksi seperti:
 - 35 – Tombol Tambah yang dapat digunakan untuk menambahkan autoreport baru. Jika
36 tombol ini diklik, sistem akan memunculkan modal yang menanyakan pada siapa report

1 ini akan dikirimkan. rancangan modal tersebut dapat dilihat pada Gambar 4.16. Bagian
2 yang ditunjukan pada poin 1 dari rancangan tersebut dapat diisi dengan daftar email
3 yang dipisah dengan tanda koma.

- 4 – Untuk setiap entri pada bagian ini memiliki dua tombol yang dapat digunakan untuk
5 mengirimkan email, dan penghapusan entri. Tombol penghapusan entri akan menampilkan
6 modal konfirmasi penghapusan, seperti yang dapat dilihat pada Gambar 4.16. Modal
7 tersebut akan menampilkan informasi singkat tentang entri yang ada dan tombol aksi.

8 **Detil Ujian: Fitur notifikasi**

9 Pada bagian notifikasi, rancangan tampilan antarmuka memiliki beberapa bagian tergantung dari
10 jenis notifikasi yang ingin disebarluaskan ke setiap peserta. Seperti yang dapat dilihat pada 4.17,
11 rancangan tampilan antarmuka terdapat tiga bagian.

12 Bagian yang ditunjukan dengan label A adalah modal pertama yang muncul pada saat tim
13 admin menekan tombol tambah. Modal akan memiliki dua buah tombol yang merepresentasikan
14 jenis yang didukung oleh fitur ini: Kata sandi (poin 1) dan lainnya (poin 2). Jika tombol untuk
15 jenis kata sandi dipilih, maka tim admin akan disajikan modal dengan label B. Sebaliknya, jika
16 tombol untuk jenis lainnya dipilih, maka sistem akan menyajikan modal dengan label C.

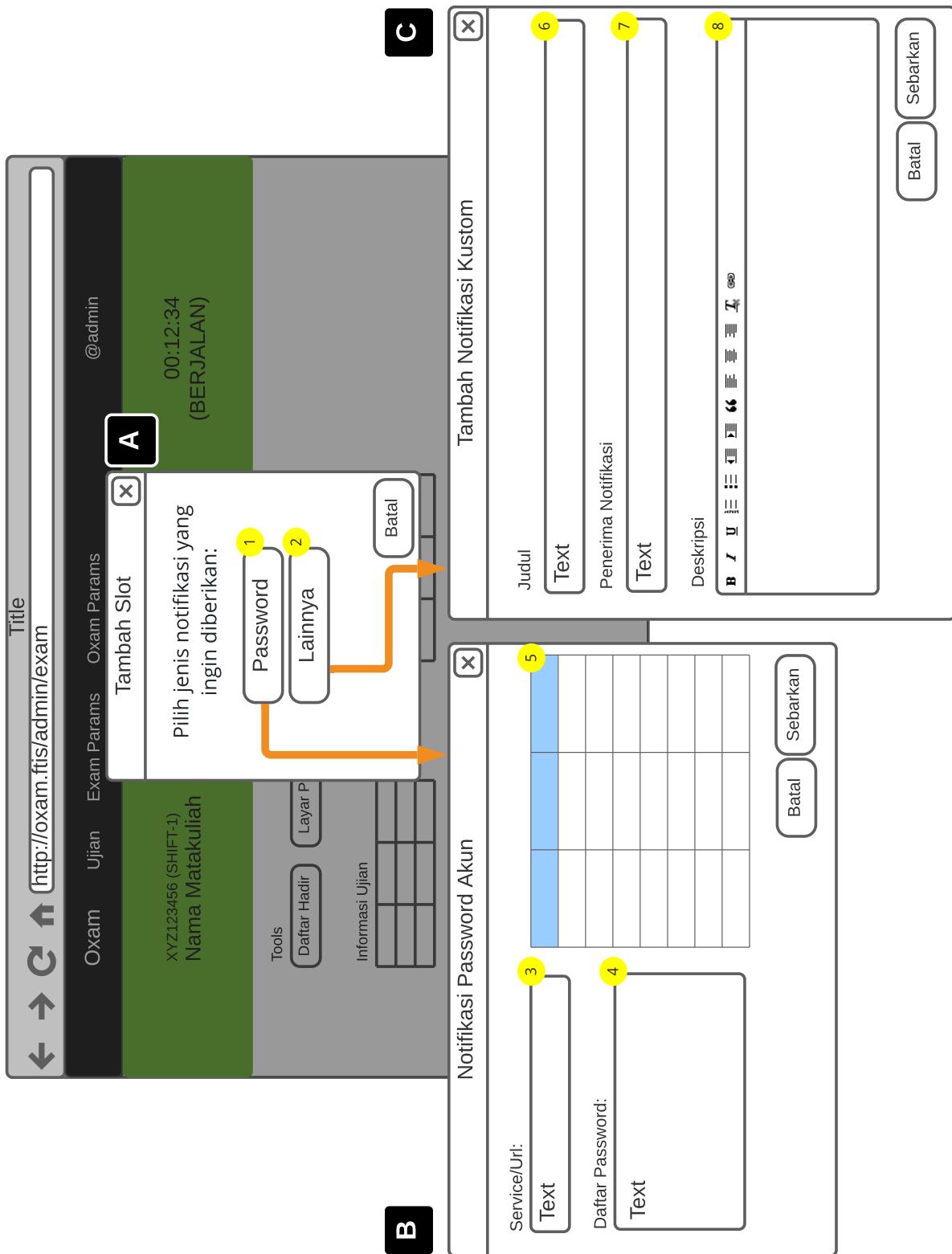
17 Modal yang ditunjukkan dengan label B adalah modal untuk mengisi informasi layanan dan
18 daftar kata sandi yang akan disebarluaskan. Bagian-bagian yang terdapat pada modal ini adalah
19 sebagai berikut

- 20 • Poin 3 menunjukan informasi tentang layanan dari kata sandi yang akan disebar.
21 • Poin 4 adalah daftar kata sandi dalam bentuk teks yang berisi informasi namapengguna dan
22 kata sandi, serta npm yang dipisah dengan karakter enter untuk setiap entrinya.
23 • Poin 5 adalah tabel daftar peserta yang akan menerima kredensial nama pengguna dan kata
24 sandi dari tim admin. Tabel ini ditambahkan untuk tim admin melakukan pengecekan sebelum
25 notifikasi disebar.

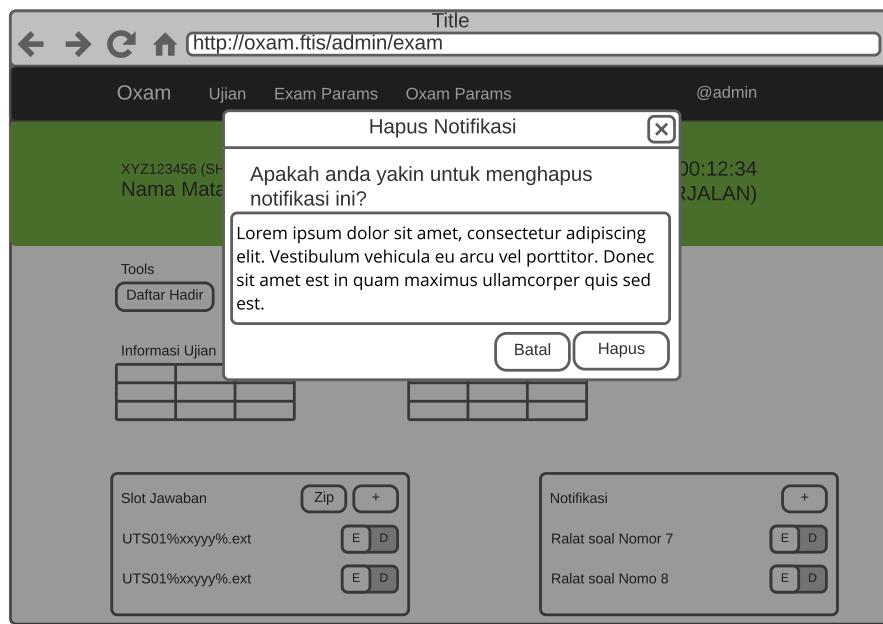
26 Kemudian, modal yang ditunjukkan dengan label C adalah modal untuk jenis notifikasi lainnya.
27 Notifikasi ini nantinya akan dapat ditargetkan untuk peserta-peserta tertentu. Bagian-bagian yang
28 terdapat pada modal ini dijelaskan sebagai berikut

- 29 • Poin 6 menunjukan bidang judul yang akan disampaikan ke peserta ujian.
30 • Poin 7 menunjukan bidang yang digunakan untuk mespesifikasikan penerima notifikasi ujian.
31 • Poin 8 adalah isi dari notifikasi tersebut.

32 Untuk menghapus notifikasi ujian, tim admin dapat menekan tombol yang nantinya akan
33 menampilkan modal untuk konfirmasi penghapusan. Rancangan tampilan modal tersebut dapat
34 diperhatikan pada Gambar 4.18. Modal akan menampilkan informasi badan notifikasi dan tombol
35 aksi.



Gambar 4.17: Rancangan antarmuka untuk beberapa tampilan modal jenis notifikasi.
 (A) Modal jenis notifikasi; (B) Notifikasi Kata Sandi; (C) Notifikasi Lainnya.



Gambar 4.18: Rancangan antarmuka untuk modal konfirmasi penghapusan notifikasi.

¹ Detil Ujian: Daftar Hadir

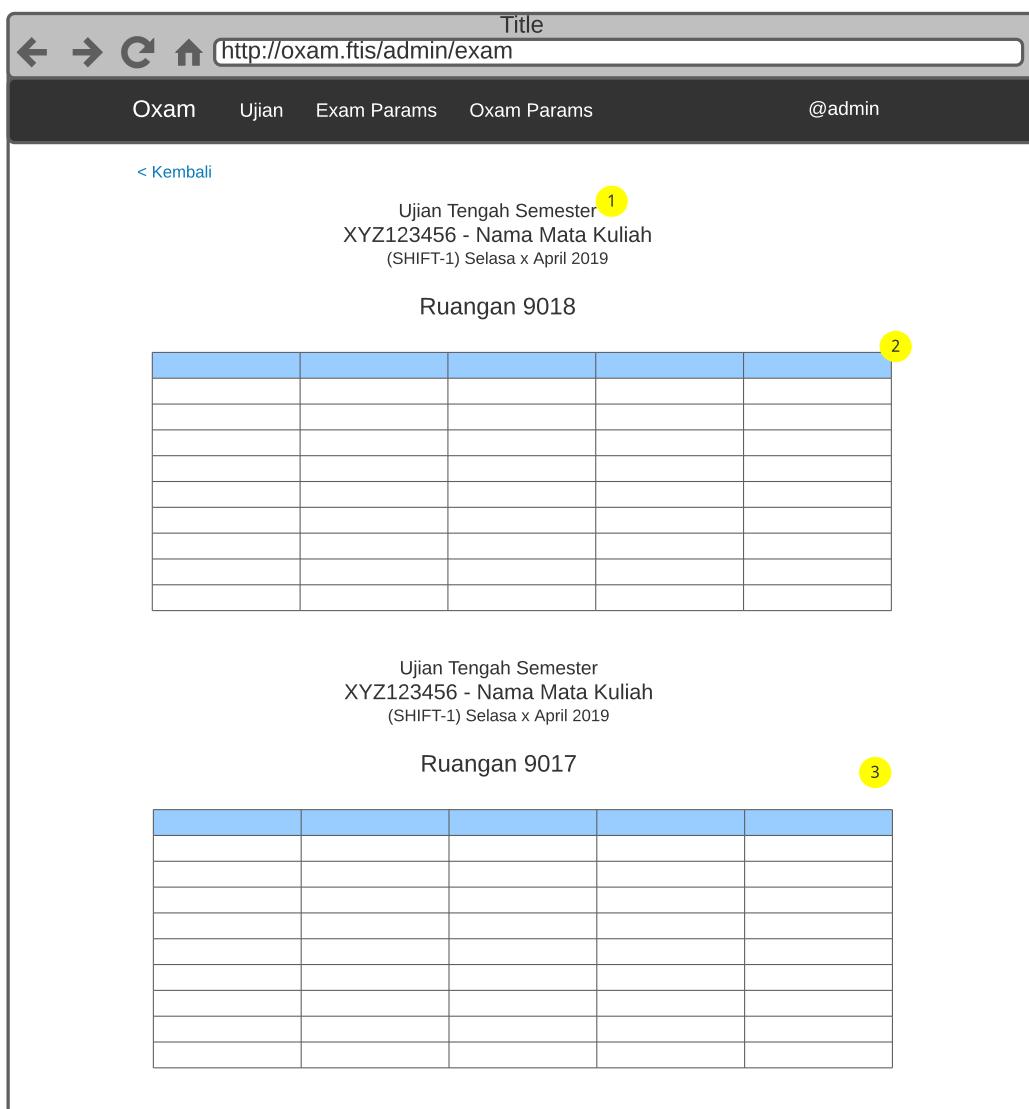
² Salah satu bagian dari kotak alat yang disediakan untuk tim admin adalah daftar hadir. Rancangan ³ tampilan daftar hadir dapat dilihat pada Gambar 4.19. Daftar hadir tersebut harus dapat dicetak ⁴ dengan pencetak. Daftar hadir ini akan terdiri dari sebuah halaman yang berisi beberapa bagian ⁵ penting, seperti

- ⁶ • Poin 1, informasi singkat ujian seperti informasi UTS/UAS, kode mata kuliah, nama mata ⁷ kuliah, shift (jika ada), tanggal mulai, dan ruangan.
- ⁸ • Poin 2, daftar peserta dengan pengurutan kolom tertentu. Jika daftar hadir ditempelkan pada ⁹ pintu, maka daftar hadir harus diurutkan berdasarkan NPMnya. Jika daftar hadir digunakan ¹⁰ untuk absensi, maka daftar tersebut harus diurutkan berdasarkan nomor komputernya.
- ¹¹ • Poin 3, daftar hadir pada ruangan lain yang setiap daftar hadirnya terdiri dari poin 1 dan 2. ¹² Daftar hadir ini harus dapat terpisah menjadi halaman baru pada saat dicetak.

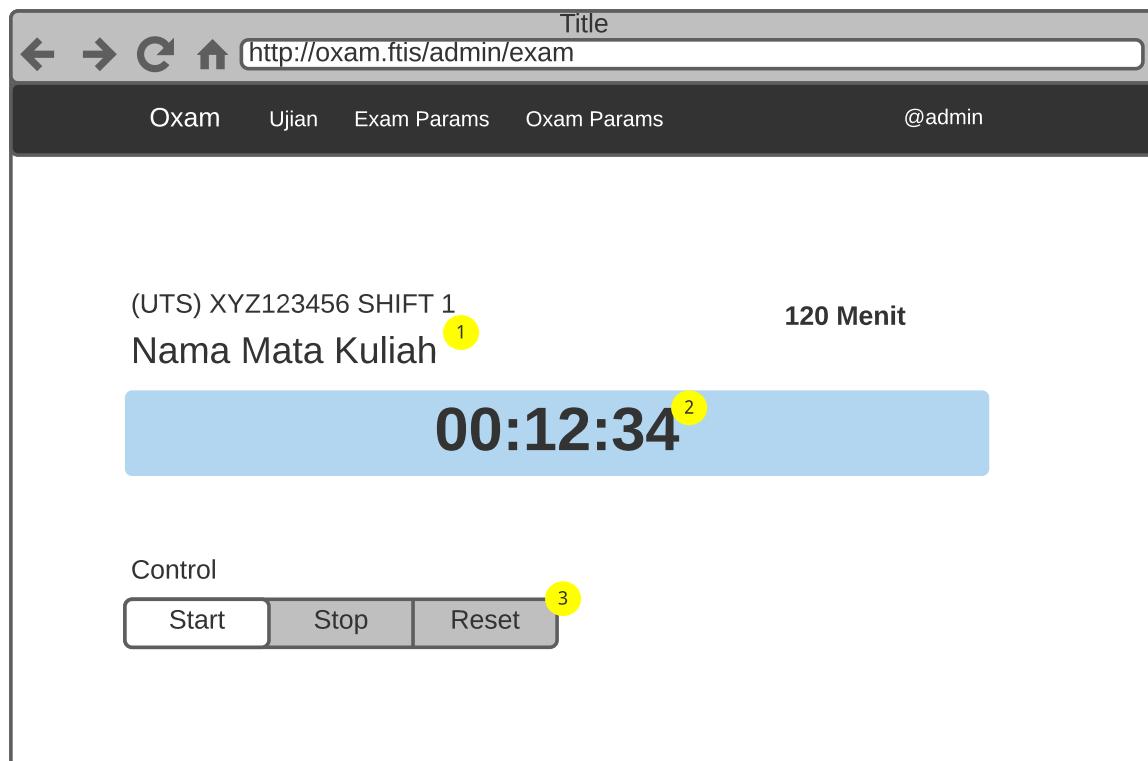
¹³ Tampilan daftar hadir ini harus dapat dicetak oleh pencetak atau *printer*. Sehingga setiap ¹⁴ halaman harus memiliki logo UNPAR dan informasi fakultas.

¹⁵ Detil Ujian: Layar Proyektor

¹⁶ Fitur berikutnya adalah tampilan timer untuk proyektor. Rancangan tampilan dapat dilihat pada ¹⁷ Gambar 4.20. Karena akun admin tidak memiliki relasi dengan lokasi ruangan, maka tampilan ¹⁸ ini ditambahkan sebagai cadangan dan digunakan hanya pada saat darurat. Tampilan ini akan ¹⁹ memiliki informasi singkat tentang ujian yang akan diadakan dan durasinya (poin 1), sisa waktu ²⁰ (poin 2) dan tombol kontrol timer (poin 3). Tombol kontrol timer ini akan terhubung dengan sistem ²¹ untuk membuka dan menutup lembar jawab.



Gambar 4.19: Rancangan tampilan untuk daftar hadir peserta ujian.

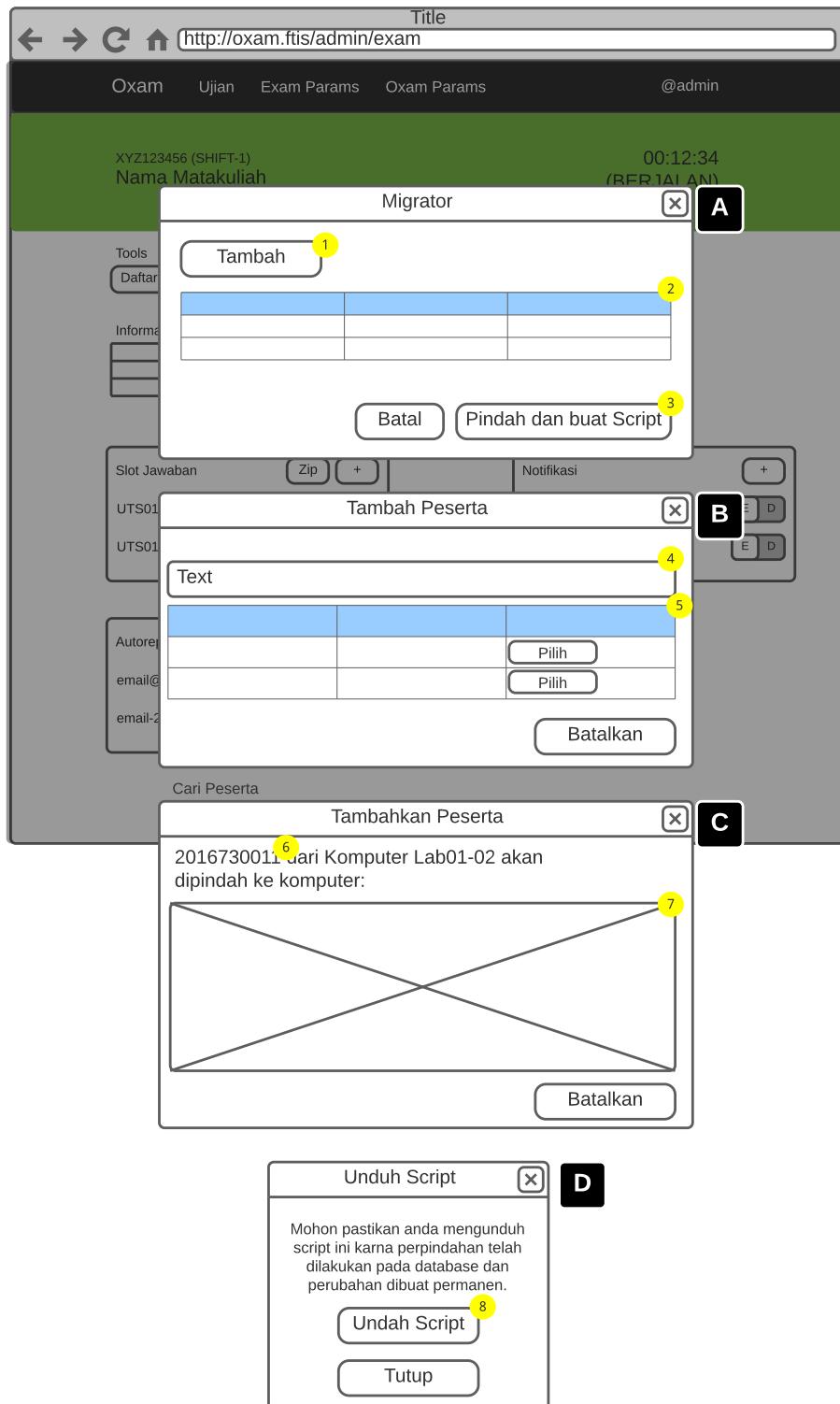


Gambar 4.20: Rancangan antarmuka untuk tampilan pada layar proyektor.

¹ Detil Ujian: Pindah Peserta

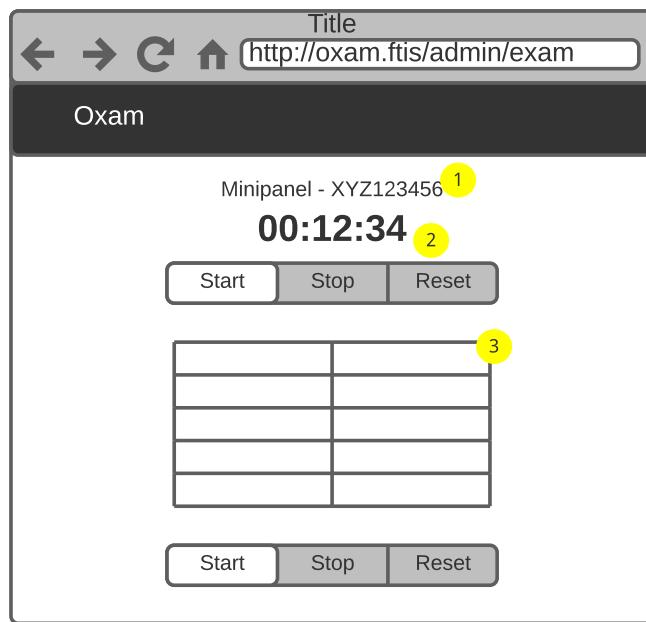
² Fitur pada kotak alat berikutnya adalah fitur untuk memindahkan peserta. Rancangan tampilan pada fitur ini dapat dilihat pada Gambar 4.21. Fitur pemindahan ini akan melalui tahapan seperti memilih peserta yang akan dipindahkan (Bagian B), memilih komputer tujuan (Bagian C), lalu eksekusi script yang diberikan oleh sistem (Bagian D). Poin-poin yang ditujukan pada sistem akan dijelaskan sebagai berikut:

- ⁷ • Modal Migrator (Bagian A)
 - ⁸ – Poin 1 adalah tombol tambah yang akan membuka Modal yang terdapat pada bagian B.
 - ⁹ – Poin 2 menunjukkan daftar peserta yang akan dipindahkan beserta komputer tujuannya.
 - ¹⁰ – Poin 3 adalah tombol eksekusi pemindahan pada level basis data dan bangkitkan sebuah *script* untuk melakukan pemindahan lembar kerja peserta.
- ¹² • Modal Tambah Peserta (Bagian B)
 - ¹³ – Poin 4 adalah bidang pencarian cepat untuk membantu tim admin mencari peserta yang ingin dipindah.
 - ¹⁵ – Poin 5 menunjukkan daftar peserta yang terdapat pada ujian ini, dengan sebuah tombol pilih pada tiap entri untuk memilih peserta bersangkutan untuk melakukan pemindahan.
- ¹⁷ • Modal Pilih Komputer (Bagian C)



Gambar 4.21: Rancangan tampilan untuk pemindahan peserta.

(A) Daftar pemindahan; (B) Pencari peserta target; (C) Pencari komputer target; (D) Konfirmasi dan pengunduhan *script* pemindahan.



Gambar 4.22: Rancangan antarmuka untuk Minipanel.

- Poin 6 menunjukkan informasi peserta mana yang akan dipindah. Bagian ini ditambahkan untuk menyakinkan admin bahwa admin telah memilih peserta yang benar.
- Poin 7 adalah peta tempat duduk ujian pada ruangan tertentu. Komputer pada peta tersebut dapat diklik untuk dipilih. Saat tempat duduk selesai dipilih, maka modal akan kembali ke Bagian A, hingga seluruh peserta yang ingin dipindah telah didaftarkan seluruhnya.

Detil Ujian: Minipanel

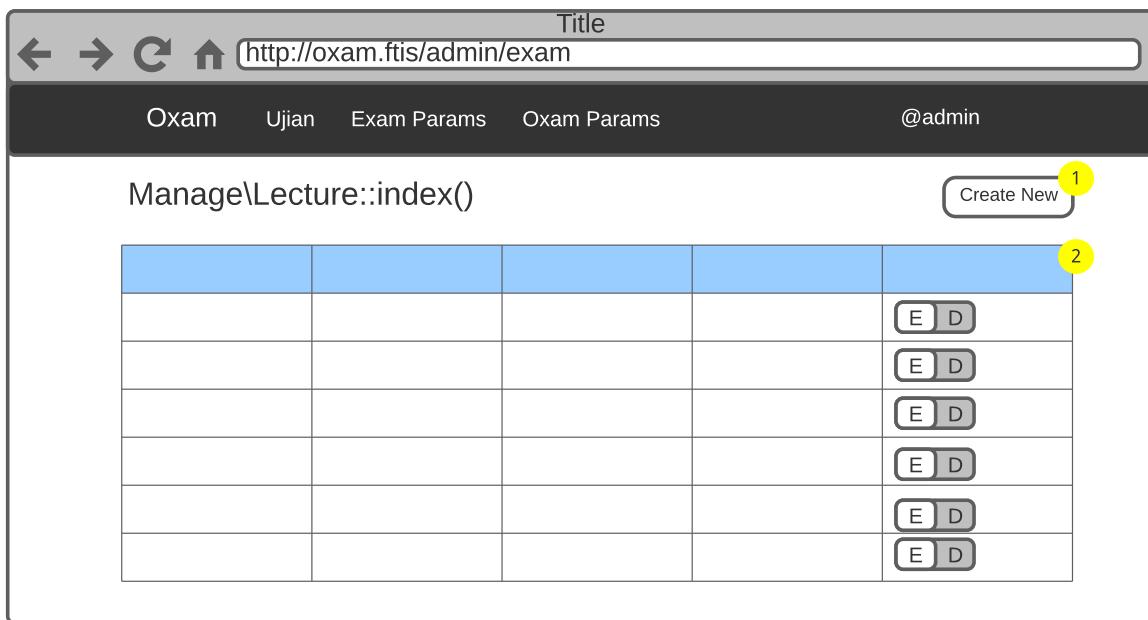
Tampilan minipanel adalah tampilan untuk perangkat bergerak yang dapat digunakan oleh tim admin untuk memicu mulainya timer ujian. Pemicu tersebut akan terhubung langsung dengan sistem back-end dan membuka lembar jawab untuk peserta. Rancangan tampilan dapat dilihat pada Gambar 4.22. Secara garis besar, rancangan tampilan memiliki informasi singkat ujian pada atas halaman, satu set timer dengan kontrolnya, serta tabel detil ujian pada bagian bawah halaman.

Bagian-bagian yang ditunjukkan oleh poin dideskripsikan sebagai berikut:

- Poin 1 mengandung informasi singkat tentang kode mata kuliah yang dipilih.
- Poin 2 adalah bagian satu set timer beserta dengan kontrolnya. Selain itu kontrol juga akan tersedia pada bagian bawah halaman untuk mempermudah memulai timer pada perangkat bergerak.
- Poin 3 adalah informasi detil ujian dalam bentuk tabel.

Pengelola Entitas: Daftar Entri

Selain ujian, tim admin dapat mengelola entri dari entitas. Rancangan antarmuka pertama untuk pengelola entri adalah daftar entri. Setiap entitas yang ada akan memiliki sebuah daftar yang



Gambar 4.23: Rancangan antarmuka untuk daftar entri entitas.

- 1 disajikan dengan tabel terdefinisi. Rancangan antarmuka tersebut dapat dilihat pada Gambar 4.23.
- 2 Seperti rancangan antarmuka lainnya, bagian yang ditunjukkan oleh poin-poin dideskripsikan sebagai berikut:
- 3
 - Poin 1 adalah tombol untuk membuat entri baru. Tombol ini akan mengarahkan admin ke halaman *editor* dengan bidang yang tidak terisi.
 - Poin 2 adalah daftar entri yang terdapat pada entitas tersebut. Setiap entri akan memiliki tombol ubah dan hapus. Tombol ubah kan mengarahkan admin ke halaman *editor* dengan bidang yang terisi dari entri terpilih. Tombol hapus akan memunculkan modal konfirmasi penghapusan entri.
- 4 Daftar entri disajikan dalam bentuk tabel yang telah didefinisikan konfigurasinya. Tiap entri yang dapat dikelola akan memiliki sebuah daftar definisi yang ditentukan oleh pengembang. Berkas definisi tersebut akan dapat menentukan aturan kolom-kolom tertentu yang ingin ditampilkan.
- 5

Pengelola Entitas: Buat dan Ubah Entri Baru
- 6 Entri dari entitas dapat diubah dengan bantuan *editor*. Rancangan tampilan *editor* tersebut dapat dilihat pada Gambar 4.24. Secara garis besar, tampilan ini akan memiliki tanggung jawab untuk membuat dan mengubah data entri baru maupun yang sudah ada. Bagian-bagian yang ditunjukkan dengan poin dideskripsikan sebagai berikut
- 7
 - Poin 1a judul menunjukkan informasi status bahwa *editor* akan melakukan perubahan pada entitas dengan id 14. Jika status editor adalah untuk membuat entri baru, maka judul akan tampil mirip seperti poin 1b.

A

Manage\Lecture::edit(14)

id
14

Name
Nama Mata Kuliah

Lecture Code
XYZ123456

Save Delete

B

Manage\Lecture::new()

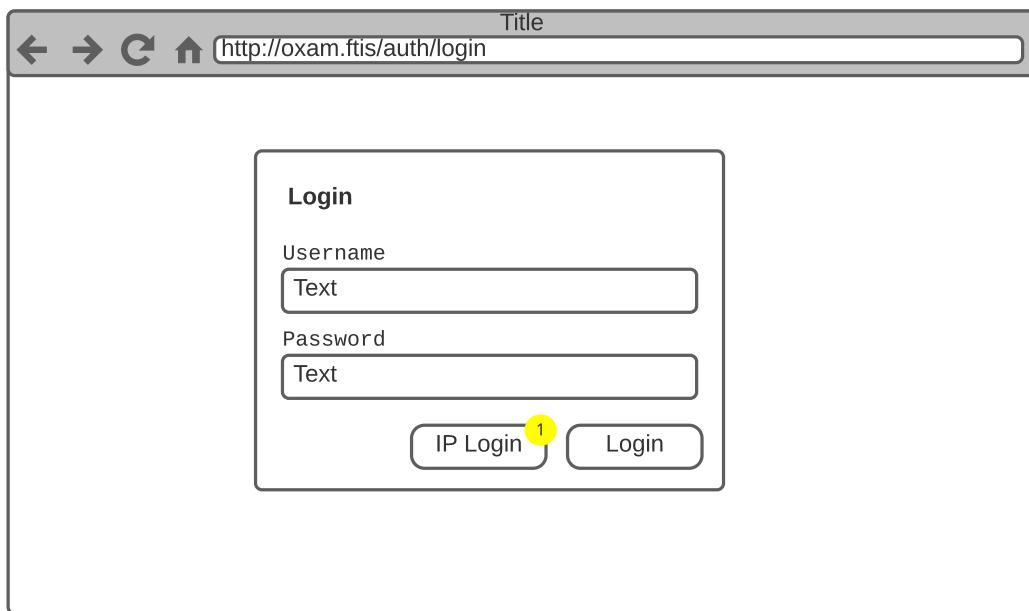
Gambar 4.24: Rancangan antarmuka untuk *editor* entri.

(A) Buat baru; (B) Ubah yang sudah ada.

- 1 • Poin 2 adalah daftar atribut yang dapat diubah atau isi nilainya, berdasarkan berkas definisi tabel.
- 2
- 3 • Poin 3 adalah tombol aksi untuk melakukan penyimpanan atau penghapusan pada entri yang saat ini dibuka. Tombol simpan akan mengarahkan kembali admin ke halaman daftar entri setelah perubahan disimpan atau entri dibuat. Tombol Hapus akan memunculkan modal konfirmasi hapus entri.
- 4
- 5
- 6



Gambar 4.25: Rancangan antarmuka untuk menghapus entri.



Gambar 4.26: Rancangan antarmuka untuk otentikasi, dengan menekankan bagian tertentu.

¹ **Pengelola Entitas: Hapus Entri**

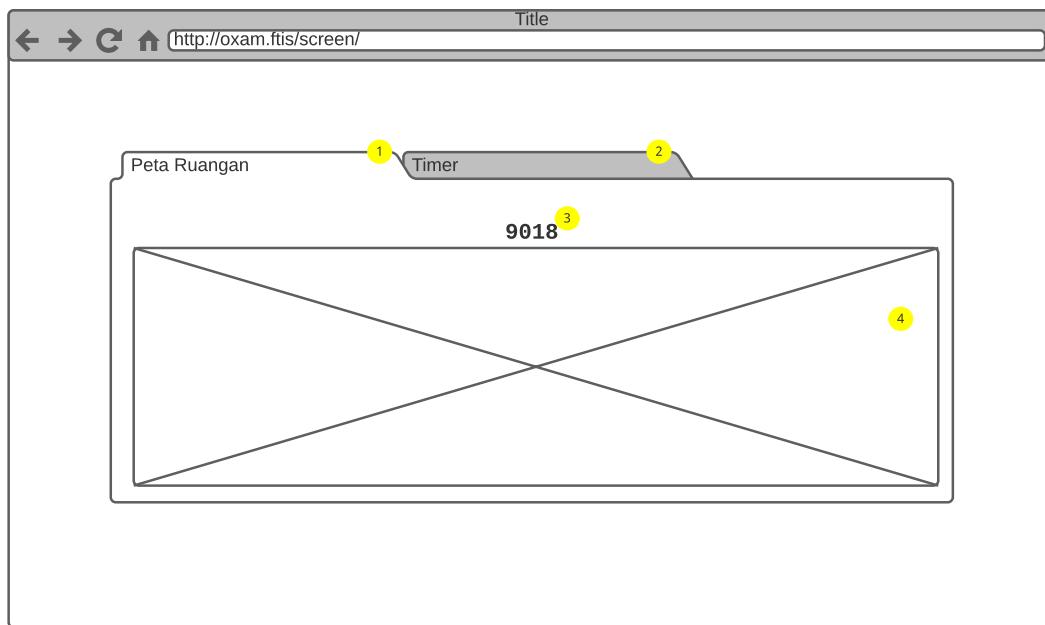
- ² Pada saat tombol hapus pada halaman daftar entri dan *editor* entri diklik, sebuah modal konfirmasi penghapusan akan dimunculkan oleh sistem. Rancangan modal tersebut dapat dilihat pada 4.25.
- ³ Secara garis besar, informasi yang dimunculkan pada modal (poin 1) adalah kolom yang ditampilkan pada halaman *editor*. Informasi tersebut ditampilkan untuk menyakinkan admin bahwa admin akan menghapus entri yang tepat.
- ⁷ Jika modal muncul pada halaman *editor*, maka setelah aksi penghapusan berhasil dilakukan, admin akan diarahkan menuju halaman daftar entri. Namun jika modal muncul pada halaman entri, admin tidak akan diarahkan kemanapun.

¹⁰ **4.1.3 Rancangan Antarmuka untuk Dosen Pengawas**

- ¹¹ Antarmuka untuk dosen pengawas akan bertanggung jawab untuk membuka dan menutup lembar jawab, serta menampilkan informasi ujian di proyektor tiap ruangan. Tampilan yang ada akan memiliki daftar ujian yang berbeda tiap ruangan, tergantung dari ujian yang diaktifkan pada ruangan tersebut. Untuk dapat mengakses antarmuka ini, Dosen Pengawas diharuskan untuk melakukan otentikasi terlebih dahulu.

¹⁶ **Otentikasi**

- ¹⁷ Tahap pertama untuk dapat mengakses layar timer, pengguna harus melakukan login terlebih dahulu. Serupa dengan halaman Login yang digunakan untuk otentikasi Admin pada Gambar 4.7, otentikasi untuk pengguna jenis ini dipertegas dengan penggunaan tombol IP Login yang ditunjukan pada poin 1 di Gambar 4.26.
- ²¹ Tombol IP Login pada Gambar 4.26 akan langsung mengarahkan pengguna ke halaman peta



Gambar 4.27: Rancangan antarmuka untuk halaman Peta Ruangan Ujian.

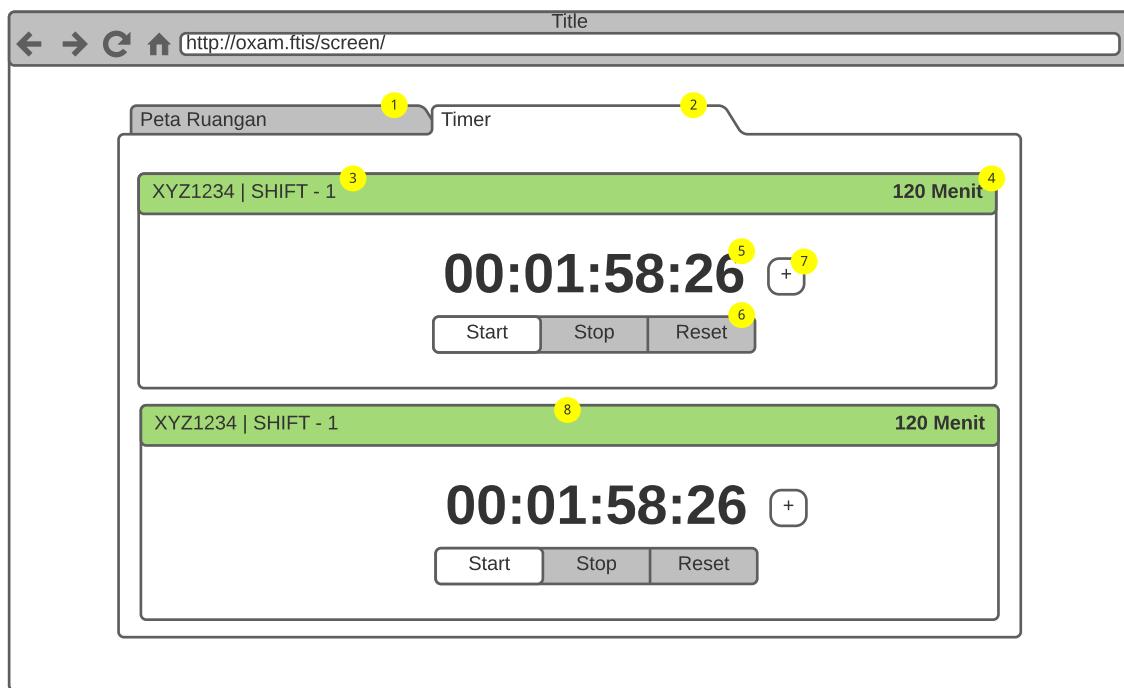
- ¹ ruangan ujian dan timer ujian. IP dari komputer yang digunakan untuk mengakses halaman ini harus diregistrasikan ke sistem untuk proses otentikasi dapat berhasil. Jika otentikasi gagal, halaman ini harus menampilkan pesan kesalahan yang mendeskripsikan bahwa IP komputer tidak teregistrasi.

⁵ Peta Ruangan

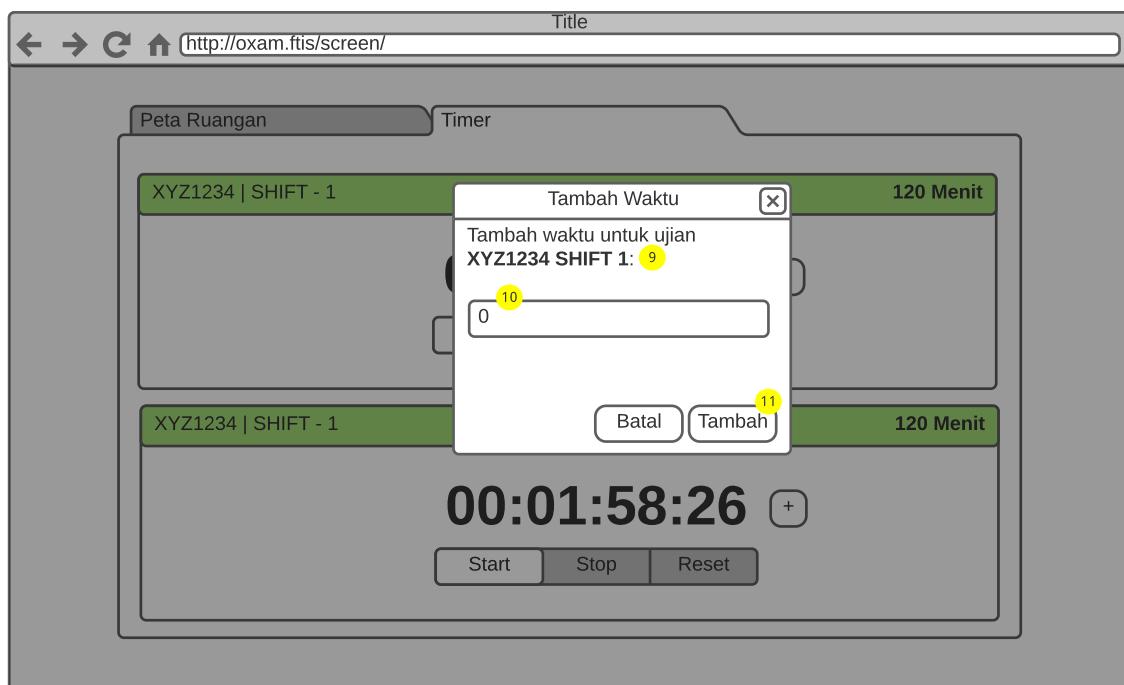
- ⁶ Setelah otentikasi berhasil, dosen pengawas akan diarahkan menuju halaman peta tempat duduk ruangan ujian. Rancangan halaman informasi peta ruangan dapat dilihat pada Gambar 4.27. Pada halaman ini, informasi tentang nomor ruangan dan peta tempat duduk ruangan tersebut akan ditampilkan secara diagram.

¹⁰ Bagian-bagian yang ditunjukkan oleh poin pada Gambar 4.27 dijelaskan sebagai berikut

- ¹¹ • Bagian yang ditunjukan oleh poin 1 dan poin 2 pada Gambar 4.27 adalah navigasi untuk menuju halaman timer dan peta ruangan. Navigasi ditambahkan untuk memudahkan dosen untuk melihat ujian yang akan berjalan dan peta ruangan yang ada saat ini. Saat pengguna pindah ke tab lain, halaman tidak akan disegarkan atau mengalami pemutaran ulang.
- ¹⁵ • Poin 3 dan 4 masing-masing menunjukkan informasi nama ruangan, dan peta tempat duduk dari ruangan tersebut. Poin 3 akan menunjukkan nama ruangan yang ada dan poin 4 menunjukkan peta ruangan ujian. Peta ini dirancang untuk membantu peserta untuk mencari tempat duduk mereka. Sistem akan menampilkan tempat duduk tempat komputer yang terdaftar pada sistem. Informasi tentang posisi komputer akan disimpan pada database.



Gambar 4.28: Rancangan antarmuka untuk halaman Timer.



Gambar 4.29: Rancangan antarmuka untuk *Overtime Ujian*.

1 Timer Ujian

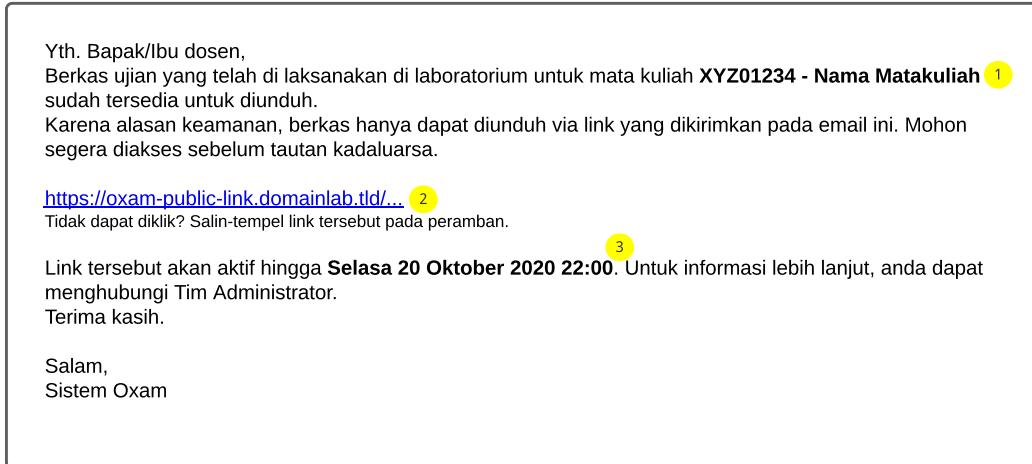
2 Rancangan halaman timer ujian dapat diperhatikan pada Gambar 4.28. Secara keseluruhan,
3 halaman timer akan menampilkan informasi ujian singkat, waktu tersisa dan tombol-tombol alat
4 untuk memanipulasi waktu ujian.

5 Mengikuti rancangan antarmuka Peta Ruangan, halaman timer memiliki bagian navigasi seperti
6 yang ditunjukkan pada poin 1 dan 2 pada Gambar 4.28. Karena saat ini pengguna sedang membuka
7 tab Timer, maka kepala tab yang aktif menunjukkan kepala tab timer. Dosen pengawas dapat melihat
8 kedua informasi tersebut dengan mudah. Bagian pada rancangan tampilan tersebut dideskripsikan
9 sebagai berikut:

- 10 • Bagian yang ditunjukkan pada poin 3 adalah informasi ujian secara singkat. Informasi yang
11 ditunjukkan berupa kode mata kuliah dan *shiftnya*. Informasi ini nantinya dapat digunakan
12 untuk membedakan ujian satu dengan yang lainnya pada tampilan multiujian.
- 13 • Bagian yang ditunjukkan pada poin 4 adalah jumlah durasi waktu yang diberikan untuk
14 ujian ini. Durasi lamanya ujian ini akan diambil dari basis data dan tidak akan berkurang
15 mengikuti tampilan timer.
- 16 • Bagian yang ditunjukkan oleh poin 5 adalah tampilan timer. Timer ini akan memiliki nilai
17 dasar total durasi dari ujian. Timer akan mulai berjalan saat kontrol yang ditunjukkan pada
18 poin 6 mulai digunakan. Pada saat timer mencapai nilai nol, maka tampilan tersebut akan
19 dipertahankan selama beberapa menit untuk memastikan dosen pengawas dapat menambahkan
20 waktu lebih sebelum lembar jawab tidak dapat diubah kembali.
- 21 • Poin 6 menunjukkan bagian kontrol dari timer ujian. Kontrol akan terdiri dari tombol aksi
22 mulai berhenti, dan setel ulang. Tombol-tombol ini akan langsung melakukan panggilan API
23 ke sistem back-end.
- 24 • Poin 7 menunjukkan tombol tambahan *overtime*. Tombol ini digunakan untuk menambahkan
25 waktu ujian jika sewaktu-waktu dibutuhkan. Tombol akan menampilkan sebuah *modal*
26 yang menyediakan sebuah formulir sederhana, menanyakan berapa banyak waktu yang ingin
27 ditambahkan. Modal tersebut dapat dilihat pada Gambar 4.29. Modal akan menampilkan
28 informasi singkat ujian (poin 9), banyak waktu yang diinginkan (poin 10), dan tombol *submit*
29 dan batalkan (poin 11).
- 30 • Poin 8 adalah contoh timer lain yang ditunjukkan jika terdapat ujian lain yang akan diadakan
31 pada ruangan yang sama. Tampilan akan memiliki bentuk tataletak yang sama, sehingga
32 yang menjadi pembeda hanyalah bagian kepala (ditunjukkan pada poin 3).

33 4.1.4 Rancangan Antarmuka Tambahan

34 Rancangan antarmuka tambahan ini muncul dari kebutuhan untuk sistem mengirimkan laporan
35 ujian dengan tambahan halaman pengunduhan. Oleh karena itu, pada bagian ini akan dibahas
36 rancangan antarmuka untuk email dan halaman pengunduhannya.



Gambar 4.30: Rancangan antarmuka untuk email laporan ujian.

¹ **Email Laporan**

² Email yang akan dikirimkan sebagai laporan akan dibuat dengan teknologi HTML, seperti pada ³ umumnya. Email akan berisi sebuah tautan menuju halaman untuk mengunduh berkas jawaban ⁴ ujian. Email juga akan berisi beberapa informasi krusial tentang masa hidup tautan tersebut ⁵ sehingga penerima laporan tidak merasa kebingungan jika tautan tidak dapat diakses.

⁶ Rancangan untuk tampilan email dapat dilihat pada Gambar 4.30. Seperti yang ditunjukkan ⁷ pada gambar, email akan memiliki sejumlah instruksi untuk mengunduh berkas jawaban, informasi ⁸ mata kuliah ujian (Poin 1 dari Gambar 4.30), tautan untuk mengunduh berkas jawaban tersebut ⁹ (Poin 2 dari Gambar 4.30), dan informasi kapan tautan tersebut akan kadaluarsa (Poin 3 dari ¹⁰ Gambar 4.30).

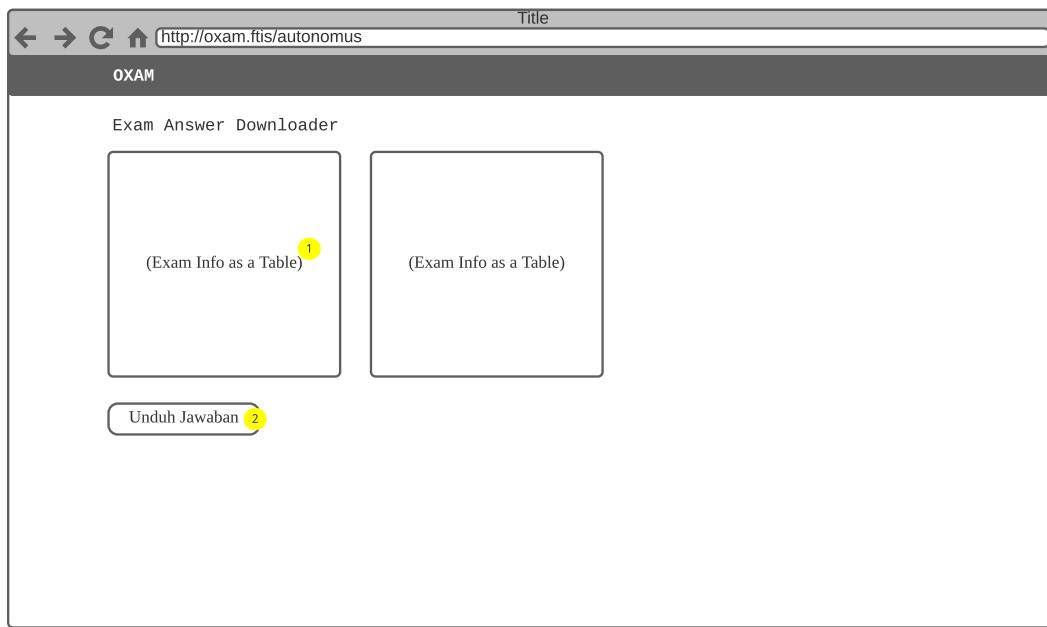
¹¹ **Halaman Pengunduhan Berkas Jawaban Ujian**

¹² Tautan yang dikirim lewat email akan mengarah pada halaman ini. Informasi singkat tentang ¹³ ujian akan ditampilkan pada halaman ini sebelum akhirnya penerima laporan dapat melakukan ¹⁴ pengunduhan hasil jawaban tersebut. Halaman ini dirancang untuk disediakan oleh sistem Oxam ¹⁵ yang dapat diakses via internet.

¹⁶ Rancangan halaman pengunduhan dapat dilihat pada Gambar 4.31. Bagian yang ditunjukkan ¹⁷ oleh Poin 1 pada Gambar 4.31 adalah informasi ujian yang disajikan dalam bentuk tabel. Tabel ini ¹⁸ nantinya akan berisi semua informasi dari entitas ujian, dengan penyesuaian tertentu. Lalu bagian ¹⁹ yang ditunjukkan pada poin 2 adalah tombol unduh jawaban ujian. Tombol ini akan memaksa ²⁰ peramban untuk melakukan pengunduhan berkas jawaban ke komputer penerima laporan.

²¹ **4.2 Perancangan Sistem Backend**

²² Sesuai dengan perannya, sistem backend akan bertanggung jawab menangani bagian-bagian yang ²³ hanya ditangani pada sisi server. Tanggung jawab tersebut mulai dari melakukan pengolahan



Gambar 4.31: Rancangan antarmuka untuk halaman pengunduhan berkas jawaban ujian.

1 data, pengelolaan basis data, serta menyediakan API yang digunakan oleh sistem frontend untuk
2 berkomunikasi.

3 Pada bagian ini akan dijelaskan perancangan sistem untuk subsistem backend. Penjelasan
4 tersebut dimulai dari pembahasan perancangan basis data, perancangan REST API serta desain
5 kelas dari subsistem ini.

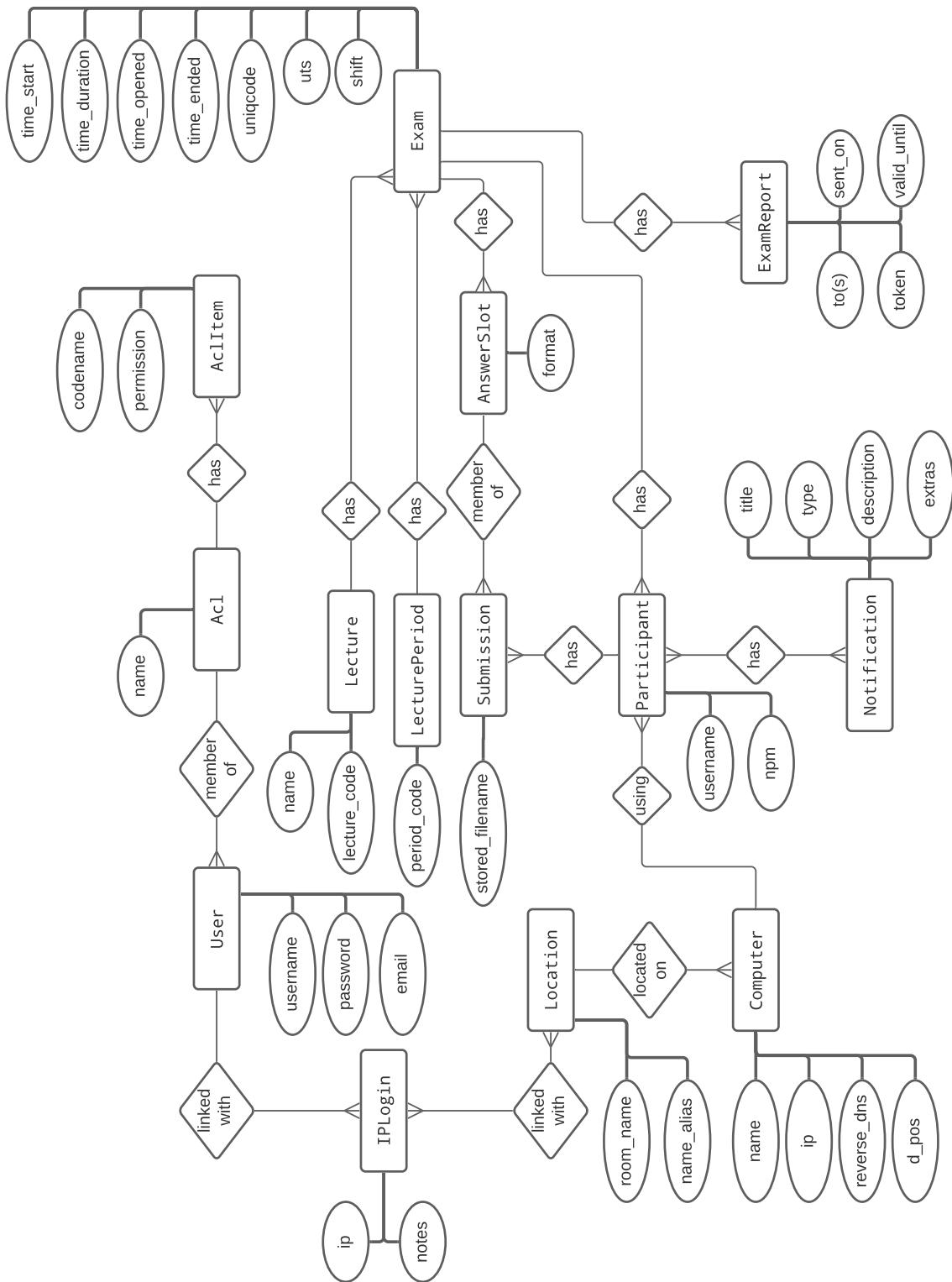
6 4.2.1 Racangan Basis Data

7 Perancangan basis data dimulai dengan merumuskan entitas yang dibutuhkan dan relasinya dengan
8 entitas lainnya. Rumusan tersebut direpresentasikan dalam bentuk diagram relasi entitas atau ERD.
9 Diagram ERD untuk aplikasi ini dapat dilihat pada Gambar 4.32. Berdasarkan kebutuhan yang
10 dianalisis pada bab sebelumnya, sistem membutuhkan empat belas entitas yang bertanggung jawab
11 untuk merepresentasikan struktur data yang digunakan. Setiap entitas yang dibuat akan dibuat
12 ulang representasinya dalam bentuk kelas pada sistem PHP dengan harapan membantu penanganan
13 dan perancangan sistem REST API.

14 Pembahasan akan dilanjutkan dengan menjelaskan satu per satu entitas yang ditunjukan pada
15 diagram ERD. Pembahasan akan dimulai dengan entitas yang penulis kategorikan sebagai esensial
16 untuk sistem terlebih dahulu, lalu dilanjutkan dengan entitas yang muncul dari analisis dari bab
17 sebelumnya.

18 User

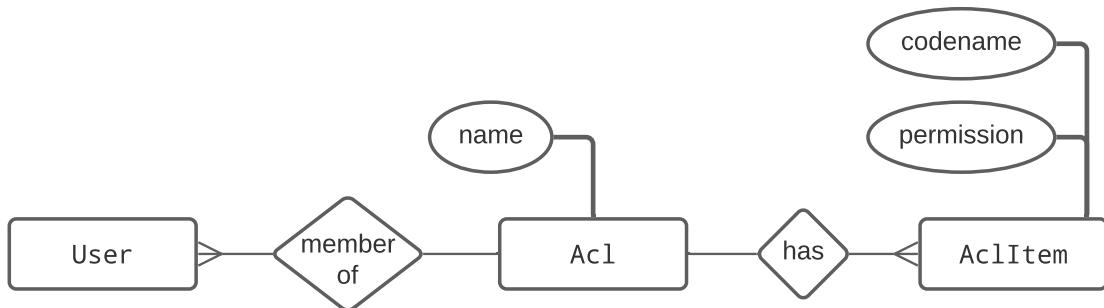
19 Entitas **User** merepresentasikan pengguna aplikasi. Entitas ini akan menyimpan informasi khusus
20 seperti nama pengguna (**username**), kata sandi (**password**) dan email. Kolom **password** pada
21 entitas ini akan dihash menggunakan algoritma Blowfish dengan bantuan *library* dari PHP dengan
22 alasan keamanan.



Gambar 4.32: Diagram *ERD* untuk sistem aplikasi yang baru.

1 Entitas ini dinilai esensial karena entitas ini digunakan untuk melakukan otentikasi menuju panel
 2 admin. Entitas ini akan memiliki perlakuan khusus pada saat data dari entitas ini ditransmisikan
 3 ke sistem frontend. Otentikasi ini penting untuk mengkhususkan peran dari setiap pengguna.

4 ACL dan ACLItem



Gambar 4.33: Potongan diagram entitas untuk ACL dan ACLItem.

5 Entitas **ACL** dan **ACLItem** adalah entitas yang menampung informasi tentang peran dan izin untuk
 6 setiap pengguna, diperjelas pada Gambar 4.33. **ACL** menyimpan grup utama dari **ACLItem**, atau
 7 dapat kita sebut sebagai peran. Sedangkan **ACLItem** menyimpan informasi izin untuk setiap nama
 8 kasus (**codename**) yang diperbolehkan.

Label	Deskripsi	Nilai Biner
C	Buat	0b0001
R	Baca	0b0010
U	Perbarui	0b0100
D	Hapus	0b1000

Tabel 4.1: Tabel representasi nilai biner pada kolom **permission** di entitas **ACLItem**.

9 Perizinan direpresentasikan dengan menggunakan sistem biner yang disimpan pada database
 10 sebagai angka. Representasi biner ini terdiri dari label CRUD, dengan C adalah Buat (*Create*), R
 11 adalah Baca (*Read*), U adalah Perbarui (*Update*) dan D adalah Hapus (*Delete*). Label tersebut
 12 kemudian diberikan tempat khusus pada bitstring sebelum akhirnya dikonversi sebagai angka.
 13 Nilai-nilai tersebut dapat dilihat pada tabel 4.1.

14 Sebagai contoh, seorang pengguna dapat **membuat**, **membaca**, **memperbarui** namun tidak
 15 dapat menghapus sebuah entri. Nilai dari tabel 4.1 digabungkan dengan operator OR untuk setiap

1 izin yang diberikan. Dengan demikian, nilai **permission** yang diberikan adalah sebagai berikut:

$$C \vee R \vee U \vee 0b0000 = K \quad (4.1a)$$

$$0b0001 \vee 0b0010 \vee 0b0100 \vee 0b0000 = K \quad (4.1b)$$

$$K = 0b0111 \quad (4.1c)$$

$$K = 7 \quad (4.1d)$$

2 Hasil kalkulasi kode izin tersebut akan disimpan sebagai 7 pada database sebagai representasi
3 kode izin 0b000.

4 Pengecekan izin **permission** dilakukan dengan memanfaatkan operator AND antara nilai pada
5 kolom **permission** dengan nilai **permission** yang diekspektasikan, lalu dibandingkan dengan nilai
6 ekspektasi. Sebagai contoh, sistem perlu mengecek apakah pengguna dapat membuat sebuah entri.
7 Sistem dapat melakukan pengecekan dengan melakukan operasi AND pada nilai **permission** yang
8 ada saat ini (0111) dengan nilai **permission** yang diekspektasi (0010), dan hasilnya dibandingkan
9 dengan hasil ekspektasi. Kalkulasi yang dilakukan dapat dilihat pada simulasi ekuasi berikut:

$$K \wedge C = C \quad (4.2a)$$

$$0b0111 \wedge C = C \quad (4.2b)$$

$$0b0111 \wedge 0b0001 = 0b0001 \quad (4.2c)$$

$$0b0001 = 0b0001 \quad (4.2d)$$

$$\text{True} \quad (4.2e)$$

10 Sedangkan jika pengecekan apakah seorang pengguna dapat melakukan penghapusan terhadap
11 sebuah entri, perhitungan perizinannya menjadi:

$$K \wedge D = D \quad (4.3a)$$

$$0b0111 \wedge D = D \quad (4.3b)$$

$$0b0111 \wedge 0b1000 = 0b1000 \quad (4.3c)$$

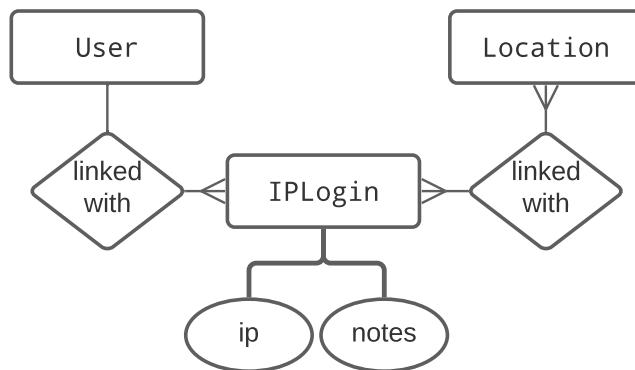
$$0b0000 = 0b1000 \quad (4.3d)$$

$$\text{False} \quad (4.3e)$$

12 Hasil dari ekuasi tersebut kemudian dapat digunakan untuk menentukan apakah seorang
13 pengguna dapat melakukan aksi yang diminta atau tidak.

14 Operasi ini terinspirasi dari **chmod** dari Linux¹. **chmod** menerima input berupa tiga angka yang
15 merepresentasikan level akses pada *resource* tertentu pada sistem. Tiga angka tersebut kemudian
16 diubah menjadi biner untuk dilihat apakah seorang pengguna dapat melakukan aksi tertentu.
17 Pendekatan ini kemudian diadaptasi pada sistem ini dengan menggunakan level akses yang berbeda

¹Lihat <https://linux.die.net/man/1/chmod>



Gambar 4.34: Potongan diagram entitas untuk IPLogin.

- ¹ (CRUD, dibanding RWX). Dengan demikian pengaturan dan pengelolaan resources pada entitas
- ² dapat lebih terkontrol dengan lebih fleksibel.

³ IPLogin

⁴ Entitas IPLogin digunakan untuk melakukan otentikasi semi-otomatis dengan memanfaatkan IP
⁵ pengguna pada lab, ditautkan dengan sebuah akun pengguna dengan peran yang terbatas (hanya
⁶ dapat melihat dan mengubah entitas tertentu), diperjelas pada Gambar 4.34. Entitas menyimpan
⁷ informasi IP, tautan pada pengguna dan lokasi-lokasi, serta catatan khusus tentang penautan
⁸ tersebut.

⁹ Entitas ini memiliki hubungan *many-to-many* dengan entitas Location. Relasi ini diharapkan
¹⁰ agar Tim Admin dapat melakukan pengaturan komputer master untuk melihat seluruh ujian yang
¹¹ aktif pada ruangan tertentu tanpa harus membuatkan akun khusus tertentu pada sistem. Tim
¹² Admin dapat dengan mudah menautkan sejumlah ruangan yang diinginkan dengan akun yang
¹³ memiliki peranan terbatas yang sudah ada sebelumnya.

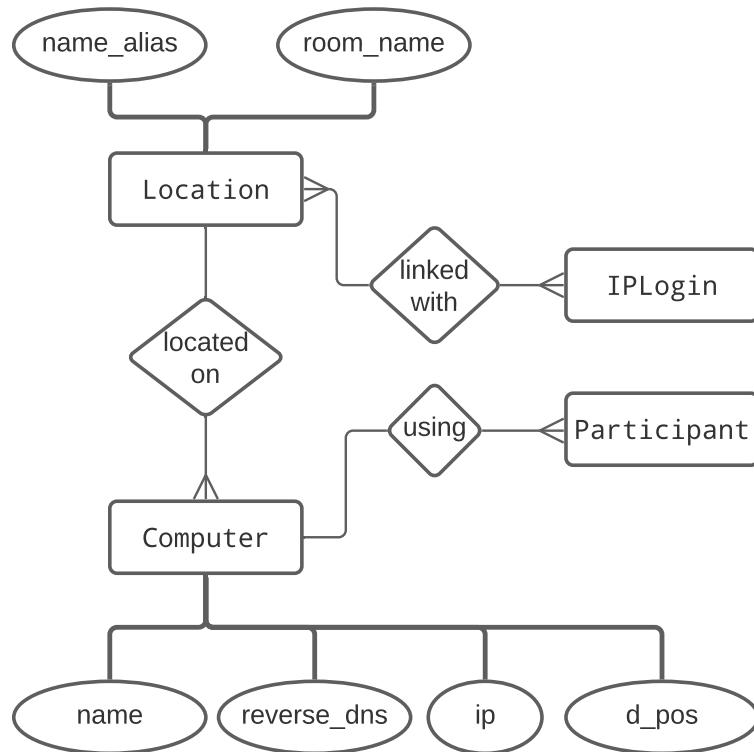
¹⁴ Entitas ini dinilai cukup esensial karena berhubungan dengan sistem otentikasi yang membatasi
¹⁵ pengguna untuk melakukan aksi tertentu di admin panel.

¹⁶ Location dan Computer

¹⁷ Entitas Location dan Computer menyimpan informasi tentang lokasi (ruangan) dan daftar pada
¹⁸ komputer tersebut sebelum nantinya dihubungkan dengan entitas lainnya. Hubungan antara kedua
¹⁹ entitas tersebut dapat diperhatikan pada potongan diagram entitas di gambar 4.35.

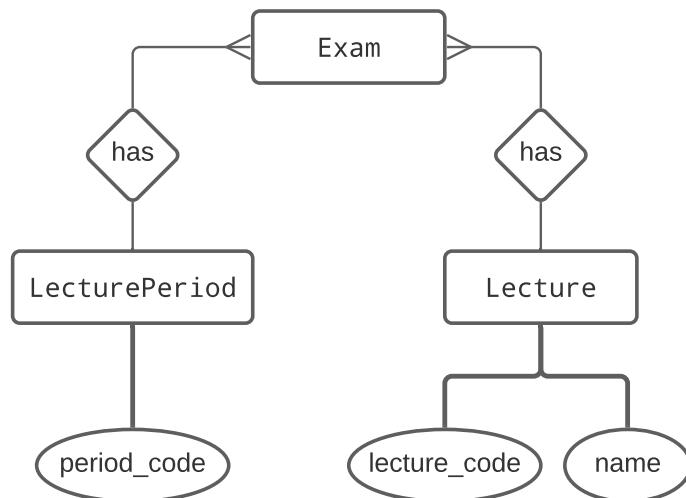
²⁰ Entitas Location menyimpan informasi lokasi dan ruangan tempat peserta dapat mengikuti
²¹ ujian. Pada entitas ini, informasi yang disimpan adalah nama ruangan tersebut dan nama lain dari
²² ruangan tersebut. Berdasarkan survei lapangan, setiap ruangan memiliki nama lain, sehingga kolom
²³ name_alias ditambahkan. Sedangkan nama ruangan disimpan pada kolom room_name.

²⁴ Entitas Computer mendefinisikan komputer yang terdapat pada lokasi tersebut. Oleh karena
²⁵ itu hubungan yang dimiliki dengan entitas Location adalah *one-to-many*. Entitas ini menyimpan
²⁶ informasi berupa nama komputer (*name*), IPnya (*ip*), nama FQDN (*Fully Qualified Domain Name*)



Gambar 4.35: Potongan diagram entitas untuk Location dan Computer.

- 1 atau domain dari komputer tersebut (*reverse_dns*), dan detil letak posisi dari komputer tersebut
- 2 pada peta ruangan (*d_pos*).
- 3 Kolom *d_pos* digunakan untuk menyimpan informasi letak komputer pada peta dalam bentuk
- 4 JSON. Bentuk format ini digunakan karena data ini digunakan hanya pada sistem frontend. Dengan
- 5 kesepakatan yang diharapkan cukup fleksibel, maka bentuk pemosian ini akan disimpan dalam
- 6 bentuk JSON.
- 7 Kolom *ip* digunakan untuk menautkan sebuah komputer dengan IP, sehingga tabel otentikasi
- 8 yang digunakan oleh peserta dapat langsung merujuk pada entitas ini. Setiap komputer peserta
- 9 yang ingin digunakan akan didaftarkan pada entitas ini. Untuk saat ini, karena survei lapangan
- 10 menunjukan bahwa versi IP yang digunakan adalah versi 4, maka fokus aplikasi saat ini akan
- 11 menggunakan IPv4.
- 12 Selain itu, terdapat kolom *reverse_dns* yang digunakan untuk menyimpan informasi nama domain
- 13 dari komputer tersebut. Kolom ini diharapkan dapat membantu Tim Admin dan pengembang
- 14 aplikasi berikutnya untuk melakukan *debugging* dan kebutuhan API lainnya. Nama domain ini
- 15 didapatkan dari LDAP yang digunakan oleh Server Windows untuk melakukan auditing objek
- 16 mereka. Komputer yang terhubung dengan Active Directory milik Lab akan terdaftar pada sistem
- 17 internal LDAP milik Windows Server yang nantinya akan diberikan domain khusus untuk komputer
- 18 tersebut.



Gambar 4.36: Potongan diagram entitas untuk **Lecture** dan **LecturePeriod**.

Nilai	Tipe Semester
1	Ganjil
2	Genap
3	Pendek

Tabel 4.2: Definisi tipe semester dengan nilainya untuk kolom **period_code** pada entitas **LecturePeriod**.

¹ Lecture dan LecturePeriod

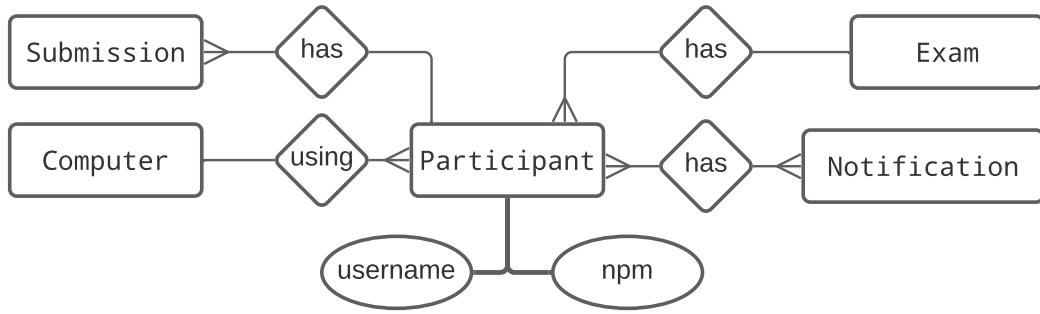
2 Entitas **Lecture** dan **LecturePeriod** menyimpan informasi tentang mata kuliah dan periode tahun
 3 ajar mata kuliah bersangkutan pada ujian tertentu. Hubungan antara kedua entitas tersebut
 4 terhadap entitas **Exam** dapat dilihat pada potongan diagram entitas di gambar 4.36. Entitas
 5 **Lecture** menyimpan informasi nama mata kuliah beserta kodennya, dan **LecturePeriod** menyimpan
 6 informasi tahun ajaran dalam bentuk kode.

7 Kode tersebut terdiri dari lima digit angka yang terdiri dari tahun ajar dan tipe semester yang
 8 sedang berjalan. Empat digit pertama adalah tahun ajar, dan satu digit terakhir adalah tipe
 9 semester. Tipe semester dan nilai representasinya dapat dilihat lebih jelas pada tabel 4.2.

10 Sebagai contoh, semester ganjil pada tahun ajar 2016-2017 direpresentasikan sebagai 20161.
 11 Empat digit pertama diambil dari tahun ajar pada tahun dimulai, 2016. Lalu berdasarkan tabel
 12 4.2, semester ganjil memiliki nilai representasi 1, sehingga kode untuk tahun ajar yang disimpan
 13 pada kolom *period_code* adalah 20161.

¹⁴ Participant

15 Entitas **Participant** digunakan untuk menyimpan informasi tentang peserta ujian di lab. Seperti
 16 pada potongan diagram relasi entitas pada gambar 4.37, entitas ini hanya memiliki dua kolom utama
 17 yang digunakan untuk merepresentasikan seorang peserta. Kolom **username** untuk menyimpan
 18 informasi username pada sistem dan **npm** untuk menyimpan informasi NPM dari peserta.



Gambar 4.37: Potongan diagram entitas untuk **Participant**.

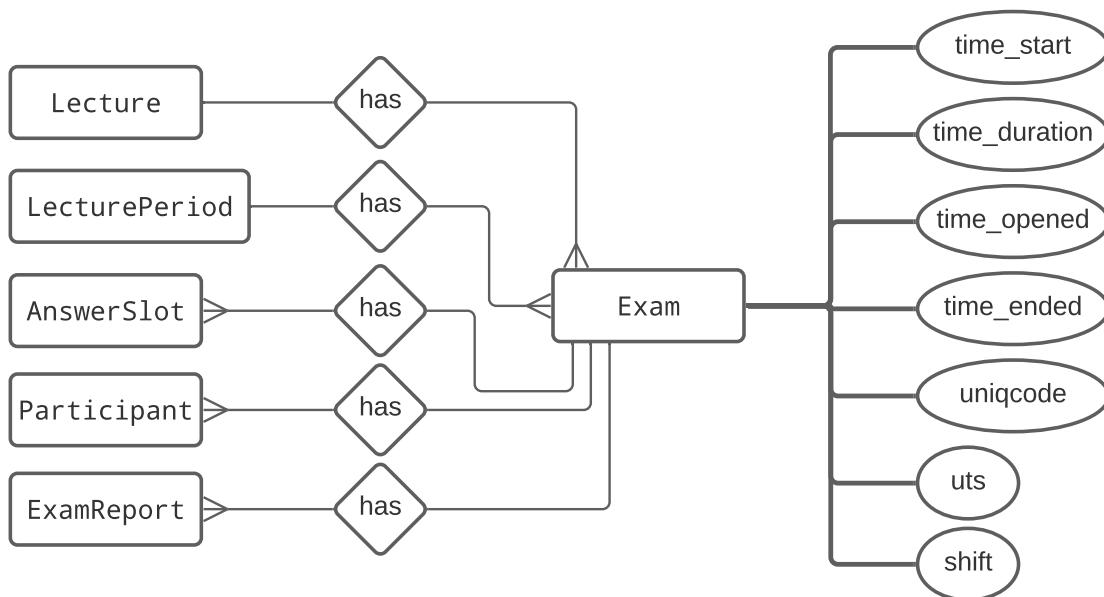
- ¹ Kolom **npm** digunakan untuk menyimpan string *literal* dari nomor pokok mahasiswa (NPM).
- ² Karena NPM mahasiswa yang aktif pada saat penelitian ini dibuat terdapat dua standar, maka untuk memperingan kinerna basis data, string *literal* NPM ikut disimpan.
- ³ Sedangkan kolom **username** digunakan untuk menyimpan username yang biasa digunakan di lab untuk login. Kolom ini akan berguna untuk membantu sistem membuat *script batch* pada tahap pemberian akses ke berkas bantuan ujian dan *workspace* peserta ujian. Pemberian akses dengan *script* membutuhkan nama pengguna yang digunakan untuk login.

⁸ **AnswerSlot**

- ⁹ Entitas **AnswerSlot** digunakan untuk menyimpan informasi tentang slot jawaban pada ujian tertentu. Entitas ini digunakan sebagai panduan untuk sistem memberikan informasi tentang format penamaan file, disimpan pada kolom **format**. Seperti yang terlihat pada diagram entitas pada 4.32, entitas ini akan terhubung secara *many-to-one* terhadap entitas **exam** dan *one to many* pada entitas **Submission**.
- ¹⁴ Jawaban yang dikirimkan oleh peserta nantinya akan disimpan pada entitas **Submission** dengan bantuan referensi format dari **AnswerSlot**. Setiap jawaban yang dikumpulkan akan dicek formatnya berdasarkan format yang diberikan pada entitas *AnswerSlot*.

¹⁷ **ExamReport**

- ¹⁸ Entitas **ExamReport** digunakan untuk membantu sistem menjadwalkan pengiriman laporan ujian. Laporan ini nantinya akan dikirimkan via email untuk mengantisipasi email diblokir oleh penyedia layanan email karena kode Java dan berkas **.class** dianggap sebagai virus.
- ²¹ Seperti yang ditunjukkan pada diagram entitas pada gambar 4.32, entitas **ExamReport** menyimpan beberapa kolom yang digunakan untuk membantu penjadwalan. Kolom pertama adalah **sent_on** yang digunakan untuk menandakan apakah ujian ini telah dikirimkan pada email (**to(s)**) ini. Kolom **valid_until** digunakan untuk menyimpan informasi kapan **token** akan kadaluwarsa (*expired*).
- ²⁶ Penggunaan token pada kasus ini dengan harapan alamat link yang dikirim oleh sistem ke dosen dapat dengan aman di akses. Kode token akan terdiri dari 16 byte acak yang dikonversi menjadi 32 digit heksadesimal string. Token ini diharapkan dibangkitkan dengan standar pengacakkan



Gambar 4.38: Potongan diagram entitas untuk Exam.

- 1 untuk kriptografi. Pengacakan dengan standar tersebut diharapkan dapat memperkecil faktor
- 2 kemungkinan serangan prediksi kode token (*guessing attack*).

3 Exam

- 4 Entitas berikutnya yang akan dibahas adalah entitas **Exam**, dapat dilihat pada potongan gambar 4.38.
- 5 Entitas ini menyimpan informasi utama untuk ujian dan jadwalnya. Informasi jadwal disimpan pada
- 6 kolom dengan *prefix time_*. Informasi dasar dari ujian tersebut disimpan dengan relasi langsung ke
- 7 tabel lain seperti **Lecture**, **LecturePeriod**, **AnswerSlot**, dan seterusnya.

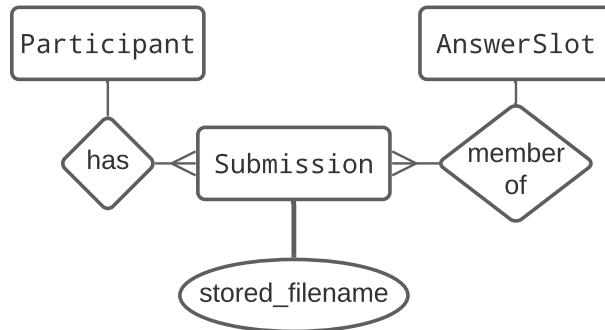
Informasi yang cukup penting pada entitas ini terdapat pada kolom **uniqcode**. Kolom ini menyimpan informasi kode unik dari sebuah ujian yang kita gunakan untuk menyimpan informasi nama folder berkas ujian. Kode ini diharapkan terdiri dengan byte acak yang dibangkitkan dengan standar pengacakan untuk kriptografi. Pengacakan dengan standar tersebut diharapkan dapat memperkecil faktor kemungkinan serangan prediksi kode unik (*guessing attack*).

Pengecekan ujian yang aktif dilakukan dengan melakukan pengecekan pada kolom **time_start** dan **time_ended**. Kolom **time_start** menandakan informasi tentang kapan ujian **akan** dimulai. Kolom ini tidak menentukan kapan lembar jawaban dapat *submit* oleh peserta ujian. Sedangkan kolom **time_ended** menandakan informasi tentang kapan lembar jawaban ditutup.

Untuk membuka lembar jawaban, kolom **time_opened** dan **time_ended** harus diisi dengan informasi tanggal kapan lembar jawaban telah dibuka. Pada keadaan bawaan, normalnya kolom tersebut seharusnya memiliki nilai **NULL**. Pada saat Dosen Pengawas melakukan aksi membuka jawaban, maka kedua kolom tersebut harus diisi sesuai dengan durasi yang diberikan pada kolom **time_duration**. Pengisian kolom **time_opened** dan **time_ended** diharapkan dapat mengurangi beban sistem untuk melakukan perhitungan secara terus menerus jika **time_ended** tidak diberikan. Selain itu, dengan mengisi dua kolom tersebut secara simultan, diharapkan dapat mengurangi

¹ potensi masalah ujian lupa ditutup.

² Submission



Gambar 4.39: Potongan diagram relasi entitas untuk **Submission**.

- ³ Entitas **Submission**, seperti yang dapat dilihat pada potongan diagram relasi entitas 4.39, menampung informasi tentang submisi yang dilakukan oleh peserta ujian. Submisi yang dikirimkan akan disimpan dengan nama berkas yang acak. Nama berkas tersebut kemudian disimpan pada database sebagai referensi untuk nantinya pelaporan yang dibantu oleh entitas **ExamReport**.
⁷ Submisi yang dikirimkan oleh peserta akan dicek formatnya dengan bantuan dari entitas **AnswerSlot**. Jika nama berkas sudah sesuai dengan format, maka data akan dimasukan pada entitas ini.

¹⁰ Notification

¹¹ Entitas **Notification** menyimpan informasi tentang notifikasi yang diberikan pada peserta. Karena beragam notifikasi dapat diberikan pada peserta, seperti informasi tentang kredensial untuk masuk ke sistem judge atau basis data; atau informasi tentang ralat soal. Oleh karena itu hubungan yang dimiliki entitas ini dengan entitas peserta adalah *many-to-many*.

¹⁵ Seperti pada diagram relasi entitas pada gambar 4.32, entitas **Notification** memiliki beberapa kolom untuk menyimpan isi notifikasi (**description**) dan judul (**title**). Kolom lainnya, seperti **type** digunakan untuk memasukan notifikasi terhadap grup tertentu. Kolom yang terakhir adalah **kolom extras**. Semua informasi yang tidak dapat direpresentasikan pada kolom akan disimpan dalam bentuk JSON pada kolom ini. Kolom ini disediakan karena kebutuhan fleksibilitas notifikasi yang mungkin akan muncul di masa yang akan datang.

²¹ 4.2.2 Rancangan REST API

²² REST API dirancang dengan pola yang seragam pada beberapa *endpoint* besar, sehingga penggunaan API dapat lebih optimal diaplikasikan dengan metode abstraksi API class pada sistem frontend.

²⁴ Perancangan REST API dimulai dengan merancang metode otentikasi yang akan digunakan untuk setiap permintaan. Lalu pembahasan dilanjutkan dengan desain URL yang digunakan dan tujuannya.

1 Otentikasi API

2 Sesuai dengan kebutuhan, Otentikasi API terbagi menjadi dua bagian besar berdasarkan pengguna-
3 annya. Otentikasi API yang pertama dilakukan dengan menggunakan IP, sedangkan yang kedua,
4 otentikasi dilakukan dengan menggunakan token.

5 Otentikasi dengan IP diperuntukkan untuk API yang berhubungan dengan ujian pada lokasi
6 yang telah didefinisikan sebelumnya. IP digunakan karena sistem sebelumnya menggunakan cara
7 ini dan telah terbukti pada survei lapangan bahwa cara ini sudah sangat efektif. Pengguna tidak
8 perlu melakukan otentikasi secara manual dengan memasukkan kredensial khusus untuk sistem.
9 Otentikasi dengan IP juga tidak memaksa klien untuk melakukan refresh secara terus menerus
10 untuk memperbarui *session* mereka, seperti pada aplikasi sebelumnya.

11 Otentikasi dengan token dilakukan dengan melakukan otentikasi manual hingga sistem mem-
12 berikan sebuah string khusus yang digunakan sebagai token untuk setiap permintaan. Otentikasi
13 manual tersebut dapat berupa memasukan kredensial login, atau dengan token khusus yang se-
14 belumnya diberikan pada email. Token ini nantinya dapat digunakan untuk mengakses berbagai
15 *resources* yang backend sediakan. Seperti tombol unduh pada halaman laporan, atau melakukan
16 CRUD pada entitas tertentu. Token yang digunakan pada sistem normalnya akan menggunakan
17 JSON Web Token atau JWT, dengan bantuan *library* dari backend.

18 Desain *Endpoint*

19 REST API dibagi menjadi empat bagian besar berdasarkan kebutuhannya. Bagian-bagian tersebut
20 memiliki teknik otentikasi yang berbeda-beda. Sebagai contoh untuk endpoint otentikasi tidak akan
21 memiliki otentikasi sama sekali, atau, otentikasi Ujian dilakukan secara tidak langsung dengan
22 menggunakan IP.

23 Seluruh endpoint akan memiliki prefiks /api/v1/ untuk memberikan fleksibilitas di masa yang
24 akan datang pada saat API akan diaplikasikan teknik *versioning*. Selain itu, dengan desain url
25 seperti ini, pengembang dapat mengetahui versi yang mereka gunakan pada API tersebut.

26 Pembahasan akan dimulai dengan endpoint untuk otentikasi terlebih dahulu, lalu dilanjutkan
27 dengan API ujian, API admin, dan API lainnya.

28 Desain *Endpoint* Otentikasi

29 Endpoint untuk otentikasi dapat di akses pada **system**. Endpoint ini akan bertugas untuk menangani
30 segala hal yang berhubungan dengan otentikasi. Mulai dari melakukan pengecekan login, mengambil
31 informasi pengguna yang sedang masuk, hingga membantu untuk menangani **IPLogin**. Endpoint
32 ini seharusnya bertanggung jawab untuk menerbitkan token JWT berdasarkan informasi pengguna
33 terkait yang terekam pada basisdata.

34 Terdapat satu buah pengecualian untuk otentikasi ujian. Otentikasi ujian dilakukan secara
35 tidak langsung dengan melakukan pengecekan IP langsung di level endpoint mereka. Endpoint
36 otentikasi tidak akan menerbitkan token untuk peserta ujian, karena peserta ujian dapat langsung
37 mengakses endpoint **exam** dengan catatan. IP komputer harus sudah terregistrasi di sistem.

38 Endpoint yang digunakan untuk otentikasi terdiri dari:

1 • POST `auth/login`

2 Bertanggung jawab untuk melakukan login, membangkitkan token, serta memberi informasi
3 singkat tentang pengguna yang masuk.

4 **Input:**

- 5 – `username`: *string*, Nama pengguna atau email.
6 – `password`: *string*, Kata sandi pengguna.

7 **Output:**

- 8 – `id_token`: *string*, Token otentikasi.
9 – `profile`: *object*, objek dari entitas `User`.

10 • POST `auth/iplogin`

11 Bertanggung jawab untuk melakukan login dengan IP, membangkitkan token, serta memberi
12 informasi singkat tentang pengguna yang tertau dengan IP tersebut.

13 **Input:** -

14 **Output:**

- 15 – `id_token`: *string*, Token otentikasi.
16 – `profile`: *object*, objek dari entitas `User`.

17 • GET `user/me`

18 Bertanggung jawab untuk memberikan informasi tentang user yang terautentikasi. **Input:** -

19 **Output:**

- 20 – `profile`: *object*, objek dari entitas `User`.

21 • POST `user/me`

22 Bertanggung jawab untuk memperbarui informasi tentang user yang terautentikasi. **Input:** -
23 Objek dari entitas `User`.

24 **Output:**

- 25 – `profile`: *object*, objek dari entitas `User`.

26 **Desain Endpoint Ujian**

27 Endpoint untuk ujian memiliki prefiks `exam`. Endpoint ini akan menggunakan otentikasi IP dan
28 tidak membutuhkan penggunanya untuk mengirimkan informasi header `Authorization` apapun.
29 Endpoint ini akan melayani segala kebutuhan peserta untuk ujian.

30 Endpoint yang melayani API untuk ujian terdiri dari:

31 • GET `info`

32 Bertanggung jawab untuk mengembalikan informasi ujian yang sedang dan akan dimulai.

33 **Input:** -

34 **Output:** objek dari entitas `Participant`

- 1 • GET notification
- 2 Bertanggung jawab untuk mengembalikan daftar notifikasi.
- 3 **Input:** -
- 4 **Output:** array objek entitas Notifikasi
- 5 • GET submission
- 6 Bertanggung jawab untuk mengembalikan status submisi pada slot jawaban tertentu.
- 7 **Input:** -
- 8 **Output:** objek dari entitas Submission
- 9 • POST submission
- 10 Bertanggung jawab untuk mengirimkan berkas jawaban (selanjutnya disebut submisi) pada slot jawaban tertentu.
- 11 **Input:**
- 12 – **file:** *multipart-form*, berkas ujian dalam bentuk blob.
- 13 – **answer_slot:** *string/number*, id dari slot jawaban yang akan disubmisi.
- 14 **Output:** objek dari entitas Submission

16 Desain *Endpoint Admin*

17 Slot endpoint yang digunakan untuk admin mengatur berbagai fitur yang tersedia pada admin
18 panel. Endpoint ini membutuhkan otentikasi pengguna level admin. Setiap entitas memiliki tabel
19 izinnya masing-masing yang dapat diperoleh dari entitas `AclItem`. Namun pada dasarnya, setiap
20 admin berhak untuk melakukan CRUD pada entitas tersebut, kecuali disebutkan sebaliknya.

21 Enpoint pada admin ini memiliki prefiks `manage`. Karena enpoint ini berhubungan langsung
22 dengan entitas, maka pada dasarnya seluruh entitas memiliki endpoint khusus dengan pola REST
23 API sebagai berikut

- 24 • GET :entity
- 25 Bertanggung jawab untuk mengembalikan daftar entri dari :entity yang disebutkan.
- 26 **Input:** -
- 27 **Output:** array objek entitas :entity
- 28 • POST :entity
- 29 Bertanggung jawab untuk membuat entri baru dari :entity yang disebutkan.
- 30 **Input:** objek (atau beberapa bagian) dari entitas entity.
- 31 **Output:** objek dari entitas :entity.
- 32 • GET :entity/:id
- 33 Bertanggung jawab untuk mengembalikan info detil entri dari :entity dengan id :id.
- 34 **Input:** -
- 35 **Output:** objek dari entitas :entity
- 36 • PUT :entity/:id
- 37 Bertanggung jawab untuk mengubah atau memperbarui entri dengan id :id pada entitas

1 entity.

2 **Input:** objek atau beberapa bagian) dari entitas :entity.

3 **Output:** objek dari entitas :entity

4 • DELETE :entity/:id
5 Bertanggung jawab untuk menghapus entri :id pada entity.
6 **Input:** -
7 **Output:** objek dari entitas :entity.

8 Selain dari pola tersebut, terdapat beberapa endpoint khusus untuk memfasilitasi aksi yang
9 dapat dilakukan pada entitas tersebut. Endpoint tersebut adalah

10 • DELETE exam/:id/start
11 Bertanggung jawab untuk melakukan reset timer ujian pada id tertentu.
12 **Input:** -
13 **Output:** object objek entitas Exam.

14 • GET exam/:id/answers
15 Bertanggung jawab untuk melakukan pengumpulan jawaban dan pengunduhan dalam bentuk
16 zip.
17 **Input:** -
18 **Output:** blob berkas zip.

19 • GET exam/:id/notifications
20 Bertanggung jawab untuk mengembalikan daftar notifikasi yang tertaut pada peserta di ujian
21 yang disebutkan.
22 **Input:** -
23 **Output:** array objek entitas Notification.

24 • GET exam/:id/participants
25 Bertanggung jawab untuk mengembalikan daftar peserta yang terdapat pada ujian ini.
26 **Input:** -
27 **Output:** array objek entitas Participant.

28 • GET exam/:id/script
29 Bertanggung jawab untuk membuat script yang digunakan untuk membuat lembar kerja
30 peserta ujian.
31 **Input:** -
32 **Output:** blob berkas zip.

33 • POST exam/:id/close
34 Bertanggung jawab untuk menghentikan timer ujian dan menutup lembar jawab peserta secara
35 manual.
36 **Input:** -
37 **Output:** object objek entitas Exam.

- 1 • POST exam/:id/move

2 Bertanggung jawab untuk memindahkan peserta ke tempat duduk lain dan membuat *script*
3 untuk lembar kerja mahasiswa yang baru.

4 **Input:**

- 5 – lists[] .participant: *string/id*, id dari peserta.
6 – lists[] .to: *string/id*, id dari komputer target.

7 **Output:** *blob* berkas zip.

- 8 • POST exam/:id/populate

9 Bertanggung jawab untuk melakukan populasi tempat duduk yang sudah dipilih dengan daftar
10 peserta yang ada.

11 **Input:**

- 12 – computers: *array* id dari entitas Computer.
13 – participants: *array* id dari entitas Participant.
14 – should_cleanup: *boolean* Hapus daftar peserta yang sudah ada.

15 **Output:** objek dari entitas Exam.

- 16 • POST exam/:id/start

17 Bertanggung jawab untuk memulai timer pada ujian.

18 **Input:** -

19 **Output:** *object* objek entitas Exam.

- 20 • POST examreport/:id/forcesend

21 Bertanggung jawab untuk melakukan pengiriman laporan ujian secara paksa ke email yang
22 terdapat pada entri :id.

23 **Input:** -

24 **Output:** -

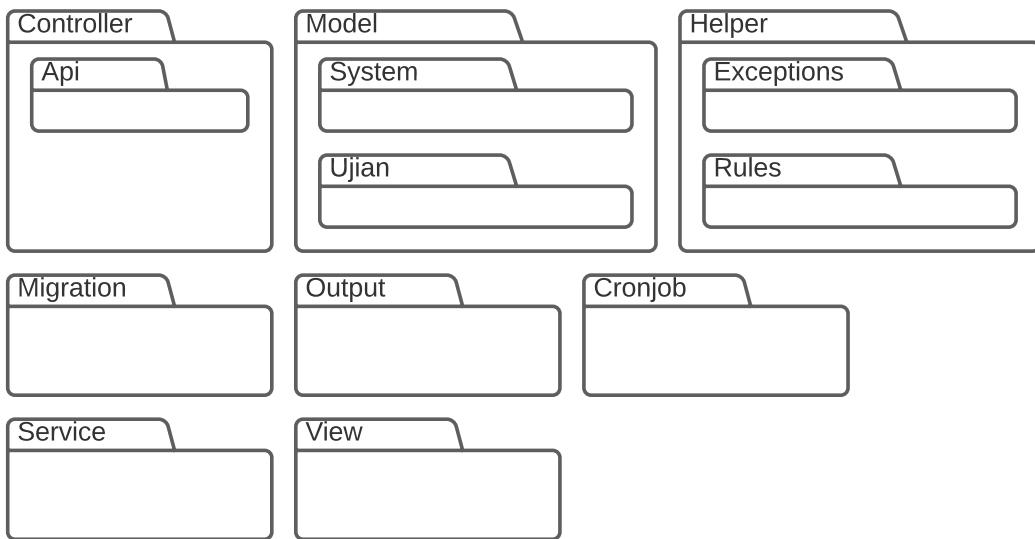
- 25 • POST notification/mass_gen

26 Bertanggung jawab untuk melakukan penyebaran notifikasi kata sandi pada daftar yang
27 diberikan pada ujian tertentu.

28 **Input:**

- 29 – lists[] .participant: *string/id* id dari entitas Participant.
30 – lists[] .username: *string* Nama pengguna layanan
31 – lists[] .password: *string* Kata sandi layanan
32 – url: *string* Alamat layanan

33 **Output:** -



Gambar 4.40: Gambaran besar dari perancangan kelas untuk sistem *back-end*.

¹ Desain *Endpoint* Lainnya

- ² Endpoint lainnya terdiri dari endpoint yang dapat digunakan dengan otentikasi khusus. Otentifikasi tersebut dapat dengan menukar token *payload* tertentu menjadi token bentuk lain. Endpoint tersebut terdiri dari

- ⁵ • POST `autonomus/examdownload`

⁶ Endpoint ini bertanggung jawab untuk mengembalikan informasi tentang token yang terhubung
⁷ dengan ujian tertentu, serta memberikan link otorisasi pengunduhan berkas jawaban.

⁸ **Input:**

- ⁹ – `token`: *string*, token dari laporan ujian yang dikirimkan via email.

¹⁰ **Output:**

- ¹¹ – `exam`: *object*, objek dari entitas `Exam`.
- ¹² – `downloadToken`: *string*, tautan lengkap untuk mengunduh ujian.

- ¹³ • GET `autonomus/examdownload/download`

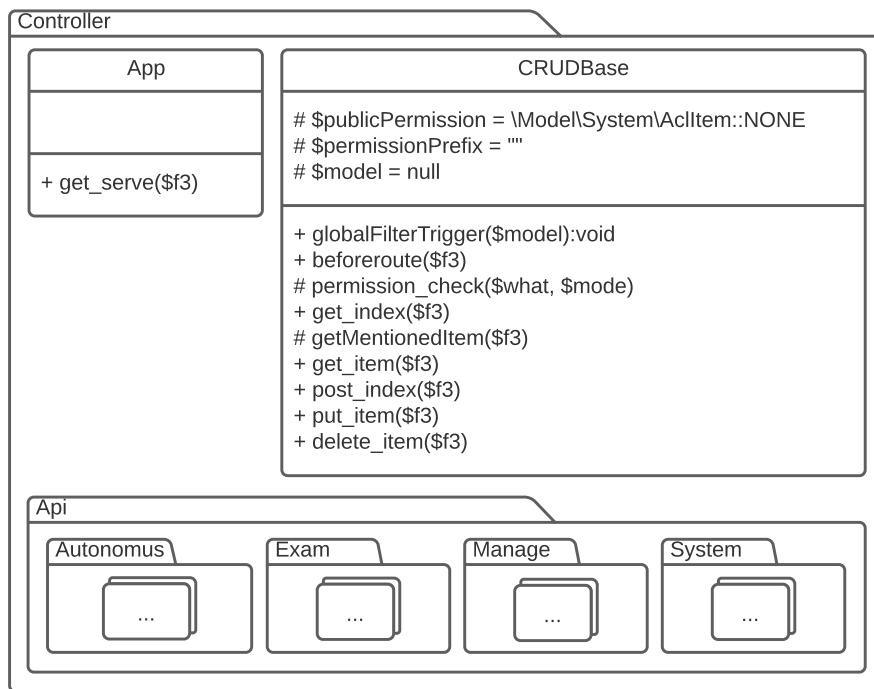
¹⁴ Endpoint ini bertanggung jawab untuk menyediakan berkas jawaban dalam bentuk zip.

¹⁵ **Input:** -

¹⁶ **Output:** *blob* berkas zip.

¹⁷ 4.2.3 Desain Kelas

- ¹⁸ Perancangan desain kelas dilakukan dengan mempertimbangkan pola yang telah diberikan oleh beberapa *library*. Karena *framework* yang digunakan cukup fleksibel, maka dibuat beberapa *namespace* sesuai dengan konteks penggunaan kelasnya masing-masing. Secara garis besar, *namespace* yang ada aplikasi dapat dilihat pada Gambar 4.40.



Gambar 4.41: Diagram kelas untuk *namespace Controller*.

1 Kemudian setiap *namespace* akan dijelaskan secara terperinci beserta dengan kelas yang terdapat
 2 pada *namespace* tersebut.

3 ***Namespace Controller***

4 *Namespace* ini akan berisi kelas-kelas yang bertanggung jawab untuk menangani permintaan.
 5 Permintaan tersebut terdiri dari permintaan dari API dan permintaan non-API. Diagram kelas
 6 untuk *namespace* ini dapat diperhatikan pada Gambar 4.41

7 *Namespace* ini terdiri dari kelas-kelas berikut:

8 • **App**

9 Kelas ini bertanggung jawab untuk menyajikan aplikasi React. Oleh karena itu sesuai dengan
 10 desain rutennya, kelas ini akan menangani permintaan pada *url*:

- 11 – GET /admin
- 12 – GET /admin/*
- 13 – GET /exam
- 14 – GET /exam/*
- 15 – GET /autonomus
- 16 – GET /autonomus/*

17 Definisi rute lengkap URL dapat dilihat pada lampiran (TODO: lampiran routings.ini).

18 Pada kelas ini terdapat fungsi:

1 – `get_serve($f3)`

2 Fungsi ini yang bertanggung jawab untuk menyajikan aplikasi React.

3 **Input:** Instansi kelas `Base` dari framework.

4 **Output:** –

5 • `CRUDBase`

6 Kelas ini akan menjadi kelas yang menangani abstraksi dari REST API. Atribut yang terdapat
7 pada kelas ini adalah:

8 – `$publicPermission`

9 Mendefinisikan permisi yang dimiliki oleh pengkonsumsi API tanpa otentikasi.

10 – `$permissionPrefix`

11 Mendefinisikan nama permisi yang tertaut pada pengkonsumsi API yang terotentikasi.

12 – `$model`

13 Mendefinisikan nama kelas model yang akan digunakan. Nama kelas yang diharapkan
14 adalah *fully qualified name*².

15 Fungsi yang terdapat pada kelas ini adalah:

16 – `globalFilterTrigger($model):void`

17 Menangani filter global untuk model yang ditautkan untuk kelas ini.

18 **Input:** instansi model

19 **Output:** –

20 – `beforeroute($f3)`

21 Menangani *event* sebelum *framework* melakukan pemanggilan fungsi utama.

22 **Input:** Instansi kelas `Base` dari framework.

23 **Output:** –

24 – `permission_check($what, $mode)`

25 Melakukan pengecekan permisi/izin pada otentikasi pengkonsumsi API. Mungkin akan
26 melempar eksepsi jika pengkonsumsi tidak memiliki izin.

27 **Input:** Nama permisi, dan mode yang seharusnya diperbolehkan (buat, baca, ubah,
28 hapus)

29 **Output:** –

30 – `get_index($f3)`

31 Menangani permintaan API untuk melakukan *listing* entri dari model

32 **Input:** Instansi kelas `Base` dari framework.

33 **Output:** –

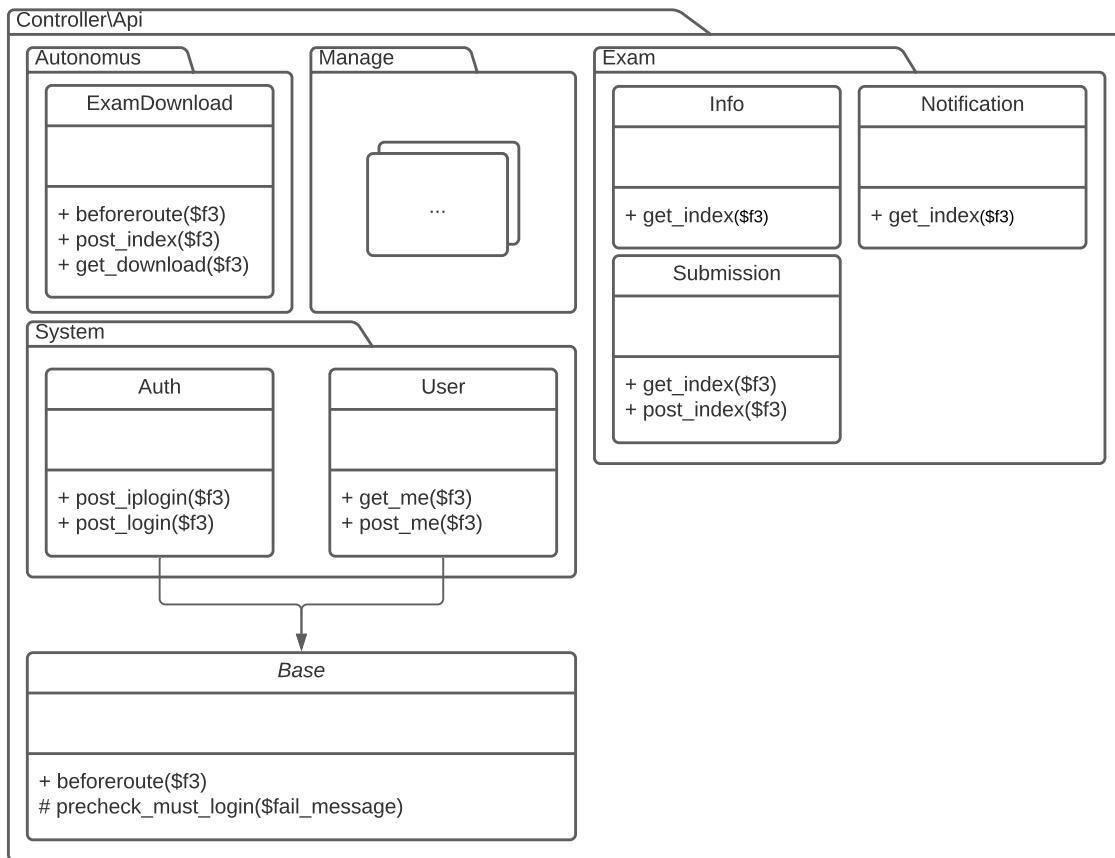
34 – `getMentionedItem($f3)`

35 Ambil entri yang disebutkan pada *url*.

36 **Input:** Instansi kelas `Base` dari framework.

37 **Output:** Instansi model yang tertaut dengan sebuah entri.

²Lihat <https://www.php.net/manual/en/language.namespaces.rules.php>



Gambar 4.42: Potongan diagram kelas untuk *namespace Controller\Api*.

1 – `get_item($f3)`

2 Menangani permintaan API untuk menampilkan infomasi lengkap entri.

3 **Input:** Instansi kelas `Base` dari framework.

4 **Output:** -

5 – `post_index($f3)`

6 Menangani permintaan API untuk membuat entri baru.

7 **Input:** Instansi kelas `Base` dari framework.

8 **Output:** -

9 – `put_item($f3)`

10 Menangani permintaan API untuk memperbarui/mengubah informasi pada entri.

11 **Input:** Instansi kelas `Base` dari framework.

12 **Output:** -

13 – `delete_item($f3)`

14 Menagnagi permintaa API untuk menghapus entri.

15 **Input:** Instansi kelas `Base` dari framework.

16 **Output:** -

17 Selain dari kelas-kelas tersebut, *namespace* ini memiliki anak *namespace* `Api`. *Namespace* tersebut
18 secara gais besar akan menangani berbagai macam permintaan API yang akan disediakan oleh

1 sistem *back-end*. Diagram kelas untuk *namespace* ini dapat dilihat pada Gambar 4.42.

2 *Namespace Api* terdiri dari:

- 3 • **Base**

4 Kelas ini bertanggung jawab untuk menangani abstraksi API dasar yang diperlukan. Kelas
5 ini memiliki fungsi:

- 6 – **beforeroute(\$f3)**

7 Menangani *event* sebelum *framework* melakukan pemanggilan fungsi utama. Melakukan
8 parsing dan pengecekan tertentu pada permintaan API.

9 **Input:** Instansi kelas **Base** dari framework.

- 10 • **Autonomus\ExamDownload**

11 Kelas ini bertanggung jawab untuk melayani API untuk pengunduhan jawaban via tautan
12 yang dikirimkan ke email. Kelas ini memiliki fungsi:

- 13 – **beforeroute(\$f3)**

14 Menangani *event* sebelum *framework* melakukan pemanggilan fungsi utama.

15 **Input:** Instansi kelas **Base** dari framework.

16 **Output:** -

- 17 – **post_index(\$f3)**

18 Menangani penukaran token dengan informasi ujian dan menerbitkan tautan untuk
19 melakukan pengunduhan.

20 **Input:** Instansi kelas **Base** dari framework.

21 **Output:** -

22 **URL:** POST autonomus/examdownload

- 23 – **get_download(\$f3)**

24 Menyediakan layanan pengunduhan berkas berupa Zip. **Input:** Instansi kelas **Base** dari
25 framework.

26 **Output:** -

27 **URL:** GET autonomus/examdownload/download

- 28 • **Exam\Info**

29 Bertanggung jawab untuk memberikan informasi tentang ujian yang tertaut pada IP pengkon-
30 sumsi API pada waktu tertentu. Kelas ini memiliki fungsi:

- 31 – **get_index(\$f3)**

32 Memberikan informasi tentang ujian yang sedang aktif. **Input:** Instansi kelas **Base** dari
33 framework.

34 **Output:** -

35 **URL:** GET exam/info

- 36 • **Exam\Notification**

37 Bertanggung jawab untuk memberikan notifikasi yang tertaut dengan peserta yang sedang
38 aktif. Kelas ini memiliki fungsi:

```
1   – get_index($f3)
2     Memberikan daftar entri notifikasi. Input: Instansi kelas Base dari framework.
3     Output: -
4     URL: GET exam/notification
5
6   • Exam\Submission
7     Bertanggung jawab untuk menangani berbagai hal seputar submisi peserta. Kelas ini memiliki
8     fungsi:
9
10    – get_index($f3)
11      Memberikan informasi tentang submisi pada slot jawaban tertentu. Input: Instansi
12      kelas Base dari framework.
13      Output: -
14      URL: GET exam/submission
15
16    – get_index($f3)
17      Menangani submisi yang dikirimkan peserta pada slot jawaban tertentu. Input: Instansi
18      kelas Base dari framework.
19      Output: -
20      URL: POST exam/submission
21
22   • System\Auth
23     Bertanggung jawab untuk menangani berbagai hal seputar otentikasi dengan kata sandi dan
24     nama pengguna.
25
26    – post_iplogin($f3)
27      Menangani permintaan untuk login dengan menggunakan IP.
28      Input: Instansi kelas Base dari framework.
29      Output: -
30      URL: POST system/auth/iplogin
31
32    – post_login($f3)
33      Menangani permintaan untuk login dengan menggunakan kombinasi nama pengguna
34      atau email, dan kata sandi.
35      Input: Instansi kelas Base dari framework.
36      Output: -
37      URL: POST system/auth/login
38
39   • System\User
40     Bertanggung jawab untuk menangani berbagai hal seputar otentikasi dengan kata sandi dan
41     nama pengguna.
42
43    – get_me($f3)
44      Menangani permintaan informasi tentang user itu sendiri.
45      Input: Instansi kelas Base dari framework.
46      Output: -
47      URL: GET system/user/me
```

1 – `post_me($f3)`
2 Menangani permintaan untuk mengubah informasi tentang user itu sendiri .
3 **Input:** Instansi kelas `Base` dari framework.
4 **Output:** -
5 **URL:** POST `system/user/me`

6 Kemudian terdapat kelas-kelas dari *namespace Controller/Api/Manage* yang memiliki banyak
7 kelas turunan dari `CRUDBase`. Diagram kelas tersebut dapat dilihat pada Gambar 4.43. *Namespace*
8 tersebut memiliki kelas-kelas berikut:

9 • **Acl**

10 Bertanggung jawab untuk menangani permintaan REST API untuk entitas `Acl`. Kelas ini
11 memiliki atribut:

- 12 – `$permissionPrefix`: Turunan dari kelas `CRUDBase`.
13 – `$model`: Turunan dari kelas `CRUDBase`.

14 • **AclItem**

15 Bertanggung jawab untuk menangani permintaan REST API untuk entitas `AclItem`. Kelas
16 ini memiliki atribut:

- 17 – `$permissionPrefix`: Turunan dari kelas `CRUDBase`.
18 – `$model`: Turunan dari kelas `CRUDBase`.

19 • **AnswerSlot**

20 Bertanggung jawab untuk menangani permintaan REST API untuk entitas `AnswerSlot`. Kelas
21 ini memiliki atribut:

- 22 – `$permissionPrefix`: Turunan dari kelas `CRUDBase`.
23 – `$model`: Turunan dari kelas `CRUDBase`.

24 • **Computer**

25 Bertanggung jawab untuk menangani permintaan REST API untuk entitas `Computer`. Kelas
26 ini memiliki atribut:

- 27 – `$permissionPrefix`: Turunan dari kelas `CRUDBase`.
28 – `$model`: Turunan dari kelas `CRUDBase`.

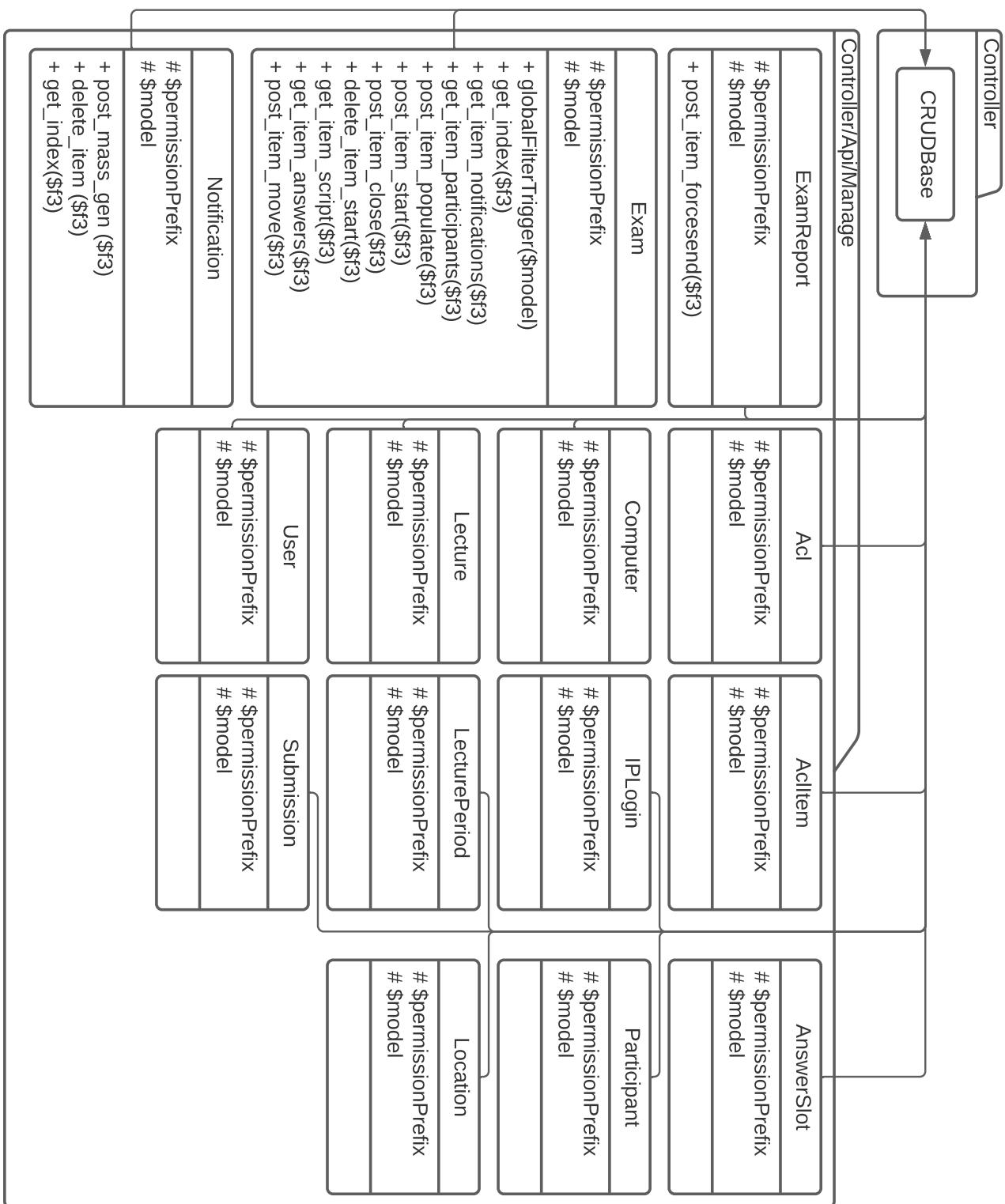
29 • **IPLogin**

30 Bertanggung jawab untuk menangani permintaan REST API untuk entitas `IPLogin`. Kelas
31 ini memiliki atribut:

- 32 – `$permissionPrefix`: Turunan dari kelas `CRUDBase`.
33 – `$model`: Turunan dari kelas `CRUDBase`.

34 • **Participant**

35 Bertanggung jawab untuk menangani permintaan REST API untuk entitas `Participant`.
36 Kelas ini memiliki atribut:

Gambar 4.43: Potongan diagram kelas untuk `namespace Controller/Api/Manage`.

- 1 – `$permissionPrefix`: Turunan dari kelas CRUDBase.
 - 2 – `$model`: Turunan dari kelas CRUDBase.
 - 3 • **Lecture**
4 Bertanggung jawab untuk menangani permintaan REST API untuk entitas `Lecture`. Kelas
5 ini memiliki atribut:

6 – `$permissionPrefix`: Turunan dari kelas CRUDBase.
7 – `$model`: Turunan dari kelas CRUDBase.
 - 8 • **LecturePeriod**
9 Bertanggung jawab untuk menangani permintaan REST API untuk entitas `LecturePeriod`.
10 Kelas ini memiliki atribut:

11 – `$permissionPrefix`: Turunan dari kelas CRUDBase.
12 – `$model`: Turunan dari kelas CRUDBase.
 - 13 • **Location**
14 Bertanggung jawab untuk menangani permintaan REST API untuk entitas `Location`. Kelas
15 ini memiliki atribut:

16 – `$permissionPrefix`: Turunan dari kelas CRUDBase.
17 – `$model`: Turunan dari kelas CRUDBase.
 - 18 • **User**
19 Bertanggung jawab untuk menangani permintaan REST API untuk entitas `User`. Kelas ini
20 memiliki atribut:

21 – `$permissionPrefix`: Turunan dari kelas CRUDBase.
22 – `$model`: Turunan dari kelas CRUDBase.
 - 23 • **Submission**
24 Bertanggung jawab untuk menangani permintaan REST API untuk entitas `Submission`. Kelas
25 ini memiliki atribut:

26 – `$permissionPrefix`: Turunan dari kelas CRUDBase.
27 – `$model`: Turunan dari kelas CRUDBase.
 - 28 • **ExamReport**
29 Bertanggung jawab untuk menangani permintaan REST API untuk entitas `ExamReport`. Kelas
30 ini memiliki atribut:

31 – `$permissionPrefix`: Turunan dari kelas CRUDBase.
32 – `$model`: Turunan dari kelas CRUDBase.
- 33 Selain itu, kelas ini memiliki fungsi:

1 – `post_item_forcesend($f3)`
2 Bertanggung jawab untuk mengirimkan laporan ke email yang terdapat pada entri secara
3 manual.

4 **Input:** Instansi kelas `Base` dari framework.

5 **Output:** -

6 **URL:** POST `manage/examreport/:id/forcesend`

7 • **Exam**

8 Bertanggung jawab untuk menangani permintaan REST API untuk entitas `Exam`. Kelas ini
9 memiliki atribut:

- 10 – `$permissionPrefix`: Turunan dari kelas `CRUDBase`.
11 – `$model`: Turunan dari kelas `CRUDBase`.

12 Selain itu, kelas ini memiliki fungsi:

13 – `globalFilterTrigger($model)`
14 Diturunkan dari kelas `CRUDBase`. Digunakan untuk melimitasi daftar entri yang
15 dikembalikan khusus untuk pengkonsumsi API yang terotentikasi via IPLLogin.

16 **Input:** Instansi dari model yang ditautkan ke kelas ini.

17 **Output:** -

18 – `get_index($f3)`
19 Mengoverride fungsi dari kelas `CRUDBase`. Digunakan untuk menampilkan subset
20 dengan kueri khusus untuk pengkonsumsi API yang terotentikasi via IPLLogin.

21 **Input:** Instansi kelas `Base` dari framework.

22 **Output:** -

23 **URL:** GET `manage/exam/:id`

24 – `get_item_notifications($f3)`
25 Menangani permintaan untuk mengembalikan daftar notifikasi yang terhubung dengan
26 peserta yang mengikuti ujian ini.

27 **Input:** Instansi kelas `Base` dari framework.

28 **Output:** -

29 **URL:** GET `manage/exam/:id/notifications`

30 – `get_item_participants($f3)`
31 Menangani permintaan untuk mengembalikan daftar peserta yang terhubung dengan
32 ujian ini.

33 **Input:** Instansi kelas `Base` dari framework.

34 **Output:** -

35 **URL:** GET `manage/exam/:id/participants`

36 – `post_item_populate($f3)`
37 Melakukan populasi tempat duduk dengan daftar peserta ujian.

38 **Input:** Instansi kelas `Base` dari framework.

39 **Output:** -

40 **URL:** POST `manage/exam/:id/populate`

```
1   - post_item_start($f3)
2     Bertanggung jawab untuk memulai timer.
3     Input: Instansi kelas Base dari framework.
4     Output: -
5     URL: POST manage/exam/:id/start
6   - post_item_close($f3)
7     Bertanggung jawab utnuk menutup timer.
8     Input: Instansi kelas Base dari framework.
9     Output: -
10    URL: POST manage/exam/:id/close
11   - delete_item_start($f3)
12     Bertanggung jawab untuk menyetel ulang timer.
13     Input: Instansi kelas Base dari framework.
14     Output: -
15     URL: DELETE manage/exam/:id/start
16   - get_item_script($f3)
17     Menangani pembuatan script untuk membuat lembar kerja ujian peserta dan menyalin
18     berkas pendukung ujian ke komputer peserta.
19     Input: Instansi kelas Base dari framework.
20     Output: -
21     URL: GET manage/exam/:id/script
22   - get_item_answers($f3)
23     Bertanggung jawab untuk mengumpulkan jawaban, lalu menggabungkannya dalam
24     sebuah archive zip.
25     Input: Instansi kelas Base dari framework.
26     Output: -
27     URL: GET manage/exam/:id/answers
28   - post_item_move($f3)
29     Menangani pemindahan peserta pada komputer tertentu.
30     Input: Instansi kelas Base dari framework.
31     Output: -
32     URL: POST manage/exam/:id/move
33   • Notification
34     Bertanggung jawab untuk menangani permintaan REST API untuk entitas Notification.
35     Kelas ini memiliki atribut:
36       - $permissionPrefix: Turunan dari kelas CRUDBase.
37       - $model: Turunan dari kelas CRUDBase.
38     Kelas ini memiliki fungsi:
39       - post_mass_gen($f3)
40         Bertanggung jawab untuk membuat notifikasi tentang kredensial akun untuk tiap peserta
```

ujian pada ujian tertentu.

Input: Instansi kelas `Base` dari framework.

Output: -

URL: POST `manage/notification/mass_gen`

- `delete_item($f3)`
Digunakan untuk menghapus item notifikasi secara paksa (mematikan fitur *soft-delete*) pada model.

Input: Instansi kelas `Base` dari framework.

Output: -

URL: DELETE `manage/notification/:id`

- `get_index($f3)`
Digunakan untuk mengembalikan daftar notifikasi yang tidak ter-*soft-delete*.

Input: Instansi kelas `Base` dari framework.

Output: -

URL: GET `manage/notification`

16 Namespace Model

17 *Namespace* ini akan berisi kelas-kelas yang bertanggung jawab merepresentasikan entitas dalam
 18 bentuk kelas dengan memanfaatkan teknologi ORM. Diagram kelas tersebut dapat dilihat pada
 19 Gambar 4.44.

20 Kelas-kelas yang terdapat pada *namespace* ini terdiri dari

- 21 • **ModelBase**

22 Kelas ini bertanggung jawab untuk menyediakan abstraksi untuk setiap model yang digunakan
 23 pada aplikasi. Atribut yang terdapat pada kelas ini adalah:

- 24 – `$validators`: Digunakan untuk menyimpan daftar validator dalam bentuk pasangan key-
 25 value, tergantung dari jenis kolom yang didefinisikan pada kelas representasi entitasnya.

26 Selain itu, fungsi yang dimiliki oleh kelas ini adalah:

- 27 – `copyfromwithfilter($data, $strict = false)`

28 Bertanggung jawab untuk melakukan penyalinan data dengan syarat yang terdapat pada
 29 definisi kelas representasi entitas.

30 **Input:** Data yang ada, apakah harus berada dalam mode ketat.

31 **Output:** -

- 32 – `_validate($data)`

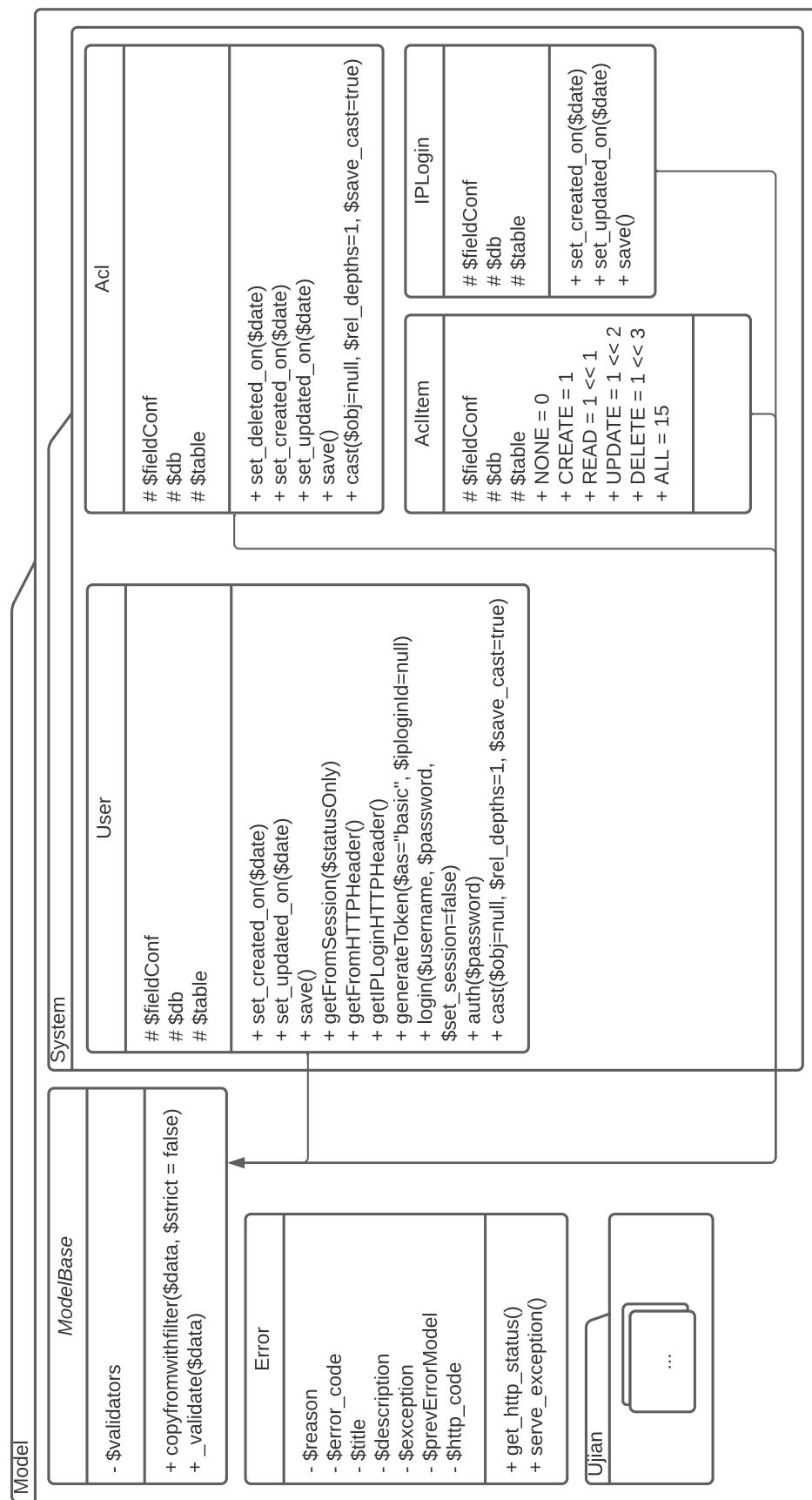
33 Bertanggung jawab untuk melakukan validasi data yang diberikan dengan tabel definisi
 34 validasi yang diberikan pada atribut `$validator`.

35 **Input:** data yang akan divalidasi

36 **Output:** `true` jika data valid, *Exception* jika terdapat pelanggaran validasi.

- 37 • **Error**

38 Bertanggung jawab untuk memodelkan pesan kesalahan yang dapat diserialisasi menjadi



Gambar 4.44: Potongan diagram kelas untuk *namespace Model*.

1 respon yang dapat dipresentasikan untuk pengkonsumsi API. Kelas ini bukan bagian dari
 2 entitas ataupun ORM, kelas ini bagian dari sistem dan siklus hidupnya. Kelas ini memiliki
 3 atribut:

- 4 – `$reason`: Menyimpan informasi alasan singkat kenapa pesan kesalahan ini diterbitkan.
- 5 – `$error_code`: Menyimpan informasi kode kesalahan untuk internal sistem.
- 6 – `$title`: Menyimpan informasi judul dari pesan kesalahan ini.
- 7 – `$description`: Menyimpan informasi detil dari kesalahan yang muncul.
- 8 – `$exception`: Objek eksepsi sebagai pendukung informasi pesan kesalahan.
- 9 – `$prevErrorModel`: Objek kesalahan sebelumnya sebagai pendukung informasi pesan
 kesalahan.
- 10 – `$http_code`: Kode HTTP yang dapat merepresentasikan pesan kesalahan ini. (Contohnya 400 untuk permintaan yang tidak valid, 500 untuk galat yang tidak terduga, dan seterusnya).

14 Selain itu, kelas ini memiliki fungsi:

- 15 – `get_http_status()`
 Getter untuk kode respon HTTP pada pesan kesalahan ini.
Input: -
Output: kode respon HTTP dalam representasi angka maupun *string*.
- 16 – `serve_exception()`
 Bertanggung jawab untuk memformat pesan kesalahan sebelum disajikan pada pengkonsumsi API.
Input: -
Output: representasi pesan kesalahan dalam bentuk pasangan *hashmap key-value*.

24 • `System\User`

25 Kelas ini merepresentasikan entitas User, bagian dari ORM, bertanggung jawab untuk
 26 memberikan token untuk otentikasi, juga mendeteksi pengguna yang terotentikasi berdasarkan
 27 informasi header. Kelas ini memiliki atribut yang diturunkan dari kelas utama ORM, F3-
 28 Cortex. Atribut-atribut tersebut terdiri dari

- 29 – `$fieldConf`
 Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel *User*.
- 30 – `$db` dengan nilai awal "DB".
 Menentukan database yang akan digunakan pada model ini.
- 31 – `$table` dengan nilai awal "system_user".
 Nama tabel yang akan digunakan pada sistem basis data.

35 Sedangkan fungsi yang terdapat pada kelas ini adalah:

- 36 – `set_created_on($date)`
 Mengkonversi bentuk tanggal kolom `created_on` native ke format basis data.

1 **Input:** Tanggal dalam bentuk *Unix time*
2 **Output:** *string* tanggal terformat.

3 – **set_updated_on(\$date)**
4 Mengkonversi bentuk tanggal kolom *updated_on native* ke format basis data.

5 **Input:** Tanggal dalam bentuk *Unix time*
6 **Output:** *string* tanggal terformat.

7 – **save()**
8 Meng-*override* kelas dari ORM. Bertanggung jawab untuk mengisi kolom *created_on*
9 dan *updated_on*.

10 **Input:** –
11 **Output:** –

12 – **getFromSession(\$statusOnly)**
13 Bertanggung jawab untuk melakukan pengecekan pengguna terotentikasi melalui *session*.
14 **Input:** Apakah bentuk kembalian yang diinginkan hanyalah statusnya saja atau objek
15 penuh
16 **Output:** *null* pada saat kosong, "user" atau objek dari kelas ini jika otentikasi
17 ditemukan.

18 – **getFromHTTPHeader()**
19 Cek otentikasi dari kepala permintaan HTTP.
20 **Input:** –
21 **Output:** Objek dari kelas ini jika terdapat otentikasi, *false* jika tidak ada.

22 – **getIPLoginHTTPHeader()**
23 Cek IPLogin dari kepala permintaan HTTP pada otentikasi yang terhubung.
24 **Input:** –
25 **Output:** Objek dari kelas IPLogin jika terdapat otentikasi, *null* jika tidak ada.

26 – **generateToken(\$as="basic", \$iploginId=null)**
27 Bangkitkan token JWT untuk pengguna ini.
28 **Input:** Peran token ini, IPLogin tertaut.
29 **Output:** *string* token.

30 – **login(\$username, \$password, \$set_session=false)**
31 Lakukan pengecekan kredensial login berdasarkan nama pengguna dan password.
32 **Input:** nama pengguna, kata sandi, apakah ingin sekaligus disimpan ke *session*
33 **Output:** Objek dari user dimaksud, atau *Exception* jika terdapat kesalahan.

34 – **auth(\$password)**
35 Lakukan verifikasi kata sandi yang dimiliki pengguna ini.
36 **Input:** Kata sandi dari pengguna.
37 **Output:** *boolean* apakah password benar atau tidak.

38 – **cast(\$obj=null, \$rel_depths=1, \$save_cast=true)**
39 Meng-*override* kelas dari ORM. Bertanggung jawab untuk merepresentasikan kelas dalam
40 bentuk *hashmap* secara aman atau tidak.
41 **Input:** Instansi yang akan di-*cast*, konfigurasi relasi, dan lakukan *casting* yang aman

1 atau tidak.

2 **Output:** *array*.

3 • **System\Acl**

4 Kelas ini merepresentasikan entitas *Access Control List* atau ACL. Kelas ini bagian dari ORM.

5 Atribut yang terdapat pada kelas ini terdiri dari:

6 – **\$fieldConf**

7 Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel *ACL*.

8 – **\$db** dengan nilai awal "DB".

9 Menentukan database yang akan digunakan pada model ini.

10 – **\$table** dengan nilai awal "system_acl".

11 Nama tabel yang akan digunakan pada sistem basis data.

12 Sedangkan fungsi yang terdapat pada kelas ini adalah:

13 – **set_deleted_on(\$date)**

14 Mengkonversi bentuk tanggal kolom *deleted_on native* ke format basis data.

15 **Input:** Tanggal dalam bentuk *Unix time*

16 **Output:** *string* tanggal terformat.

17 – **set_created_on(\$date)**

18 Mengkonversi bentuk tanggal kolom *created_on native* ke format basis data.

19 **Input:** Tanggal dalam bentuk *Unix time*

20 **Output:** *string* tanggal terformat.

21 – **set_updated_on(\$date)**

22 Mengkonversi bentuk tanggal kolom *updated_on native* ke format basis data.

23 **Input:** Tanggal dalam bentuk *Unix time*

24 **Output:** *string* tanggal terformat.

25 – **save()**

26 Meng-*override* kelas dari ORM. Bertanggung jawab untuk mengisi kolom *created_on*
27 dan *updated_on*.

28 **Input:** -

29 **Output:** -

30 – **cast(\$obj=null, \$rel_depths=1, \$save_cast=true)**

31 Meng-*override* kelas dari ORM. Bertanggung jawab untuk merepresentasikan kelas dalam
32 bentuk *hashmap* secara aman atau tidak.

33 **Input:** Instansi yang akan di-*cast*, konfigurasi relasi, dan lakukan *casting* yang aman
34 atau tidak.

35 **Output:** *array*.

36 • **System\AclItem**

37 Kelas ini adalah kelas terkait ORM yang merepresentasikan entitas AclItem. Kelas ini memiliki
38 daftar atribut:

- 1 – **\$fieldConf**
2 Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel *ACLItem*.
- 3 – **\$db** dengan nilai awal "DB".
4 Menentukan database yang akan digunakan pada model ini.
- 5 – **\$table** dengan nilai awal "system_aclitem".
6 Nama tabel yang akan digunakan pada sistem basis data.
- 7 – **NONE**: Nilai konstanta tidak memiliki izin apapun.
- 8 – **CREATE**: Nilai konstanta untuk izin membuat.
- 9 – **READ**: Nilai konstanta untuk izin membaca/melihat.
- 10 – **UPDATE**: Nilai konstanta untuk izin mengubah/memperbarui.
- 11 – **DELETE**: Nilai konstanta untuk izin menghapus.
- 12 – **ALL**: Nilai konstanta untuk semua izin diperbolehkan.

- 13 • **System\IPLogin**

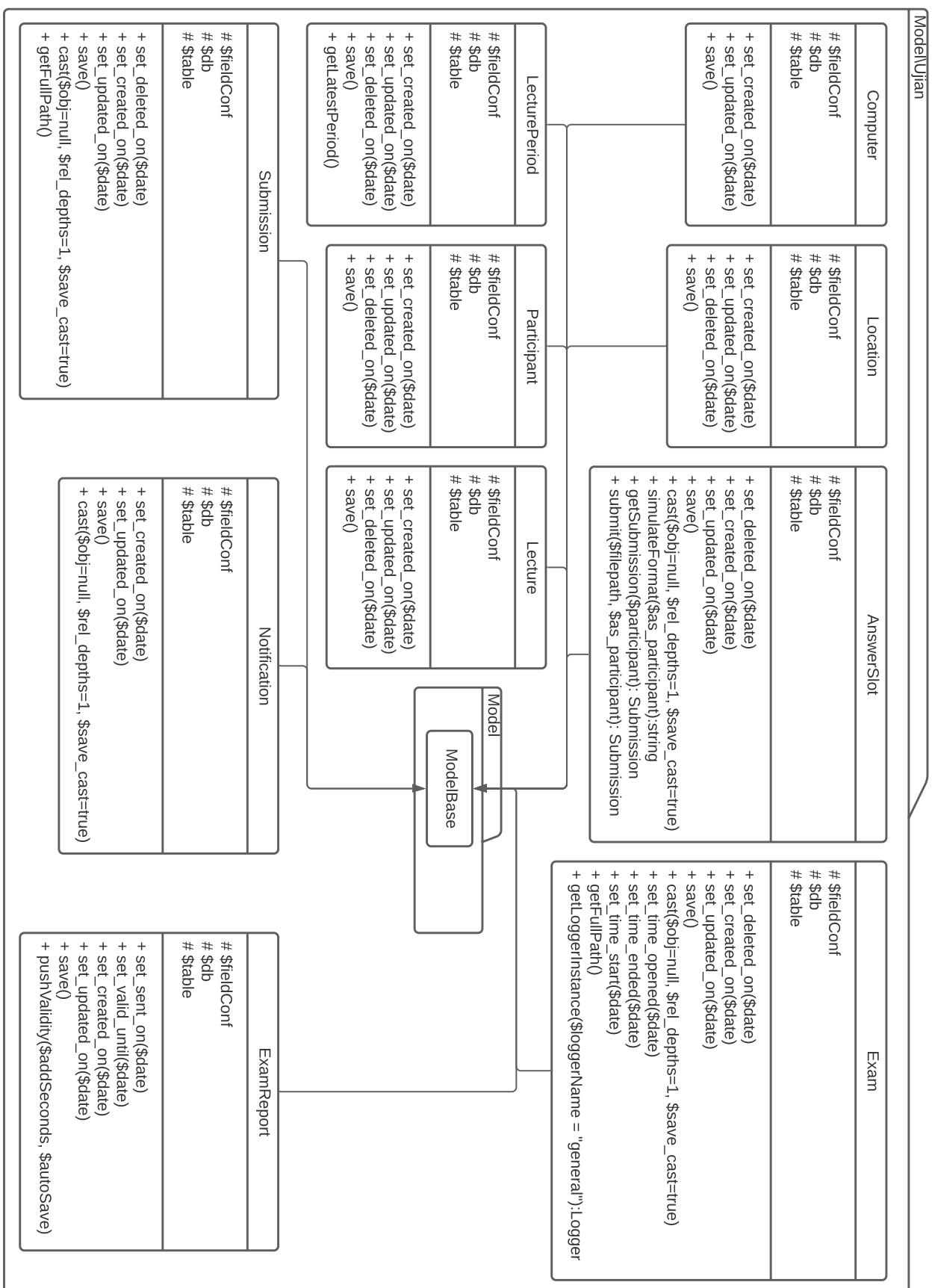
Kelas ini adalah kelas terkait ORM yang merepresentasikan entitas IPlogin. Atribut yang terdapat pada kelas ini terdiri dari:

- 16 – **\$fieldConf**
17 Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel *ACL*.
- 18 – **\$db** dengan nilai awal "DB".
19 Menentukan database yang akan digunakan pada model ini.
- 20 – **\$table** dengan nilai awal "system_iplogin".
21 Nama tabel yang akan digunakan pada sistem basis data.

Sedangkan fungsi yang terdapat pada kelas ini adalah:

- 23 – **set_created_on(\$date)**
24 Mengkonversi bentuk tanggal kolom *created_on* native ke format basis data.
25 **Input:** Tanggal dalam bentuk *Unix time*
26 **Output:** string tanggal terformat.
- 27 – **set_updated_on(\$date)**
28 Mengkonversi bentuk tanggal kolom *updated_on* native ke format basis data.
29 **Input:** Tanggal dalam bentuk *Unix time*
30 **Output:** string tanggal terformat.
- 31 – **save()**
32 Meng-override kelas dari ORM. Bertanggung jawab untuk mengisi kolom *created_on* dan *updated_on*.
33 **Input:** -
34 **Output:** -

36 Namespace Model\Ujian memiliki beberapa kelas ORM yang berhubungan langsung dengan
37 ujian. Diagram kelas untuk *namespace* tersebut dapat dilihat pada Gambar 4.45. Kelas-kelas pada
38 *namespace* ini terdiri dari

Gambar 4.45: Potongan diagram kelas untuk *namespace Model/ujian*.

- 1 • Computer

2 Kelas ini adalah kelas terkait ORM yang merepresentasikan entitas Computer. Atribut yang
3 terdapat pada kelas ini terdiri dari:

4 – \$fieldConf

5 Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel *Computer*.

6 – \$db dengan nilai awal "DB".

7 Menentukan database yang akan digunakan pada model ini.

8 – \$table dengan nilai awal "ujian_computer".

9 Nama tabel yang akan digunakan pada sistem basis data.

10 Sedangkan fungsi yang terdapat pada kelas ini adalah:

11 – set_created_on(\$date)

12 Mengkonversi bentuk tanggal kolom *created_on* *native* ke format basis data.

13 **Input:** Tanggal dalam bentuk *Unix time*

14 **Output:** *string* tanggal terformat.

15 – set_updated_on(\$date)

16 Mengkonversi bentuk tanggal kolom *updated_on* *native* ke format basis data.

17 **Input:** Tanggal dalam bentuk *Unix time*

18 **Output:** *string* tanggal terformat.

19 – save()

20 Meng-*override* kelas dari ORM. Bertanggung jawab untuk mengisi kolom *created_on*
21 dan *updated_on*.

22 **Input:** -

23 **Output:** -

24 • Location

25 Kelas ini adalah kelas terkait ORM yang merepresentasikan entitas Location. Atribut yang
26 terdapat pada kelas ini terdiri dari:

27 – \$fieldConf

28 Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel *Location*.

29 – \$db dengan nilai awal "DB".

30 Menentukan database yang akan digunakan pada model ini.

31 – \$table dengan nilai awal "ujian-computer".

32 Nama tabel yang akan digunakan pada sistem basis data.

33 Sedangkan fungsi yang terdapat pada kelas ini adalah:

34 – set_created_on(\$date)

35 Mengkonversi bentuk tanggal kolom *created_on* *native* ke format basis data.

36 **Input:** Tanggal dalam bentuk *Unix time*

37 **Output:** *string* tanggal terformat.

```
1   – set_updated_on($date)
2     Mengkonversi bentuk tanggal kolom updated_on native ke format basis data.
3     Input: Tanggal dalam bentuk Unix time
4     Output: string tanggal terformat.
5   – set_deleted_on($date)
6     Mengkonversi bentuk tanggal kolom deleted_on native ke format basis data.
7     Input: Tanggal dalam bentuk Unix time
8     Output: string tanggal terformat.
9   – save()
10    Meng-override kelas dari ORM. Bertanggung jawab untuk mengisi kolom created_on,
11      updated_on dan deleted_on.
12    Input: -
13    Output: -
```

- 14 • **AnswerSlot**

15 Kelas ini adalah kelas terkait ORM yang merepresentasikan entitas AnswerSlot. Atribut yang
16 terdapat pada kelas ini terdiri dari:

```
17   – $fieldConf
18     Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel AnswerSlot.
19   – $db dengan nilai awal "DB".
20     Menentukan database yang akan digunakan pada model ini.
21   – $table dengan nilai awal "ujian-computer".
22     Nama tabel yang akan digunakan pada sistem basis data.
```

23 Sedangkan fungsi yang terdapat pada kelas ini adalah:

```
24   – set_created_on($date)
25     Mengkonversi bentuk tanggal kolom created_on native ke format basis data.
26     Input: Tanggal dalam bentuk Unix time
27     Output: string tanggal terformat.
28   – set_updated_on($date)
29     Mengkonversi bentuk tanggal kolom updated_on native ke format basis data.
30     Input: Tanggal dalam bentuk Unix time
31     Output: string tanggal terformat.
32   – set_deleted_on($date)
33     Mengkonversi bentuk tanggal kolom deleted_on native ke format basis data.
34     Input: Tanggal dalam bentuk Unix time
35     Output: string tanggal terformat.
36   – save()
37     Meng-override kelas dari ORM. Bertanggung jawab untuk mengisi kolom created_on,
38      updated_on dan deleted_on.
39     Input: -
40     Output: -
```

```
1   - cast($obj=null, $rel_depths=1, $save_cast=true)
2     Meng-override kelas dari ORM. Bertanggung jawab untuk merepresentasikan kelas dalam
3     bentuk hashmap secara aman atau tidak.
4     Input: Instansi yang akan di-cast, konfigurasi relasi, dan lakukan casting yang aman
5     atau tidak.
6     Output: array.
7   - simulateFormat($as_participant)
8     Bertanggung jawab untuk membangkitkan nama berkas yang ditunjukkan untuk slot
9     jawaban ini pada peserta tertentu.
10    Input: Detil peserta dalam kelas Participant.
11    Output: string nama berkas jawaban.
12   - getSubmission($participant)
13     Bertanggung jawab untuk mengembalikan instansi Submission untuk peserta tertentu.
14     Input: Instansi objek peserta.
15     Output: objek Submission
16   - submit($filepath, $as_participant)
17     Melakukan submisi untuk slot jawaban ini dengan informasi lokasi berkas dan informasi
18     peserta.
19     Input: Lokasi berkas, objek Participant.
20     Output: objek Submission
```

21 • **Exam**

22 Kelas ini adalah kelas terkait ORM yang merepresentasikan entitas *Exam*. Atribut yang
23 terdapat pada kelas ini terdiri dari:

```
24   - $fieldConf
25     Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel Exam.
26   - $db dengan nilai awal "DB".
27     Menentukan database yang akan digunakan pada model ini.
28   - $table dengan nilai awal "ujian-exam".
29     Nama tabel yang akan digunakan pada sistem basis data.
```

30 Sedangkan fungsi yang terdapat pada kelas ini adalah:

```
31   - set_created_on($date)
32     Mengkonversi bentuk tanggal kolom created_on native ke format basis data.
33     Input: Tanggal dalam bentuk Unix time
34     Output: string tanggal terformat.
35   - set_updated_on($date)
36     Mengkonversi bentuk tanggal kolom updated_on native ke format basis data.
37     Input: Tanggal dalam bentuk Unix time
38     Output: string tanggal terformat.
```

```

1   - set_deleted_on($date)
2     Mengkonversi bentuk tanggal kolom deleted_on native ke format basis data.
3     Input: Tanggal dalam bentuk Unix time
4     Output: string tanggal terformat.
5   - save()
6     Meng-override kelas dari ORM. Bertanggung jawab untuk mengisi kolom created_on,
7     updated_on dan deleted_on.
8     Input: -
9     Output: -
10    - cast($obj=null, $rel_depths=1, $save_cast=true)
11      Meng-override kelas dari ORM. Bertanggung jawab untuk merepresentasikan kelas dalam
12      bentuk hashmap secara aman atau tidak.
13      Input: Instansi yang akan di-cast, konfigurasi relasi, dan lakukan casting yang aman
14      atau tidak.
15      Output: array.
16    - set_time_opened($date)
17      Mengkonversi bentuk tanggal kolom time_opened native ke format basis data.
18      Input: Tanggal dalam bentuk Unix time
19      Output: string tanggal terformat.
20    - set_time_ended($date)
21      Mengkonversi bentuk tanggal kolom time_ended native ke format basis data.
22      Input: Tanggal dalam bentuk Unix time
23      Output: string tanggal terformat.
24    - set_time_start($date)
25      Mengkonversi bentuk tanggal kolom time_start native ke format basis data.
26      Input: Tanggal dalam bentuk Unix time
27      Output: string tanggal terformat.
28    - getFullPath()
29      Bertanggung jawab untuk mengembalikan alamat lengkap dari ujian ini.
30      Input: -
31      Output: string alamat lengkap menuju folder ujian ini.
32    - getLoggerInstance($loggerName = "general")
33      Bertanggung jawab untuk mengembalikan objek Logger yang dapat digunakan untuk
34      melakukan logging
35      Input: Nama label untuk log yang ingin dibuat
36      Output: Objek Logger
37
38  • LecturePeriod
39    Kelas ini adalah kelas terkait ORM yang merepresentasikan entitas LecturePeriod. Atribut
40    yang terdapat pada kelas ini terdiri dari:
41
42    - $fieldConf
43      Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel LecturePeriod.

```

- 1 – `$db` dengan nilai awal "DB".
2 Menentukan database yang akan digunakan pada model ini.
- 3 – `$table` dengan nilai awal "ujian_computer".
4 Nama tabel yang akan digunakan pada sistem basis data.

5 Sedangkan fungsi yang terdapat pada kelas ini adalah:

- 6 – `set_created_on($date)`
7 Mengkonversi bentuk tanggal kolom `created_on` native ke format basis data.
8 **Input:** Tanggal dalam bentuk *Unix time*
9 **Output:** *string* tanggal terformat.
- 10 – `set_updated_on($date)`
11 Mengkonversi bentuk tanggal kolom `updated_on` native ke format basis data.
12 **Input:** Tanggal dalam bentuk *Unix time*
13 **Output:** *string* tanggal terformat.
- 14 – `set_deleted_on($date)`
15 Mengkonversi bentuk tanggal kolom `deleted_on` native ke format basis data.
16 **Input:** Tanggal dalam bentuk *Unix time*
17 **Output:** *string* tanggal terformat.
- 18 – `save()`
19 Meng-*override* kelas dari ORM. Bertanggung jawab untuk mengisi kolom `created_on`,
20 `updated_on` dan `deleted_on`.
21 **Input:** -
22 **Output:** -
- 23 – `getLatestPeriod()`
24 Bertanggung jawab untuk mengambil periode terbaru pada tahun aktif. Jika tidak
25 tersedia pada basis data, maka fungsi ini akan membuat periode yang baru.
26 **Input:** -
27 **Output:** Objek *LecturePeriod* berisi baris tertentu.

28 • **Participant**

29 Kelas ini adalah kelas terkait ORM yang merepresentasikan entitas Participant. Atribut yang
30 terdapat pada kelas ini terdiri dari:

- 31 – `$fieldConf`
32 Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel *Participant*.
- 33 – `$db` dengan nilai awal "DB".
34 Menentukan database yang akan digunakan pada model ini.
- 35 – `$table` dengan nilai awal "ujian_participant".
36 Nama tabel yang akan digunakan pada sistem basis data.

37 Sedangkan fungsi yang terdapat pada kelas ini adalah:

```
1   – set_created_on($date)
2     Mengkonversi bentuk tanggal kolom created_on native ke format basis data.
3     Input: Tanggal dalam bentuk Unix time
4     Output: string tanggal terformat.
5   – set_updated_on($date)
6     Mengkonversi bentuk tanggal kolom updated_on native ke format basis data.
7     Input: Tanggal dalam bentuk Unix time
8     Output: string tanggal terformat.
9   – set_deleted_on($date)
10    Mengkonversi bentuk tanggal kolom deleted_on native ke format basis data.
11    Input: Tanggal dalam bentuk Unix time
12    Output: string tanggal terformat.
13   – save()
14     Meng-override kelas dari ORM. Bertanggung jawab untuk mengisi kolom created_on,
15     updated_on dan deleted_on.
16     Input: -
17     Output: -
```

18 • **Lecture**

19 Kelas ini adalah kelas terkait ORM yang merepresentasikan entitas Lecture. Atribut yang
20 terdapat pada kelas ini terdiri dari:

```
21   – $fieldConf
22     Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel Lecture.
23   – $db dengan nilai awal "DB".
24     Menentukan database yang akan digunakan pada model ini.
25   – $table dengan nilai awal "ujian_lecture".
26     Nama tabel yang akan digunakan pada sistem basis data.
```

27 Sedangkan fungsi yang terdapat pada kelas ini adalah:

```
28   – set_created_on($date)
29     Mengkonversi bentuk tanggal kolom created_on native ke format basis data.
30     Input: Tanggal dalam bentuk Unix time
31     Output: string tanggal terformat.
32   – set_updated_on($date)
33     Mengkonversi bentuk tanggal kolom updated_on native ke format basis data.
34     Input: Tanggal dalam bentuk Unix time
35     Output: string tanggal terformat.
36   – set_deleted_on($date)
37     Mengkonversi bentuk tanggal kolom deleted_on native ke format basis data.
38     Input: Tanggal dalam bentuk Unix time
39     Output: string tanggal terformat.
```

- 1 – **save()**
2 Meng-*override* kelas dari ORM. Bertanggung jawab untuk mengisi kolom *created_on*,
3 *updated_on* dan *deleted_on*.
4 **Input:** -
5 **Output:** -
- 6 • **Submission**
7 Kelas ini adalah kelas terkait ORM yang merepresentasikan entitas Submission. Atribut yang
8 terdapat pada kelas ini terdiri dari:
9 – **\$fieldConf**
10 Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel *Submission*.
11 – **\$db** dengan nilai awal "DB".
12 Menentukan database yang akan digunakan pada model ini.
13 – **\$table** dengan nilai awal "ujian_submission".
14 Nama tabel yang akan digunakan pada sistem basis data.
- 15 Sedangkan fungsi yang terdapat pada kelas ini adalah:
16 – **set_created_on(\$date)**
17 Mengkonversi bentuk tanggal kolom *created_on* *native* ke format basis data.
18 **Input:** Tanggal dalam bentuk *Unix time*
19 **Output:** *string* tanggal terformat.
20 – **set_updated_on(\$date)**
21 Mengkonversi bentuk tanggal kolom *updated_on* *native* ke format basis data.
22 **Input:** Tanggal dalam bentuk *Unix time*
23 **Output:** *string* tanggal terformat.
24 – **set_deleted_on(\$date)**
25 Mengkonversi bentuk tanggal kolom *deleted_on* *native* ke format basis data.
26 **Input:** Tanggal dalam bentuk *Unix time*
27 **Output:** *string* tanggal terformat.
28 – **save()**
29 Meng-*override* kelas dari ORM. Bertanggung jawab untuk mengisi kolom *created_on*,
30 *updated_on* dan *deleted_on*.
31 **Input:** -
32 **Output:** -
33 – **cast(\$obj=null, \$rel_depths=1, \$save_cast=true)**
34 Meng-*override* kelas dari ORM. Bertanggung jawab untuk merepresentasikan kelas dalam
35 bentuk *hashmap* secara aman atau tidak.
36 **Input:** Instansi yang akan di-*cast*, konfigurasi relasi, dan lakukan *casting* yang aman
37 atau tidak.
38 **Output:** *array*.

1 – `getFullPath()`
 2 Bertanggung jawab untuk mengembalikan alamat lengkap menuju berkas submisi ini.
 3 **Input:** -
 4 **Output:** *string* alamat lengkap menuju berkas submisi

5 • **Notification**

6 Kelas ini adalah kelas terkait ORM yang merepresentasikan entitas Notification. Atribut yang
 7 terdapat pada kelas ini terdiri dari:

8 – `$fieldConf`
 9 Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel *Notification*.
 10 – `$db` dengan nilai awal "DB".
 11 Menentukan database yang akan digunakan pada model ini.
 12 – `$table` dengan nilai awal "ujian_notification".
 13 Nama tabel yang akan digunakan pada sistem basis data.

14 Sedangkan fungsi yang terdapat pada kelas ini adalah:

15 – `set_created_on($date)`
 16 Mengkonversi bentuk tanggal kolom `created_on` native ke format basis data.
 17 **Input:** Tanggal dalam bentuk *Unix time*
 18 **Output:** *string* tanggal terformat.
 19 – `set_updated_on($date)`
 20 Mengkonversi bentuk tanggal kolom `updated_on` native ke format basis data.
 21 **Input:** Tanggal dalam bentuk *Unix time*
 22 **Output:** *string* tanggal terformat.
 23 – `set_deleted_on($date)`
 24 Mengkonversi bentuk tanggal kolom `deleted_on` native ke format basis data.
 25 **Input:** Tanggal dalam bentuk *Unix time*
 26 **Output:** *string* tanggal terformat.
 27 – `save()`
 28 Meng-*override* kelas dari ORM. Bertanggung jawab untuk mengisi kolom `created_on`,
 29 `updated_on` dan `deleted_on`.
 30 **Input:** -
 31 **Output:** -
 32 – `cast($obj=null, $rel_depths=1, $save_cast=true)`
 33 Meng-*override* kelas dari ORM. Bertanggung jawab untuk merepresentasikan kelas dalam
 34 bentuk *hashmap* secara aman atau tidak.
 35 **Input:** Instansi yang akan di-*cast*, konfigurasi relasi, dan lakukan *casting* yang aman
 36 atau tidak.
 37 **Output:** *array*.

38 • **ExamReport**

39 Kelas ini adalah kelas terkait ORM yang merepresentasikan entitas ExamReport. Atribut
 40 yang terdapat pada kelas ini terdiri dari:

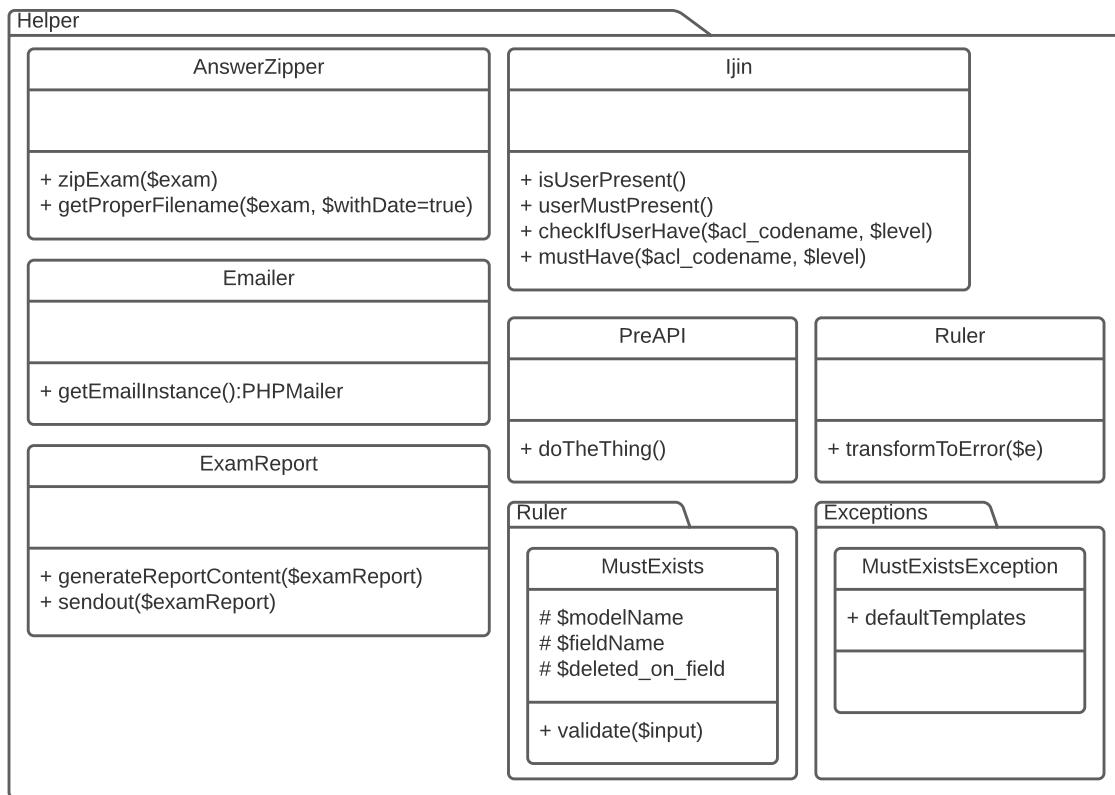
- 1 – `$fieldConf`
- 2 Digunakan untuk mendefinisikan kolom-kolom yang terdapat pada tabel *ExamReport*.
- 3 – `$db` dengan nilai awal "DB".
- 4 Menentukan database yang akan digunakan pada model ini.
- 5 – `$table` dengan nilai awal "ujian_examreport".
- 6 Nama tabel yang akan digunakan pada sistem basis data.

7 Sedangkan fungsi yang terdapat pada kelas ini adalah:

- 8 – `set_sent_on($date)`
9 Mengkonversi bentuk tanggal kolom `sent_on` native ke format basis data.
10 **Input:** Tanggal dalam bentuk *Unix time*
11 **Output:** *string* tanggal terformat.
- 12 – `set_valid_until($date)`
13 Mengkonversi bentuk tanggal kolom `valid_until` native ke format basis data.
14 **Input:** Tanggal dalam bentuk *Unix time*
15 **Output:** *string* tanggal terformat.
- 16 – `set_created_on($date)`
17 Mengkonversi bentuk tanggal kolom `created_on` native ke format basis data.
18 **Input:** Tanggal dalam bentuk *Unix time*
19 **Output:** *string* tanggal terformat.
- 20 – `set_updated_on($date)`
21 Mengkonversi bentuk tanggal kolom `updated_on` native ke format basis data.
22 **Input:** Tanggal dalam bentuk *Unix time*
23 **Output:** *string* tanggal terformat.
- 24 – `save()`
25 Meng-*override* kelas dari ORM. Bertanggung jawab untuk mengisi kolom `created_on`,
26 `updated_on` dan `deleted_on`.
27 **Input:** -
28 **Output:** -
- 29 – `pushValidity($addSeconds, $autoSave)`
30 Menambahkan waktu validitas dari token yang terdapat pada entri ini.
31 **Input:** Jumlah waktu dalam detik, jalankan simpan otomatis.
32 **Output:** -

33 *Namespace Helper*

- 34 Kelas-kelas yang terdapat pada *namespace* ini akan menampung berbagai fungsi pembantu yang
35 dapat digunakan tanpa harus diinstansiasi atau kelas tambahan untuk perkakas validasi. Diagram
36 kelas untuk *namespace* ini dapat diperhatikan pada Gambar 4.46.
- 37 Kelas-kelas yang terdapat pada *namespace* ini tediri dari
- 38 • **AnswerZipper** Kelas ini bertanggung jawab untuk membuat *archive zip* dari ujian tertentu.
39 Fungsi pada kelas ini terdiri dari:



Gambar 4.46: Diagram kelas untuk *namespace Helper*.

1 – `zipExam($exam)`

2 Buat *archive* untuk ujian tertentu.

3 **Input:** Objek dari *Exam*.

4 **Output:** nama berkas zip.

5 – `getProperFilename($exam, $withDate=true)`

6 Bangkitkan nama berkas yang tepat untuk ujian tertentu.

7 **Input:** objek *exam*, opsi untuk menambahkan tanggal.

8 **Output:** *string* nama berkas.

- 9 • **Emailer** Kelas ini digunakan untuk membuat dan mengkonfigurasi objek *PHPMailer*. Fungsi
10 pada kelas ini terdiri dari:

11 – `getEmailInstance()`

12 Buat objek *PHPMailer* dan konfigurasi objek tersebut berdasarkan konfigurasi sistem.

13 **Input:** –

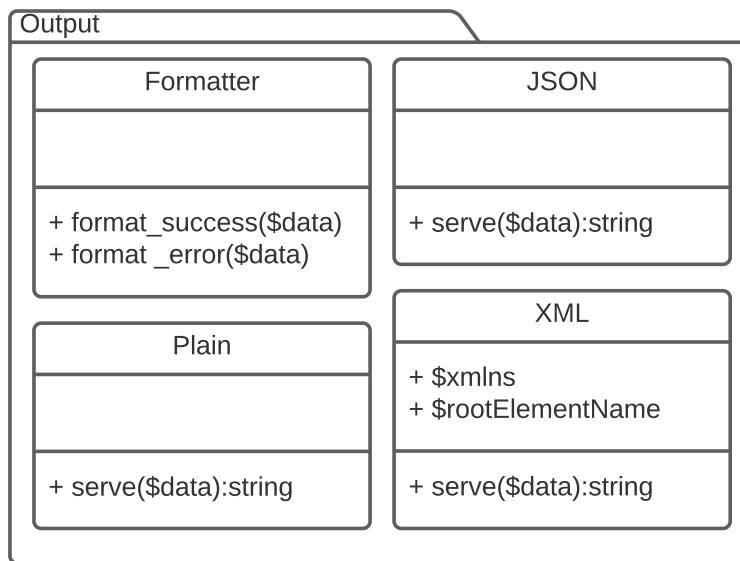
14 **Output:** Objek dari *PHPMailer*.

- 15 • **Ijin** Kelas ini membantu pengecekan perizinan dengan entitas ACL dan ACLItem.

16 – `isUserPresent()`

17 Melakukan pengecekan apakah kode otentikasi tersedia dan valid.

- 1 **Input:** -
2 **Output:** *boolean* kehadiran otentikasi yang valid.
- 3 – `userMustPresent()`
4 Memastikan bahwa token otentikasi ada.
- 5 **Input:** -
6 **Output:** -, namun dapat melempar eksepsi jika token tidak tersedia.
- 7 – `checkIfUserHave($acl_codename, $level)`
8 Melakukan pengecekan apakah token otentikasi memiliki izin untuk melakukan suatu
9 aksi pada *codename* tertentu.
- 10 **Input:** nama kode izin, perizinan yang seharusnya dimiliki
11 **Output:** *boolean* apakah token memiliki permisi tersebut.
- 12 – `mustHave($acl_codename, $level)`
13 Memastikan bahwa token memiliki izin untuk melakukan sesuatu
- 14 **Input:** nama kode izin, perizinan yang seharusnya dimiliki
15 **Output:** -, namun dapat melempar eksepsi jika token tidak memiliki izin.
- 16 • PreAPI Macro untuk API.
- 17 – `doTheThing()`
18 Melakukan transformasi badan permintaan API menjadi JSON. Dilakukan hanya pada
19 HTTP method POST dan PUT.
- 20 **Input:** -
21 **Output:** -
- 22 • Ruler Macro untuk *library* validasi menjadi eksepsi kelas Model/Error.
- 23 – `transformToError($e)`
24 Melakukan transformasi eksepsi dari *library* validasi, menjadi Model/Error
- 25 **Input:** -
26 **Output:** -
- 27 • Ruler\MustExists Kelas peraturan untuk validasi tambahan. Memastikan apakah suatu
28 entri terdapat pada sebuah tabel. Kelas ini memiliki atribut sebagai berikut:
- 29 – `$modelName`: Nama model
30 – `$fieldName`: Nama kolom kunci primer dari model
31 – `$deleted_on_field`: Nama kolom *soft delete* dari model
- 32 Serta fungsi sebagai berikut:
- 33 – `validate($input)`
34 Melakukan validasi bahwa data yang diberikan terdapat pada tabel.
- 35 **Input:** input data
36 **Output:** *boolean* kebenaran dari validasi tersebut.
- 37 • Exceptions\MustExistsException Eksepsi oleh aturan *MustExists*.



Gambar 4.47: Diagram kelas untuk *namespace Output*.

– `$defaultTemplates`: Pesan kesalahan pada saat validasi gagal.

Serta fungsi sebagai berikut:

– `validate($input)`

Melakukan validasi bahwa data yang diberikan terdapat pada tabel.

Input: input data

Output: boolean kebenaran dari validasi tersebut.

7 Namespace Output

Namespace output menyimpan kelas yang membantu untuk formatisasi API. Format yang didukung saat ini berbentuk JSON, XML, dan teks biasa. Rancangan diagram kelas ini dapat dilihat pada Gambar 4.47.

Kelas-kelas yang terdapat pada *namespace* ini terdiri dari

- **Formatter**

Kelas ini bertanggung jawab untuk melakukan formating utama untuk jenis respon sukses dan gagal. Kelas ini memiliki fungsi:

– `format_success($data)`

Melakukan format untuk jenis respon sukses.

Input: input data

Output: array.

– `format_error($data)`

Melakukan format untuk jenis respon gagal.

Input: objek dari `model/Error`

Output: array.

1 • **JSON**

2 Kelas ini bertanggung jawab untuk melakukan performatan data ke bentuk JSON. Kelas ini
3 memiliki fungsi:

4 – **serve(\$data)**

5 Merepresentasikan data dalam bentuk JSON.

6 **Input:** input data

7 **Output:** *string*.

8 • **Plain**

9 Kelas ini bertanggung jawab untuk melakukan performatan data ke bentuk teks biasa. Kelas
10 ini memiliki fungsi:

11 – **serve(\$data)**

12 Merepresentasikan data dalam bentuk teks.

13 **Input:** input data

14 **Output:** *string*.

15 • **XML**

16 Kelas ini bertanggung jawab untuk melakukan performatan data ke bentuk JSON. Kelas ini
17 memiliki atribut:

18 – **\$xmlns**: *Namespace* yang digunakan untuk mendefinisikan XML.

19 – **\$rootElementName**: Nama elemen pangkal yang akan digunakan.

20 Kelas ini memiliki fungsi:

21 – **serve(\$data)**

22 Merepresentasikan data dalam bentuk XML.

23 **Input:** input data

24 **Output:** *string*.

25 **4.2.4 Namespace Service, Cronjob dan View**

26 Diagram kelas untuk beberapa *Namespace* ini dapat dilihat pada Gambar 4.48. Kelas-kelas pada
27 *namespace* ini terdiri dari:

28 • **Service\Batgenerator**

29 Kelas ini bertanggung jawab untuk membuat *script bat* untuk ujian dan migrasi. Atribut
30 yang terdapat pada kelas ini:

31 – **\$zipname**: Nama berkas zip sebelumnya

32 Fungsi yang terdapat pada kelas ini:

33 – **generate(\$ujian)**

34 Bertanggung jawab untuk melakukan pembuatan *script bat* untuk ujian.

35 **Input:** Objek Exam.

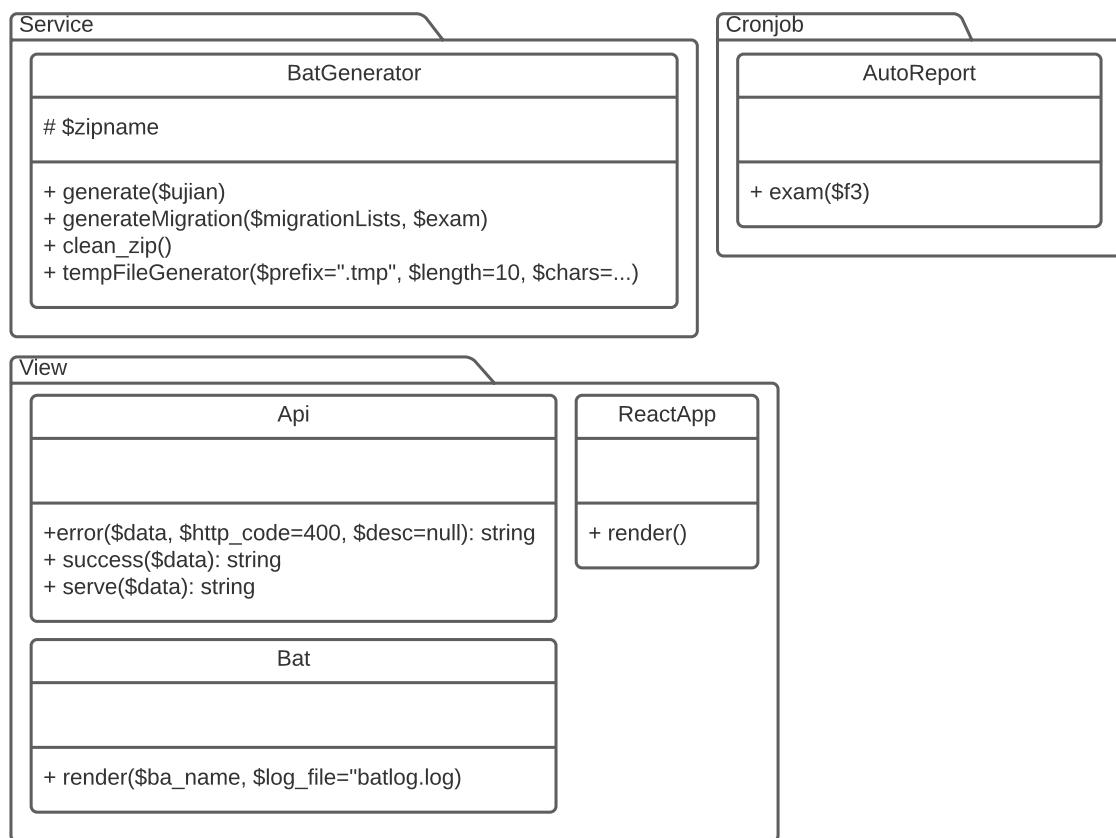
36 **Output:** *string* nama berkas zip.

- 1 – `generateMigration($migrationLists, $exam)`
 2 Bertanggung jawab untuk melakukan pembuatan *script* bat untuk perpindahan peserta.
 3 **Input:** Daftar migrasi, dan objek `Exam`
 4 **Output:** *string* nama berkas zip.
 - 5 – `clean_zip()`
 6 Menghapus berkas zip
 7 **Input:** -
 8 **Output:** -
 - 9 – `tempFileGenerator($prefix=".tmp", $length=10, $chars=...)`
 10 Membangkitkan nama sementara untuk zip.
 11 **Input:** prefixks untuk nama berkas sementara, panjang teks acak, dan daftar karakter
 12 yang ingin digunakan.
 13 **Output:** *string* nama berkas sementara.
- 14 • **Cronjob\Autoreport**
 15 Kelas ini bertanggung jawab untuk melakukan tugas *cron* untuk mengirimkan laporan. Fungsi
 16 yang terdapat pada kelas ini adalah:
- 17 – `exam($f3)`
 18 Bertanggung jawab untuk mengkompilasi ujian yang telah selesai, dan mengirimkan email
 19 ke daftar yang disebutkan pada entri. **Input:** Instansi kelas `Base` dari framework.
 20 **Output:** -
- 21 • **View\Api**
 22 Kelas ini didedikasikan untuk menyediakan API dengan format tertentu yang didukung oleh
 23 kelas yang terdapat pada namespace `Output`. Fungsi yang terdapat pada kelas ini adalah:
- 24 – `error($data, $http_code=400, $desc=null)`
 25 Bertanggung jawab untuk melakukan peformatan *error*.
 26 **Input:** objek `Error`, kode respon HTTP, dan deskripsinya.
 27 **Output:** *string*.
 - 28 – `success($data)`
 29 Bertanggung jawab untuk melakukan peformatan data untuk respon sukses.
 30 **Input:** *array* data.
 31 **Output:** *string*.
 - 32 – `serve($data)`
 33 Menyajikan data ke pengkonsumsi API, sesuai dengan format yang diminta oleh peng-
 34 konsumsi API, seperti JSON, XML, ataupun teks biasa.
 35 **Input:** *string* data.
 36 **Output:** *string*.
- 37 • **View\Bat**
 38 Melakukan pembuatan *script* bat berdasarkan *template* yang telah diberikan, dengan meman-
 39 faatkan *templating engine* yang disediakan oleh *framework*. Fungsi yang terdapat pada kelas
 40 ini adalah:

- 1 – `render($ba_name, $log_file="batlog.log")`
- 2 *Render template yang diberikan, dan kembalikan sebagai string.*
- 3 **Input:** Nama *template*, nama log yang akan digenerasi.
- 4 **Output:** *string*.
- 5 • **View\Reactapp**
- 6 Bertanggung jawab untuk menyajikan aplikasi React untuk tampilan publik.
- 7 – `render()`
- 8 Sajikan hasil *build* dari React.
- 9 **Input:** -
- 10 **Output:** -

11 4.3 Perancangan Sistem CI/CD dan Unit Testing

- 12 Berdasarkan hasil analisis pada bab sebelumnya, penelitian membutuhkan sebuah sistem yang dapat
- 13 melakukan *build* secara otomatis sebelum dapat di-deploy ke server produksi. Sistem *build* tersebut
- 14 akan direncanakan untuk diimplementasi pada sistem CI/CD yang terintegrasi dengan tempat
- 15 penyimpanan repositori kode sumber. Kode sumber tersebut nantinya akan ditrack oleh sistem
- 16 repositori sehingga pada saat kode mengalami perubahan, sistem CI/CD akan secara otomatis
- 17 berjalan. Sistem ini dirancang untuk melakukan hal berikut secara sekuensial:
 - 18 • Melakukan penyegaran dependensi.
 - 19 • Melakukan *build* dengan level produksi.
 - 20 • Menyimpan hasil *build* pada repositori.
 - 21 • Unggah perubahan pada repositori kembali ke layanan penyimpanan.
 - 22 • Picu sistem *deploy* untuk melakukan *deployment*.
- 23 Hal tersebut dilakukan untuk memastikan hasil *build* dapat dikirimkan pada server untuk
- 24 disediakan ke pengguna. Sistem *deployment* yang digunakan diimplementasi dengan bantuan script
- 25 yang dijalankan via SSH. *Script* tersebut dirancang akan melakukan pengunduhan perubahan kode,
- 26 lalu melakukan penyegaran dependensi versi server, dan mengabari kembali sistem CI/CD bahwa
- 27 *deployment* telah berhasil dilakukan.
- 28 Selain melakukan *build*, penelitian juga akan menambahkan sistem tes unit otomatis. Sistem tes
- 29 ini direncakan untuk melakukan pengecekan terhadap komponen-komponen krusial pada aplikasi.
- 30 Karena konsekuensi yang ditanggung aplikasi cukup besar, maka sistem tes ditambahkan untuk
- 31 mencegah kesalahan yang sangat fatal. Sistem tes unit ini direncakan berjalan bersama sistem *build*.
- 32 Sehingga kode akan di tes setiap kali kode baru diunggah ke tempat penyimpanan.



Gambar 4.48: Diagram kelas untuk *namespace Service, Cronjob dan View*.

1

BAB 5

2

IMPLEMENTASI DAN PENGUJIAN

3

5.1 Implementasi

4

5.1.1 Lingkungan Implementasi Perangkat Lunak

5

Aplikasi pendukung sistem ujian Oxam akan diimplementasi dengan spesifikasi berikut:

6

- *Runtime Environment*: Docker versi 19.03.8, `build afacb8b7f0`, dengan konfigurasi kontainer:

7

- backend

8

`php:7.3.8-apache`

9

Berkas `Dockerfile` dapat dilihat pada lampiran

10

- Bahasa Pemrograman: PHP v7.3, JavaScript (Babel Preset React).

11

- Basis data: MySQL versi 5.7

12

- CI/CD: GitLab CI

13

- *Library* lain yang digunakan:

14

- Backend: `chez14/f3-ilgar`

15

Library ini digunakan untuk membuat database dengan merepresentasikan setiap update menjadi sebuah paket migrasi. Pada aplikasi ini, `chez14/f3-ilgar` digunakan untuk membantu mengelola basis data.

16

- Backend: `lcobucci/jwt`

17

Library ini bertugas untuk membuat token dalam bentuk JWT, melakukan validasi, dan menekstrak data dari token tersebut. `lcobucci/jwt` digunakan pada saat otentikasi untuk API.

18

- Backend: `psx/openssl`

19

`psx/openssl` adalah *library* yang membungkus fungsi OpenSSL yang tersedia pada PHP. *Library* ini akan membuat kode menjadi lebih bersih karena *overhead* harus ditulis pada saat menginisialisasi OpenSSL menjadi otomatis pada saat kelas dari *library* ini diinstansiasi. *Library* ini bertugas untuk membantu *library* `lcobucci/jwt` menghasilkan kunci asimetrik dengan algoritma RSA atau ECDSA.

20

- Backend: `respect/validation`

21

Library ini bertugas untuk melakukan validasi yang cukup kompleks. *Library* ini dapat

diturunkan menjadi fungsi validasi yang lain. Pada Aplikasi ini, *library* ini digunakan untuk membantu validasi data yang diberikan dari klien lewat subsistem Frontend.

– Backend: **adldap2/adldap2**

Library adldap2/adldap2 adalah *wrapper* dari fungsi LDAP yang PHP sediakan. *Libray* ini nantinya akan bertugas membantu Backend melakukan *query* pada LDAP milik Lab Komputasi untuk *resolve username* login peserta menjadi nama lengkap.

– Backend: **xfra35/f3-cron**

Library ini bertugas untuk melakukan penjadwalan *cron* yang terdapat pada sistem. Sistem operasi cukup melakukan pemanggilan pada *endpoint* yang disediakan oleh *library* ini, lalu *library* ini akan menjadwalkan cronjob yang ada. Pada aplikasi ini, *library* ini digunakan untuk mengatur *cronjob* pengiriman laporan berkas jawaban ke dosen koordinator.

– Backend: **phpmailer/phpmailer**

phpmailer/phpmailer bertugas untuk membantu pengiriman email melalui protokol SMTP. Pada aplikasi Oxam, *phpmailer/phpmailer* bertugas untuk membantu mengirimkan email laporan berkas jawaban ke dosen koordinator.

– Backend: **monolog/monolog**

Library ini bertugas untuk membuat kelas yang digunakan untuk mencatat log yang terjadi pada sistem. Pada aplikasi ini, *monolog/monolog* digunakan untuk membuat log tentang ujian, seperti peserta mengunggah jawaban, dan seterusnya.

– Frontend: **@fortawesome/***

Library ini berisi ikon-ikon yang digunakan untuk merepresentasikan fungsi tombol dalam bentuk gambar. Pada aplikasi ini, *@fortawesome/** digunakan pada banyak halaman untuk mempermudah pendesainan beberapa tombol menjadi lebih ringkas.

– Frontend: **axios**

Library ini digunakan untuk melakukan permintaan *ajax* dengan penanganan *event* berbentuk *Promise*. Pada aplikasi ini, *axios* digunakan untuk melakukan komunikasi antara subsistem Frontend dan Backend. *Library* ini kemudian diturunkan dengan menambahkan beberapa *event* khusus seperti menambahkan token otentifikasi pada saat sistem frontend mendeteksi token pada *cookie* peramban pengguna.

– Frontend: **bootstrap**

Library ini digunakan untuk membuat tampilan web menjadi lebih seragam, lebih mudah diprediksi, dan konsisten antar perangkat. Pada aplikasi ini, *bootstrap* digunakan pada seluruh halaman aplikasi untuk membuat *look-and-feel* yang seragam, responsif, dan rapi.

– Frontend: **date-fns**

Library ini bertugas untuk melakukan manipulasi tanggal, *parsing* dan *formatting*. Pada aplikasi ini, *date-fns* digunakan untuk *parsing* tanggal.

– Frontend: **js-file-download**

js-file-download berfungsi untuk memicu *file picker* untuk menyimpan berkas.

js-file-download akan menerima sebuah *blob* binary dari berkas, lalu meminta peramban untuk menampilkan pemilih berkas. Pada aplikasi ini, *js-file-download* digunakan

- 1 untuk membantu peserta, dosen koordinator, dan admin mengunduh berkas jawaban,
2 dan *script*.
- 3 – Frontend: **mobx**
4 *Library* ini adalah manajemen *state* yang digunakan untuk berbagai *state* antar komponen
5 pada satu aplikasi. Aplikasi Oxam menggunakan **mobx** untuk menyimpan *state* entitas
6 yang akan digunakan antar halaman.
- 7 – Frontend: **mobx-react**
8 *Library* **mobx-react** adalah *binding* yang mengkoordinasi komponen antar React dengan
9 *state management* **mobx**. Aplikasi Oxam menggunakan *library* ini untuk membantu
10 mengkoordinasikan komponen-komponen yang bergantung pada *state* yang disimpan
11 oleh **mobx**.
- 12 – Frontend: **moment**
13 *Library* ini digunakan untuk memanipulasi tanggal dan melakukan *formatting*. Aplikasi
14 ini menggunakan *library* ini karena *library* **react-timekeeper** membutuhkan **moment**.
- 15 – Frontend: **opensans-npm-webfont**
16 *Package* ini menyimpan berkas font yang digunakan pada web. Font OpenSans ini
17 digunakan pada aplikasi untuk memperjelas bentuk tulisan yang akan ditampilkan
18 dilayar peserta dan proyektor. OpenSans memiliki beberapa jenis ketebalan yang cocok
19 digunakan pada konteks tertentu.
- 20 – Frontend: **prop-types**
21 **prop-types** adalah *library* yang digunakan oleh React untuk mevalidasi data properti
22 yang diberikan oleh *parent* komponen. Aplikasi ini menggunakan *library* **prop-types**
23 untuk membuat beberapa komponen yang akan digunakan ulang pada beberapa halaman.
- 24 – Frontend: **query-string**
25 **query-string** digunakan untuk *parsing* bagian *search* pada url dari peramban. *Library*
26 ini akan memecah *string* tersebut lalu mengubahnya menjadi kelas yang dapat digunakan
27 oleh pengembang. Aplikasi ini menggunakan *library* ini untuk melakukan pengecekan
28 *return-path* pada beberapa halaman, terutama halaman *editor* entitas.
- 29 – Frontend: **react-countdown**
30 **react-countdown** digunakan untuk menampilkan jam berbentuk *countdown*. Aplikasi
31 Oxam menggunakan *library* ini untuk menampilkan waktu yang tersisa untuk ujian.
- 32 – Frontend: **react-dates**
33 *Library* ini menyediakan komponen formulir berbentuk pemilih tanggal dalam format
34 kalender. Aplikasi Oxam menggunakan *library* ini untuk membuat formulir pemilih
35 tanggal pada bidang bertipe tanggal.
- 36 – Frontend: **react-dropzone**
37 **react-dropzone** menyediakan *binding* yang dapat digunakan untuk menangani *drag-drop*
38 berkas pada suatu elemen HTML. Pada aplikasi ini, *library* **react-dropzone** digunakan
39 untuk menangani *event* peserta mengunggah berkas dengan melakukan *drag-drop* pada
40 slot jawaban.

1 – Frontend: **react-if**

2 *react-if* menyediakan komponen percabangan, *If-else*. Penggunaan komponen tersebut
3 akan meningkatkan kejelasan kode karena bentuk percabangan yang diberikan menjadi
4 lebih jelas. Aplikasi ini menggunakan **react-if** pada berbagai halaman untuk mengubah
5 beberapa komponen pada kasus tertentu.

6 – Frontend: **react-router-dom**

7 *Library* ini digunakan untuk memetakan url tertentu pada frontend ke komponen tertentu.
8 Aplikasi menggunakan *library* ini untuk memetakan alamat untuk login, lembar jawab
9 ujian, halaman admin dan dosen, serta sebagainya ke komponennya masing-masing.

10 – Frontend: **react-stepper-horizontal**

11 *Library* ini adalah komponen UI yang digunakan untuk memvisualisasikan jumlah langkah
12 yang ada. Aplikasi menggunakan komponen ini untuk menampilkan jumlah langkah
13 yang ada untuk membuat suatu ujian.

14 – Frontend: **react-timekeeper**

15 *Library* ini digunakan untuk menampilkan pemilih jam. Komponen UI ini akan
16 menampilkan jam dan pengguna dapat memilih angka yang terdapat pada tampilan ter-
17 sebut untuk kemudian dapat digunakan sebagai *input* jam. Aplikasi ini menggunakan
18 **react-timekeeper** untuk membantu mengisi bidang formulir tertentu.

19 – Frontend: **reactstrap**

20 *reactstrap* adalah *binding* javascript untuk **bootstrap**. *Library* ini akan menggantikan
21 javascript bawaan dari **bootstrap** yang menggunakan jQuery, menjadi sepenuhnya meng-
22 gunakan *library* dari React. Aplikasi ini menggunakan *library* ini untuk menhidupkan
23 fungsionalitas beberapa komponen UI yang disediakan oleh **bootstrap**.

24 – Frontend: **sass**

25 *Library sass* digunakan untuk melakukan *build* pada kode SASS menjadi CSS. Aplikasi ini
26 menggunakan **sass** untuk memperkecil hasil CSS yang dibuat, serta membuat beberapa
27 penggayaan khusus untuk komponen tertentu.

28 – Frontend: **suneditor-react**

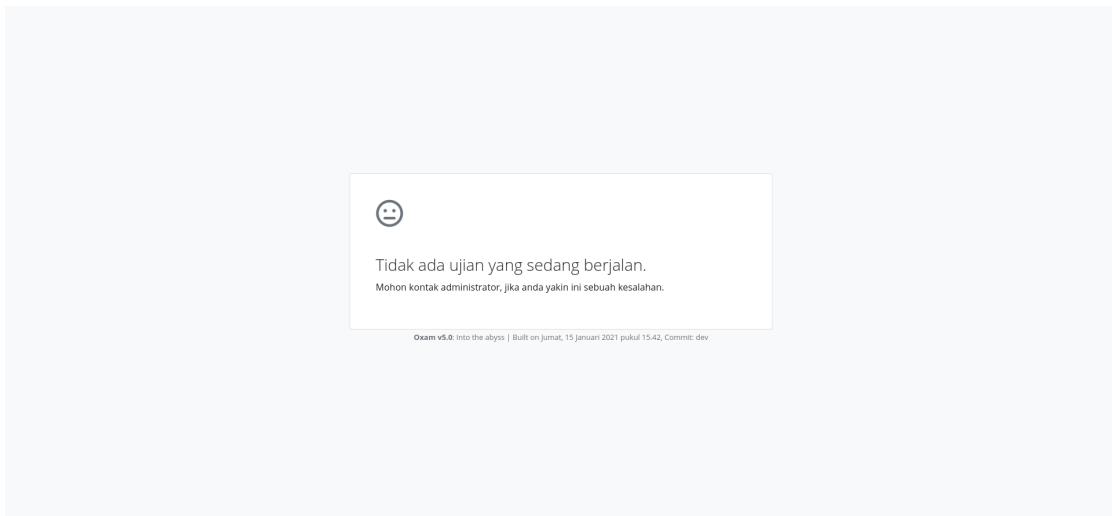
29 *Library* ini menyediakan *binding* untuk React terhadap *library suneditor*. **suneditor**
30 adalah WYSIWYG (*What you see is what you get*) HTML Editor. Aplikasi ini meng-
31 gunakan *library* ini untuk membuat badan notifikasi yang nantinya akan disebar untuk
32 setiap peserta.

33 **5.1.2 Hasil Implementasi**

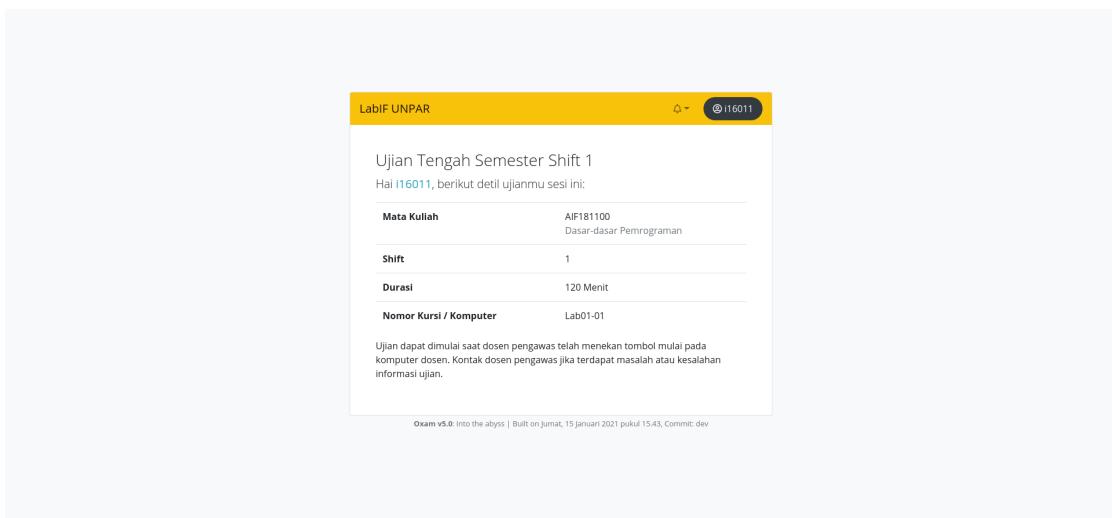
34 Pada bagian ini hasil implementasi sistem akan dijelaskan dengan bantuan tampilan antarmuka
35 yang telah diimplementasi dalam Reactjs.

36 **5.1.3 Halaman untuk Peserta**

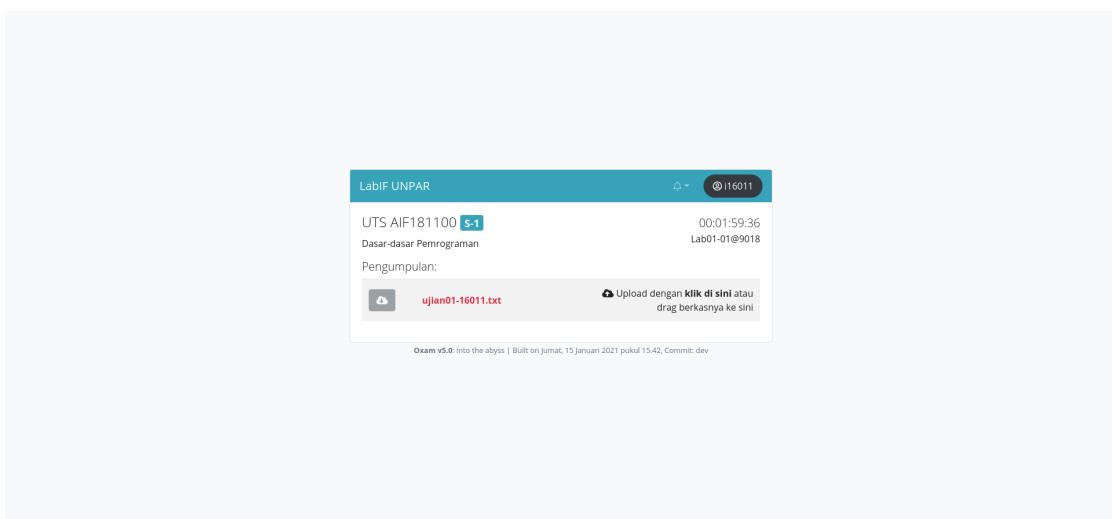
37 Tangkapan layar untuk halaman peserta dapat dilihat pada Gambar 5.1, 5.2, 5.3, peringatan
38 kesalahan nama berkas pada Gambar 5.4, serta fitur notifikasi pada Gambar 5.5.



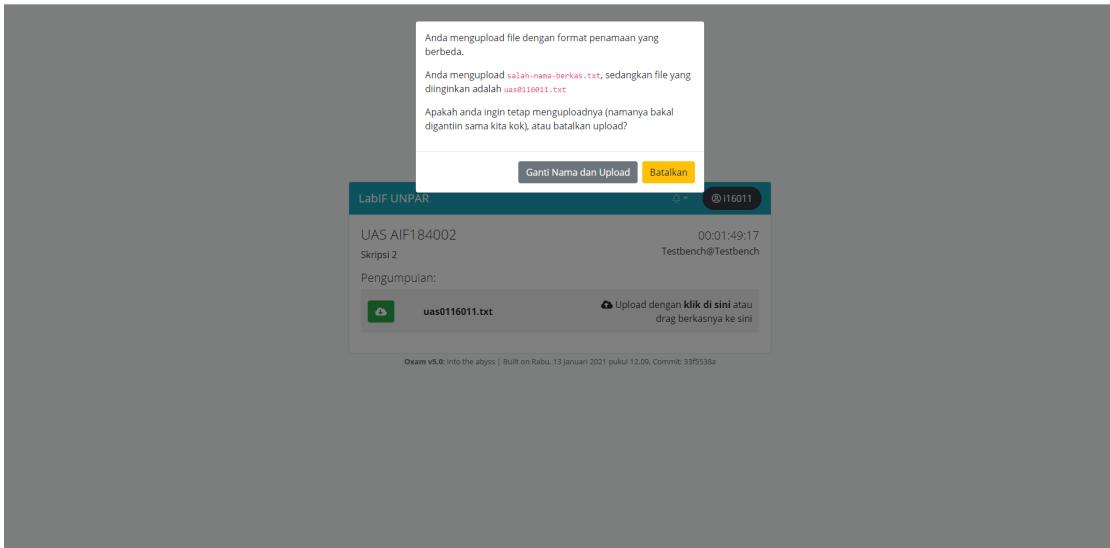
Gambar 5.1: Tangkapan layar dari halaman ujian, dengan ujian yang tidak aktif.



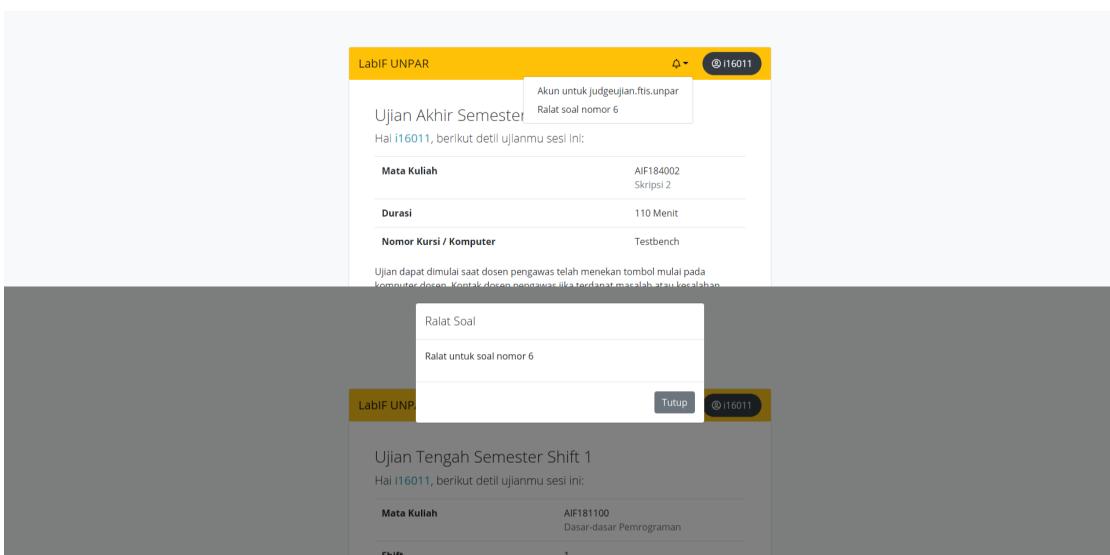
Gambar 5.2: Tangkapan layar dari halaman ujian, dengan ujian yang akan aktif.



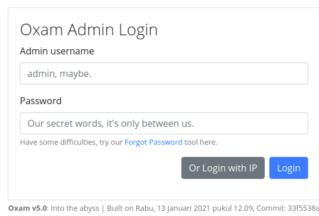
Gambar 5.3: Tangkapan layar dari halaman ujian, dengan ujian yang sedang aktif.



Gambar 5.4: Tangkapan layar dari halaman ujian untuk peringatan nama berkas yang tidak sesuai.



Gambar 5.5: Tangkapan layar dari halaman ujian, bagian notifikasi.



Gambar 5.6: Tangkapan layar halaman otentikasi

1 Pada tangkapan layar halaman ujian yang tidak aktif (Gambar 5.1), peserta akan diberikan
2 sebuah pesan yang memberitahukan bahwa tidak ada ujian yang aktif pada komputer tersebut.
3 Selain itu, terdapat sebuah pesan status penting yang menandakan versi Oxam yang sedang aktif,
4 dan jam terakhir *build* pada subsistem frontend dibuat.

5 Tangkapan layar halaman ujian yang akan aktif pada Gambar 5.2 menampilkan informasi
6 penting tentang ujian yang akan aktif. Tangakapan layar halaman ujian berikutnya adalah pada
7 saat ujian sedang berjalan, dan lembar jawaban sedang dibuka. Tangkapan layar tersebut dapat
8 dilihat pada Gambar 5.3.

9 Jika peserta mengunggah berkas jawaban dengan nama berkas yang salah, peserta akan diberikan
10 pesan peringatan untuk mengkonfirmasi perubahan nama berkas secara otomatis oleh sistem,
11 tampilan tersebut dapat dilihat pada Gambar 5.4.

12 Tangkapan berikutnya adalah bagian bagian notifikasi. Tampilan tersebut dapat dilihat pada
13 Gambar 5.5.

14 **5.1.4 Halaman untuk Tim Admin**

15 Halaman untuk Tim Admin memiliki beberapa bagian besar yang akan dipisah berdasarkan fungsi
16 utamanya. Kelompok pertama yang akan dibahas adalah bagian otentikasi. Kemudian pembahasan
17 akan maju ke bagian ujian, lalu yang terakhir editor entitas.

18 **Halaman Otentikasi**

19 Implementasi halaman otentikasi dapat dilihat pada Gambar 5.6.

20 **Halaman untuk Ujian**

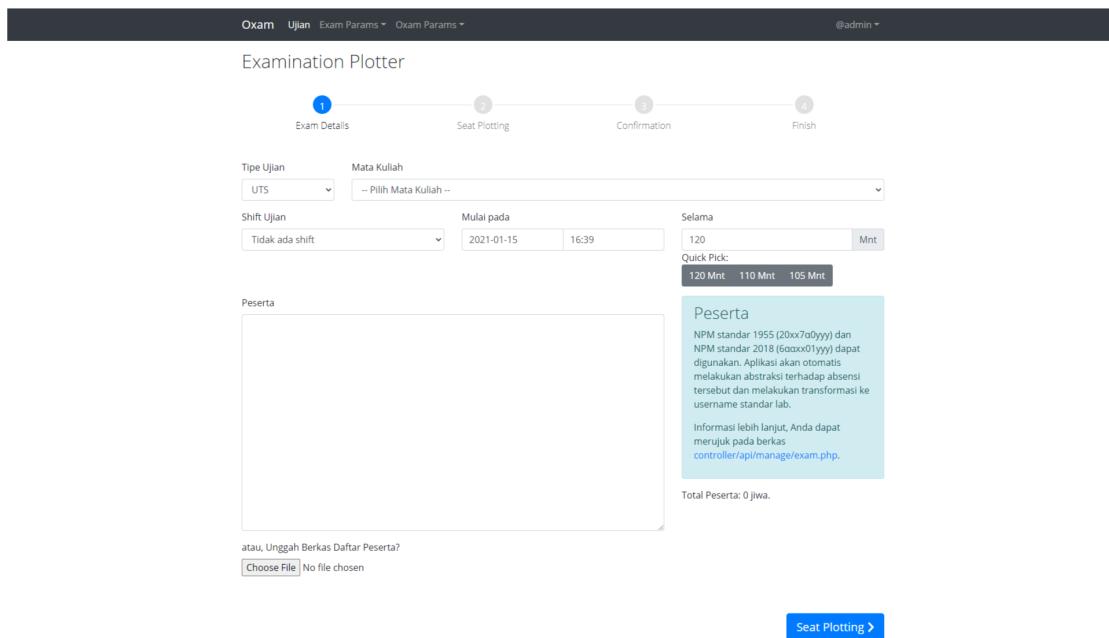
21 Tangkapan layar untuk ujian pertama adalah halaman daftar ujian, dapat dilihat pada gambar 5.7.

Lecture Info	Shift	Duration (Start-End)	
Skripsi 2 (AIF184002) 20202 UAS		2021-01-13 13:49:49 - 2021-01-13 13:57:38	Durasi: 110 Menit Dimulai pada: 1970-01-01 07:32:50
Fisika Komputasi (PHY181024) 20202 UTS		2021-01-13 13:00:27 - 2021-01-13 13:18:34	Durasi: 120 Menit Dimulai pada: 2021-01-13 13:03:35
Dasar-dasar Pemrograman (AIF181100) 20202 UTS	1	2021-01-08 14:58:22 - 2021-01-08 17:06:13	Durasi: 120 Menit Dimulai pada: 2021-01-08 15:06:13
Manajemen Informatika dan Basis Data (AIF182302) 20192 UTS	2	2020-03-16 13:00:00 -	Durasi: 110 Menit Dimulai pada: (belum dimulai)
Manajemen Informatika dan Basis Data (AIF182302) 20192 UTS	1	2020-03-16 11:00:00 -	Durasi: 110 Menit Dimulai pada: (belum dimulai)
Kapita Selekta Fisika Medis (PHY183304) 20192 UTS		2020-03-12 14:00:00 - 2020-03-12 15:56:19	Durasi: 110 Menit Dimulai pada: 2020-03-12 15:52:50
Penulisan Ilmiah (AIF183002) 20192 UTS		2020-03-12 13:20:00 - 2020-03-12 16:27:00	Durasi: 120 Menit Dimulai pada: 2020-03-12 14:09:14
Sertifikasi Administrasi Jaringan Komputer 2 (AIF183236)		2020-03-12 08:00:00 - 2020-03-12 09:50:26	Durasi: 110 Menit Dimulai pada: 2020-03-12 09:50:26

Gambar 5.7: Tangkapan layar daftar ujian untuk admin.

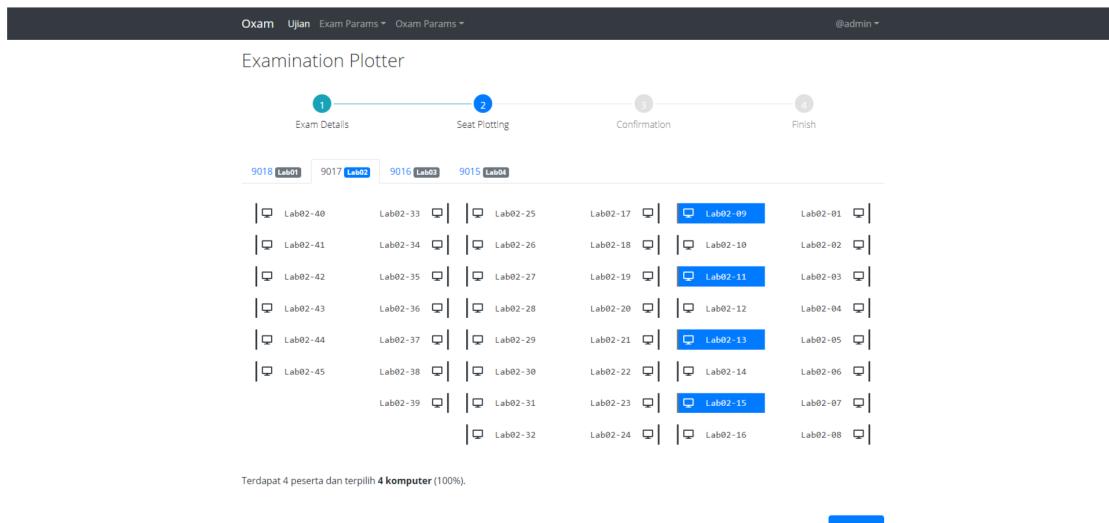
Lecture Info	Shift	Duration (Start-End)	
Skripsi 2 (AIF184002) 20202 UAS		2021-01-13 13:49:49 - 2021-01-13 13:57:38	Durasi: 110 Menit Dimulai pada: 1970-01-01 07:32:50
Fisika Komputasi (PHY181024) 20202 UTS		2021-01-13 13:00:27 - 2021-01-13 13:18:34	Durasi: 120 Menit Dimulai pada: 2021-01-13 13:03:35
Dasar-dasar Pemrograman (AIF181100) 20202 UTS	1	2021-01-08 14:58:22 - 2021-01-08 17:06:13	Durasi: 120 Menit Dimulai pada: 2021-01-08 15:06:13
Manajemen Informatika dan Basis Data (AIF182302) 20192 UTS	2	2020-03-16 13:00:00 -	Durasi: 110 Menit Dimulai pada: (belum dimulai)
Manajemen Informatika dan Basis Data (AIF182302) 20192 UTS	1	2020-03-16 11:00:00 -	Durasi: 110 Menit Dimulai pada: (belum dimulai)
Kapita Selekta Fisika Medis (PHY183304) 20192 UTS		2020-03-12 14:00:00 - 2020-03-12 15:56:19	Durasi: 110 Menit Dimulai pada: 2020-03-12 15:52:50
Penulisan Ilmiah (AIF183002) 20192 UTS		2020-03-12 13:20:00 - 2020-03-12 16:27:00	Durasi: 120 Menit Dimulai pada: 2020-03-12 14:09:14
Sertifikasi Administrasi Jaringan Komputer 2 (AIF183236)		2020-03-12 08:00:00 - 2020-03-12 09:50:26	Durasi: 110 Menit Dimulai pada: 2020-03-12 09:50:26

Gambar 5.8: Tangkapan layar konfirmasi penghapusan ujian.

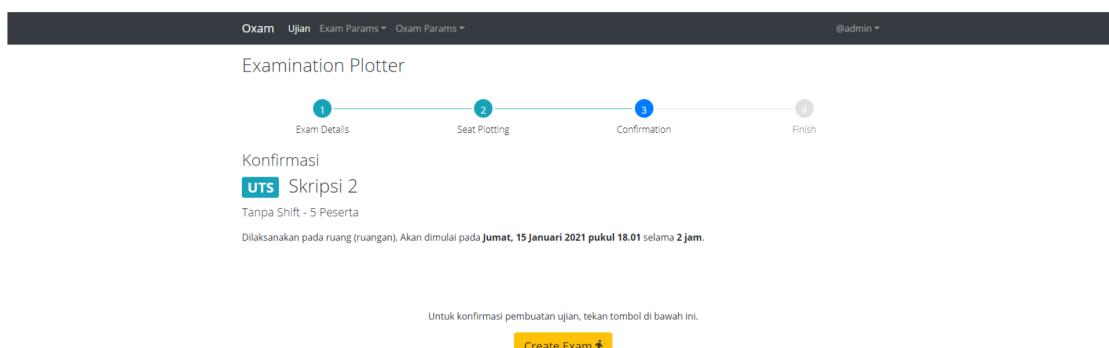


Gambar 5.9: Tangkapan layar untuk membuat ujian, langkah pertama.

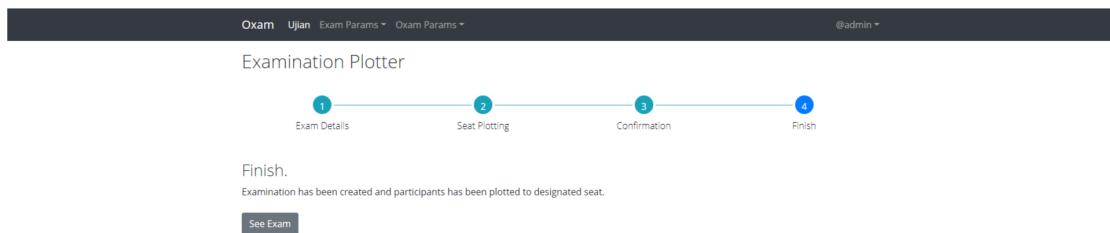
- 1 Tampilan konfirmasi penghapusan ujian dapat dilihat pada Gambar 5.8. Tampilan tersebut
- 2 terdiri dari informasi singkat tentang ujian dan tombol konfirmasi penghapusan dan pembatalan.
- 3 Tampilan berikutnya adalah tampilan untuk membuat ujian baru, tampilan tersebut dapat
- 4 dilihat pada Gambar 5.9, langkah kedua pada Gambar 5.10, ketiga pada Gambar 5.11, dan langkah
- 5 terakhir pada Gambar 5.12. Pada langkah terakhir, tangkapan layar hanya akan menunjukkan
- 6 konfirmasi ujian telah berhasil dibuat dan pengguna diberikan sebuah tombol navigasi untuk melihat
- 7 ujian.
- 8 Halaman detil ujian akan memiliki beberapa tombol peralatan yang dapat dilihat pada Gambar
- 9 5.13.
- 10 Tampilan absensi untuk peserta dapat dilihat pada Gambar 5.14 dan 5.15. Tangkapan layar
- 11 untuk absensi yang ditempel ke pintu dapat dilihat pada Gambar 5.14. Sedangkan absen untuk
- 12 tanda tangan peserta ujian dapat dilihat pada Gambar 5.15.
- 13 Tampilan berikutnya adalah tampilan panel admin mini, dapat dilihat pada Gambar 5.16.
- 14 Fitur untuk memindahkan tempat duduk peserta dapat dilihat pada gambar 5.17, 5.18, 5.19,
- 15 dan 5.20.
- 16 Tampilan layar proyektor untuk peran admin dapat dilihat pada Gambar 5.21.
- 17 Kemudian, tangkapan layar berikutnya adalah fitur untuk slot jawaban. Pada Gambar 5.22
- 18 bagian A, modal akan menanyakan format jawaban yang ingin ditambahkan dan tombol aksinya.
- 19 Sedangkan bagian B menunjukkan modal konfirmasi penghapusan slot jawaban tersebut serta
- 20 tombol aksinya.
- 21 Hasil implementasi untuk fitur notifikasi dapat dilihat pada Gambar 5.23 dan 5.24. Sedangkan
- 22 pada Gambar 5.24, modal konfirmasi penghapusan akan menampilkan informasi badan notifikasi
- 23 dan tombol aksi penghapusan atau pembatalan.



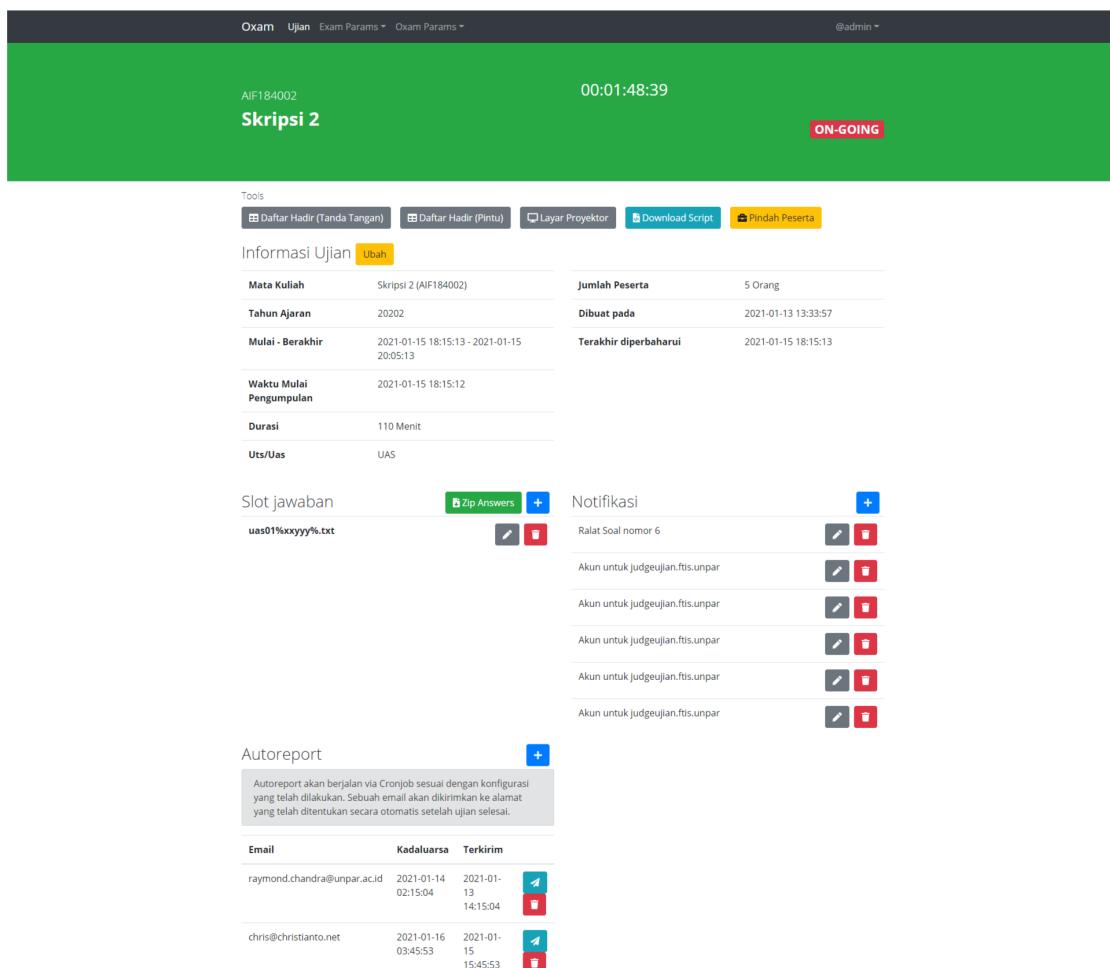
Gambar 5.10: Tangkapan layar untuk membuat ujian, langkah kedua.



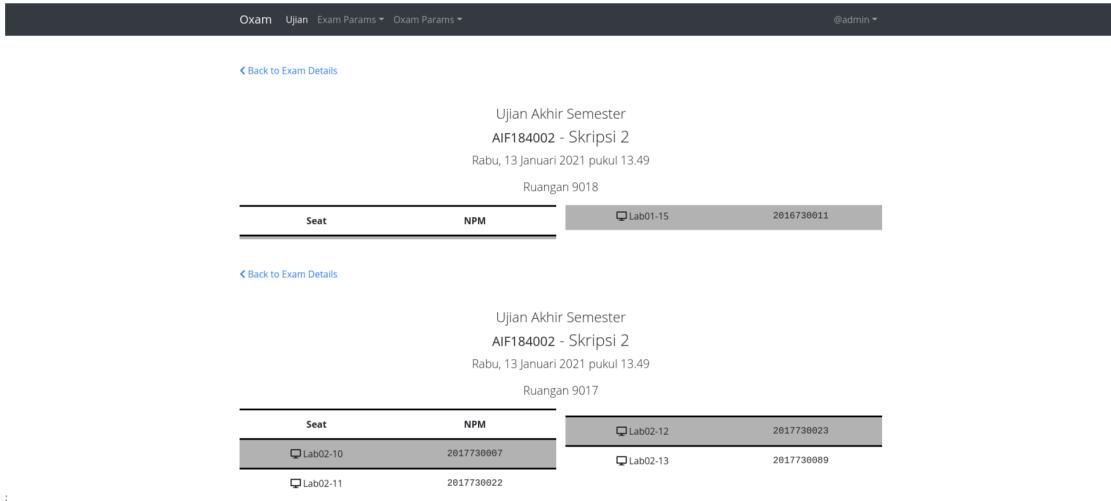
Gambar 5.11: Tangkapan layar untuk membuat ujian, langkah ketiga.



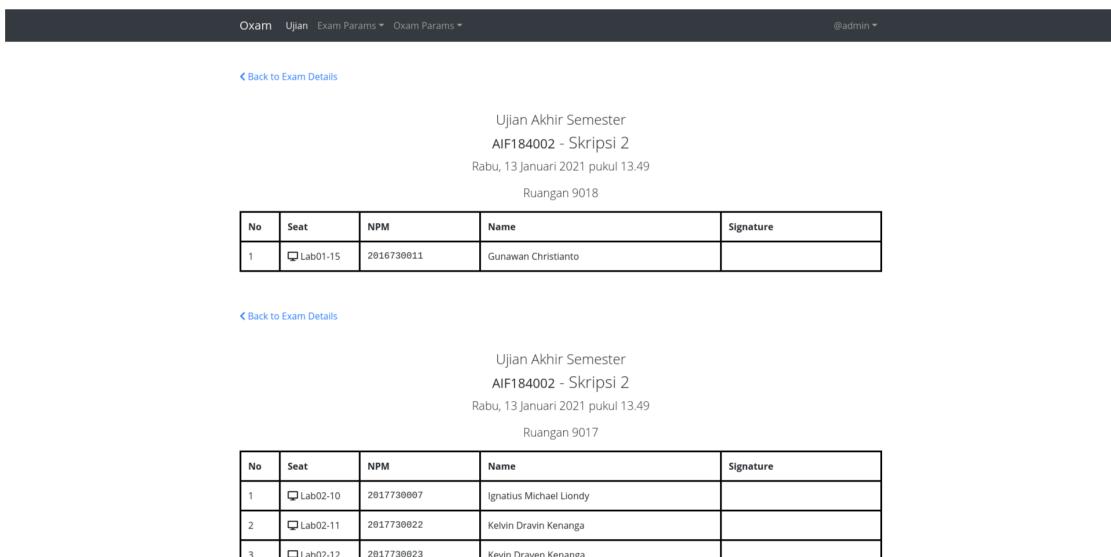
Gambar 5.12: Tangkapan layar untuk membuat ujian, langkah keempat.



Gambar 5.13: Tangkapan layar untuk detil ujian



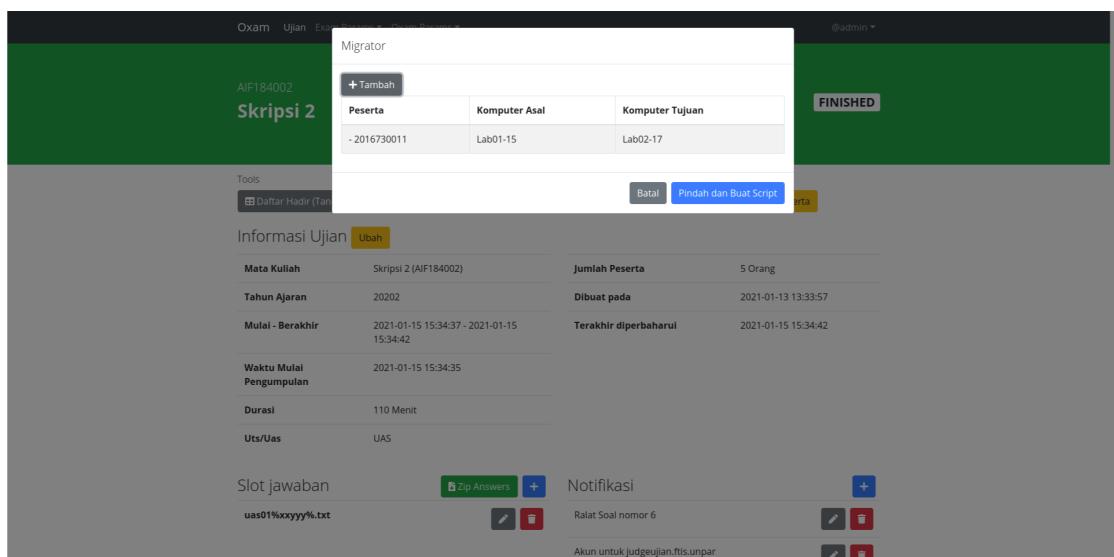
Gambar 5.14: Tangkapan layar absensi untuk ditempel pada pintu.



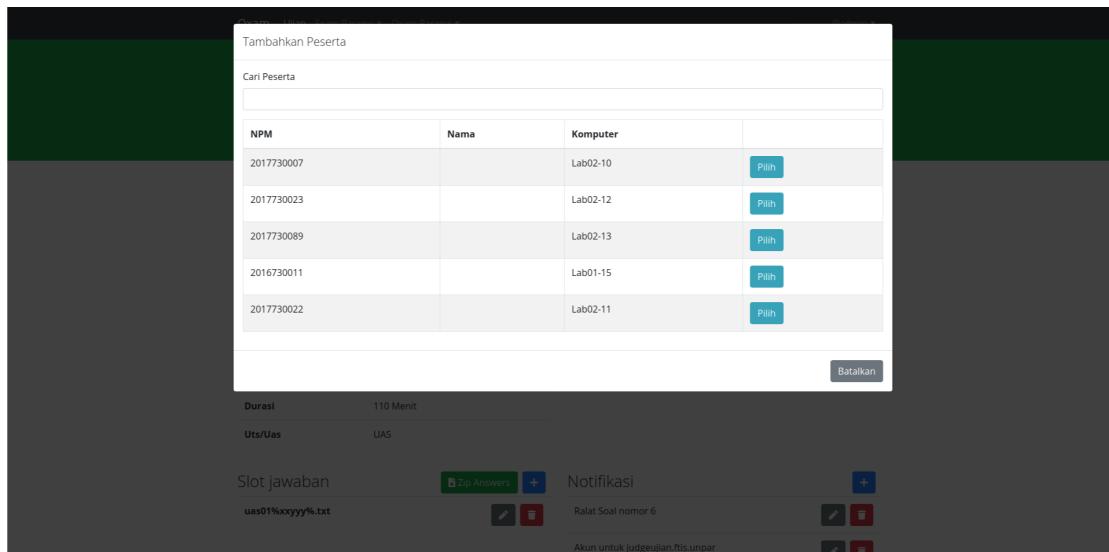
Gambar 5.15: Tangkapan layar absensi untuk tanda tangan peserta.



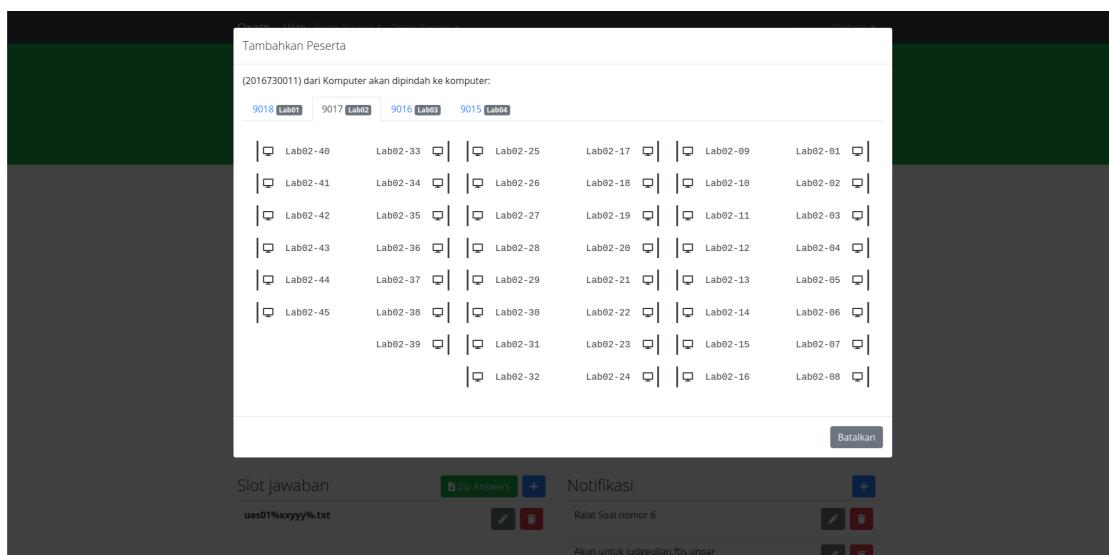
Gambar 5.16: Tangkapan layar panel perangkat bergerak.



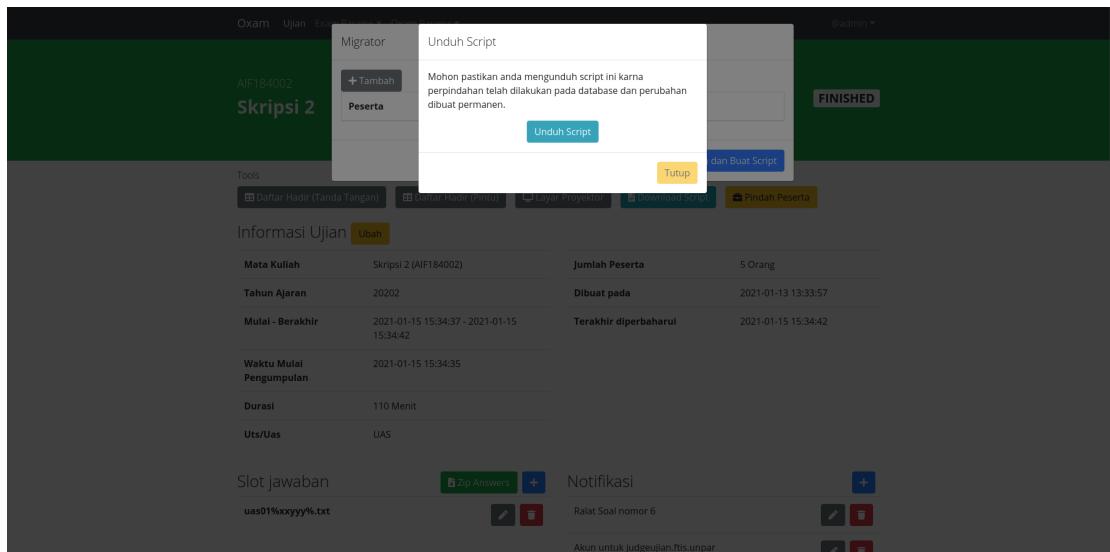
Gambar 5.17: Tangkapan layar untuk pemindah peserta.



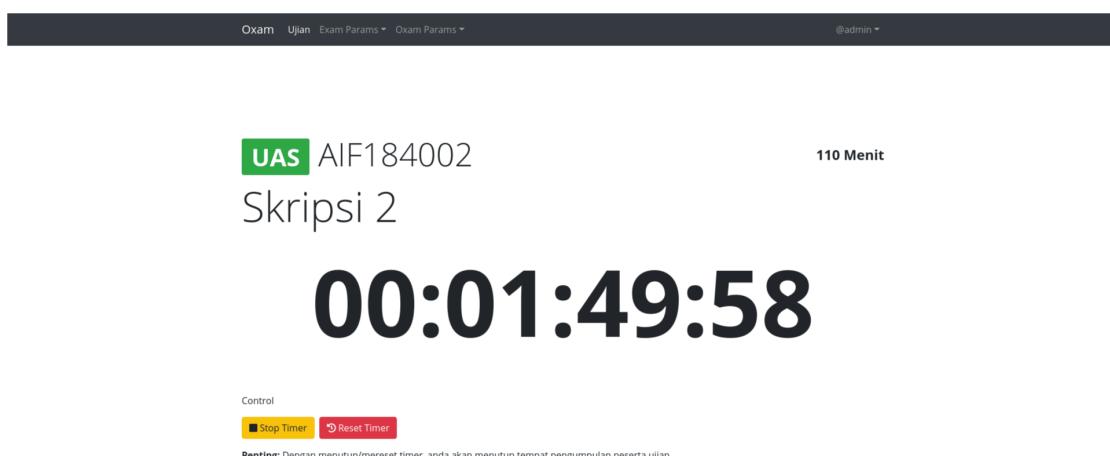
Gambar 5.18: Tangkapan layar untuk pemindah peserta, langkah memilih peserta.



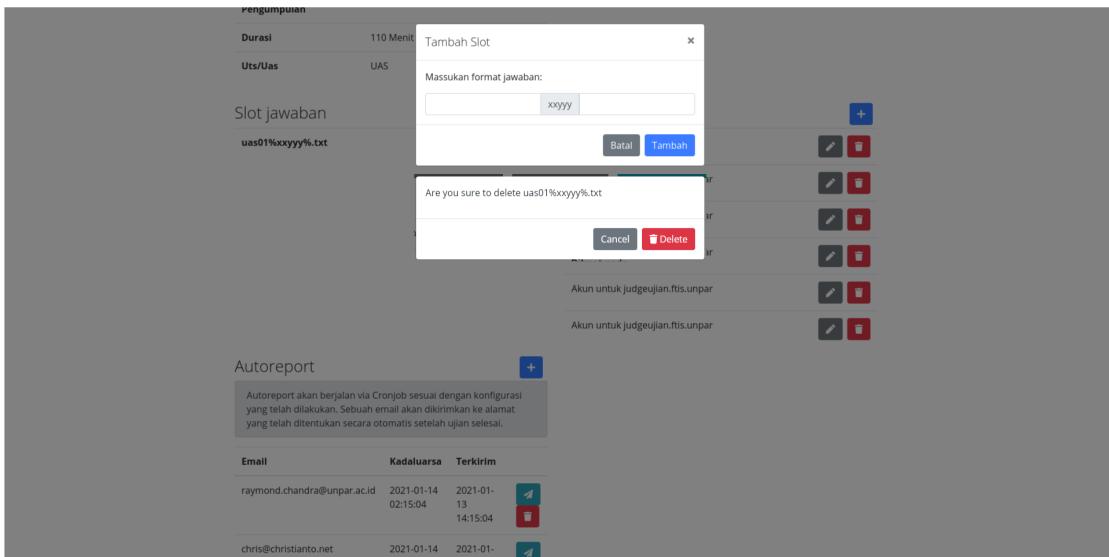
Gambar 5.19: Tangkapan layar untuk pemindah peserta, langkah memilih komputer.



Gambar 5.20: Tangkapan layar untuk pemindah peserta, langkah mengunduh *script*.

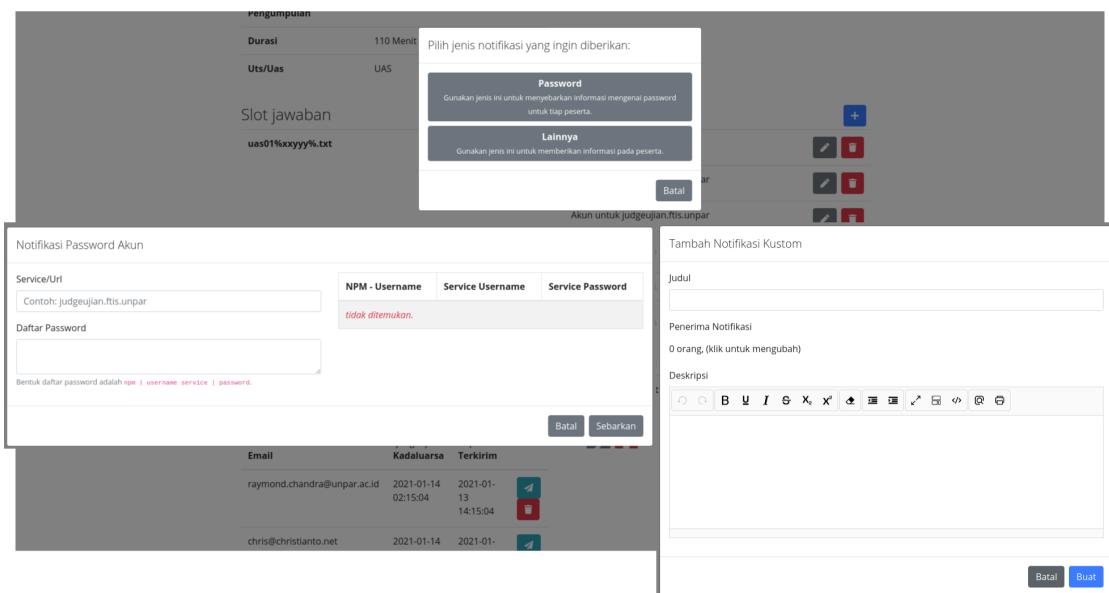


Gambar 5.21: Tangkapan layar untuk layar projektor Admin.



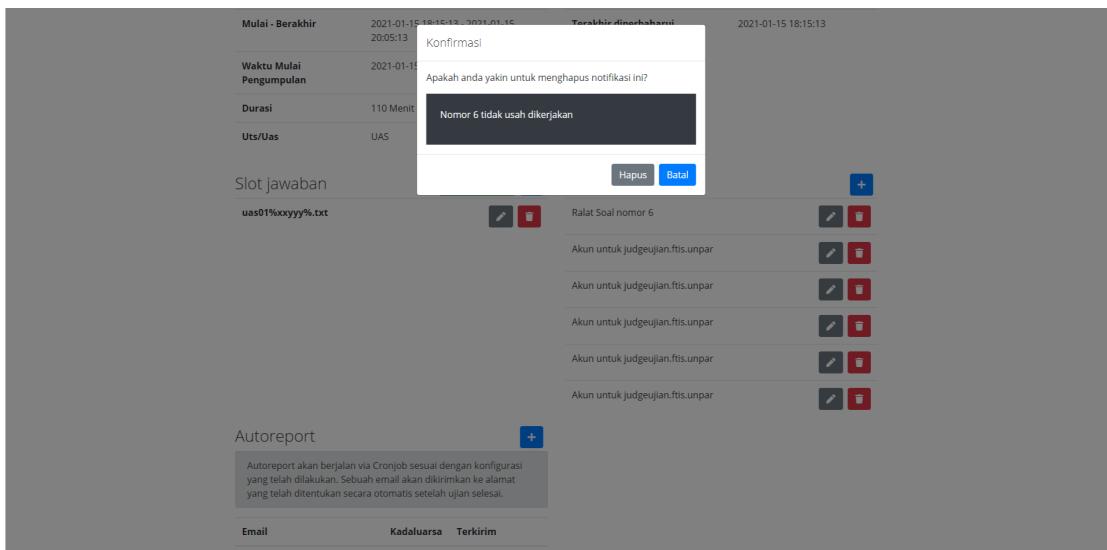
Gambar 5.22: Tangkapan layar untuk slot jawaban.

(A) untuk menambahkan slot jawaban, (B) untuk menghapus slot jawaban.



Gambar 5.23: Tangkapan layar untuk fitur notifikasi.

(A) modal pemilihan jenis. (B) modal notifikasi kata sandi. (C) modal untuk notifikasi lainnya.



Gambar 5.24: Tangkapan layar untuk konfirmasi penghapusan notifikasi.

Manage\Lecture::index()			
ID	Name	Lecture Code	updated_on
1	Dasar-dasar Pemrograman	AIF181100	2020-03-06 18:54:24
2	Desain dan Analisis Algoritma	AIF182106	2020-03-06 15:49:14
3	Sertifikasi Administrasi Jaringan Komputer 2	AIF183236	2020-03-06 15:49:14
4	Sertifikasi Administrasi Jaringan Komputer 4	AIF184222	2020-03-06 15:49:14
5	Kapita Selektiva Fisika Medis	PHY183304	2020-03-06 15:49:14
6	Penulisan Ilmiah	AIF183002	2020-03-06 15:49:14
7	Manajemen Informasi dan Basis Data	AIF182302	2020-03-06 15:49:14
8	Sertifikasi Cyber Ops	AIF183240	2020-03-06 15:49:14
9	Algoritma dan Struktur Data	AIF182101	2020-03-06 15:49:14
10	Komputasi Matematika	AMS182704	2020-03-06 15:49:14
11	Fisika Komputasi	PHY181024	2020-03-06 15:49:14
12	Metode Numerik	AIF183153	2020-03-06 15:49:14

Gambar 5.25: Tangkapan layar untuk daftar entri pada entitas.

1 Pengelolaan Entitas

2 Hasil implementasi untuk fitur pengelolaan entitas dapat dilihat pada Gambar 5.25, 5.26, dan 5.27.

3 5.1.5 Halaman untuk Dosen Pengawas / Layar Projektor

4 Fitur yang digunakan untuk dosen pengawas dapat dilihat hasil implementasinya pada Gambar 5.28, 5.29, dan 5.30. Gambar 5.28 menunjukkan denah tempat duduk, dengan tampilan tab timer pada Gambar 5.29. Fitur penambahan waktu ujian dapat dilihat pada 5.30.

7 5.1.6 Halaman Tambahan untuk Dosen Koordinator

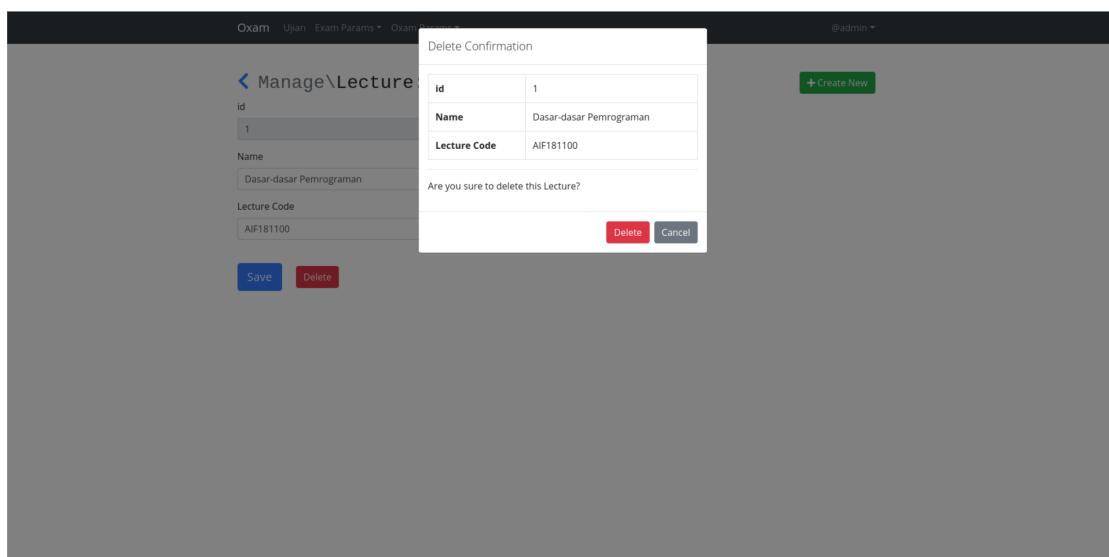
8 Fitur yang diimplementasi lainnya adalah fitur untuk dosen koordinator mengunduh jawaban peserta ujian. Tampilan email dapat dilihat pada Gambar 5.31 dan halaman pengunduhnya pada Gambar

Manage\Lecture::edit(1)

id	1
Name	Dasar-dasar Pemrograman
Lecture Code	AIF181100

Save Delete

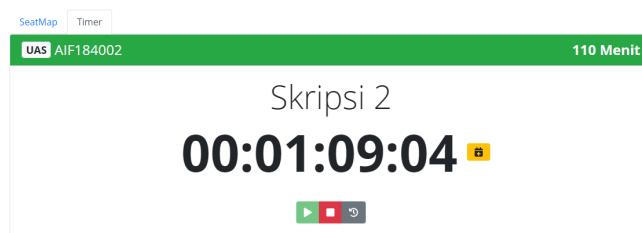
Gambar 5.26: Tangkapan layar untuk *editor* entri



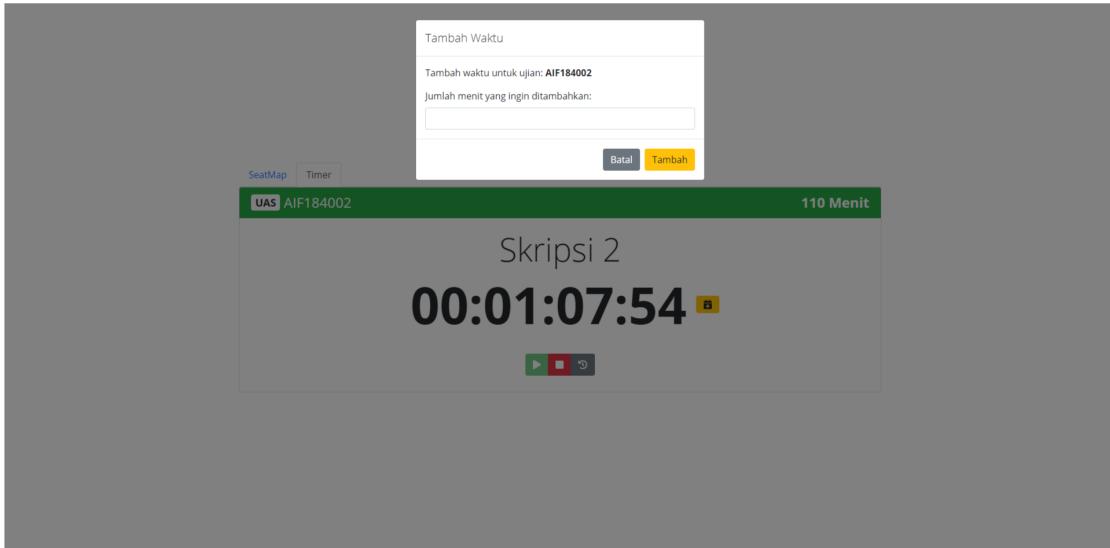
Gambar 5.27: Tangkapan layar untuk konfirmasi penghapusan entri.



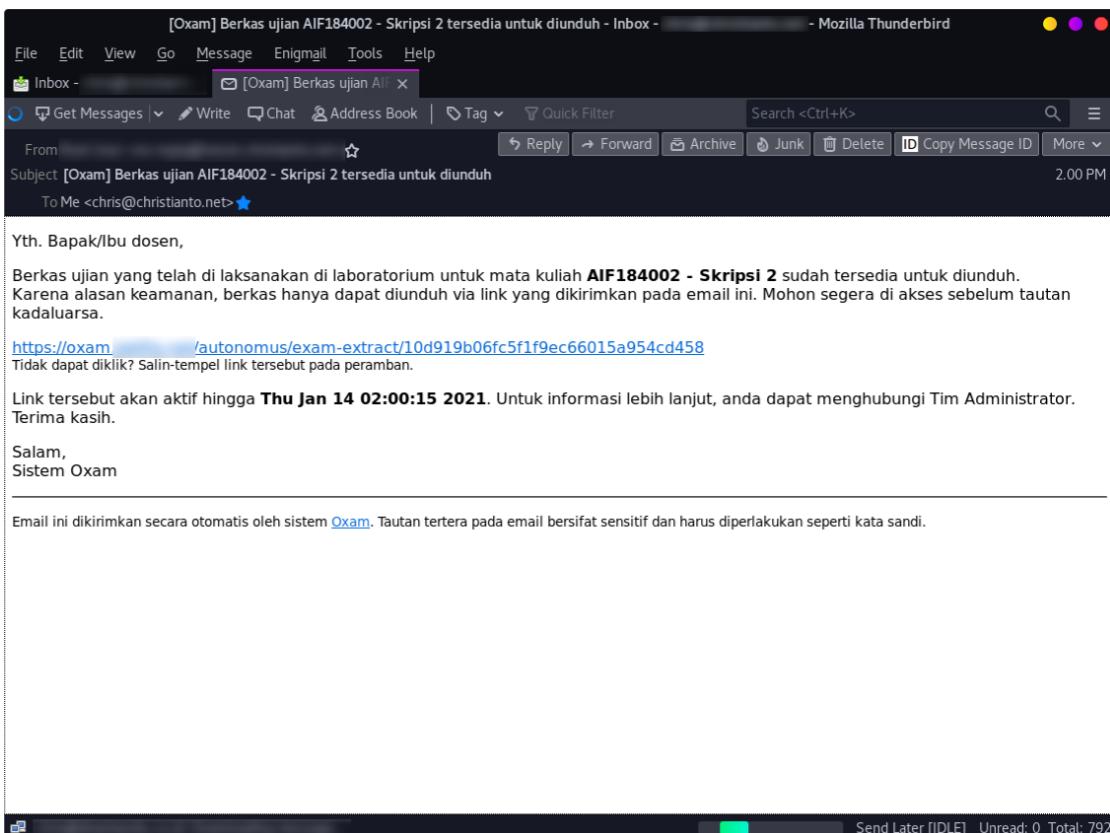
Gambar 5.28: Tangkapan layar untuk tampilan denah tempat duduk ruangan ujian.



Gambar 5.29: Tangkapan layar untuk tampilan timer untuk ditampilkan pada proyektor.



Gambar 5.30: Tangkapan layar untuk tampilan untuk modal penambahan waktu lebih.



Gambar 5.31: Tangkapan layar untuk tampilan email yang akan diterima oleh dosen.

The screenshot shows a web-based application for managing exam answers. At the top, there's a dark header bar with the text "Oxam (autonomous-mode)" on the left and "Admin" on the right. Below this is a table titled "Exam Answer Downloader" containing various parameters of the exam:

Mata Kuliah	Skripsi 2 (AIF184002)	Jumlah Peserta	5 Orang
Tahun Ajaran	20202	Dibuat pada	2021-01-13 13:33:57
Mulai - Berakhir	2021-01-15 15:34:37 - 2021-01-15 15:34:42	Terakhir diperbaharui	2021-01-15 15:34:42
Waktu Mulai Pengumpulan	2021-01-15 15:34:35		
Durasi	110 Menit		
Uts/Uas	UAS		

Below the table is a green button labeled "Unduh Jawaban". At the bottom of the page, there's a small footer note: "Oxam v5.0: Into the abyss | Built on Rabu, 13 Januari 2021 pukul 12.09, Commit: 33f5f538a".

Gambar 5.32: Tangkapan layar untuk tampilan halaman pengunduhan

¹ 5.32.

² **5.1.7 Bentuk Struktur Direktori Penyimpanan Ujian**

³ Implementasi untuk pengamanan berkas ujian dilakukan dengan mengacak nama *folder* dari
⁴ tempat penyimpanan ujian. Pada Gambar 5.33, berkas jawaban ujian disimpan pada *folder*
⁵ *uploads/secured-assets* dengan nama folder yang diacak.

⁶ **5.2 Pengujian Experimental**

⁷ Pengujian experimental dilakukan dengan mendemokan aplikasi pada tim Admin secara langsung.
⁸ Pada pengujian tersebut, Tim Admin akan membuat sebuah mata kuliah dan ujian, lalu mencoba
⁹ fitur-fitur yang tersedia pada sistem Oxam yang baru. Keberhasilan tersebut dilakukan dengan
¹⁰ memberikan kuisioner yang kemudian Tim Admin isi.

¹¹ Kuisioner yang akan diberikan pada Tim admin terdiri dari pertanyaan-pertanyaan berikut:

¹² 1. Membuat dan mengelola data untuk ujian (mata kuliah, memasukkan peserta, dst) menjadi
¹³ lebih mudah.

¹⁴ Pertanyaan dibuat dalam bentuk skala 1-4, dengan 1 sangat tidak setuju dan 5 sangat setuju.

¹⁵ Pertanyaan ditujukan untuk mengetahui apakah pengaturkan dan pengelolaan data untuk
¹⁶ ujian dapat diatur dengan mudah.

¹⁷ 2. Mengambil dan mengirimkan jawaban menjadi lebih mudah.

¹⁸ Pertanyaan dibuat dalam bentuk skala 1-4, dengan 1 sangat tidak setuju dan 5 sangat setuju.

¹⁹ Pertanyaan ditujukan untuk mengetahui proses pengambilan dan pengiriman jawaban menjadi
²⁰ lebih baik.

²¹ 3. Membuat ujian menjadi lebih mudah.

²² Pertanyaan dibuat dalam bentuk skala 1-4, dengan 1 sangat tidak setuju dan 5 sangat setuju.

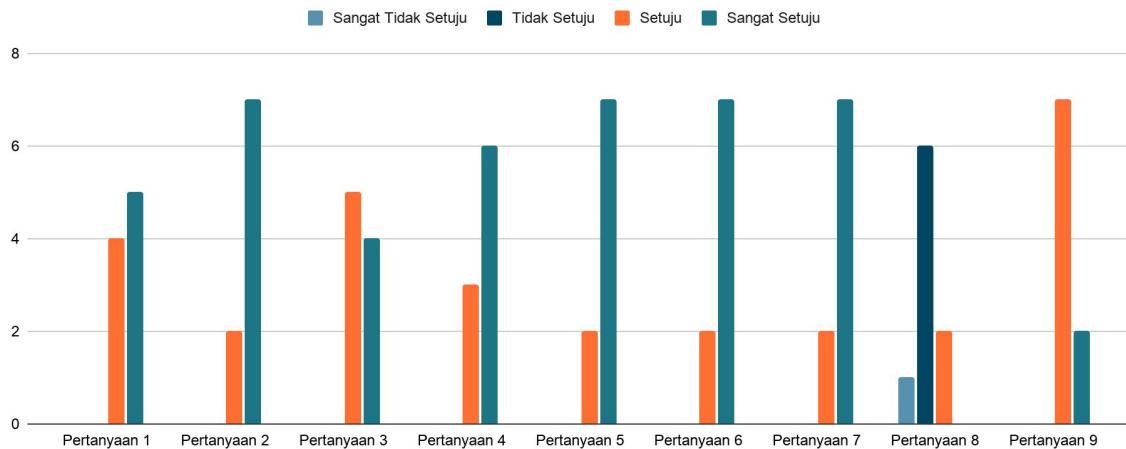
²³ Pertanyaan ditujukan untuk mengetahui apakah pembuatan ujian menjadi lebih baik.

```
hayashi@leo:/var/www/docker-apps/oxam-prod/backend/app$ tree -d
.
├── config
├── controller
│   └── api
│       ├── autonomus
│       ├── exam
│       ├── manage
│       └── internal
│           └── system
├── cronjob
├── helper
│   └── exceptions
│       └── rules
├── migration
├── model
│   └── system
│       └── ujian
├── output
├── service
└── ui
    ├── _bat
    ├── emails
    └── notifications
└── uploads
    └── secured_assets
        ├── 38aff7dfe7c420177e4f2b36a8105f4a
        ├── 562e21ebc197d4622b967807824ae33f
        ├── 632edc97ac9ee845f400ca7ec7a4bb98
        ├── 67cb0188aa5996889f9e048a39e7c5ed
        ├── 6e18d8db0263761a8920eeac10a00d70
        ├── 6f74e4ce3b3249966ecb541464998264
        ├── 92d94eb5ac8f6cef209a3522d9df6b28
        ├── 95e199cf63a836696d2ee23718ae8d3a
        ├── 9a852e3c6738924afbc9d1910872d7b
        ├── a3c3d62508a7ef5cca26d63611b58c0f
        ├── aae5df4ec3702f32f2d7cfc0d338bf69
        ├── b072805ba27d958a79a2a5ffcfb18648
        ├── b15cec6b36c8faf8f207b32fc191337e
        ├── f9568edfeccda81a5854ee92550dfb0d2
        └── fb82c8937e083bc60314e544e52f6ced
└── view
```

Gambar 5.33: Tampilan daftar direktori pada sistem backend aplikasi Oxam.

- 1 4. Upload Jawaban untuk peserta lebih mudah.
Pertanyaan dibuat dalam bentuk skala 1-4, dengan 1 sangat tidak setuju dan 5 sangat setuju. Pertanyaan ditujukan untuk mengetahui apakah formulir pengunggahan jawaban untuk peserta menjadi lebih baik.
- 2 5. Informasi (*password* dan ralat soal) untuk peserta menjadi lebih mudah.
Pertanyaan dibuat dalam bentuk skala 1-4, dengan 1 sangat tidak setuju dan 5 sangat setuju. Pertanyaan ditujukan untuk mengetahui pengalaman penyebaran kredensial untuk layanan tertentu untuk peserta dan informasi lainnya menjadi lebih mudah atau tidak.
- 3 6. Mengakses Admin panel lebih mudah.
Pertanyaan dibuat dalam bentuk skala 1-4, dengan 1 sangat tidak setuju dan 5 sangat setuju. Pertanyaan ditujukan untuk mengetahui apakah penggunaan admin panel menjadi lebih mudah.
- 4 7. Memindahkan peserta ke komputer lain menjadi lebih mudah.
Pertanyaan dibuat dalam bentuk skala 1-4, dengan 1 sangat tidak setuju dan 5 sangat setuju. Pertanyaan ditujukan untuk mengetahui apakah aplikasi Oxam yang baru dapat memindahkan peserta ke komputer lain dengan lebih mudah.
- 5 8. Aplikasi lebih *buggy*.

Hasil Kuisioner Pasca Demo



Gambar 5.34: Hasil kuisioner pasca demo pada pertanyaan berbentuk skala.

- 1 Pertanyaan dibuat dalam bentuk skala 1-4, dengan 1 sangat tidak setuju dan 5 sangat setuju.
 2 Pertanyaan ditujukan untuk mengetahui apakah aplikasi memiliki masalah *bug*.
 3 9. Bagaimana pengalaman secara keseluruhan Anda dengan Oxam yang baru?
 4 Pertanyaan dibuat dalam bentuk skala 1-4, dengan 1 sangat baik dan 5 sangat baik.
 5 Pertanyaan ditujukan untuk mengetahui kepuasan terhadap sistem Oxam yang baru.
 6 10. Apakah Anda memiliki kritik dan saran lainnya? Apa yang dapat aplikasi ini perbaiki di
 7 kemudian hari?
 8 Pertanyaan dibuat dalam bentuk paragraf panjang. Pertanyaan ini bertujuan untuk mengeta-
 9 hui kekurangan dan saran yang dapat sistem ini perbaiki pada iterasi berikutnya.
 10 Kuisioner tersebut lalu diberikan pada Tim Admin yang pernah menggunakan sistem Oxam
 11 yang lama yang kemudian dibandinkan dengan pengalaman mereka dalam menggunakan sistem
 12 Oxam yang lama. Kuisioner tersebut mendapatkan sembilan respon dan didapatkan data yang
 13 dapat dilihat pada Gambar 5.34.
 14 Hasil kuisioner tersebut dianalisis untuk menilai tingkat kepuasan dalam menggunakan aplikasi.

15 5.3 Pengujian Fungsional

- 16 Pengujian fungsional dilakukan dengan bantuan *Unit Testing*. Tes tersebut akan dilakukan setiap
 17 kali kode diunggah ke repositori penyimpanan kode sumber dengan bantuan CI/CD. Tes tersebut
 18 terdiri dari beberapa skenario dan langkah yang menentukan sebuah fitur dapat berjalan sesuai
 19 ekspektasi atau tidak. Tes tersebut terdiri dari
 20 1. Otentikasi
 21 Memastikan fitur kelas utama otentikasi dapat berjalan dengan semestinya.
 22

Test Step	Preconditions	Ekspektasi Hasil	Hasil Didapatkan	Status
Login dengan kata sandi yang salah harus ditolak.	Username benar dan kata sandi salah.	Mengeluarkan Eksepsi	Eksepsi didapatkan	Lolos
Login dengan username yang salah harus ditolak.	Username salah dan kata sandi benar.	Mengeluarkan Eksepsi	Eksepsi didapatkan	Lolos
Login dengan kredensial yang benar harus diterima.	Username dan kata sandi benar	Kelas mengembalikan nilai true	Kelas mengembalikan nilai true	Lolos

¹ 2. Ujian tanpa *shift*

² Memastikan fungsi yang bertugas untuk memetakan ujian dasar dapat berjalan sesuai ekspektasi.

³

Test Step	Preconditions	Ekspektasi Hasil	Hasil Didapatkan	Status
Ujian yang aktif hanya akan muncul pada saat <i>flag upcoming</i> dimatikan.				
Melakukan pemanggilan model dengan <i>query flag upcoming</i> aktif.	Ujian sedang aktif.	Tidak mengeluarkan ujian apapun	Ujian tidak didapatkan	Lolos
Melakukan pemanggilan model dengan <i>query flag upcoming</i> tidak aktif.	Ujian sedang aktif.	Mengeluarkan sebuah ujian yang baru saja dibuat	Ujian didapatkan, ujian tersebut se-suai dengan ujian yang dibuat diawal.	Lolos
Ujian yang akan aktif hanya akan didapatkan dengan <i>flag upcoming</i> aktif.				
Melakukan pemanggilan model dengan <i>query flag upcoming</i> aktif.	Ujian sedang aktif.	Mengeluarkan sebuah ujian yang baru saja dibuat	Ujian didapatkan, ujian tersebut se-suai dengan ujian yang dibuat diawal.	Lolos
Melakukan pemanggilan model dengan <i>query flag upcoming</i> tidak aktif.	Ujian sedang aktif.	Tidak mengeluarkan ujian apapun	Ujian tidak didapatkan	Lolos
Ujian yang telah selesai tidak akan muncul dimanapun.				
Melakukan pemanggilan model dengan <i>query flag upcoming</i> aktif.	Ujian sedang aktif.	Tidak mengeluarkan ujian apapun	Ujian tidak didapatkan	Lolos
Melakukan pemanggilan model dengan <i>query flag upcoming</i> tidak aktif.	Ujian sedang aktif.	Tidak mengeluarkan ujian apapun	Ujian tidak didapatkan	Lolos

1 3. Ujian dengan *shift*

2 Memastikan fungsi yang bertugas untuk memetakan ujian dengan *shift* dapat berjalan sesuai
3 ekspektasi.

4

<i>Test Step</i>	<i>Preconditions</i>	Ekspektasi Hasil	Hasil Didapatkan	Status
Kedua <i>shift</i> akan aktif.				
Melakukan pemanggilan model dengan <i>query flag upcoming</i> aktif.	<i>Shift 1</i> dan <i>2</i> akan aktif	Ujian <i>shift 1</i> didapatkan.	Ujian <i>shift 1</i> didapatkan.	Lolos
Melakukan pemanggilan model dengan <i>query flag upcoming</i> tidak aktif.	<i>Shift 1</i> dan <i>2</i> akan aktif	Tidak ada ujian yang didapatkan.	Tidak ada ujian yang didapatkan.	Lolos
Ujian pada <i>shift 1</i> aktif dan <i>shift 2</i> akan aktif.				
Melakukan pemanggilan model dengan <i>query flag upcoming</i> aktif.	<i>Shift 1</i> aktif dan <i>shift 2</i> akan aktif	Ujian <i>shift 2</i> didapatkan.	Ujian <i>shift 2</i> didapatkan.	Lolos
Melakukan pemanggilan model dengan <i>query flag upcoming</i> tidak aktif.	<i>Shift 1</i> aktif dan <i>shift 2</i> akan aktif	Ujian <i>shift 1</i> didapatkan.	Ujian <i>shift 1</i> didapatkan.	Lolos
Ujian pada <i>shift 1</i> selesai dan <i>shift 2</i> akan dimulai.				
Melakukan pemanggilan model dengan <i>query flag upcoming</i> aktif.	<i>Shift 1</i> selesai dan <i>shift 2</i> akan aktif	Ujian <i>shift 2</i> didapatkan.	Ujian <i>shift 2</i> didapatkan.	Lolos
Melakukan pemanggilan model dengan <i>query flag upcoming</i> tidak aktif.	<i>Shift 1</i> selesai dan <i>shift 2</i> akan aktif	Tidak ada ujian yang didapatkan.	Tidak ada ujian yang didapatkan.	Lolos
Ujian pada <i>shift 1</i> selesai dan <i>shift 2</i> sedang aktif.				
Melakukan pemanggilan model dengan <i>query flag upcoming</i> aktif.	<i>Shift 1</i> selesai dan <i>shift 2</i> aktif	Tidak ada ujian yang didapatkan.	Tidak ada ujian yang didapatkan.	Lolos
Melakukan pemanggilan model dengan <i>query flag upcoming</i> tidak aktif.	<i>Shift 1</i> selesai dan <i>shift 2</i> aktif	Ujian <i>shift 2</i> didapatkan.	Ujian <i>shift 2</i> didapatkan.	Lolos
<i>Test Step</i>	<i>Preconditions</i>	Ekspektasi Hasil	Hasil Didapatkan	Status
Ujian pada <i>shift 1</i> dan <i>2</i> selesai.				
Melakukan pemanggilan model dengan <i>query flag upcoming</i> aktif.	<i>Shift 1</i> selesai dan <i>shift 2</i> selesai	Tidak ada ujian yang didapatkan.	Tidak ada ujian yang didapatkan.	Lolos

Parameter	Rata-rata Nilai kepuasan (skala 1 hingga 5)
Pengelolaan data untuk ujian (Nomor 1)	3.56
Pengiriman berkas jawaban ujian (Nomor 2)	3.78
Pembuatan ujian (Nomor 3)	3.8
Pengunggahan data untuk ujian (Nomor 4)	3.67
Pendistribusian informasi layanan dan notifikasi peserta (Nomor 5)	3.78
Penggunaan Admin Panel (Nomor 6)	3.78
Migrasi peserta (Nomor 7)	3.78
Kestabilan aplikasi (Nomor 8)	2.11
Kepuasan secara keseluruhan (Nomor 9)	3.22

Tabel 5.4: Rata-rata nilai kepuasan berdasarkan kuisioner yang telah dilakukan.

Test Step	Preconditions	Ekspektasi Hasil	Hasil Didapatkan	Status
Melakukan pemanggilan model dengan <i>query flag upcoming</i> tidak aktif.	<i>Shift 1</i> selesai dan <i>shift 2</i> selesai	Tidak ada ujian yang didapatkan.	Tidak ada ujian yang didapatkan.	Lolos

¹ 5.3.1 Analisis Hasil Pengujian

- ² Berdasarkan kuisioner yang telah dilakukan, didapatkan rata-rata nilai kepuasan terhadap sistem
- ³ yang baru dengan rincian yang dapat dilihat pada tabel [5.4](#).
- ⁴ Dengan hasil umpan balik yang diberikan, aplikasi Oxam yang baru dinilai dapat meningkatkan
- ⁵ pengalaman memanajemen ujian di lab komputasi dengan nilai keseluruhan 3.22 poin dari 4.
- ⁶ Stabilitas aplikasi memiliki nilai kepuasan terendah pada 2.11 poin dari 4 dan pengalaman dalam
- ⁷ pembuatan ujian menjadi pengalaman terbaik pada aplikasi ini dengan nilai 3.8 poin dari 4.
- ⁸ Dilanjutkan dengan pengujian fungsional yang telah dilakukan, hasil dari unit testing menun-
- ⁹ jukkan bahwa fungsi sistem dapat berjalan dengan baik. *Query* yang digunakan untuk menentukan
- ¹⁰ ujian yang aktif dan tidak, dapat berjalan sesuai ekspektasi.

¹

BAB 6

²

KESIMPULAN DAN SARAN

³ 6.1 Kesimpulan

⁴ Aplikasi pendukung ujian yang ada saat ini perlu dibuat ulang karena beberapa masalah. Aplikasi
⁵ pendukung baru telah berhasil dibangun ulang untuk mengatasi hal-hal masalah tersebut. Aplikasi
⁶ pendukung diimplementasi dengan teknologi aplikasi berbasis web yang dibantu dengan teknik
⁷ REST API yang dapat dimanfaatkan untuk komunikasi antara sistem frontend dengan backend.
⁸ Sistem frontend diimplementasi dengan bantuan *library* Reactjs, dengan penggayaan yang dibantu
⁹ dengan **bootstrap**. Sistem backend dibantu dengan *framework* FatFree Framework dan *library*
¹⁰ pendukungnya. Sistem kemudian diimplementasikan diatas sistem Docker yang berjalan pada server
¹¹ lab. Selain itu, pada lingkungan pengembangan, terdapat sistem CI/CD yang berjalan diatas Gitlab
¹² CI yang digunakan untuk melakukan *build* utnuk sistem frontend dan *unit testing* pada sistem
¹³ backend.

¹⁴ Pengujian yang dilakukan pada aplikasi membuktikan bahwa aplikasi dapat memenuhi kebutuhan
¹⁵ baru yang muncul dari sistem yang saat ini berjalan. Hal ini didukung dengan kuisioner yang
¹⁶ dilakukan pada Tim Administator yang berpengalaman dengan sistem yang saat ini berjalan, dengan
¹⁷ hasil yang cukup memuaskan, 3.22 poin dari 4.

¹⁸ 6.2 Saran

¹⁹ Dibantu dengan kuisioner yang telah dilakukan pada saat pengujian, saran untuk penelitian
²⁰ selanjutnya adalah sebagai berikut:

²¹ 1. Aplikasi dapat mendukung pengujian daring.

²² 2. Perbaikan terhadap beberapa tampilan antarmuka.

²³ Perbaikan tersebut dapat meliputi penggantian satuan untuk halaman konfirmasi pembuatan
²⁴ ujian dari jam ke menit, teks informasi dan perintah, serta ubah pemilihan peserta pada
²⁵ notifikasi menjadi tombol khusus. Selain itu mengganti beberapa judul bagian menjadi lebih
²⁶ jelas, seperti judul untuk *editor* entri pada pengelola entitas.

²⁷ 3. Sehabis waktu ujian habis, peserta seharusnya diberikan tampilan waktu ujian telah habis.

²⁸ 4. Fitur notifikasi dibuat terintegrasi dengan peramban sehingga notifikasi muncul pada sistem
²⁹ operasi juga.

- 1 5. Fitur pengiriman email notifikasi seharusnya memunculkan informasi kapan email akan dikirim dan status pengirimannya.
- 2
- 3 6. Memberikan informasi sisa notifikasi penggerjaan di taskbar Windows.
- 4
- 5 7. Pada fitur perpindahan peserta, tombol pengunduhannya dibuat dapat diunduh ulang pada saat modal tertutup.
- 6
- 7 8. Fitur perpindahan dan *plotting* tempat duduk seharusnya menampilkan komputer yang sedang digunakan atau sedang ada masalah.
- 8
- 9 9. Eksekusi *script* yang otomatis, tidak perlu diunduh terlebih dahulu.
- 10 10. Sistem memberikan peringatan jika terdapat peserta yang duduk bersebelahan.

DAFTAR REFERENSI

- [1] `kenjis` (2017) Php framework benchmark. <https://github.com/kenjis/php-framework-benchmark>. 28 April 2020.
- [2] (2020) *Pedoman Pelaksanaan Ujian di Labkomp*.
- [3] Dictionary, O. (2019) Oxford dictionary: Definition of back end. https://www.lexico.com/en/definition/back_end. 19 November 2019.
- [4] Dictionary, O. (2019) Oxford dictionary: Definition of front end. https://www.lexico.com/en/definition/front_end. 19 November 2019.
- [5] NPM (2020) About packages and modules - npm docs. <https://docs.npmjs.com/about-packages-and-modules#about-packages>. 26 Desember 2020.
- [6] NodeJs (2020) Modules: Commonjs modules. https://nodejs.org/api/modules.html#modules_modules_commonjs_modules. 26 Desember 2020.
- [7] Reactjs (2020) Rfcs for changes to react. <https://github.com/reactjs/rfcs>. 26 Desember 2020.
- [8] sn0opy, t. (2019) Api reference. <https://fatfreeframework.com/3.6/api-reference>. 20 November 2019.
- [9] Reactjs (2019) React - a javascript library for building user interfaces. <https://reactjs.org>. 19 November 2019.
- [10] Typescript dan Contributors (2020) React. <https://www.typescriptlang.org/docs/handbook/react.html>. 27 December 2020.
- [11] Reactjs (2019) Virtual dom and internals. <https://reactjs.org/docs/faq-internals.html>. 19 November 2019.
- [12] Reactjs (2019) Thinking in react. <https://reactjs.org/docs/thinking-in-react.html>. 28 April 2020.
- [13] Braunstein, M. (2018) *Health Informatics on FHIR: How HL7's New API is Transforming Healthcare*. Springer International Publishing.
- [14] Fielding, R. T. (2000) Architectural Styles and the Design of Network-based Software Architectures. Disertasi. University of California, Irvine.
- [15] Fielding, R. T. dan Reschke, J. F. (2014) Hypertext transfer protocol (http/1.1): Semantics and content. RFC 7231. RFC Editor, <http://www.rfc-editor.org>.
- [16] Richardson, S., Leonard; Ruby (2018) *RESTful Web Services*. Sebastopol, California: O'Reilly Media.
- [17] GitLab (2020) Introduction to ci/cd methodologies. <https://docs.gitlab.com/ee/ci/introduction/index.html{#}introduction-to-cicd-methodologies>. 4 Mei 2020.

- [18] GitLab (2020) Gitlab ci/cd. <https://docs.gitlab.com/ee/ci/>. 4 Mei 2020.
- [19] Docker (2020) What is a container? <https://www.docker.com/resources/what-container>. 28 April 2020.
- [20] StackOverflow (2018) Developer survey results 2018. <https://insights.stackoverflow.com/survey/2018>. 10 September 2019.
- [21] StackOverflow (2019) Developer survey results 2019. <https://insights.stackoverflow.com/survey/2019>. 28 April 2020.

LAMPIRAN A

CHECKLIST PERSIAPAN UJIAN

Checklist Persiapan Ujian di Laboratorium Komputasi FTIS

Nama Mata Kuliah :
Semester / Tahun Akademik :

Mohon tandai beberapa hal berikut setelah di konfirmasi kepada admin labkom yang bertugas.

No	Hal yang wajib diperiksa koordinator	Status (beri tanda ✓ pada kotak)
1	Posisi peserta sudah diacak dan dicetak.	<input type="checkbox"/> Form tanda tangan <input type="checkbox"/> Pintu depan <input type="checkbox"/> Pintu ruangan
2	Sudah mendapatkan berita acara ujian.	<input type="checkbox"/> Sudah
3	Soal dan file bantuan sudah lengkap (+password) sesuai dengan yang seharusnya.	<input type="checkbox"/> Sudah diterima admin <input type="checkbox"/> Sudah didistribusikan
4	Tempat pengumpulan sudah dibuat dan sudah berkoordinasi mengenai nama file pengumpulan.	<input type="checkbox"/> Sudah
5	Durasi ujian di proyektor sudah sesuai dengan durasi di soal.	<input type="checkbox"/> Sudah
6	Menutup jalur ke server (+ drive z) dan/atau internet.	<input type="checkbox"/> Sudah
7	Layanan eksternal yang digunakan sudah diperiksa.	<input type="checkbox"/> Judge <input type="checkbox"/> Database Server <input type="checkbox"/> Lainnya:

Catatan :

1. Lembar ini diisi untuk setiap mata kuliah yang melakukan ujian di Labkom.
2. Bila ujian diadakan 2 shift cukup mengisi 1 kali saja.
3. Jawaban ujian akan segera dikirimkan ke alamat email dosen koordinator setelah ujian selesai diadakan.

Dosen Koordinator

Tertanda,
Admin 1

Admin 2

.....
Tgl : _____

.....
Tgl : _____

.....
Tgl : _____

LAMPIRAN B

KODE PROGRAM

Listing B.1 .gitlab-ci.yml

```
1 stages:
2   - test
3   - prebuild
4   - build
5   - deploy
7 .exceptions: &dont_run_things_in_this_job
8   except:
9     - "build/staging"
10    - "build/production"
12 # variable buat service
13 variables:
14   MYSQL_ROOT_PASSWORD: root
15   MYSQL_USER: homestead
16   MYSQL_PASSWORD: secret
17   MYSQL_DATABASE: homestead
18   DB_HOST: mysql
20 # test buat frontend
21 test:frontend:
22   stage: test
23   image: node:12.18
24   before_script:
25     - cd frontend
26     - yarn install
27   script:
28     - CI=true yarn run test
29     - CI=true REACT_APP_BUILD_COMMIT=$CI_COMMIT_SHORT_SHA REACT_APP_BUILD_DATE=$(date +"%F %T") yarn run build
30   <<: *dont_run_things_in_this_job
32 #
33 # build frontend, buat testing doang
34 # build:frontend:
35 #   stage: prebuild
36 #   image: node:12.18
37 #   before_script:
38 #     - cd frontend
39 #     - yarn install
40 #   script:
41 #     - CI=true yarn run build
42 #     - mv ./build/* ../backend/public_html/
43 #   artifacts:
44 #     paths:
45 #       - frontend/build/*
46 #     <<: *dont_run_things_in_this_job
47 #   cache:
48 #     key: yarn-test-cache
49 #     paths:
50 #       - frontend/node_modules/
52 .backend_test_template: &backend_tester
53   stage: test
54   services:
55     - mysql:5.7
56   before_script:
57     - cd backend
58     - composer install --prefer-dist --no-ansi --no-interaction --no-progress
59   script:
60     - composer run-script pretest
61     - composer run-script test
62   <<: *dont_run_things_in_this_job
64 # test buat support php7.3
65 test:backend:php7.3:
66   image: edbizarro/gitlab-ci-pipeline-php:7.3-alpine
67   <<: *backend_tester
68   <<: *dont_run_things_in_this_job
69   cache:
70     key: php7-test-cache
71     paths:
72       - backend/vendor/
74 # =====
```

```

75 # test integrasi doang
76 # finalisasi:
77 #   stage: integration-test
78 #   dependencies:
79 #     - build:frontend
80 #     - test:backend:php7.1

82 .deployable: &deployable
83   stage: build
84   image: node:12.18
85   dependencies: []
86   before_script:
87     - apt-get update -y
88     - "which ssh-agent || ( apt-get install openssh-client -y )"
89     - "which sshpass || ( apt-get install sshpass -y )"
90     - eval $(ssh-agent -s)
91     - ssh-add <(echo "$GIT_SSH_PRIV_KEY")"
92     - mkdir -p ~/.ssh
93     - cat ./gitlab/known-hosts >> ~/.ssh/known_hosts
94     - git config --global user.email "repurikaqbot.christianto.net"
95     - git config --global user.name "Replica"
96     - git remote set-url origin "git@gitlab.com:${CI_PROJECT_PATH}.git"
97     - "which gpg || ( apt-get install gnupg2 -y )"
98     - echo "${GIT_PGP_PRIV_KEY}" | gpg --no-tty --batch --import
99     - git config user.signingkey 221D5347423ABB8694FEF18CB728857822EEB347
100   script:
101     - (git checkout $DEPLOY_TO_BRANCH && git pull) || git checkout -b $DEPLOY_TO_BRANCH
102     - git merge origin/master -X theirs
103     - rm -rfv ./backend/public_html
104     - mkdir ./backend/public_html
105     - git checkout -q backend/public_html/index.php
106     - git checkout -q backend/public_html/.htaccess
107     - cd frontend
108     - npm install -g json
109     - json -I -f package.json -e "this.homepage=\"${DEPLOY_HOMEPAGE}\""
110     - yarn install
111     - CI=true REACT_APP_BUILD_COMMIT=$CI_COMMIT_SHORT_SHA REACT_APP_BUILD_DATE=$(date +"%F %T") yarn run build
112     - mv build/* ../backend/public_html/
113     - git add --all
114     - (git commit -a -S -m "CI/CD build for ${CI_COMMIT_SHA}") || echo "Nothing's need to be committed."
115     - git push --force -u origin $DEPLOY_TO_BRANCH
116   only:
117     - master
118   cache:
119     key: production-cache
120     paths:
121       - frontend/node_modules

123 # =====
124 # deploy ke development server
125 staging:build:
126   <<: *deployable
127   variables:
128     DEPLOY_HOMEPAGE: "https://bleeding-edge.oxam.ftis"
129     DEPLOY_TO_BRANCH: "build/staging"
130   environment:
131     name: "Leo - Staging"
132     url: "https://bleeding-edge.oxam.ftis/"

135 staging:deploy:
136   image: alpine:3.11
137   stage: deploy
138   dependencies:
139     - staging:build
140   script:
141     - "which ssh-agent || ( apk update && apk add openssh-client )"
142     - mkdir -p ~/.ssh
143     - cat ./gitlab/known-hosts >> ~/.ssh/known_hosts
144     - eval $(ssh-agent -s)
145     - cat $SSH_DEPLOY_KEY | tr -d '\r' | ssh-add -
146     - ssh $SSH_HOST "cd ${DEPLOY_STAGGING} && git pull && sh ./gitlab/post-pull.sh"
147   only:
148     - master
149   environment:
150     name: "Leo - Staging"
151     url: "https://bleeding-edge.oxam.ftis/"
152 # =====
153 # deploy ke production server
154 production:build:
155   <<: *deployable
156   variables:
157     DEPLOY_HOMEPAGE: "https://oxam.ftis"
158     DEPLOY_TO_BRANCH: "build/production"
159   environment:
160     name: "Leo - Production"
161     url: "https://oxam.ftis/"

163 production:deploy:
164   image: alpine:3.11
165   stage: deploy
166   dependencies:
167     - production:build
168   script:
169     - "which ssh-agent || ( apk update && apk add openssh-client )"
170     - mkdir -p ~/.ssh
171     - cat ./gitlab/known-hosts >> ~/.ssh/known_hosts
172     - eval $(ssh-agent -s)
173     - cat $SSH_DEPLOY_KEY | tr -d '\r' | ssh-add -

```

```

174     - ssh $SSH_HOST "cd ${DEPLOY_PRODUCTION} && git pull && sh ./gitlab/post-pull.sh"
175   only:
176     - master
177   when: manual
178   environment:
179     name: "Leo - Production"
180     url: "https://oxam.ftis/"

```

Listing B.2 .env.development

```

1 MYSQL_ROOT_PASSWORD=root
2 MYSQL_DATABASE=dev_oxam
3 MYSQL_HOST=mysql
4 MYSQL_USER=dev_oxam
5 MYSQL_PASSWORD=
6 MYSQL_ALLOW_EMPTY_PASSWORD=yes
7 PMA_USER=root
8 PMA_PASSWORD=root

10 XDEBUG_CONFIG="remote_host=172.17.0.1 remote_enable=1 remote_port=9000"

```

Listing B.3 app.php

```

1 <?php
2   //set on error
3   \F3::instance()->set('ONERROR', function($f3) {
4     if($f3->exists('EXCEPTION') && $f3->get('EXCEPTION') instanceof \Model\Error) {
5       $exc = $f3->get('EXCEPTION');
6       $f3->status($exc->get_http_status());
7       return \View\Api::error($exc);
8     }
9     $error = new \Model\Error(
10       $f3->get('ERROR.status'),
11       $f3->get('ERROR.text'),
12       "HTTP". $f3->get('ERROR.code'),
13       null,
14       $f3->get('ERROR.code'),
15       null,
16       $f3->get('EXCEPTION')
17     );
18
19     return \View\Api::error($error);
20   });
21
22   $user = \Model\System\User::getFromHTTPHeader();
23   if($user) {
24     \F3::instance()->set('SYSTEM.USER', $user);
25   }

```

Listing B.4 config.ini

```

1 ; This applies for general settings over the site.
2 ; Change this without knowing what it do, then regret it later.
3 [globals]
4 ; This variable are intended to set all time settings
5 ; relatively to this timezone. As the site are based on Bandung,
6 ; we use GMT+7, or internationally, Jakarta Timezone.
7 TZ=Asia/Jakarta

9 ; This will control the debug info
10 DEBUG=3

12 ; Locate the cache space
13 TEMP=_tmp/

15 ; Locate some dependencies that Composer didn't know
16 AUTOLOAD=app/

18 ; This variable are used to give a upload cache place,
19 ; so when the user gives us files, it will be stored on it.
20 UPLOADS=app/uploads/

22 ; UI are the place of our html templates.
23 UI="public_html/app/ui/"

26 LOGS=logs/

28 ; STOP HERE.
29 ; This is part of system cycle.
30 [configs]
31 app/config/ilgar.ini=true
32 app/config/site.ini=true
33 app/config/routings.ini=true
34 app/config/cron.ini=true
35 app/config/security.generated.ini=true
36 app/config/npmtranspile.ini=true

```

Listing B.5 cron.ini

```

1 [CRON]
2 log = TRUE
3 web = FALSE

5 [CRON.jobs]
6 AutoSendEmail = \Cronjob\AutoReport->exam, @hourly

```

Listing B.6 ilgar.ini

```

1 [ILGAR]
2 info=app/config/migration.json
3 path=app/migration/
4 ; prefix="Migration\
5 access_path="GET @ilgar: /-/migrate"

```

Listing B.7 npmtranspile.ini

```

1 [NPM_TRANSPILE]
2 ##THE ONE THAT WILL BE USED IS THE 2018 VERSION ONE.
3 16=m
4 17=f
5 18=i

```

Listing B.8 routings.ini

```

1 [routes]
2 GET / = Controller\App->get_serve

4 GET /admin = Controller\App->get_serve
5 GET /exam = Controller\App->get_serve
6 GET /autonomus = Controller\App->get_serve
7 GET /admin/* = Controller\App->get_serve
8 GET /exam/* = Controller\App->get_serve
9 GET /autonomus/* = Controller\App->get_serve

11 ; Plain URL
12 ; This will be treated as normal JSON.
13 GET /api/v1/@interface@module=Controller\Api@interface@Module->get_index
14 POST /api/v1/@interface@module=Controller\Api@interface@Module->post_index
15 PUT /api/v1/@interface@module=Controller\Api@interface@Module->put_index
16 DELETE /api/v1/@interface@module=Controller\Api@interface@Module->delete_index

18 GET /api/v1/@interface@Module@func=Controller\Api@interface@Module->get_func
19 POST /api/v1/@interface@Module@func=Controller\Api@interface@Module->post_func
20 PUT /api/v1/@interface@Module@func=Controller\Api@interface@Module->put_func
21 DELETE /api/v1/@interface@Module@func=Controller\Api@interface@Module->delete_func

23 ## CRUD
24 GET /api/v1/manage@Module=Controller\Api\Manage@Module->get_index
25 POST /api/v1/manage@Module=Controller\Api\Manage@Module->post_index

27 GET /api/v1/manage@Module@id=Controller\Api\Manage@Module->get_item
28 PUT /api/v1/manage@Module@id=Controller\Api\Manage@Module->put_item
29 DELETE /api/v1/manage@Module@id=Controller\Api\Manage@Module->delete_item

31 GET /api/v1/manage@Module@id=func=Controller\Api\Manage@Module->get_item_func
32 POST /api/v1/manage@Module@id=func=Controller\Api\Manage@Module->post_item_func
33 PUT /api/v1/manage@Module@id=func=Controller\Api\Manage\Module->put_item_func
34 DELETE /api/v1/manage@Module@id=func=Controller\Api\Manage@Module->delete_item_func

```

Listing B.9 site.example.ini

```

1 [database]
2 ; This is database settings.
3 ; we're using dsn to connect to the database.
4 ; we use SQL-based database. So, if you need other langs,
5 ; ask us to change some code and files.
6 ; refrence link : https://fatfreeframework.com/3.6/sql
7 dsn="mysql:host=your_database_server_here;port=3306;dbname=your_database_name_here;charset=utf8"
8 username=your_database_username
9 password=your_database_password

11 [SECURITY]
12 issuer="oxam.ftis"
13 expiration=3600

15 [SMTP]
16 host=
17 username=
18 password=
19 ;from=
20 ;replyto=

22 [globals]
23 BASE_URL_PUBLIC="http://oxam.localhost/"

25 [ldap_provider]
26 account_suffix=@ftis.unpar
27 hosts='ftis.unpar', '10.100.79.11'
28 base_dn='ftis.unpar'

```

```

29 admin_username=kambing
30 admin_password=kepo-ih
32 [dev_setting]
33 query_ldap=false

```

Listing B.10 config.php

```

1 <?php
2     chdir(dirname(__DIR__));
3     // Config Loader
4     \F3::instance()->config("app/config/config.ini");
5
6     //connect to SQL Database.
7     \F3::set('SYSTEM.DB', false);
8     if(\F3::instance()->exists('database')) {
9         try {
10             \F3::set('DB', new \DB\SQL(\F3::get('database.ds'), [
11                 'username' => \F3::get('database.username'),
12                 'password' => \F3::get('database.password'), [
13                     PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
14                     PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8',
15                 ]
16             ]);
17             \F3::set('SYSTEM.DB', true);
18         } catch (PDOException $e) {
19             \F3::set('DB', null);
20         }
21     }
22
23 \Chez14\Ilgar\Boot::now();
24
25 // some constants
26 \F3::set('CZ_PUBLIC_HTML', realpath(__DIR__ . '../public_html/'));
27
28 //LDAP Connect
29 if(\F3::get('dev_setting.query_ldap')) {
30     $ad = new \Adldap\Adldap();
31     $ad->addProvider(new \Adldap\Connections\Provider(\F3::get('ldap_provider')));
32
33     \F3::set('LDAP.base', $ad);
34     try {
35         \F3::set('LDAP.provider', $ad->connect());
36     } catch (\Exception $e) {
37         \F3::set('LDAP.provider', null);
38         \F3::set('dev_setting.query_ldap', false);
39     }
40 }

```

Listing B.11 examdownload.php

```

1 <?php
3 namespace Controller\Api\Autonomus;
5
6 use DateInterval;
7 use DateTimeImmutable;
8 use Helper\AnswerZipper;
9 use InvalidArgumentException;
10 use Lcobucci\JWT\Builder;
11 use Lcobucci\JWT\Parser;
12 use Lcobucci\JWT\Signer\Key;
13 use Lcobucci\JWT\Signer\Keychain;
14 use Lcobucci\JWT\Signer\Rsa\Sha256;
15 use Lcobucci\JWT\ValidationData;
16 use Model\Error asModelError;
17 use Model\Ujian\ExamReport;
18 use Respect\Validation\Exceptions\NestedValidationException;
19 use View\Api;
20
21 class ExamDownload extends \Prefab
22 {
23
24     public function beforeroute($f3)
25     {
26         \Helper\PreAPI::doTheThing();
27     }
28
29     function __construct()
30     {
31         v::with("\Helper\Rules");
32     }
33
34     /**
35      * See info and generate download token for current exam.
36      *
37      * @param \Base $f3
38      * @return void
39      */
40     function post_index(\Base $f3)
41     {
42         try {
43             $validator = v::key("token", v::notOptional()->mustExists("\Model\Ujian\ExamReport", 'token'));
44             $validator->assert($f3->POST);
45         }
46     }

```

```

45     // TODO: Limit with Google Auth Token (reconsult)
46
47     $reportExam = new ExamReport();
48     $reportExam->load(["token=?", "valid_until=?"], $f3->POST['token'], date('Y-m-d H:i:s')));
49
50     if ($reportExam->loaded() === 0) {
51         throw new InvalidArgumentException("Token no longer valid.");
52     }
53
54     $exam = $reportExam->exam;
55
56     $logger = $exam->getLoggerInstance();
57     $logger->warn("Autoreport link from " . $reportExam->_id . " is being accessed via " . $f3->IP);
58
59     // generate JWT Token
60     $issued = new DateTimeImmutable();
61     $expiration = $issued->add(DateInterval::createFromString(\Base::instance()->get('SECURITY.expiration') . " seconds"));
62     $signer = new Sha256();
63     $keychain = new Key("file://" . \Base::instance()->get('SECURITY.privatekey_path'));
64     $token = (new Builder())
65         ->issuedAt($issued)
66         ->expiresAt($expiration)
67         ->withClaim('uid', $reportExam->_id)
68         ->getToken($signer, $keychain);
69
70     $link = [$f3->get("BASE_URL_PUBLIC"), "api", "v1", "autonomus", "examdownload", "download"];
71     $link = implode("/", array_map(function ($parts) {
72         return trim($parts, "\\\\");
73     }, $link));
74     $link .= "?token=$token";
75
76     return Api::success([
77         "exam" => $exam->cast(),
78         "downloadToken" => $link
79     ]);
80 } catch (NestedValidationException $e) {
81     throw \Helper\Ruler::transformToError($e);
82 } catch (InvalidArgumentException $e) {
83     throw new ModelError("Invalid Input", $e->getMessage(), "X400", "Exception", 400);
84 }
85
86 /**
87 * Zip and download the answer file for generated exam.
88 *
89 * @param \Base $f3
90 * @return void
91 */
92 function get_download(\Base $f3)
93 {
94     try {
95         $validator = v::key("token", v::notOptional());
96         $validator->assert($f3->GET);
97     } catch (NestedValidationException $e) {
98         throw \Helper\Ruler::transformToError($e);
99     }
100
101     $token = (new Parser())->parse($f3->GET['token']);
102     // die(var_dump($token));
103
104     $data = new ValidationData(); // It will use the current time to validate (iat, nbf and exp)
105     $data->setIssuer(\Base::instance()->get('SECURITY.issuer'));
106
107
108     $signer = new Sha256();
109     $keychain = new Keychain();
110
111     if (!$token->validate($data) || !$token->verify(
112         $signer,
113         $keychain->getPublicKey("file://" . \Base::instance()->get('SECURITY.publickey_path'))
114     )) {
115         throw new ModelError("Bad Token", "The given token is invalid, expired, or Report Token no longer exists.", 403, "Bad Token", 403);
116     }
117
118     // Token has been validated. Will now continue zip the answer and serve
119     // them as is.
120     $examReport = new ExamReport();
121     $examReport->load(["id=?", $token->getClaim('uid')]);
122     if (!$examReport->loaded()) {
123         throw new ModelError("Bad Token", "The given token is invalid, expired, or Report Token no longer exists.", 403, "Bad Token", 403);
124     }
125
126     $logger = $examReport->exam->getLoggerInstance();
127     $logger->warn("Exam answer is being downloaded via ExamReport#" . $examReport->_id . " via " . $f3->IP);
128
129     $zipname = AnswerZipper::zipExam($examReport->exam);
130     $filename = AnswerZipper::getProperFilename($examReport->exam);
131
132     $logger->info("Answer filename is " . $filename);
133
134     $filename .= ".zip";
135
136     header("X-Filename: $filename");
137     \Web::instance()->send(
138         $zipname,
139         null,
140

```

```

141     0,
142     true,
143     $filename
144   );
145   if (is_file($zipname)) {
146     unlink($zipname);
147   }
148 }
149 }
```

Listing B.12 base.php

```

1 <?php
2 Namespace Controller\Api;
3
4 abstract class Base extends \Prefab {
5
6   public function beforeroute($f3){
7     \Helper\PreAPI::doTheThing();
8   }
9
10  protected function precheck_must_login($fail_message = "You need to login before doing anything with Manage API."){
11    $user = \Model\System\User::getFromHTTPHeader(true);
12    if(!$user){
13      throw new \Model\Error(
14        "You must login before you're able to do that.",
15        $fail_message,
16        "XXX01",
17        "No token nor session of user detected",
18        401
19      );
20    }
21  }
22 }
```

Listing B.13 info.php

```

1 <?php
2
3 namespace Controller\Api\Exam;
4
5 class Info extends \Prefab
6 {
7   public function get_index($f3)
8   {
9     header("X-IP: " . $f3->get("IP"));
10    $participant = \Model\Ujian\Participant::getAnyParticipant();
11    if ($participant) {
12      $parObj = $participant->cast(null, 2);
13      $parObj['exam']['answer_slot'] = array_map(function ($answer_slot) use ($participant) {
14        if ($answer_slot->deleted_on) {
15          return null;
16        }
17        return $answer_slot->cast(null, 0, true, $participant);
18      }, (array) $participant->exam->answer_slot);
19
20      // prevent other participant info leaking.
21      $parObj['exam'][ 'participants' ] = [];
22      return \View\Api::success($parObj);
23    }
24    return \View\Api::success([]);
25  }
26 }
```

Listing B.14 notification.php

```

1 <?php
2
3 namespace Controller\Api\Exam;
4
5 use View\Api;
6
7 class Notification extends \Prefab
8 {
9   public function get_index($f3)
10  {
11    $participant = \Model\Ujian\Participant::getAnyParticipant();
12
13    if (!$participant) {
14      return Api::success([]);
15    }
16
17    // Cast the notification without the participant field to minimize the
18    // recursive effect.
19    return Api::success($participant->notifications->castAll([
20      "participant" => 0
21    ]));
22  }
23 }
```

Listing B.15 submission.php

```

1 <?php
3 namespace Controller\Api\Exam;
5 use Model\Error;
7 class Submission extends \Prefab
8 {
9     public function get_submit($f3)
10    {
11        if (!$f3->exists("GET.answer_slot")) {
12            throw new \Model\Error(
13                "Incomplete Request",
14                "We need the `answer_slot` ID to do that",
15                "ES03"
16            );
17        }
19        $participant = \Model\Ujian\Participant::getActiveParticipant();
20        if (!$participant) {
21            throw new \Model\Error(
22                "Unexpected Upload",
23                "Answer submission has been closed. Please check your exam time.",
24                "ES07"
25            );
26        }
28        $answer_slot = new \Model\Ujian\AnswerSlot();
29        $answer_slot->has('exam', ['id=?', $participant->exam->_id]);
30        $answer_slot->load(['id = ?', $f3->get("GET.answer_slot")]);
32        if ($answer_slot->dry()) {
33            throw new \Model\Error(
34                "Bad Request",
35                "The given `answer_slot` is invalid.",
36                "ES04"
37            );
38        }
40        $submission = $answer_slot->getSubmission($participant);
41        if ($submission->dry()) {
42            throw new Error("No submission found", "Submission not found", "HTTP404", "no reason", 404);
43        }
44        if ($f3->exists("GET.force_download")) {
45            \Web::instance()->send(
46                $submission->getFullPath(),
47                null,
48                0,
49                true,
50                $submission->answer_slot->simulateFormat($participant)
51            );
52        } else {
53            return \View\Api::success($submission->cast());
54        }
55    }
57    public function post_submit($f3)
58    {
59        if (!$f3->exists("POST.answer_slot")) {
60            throw new \Model\Error(
61                "Incomplete Request",
62                "We need the `answer_slot` ID to do that",
63                "ES03"
64            );
65        }
67        $participant = \Model\Ujian\Participant::getActiveParticipant();
68        if (!$participant) {
69            throw new \Model\Error(
70                "Unexpected Upload",
71                "Answer submission has been closed. Please check your exam time.",
72                "ES07"
73            );
74        }
77        $answer_slot = new \Model\Ujian\AnswerSlot();
78        $answer_slot->has('exam', ['id=?', $participant->exam->_id]);
79        $answer_slot->load(['id = ? and deleted_on = ?', $f3->get("POST.answer_slot"), null]);
81        if ($answer_slot->dry()) {
82            throw new \Model\Error(
83                "Bad Request",
84                "The given `answer_slot` is invalid.",
85                "ES04"
86            );
87        }
89        $web = \Web::instance();
90        $files = $web->receive(
91            function ($file, $formFieldName) use ($answer_slot, $participant) {
92                return ($basename($file['name']) == $answer_slot->simulateFormat($participant));
93            },
94            true,
95            false
96        );

```

```

98     if (count($files) > 1) {
99         throw new \Model\Error(
100             "Bad news",
101             "Given answer is not singular. Send one-by-one please.",
102             "ES05"
103         );
104     }
105
106     if (!isset($files[array_keys($files)[0]])) {
107         throw new \Model\Error(
108             "Invalid Filename",
109             "Given filename is not respecting slot format. Got: " . array_keys($files)[0],
110             "ES06"
111         );
112     }
113     $file = array_keys($files)[0];
114     $submission = $answer_slot->submit($file, $participant);
115     $submission->touch();
116     $submission->save();
117     return \View\Api::success($submission->cast());
118 }
119 }
```

Listing B.16 acl.php

```

1 <?php
3 namespace Controller\Api\Manage;
5 use Controller\CRUDBase;
7 class Acl extends CRUDBase
8 {
9     protected $permissionPrefix = "manage-system-acl";
10    protected $model = "\Model\System\Acl";
11 }
```

Listing B.17 aclitem.php

```

1 <?php
3 namespace Controller\Api\Manage;
5 use Controller\CRUDBase;
7 class AclItem extends CRUDBase
8 {
9     protected $permissionPrefix = "manage-system-aclitem";
10    protected $model = "\Model\System\AclItem";
11 }
```

Listing B.18 answerslot.php

```

1 <?php
3 namespace Controller\Api\Manage;
5 use Controller\CRUDBase;
7 class AnswerSlot extends CRUDBase
8 {
9     protected $permissionPrefix = "manage-ujian-answerslot";
10    protected $model = "\Model\Ujian\AnswerSlot";
11 }
```

Listing B.19 computer.php

```

1 <?php
3 namespace Controller\Api\Manage;
5 use Controller\CRUDBase;
7 class Computer extends CRUDBase
8 {
9     protected $permissionPrefix = "manage-ujian-computer";
10    protected $model = "\Model\Ujian\Computer";
11 }
```

Listing B.20 exam.php

```

1 <?php
3 namespace Controller\Api\Manage;
5 use Controller\CRUDBase;
6 use Exception\NotParseable;
```

```

7  use Helper\AnswerZipper;
8  use InvalidArgumentException;
9  use Model\Error;
10 use Model\System\AclItem;
11 use Model\System\User;
12 use Model\Ujian\Notification;
13 use Model\Ujian\Participant;
14 use Respect\Validation\Exceptions\NestedValidationException;
15 use Respect\Validation\Validator as v;

17 class Exam extends CRUDBase
18 {
19     protected $permissionPrefix = "manage-ujian-exam";
20     protected $model = "\Model\Ujian\Exam";

22     public function globalFilterTrigger($model): void
23     {
24         // Check if user is ILogin user.
25         $iplogin = User::getIPLoginHTTPHeader();
26         if ($iplogin === null) {
27             return;
28         }

29         // get room:
30         $location = array_map(function ($data) {
31             return $data->_id;
32         }, (array)$iplogin->locations);

33         $model->has('participants.computer', ["location in (?)", $location]);
34     }

35     /**
36      * List all object in a model.
37      * May support search and paging. But not sure.
38      *
39      * @param object $f3 FatFree Object
40      * @return void
41      */
42     public function get_index($f3)
43     {
44         // permission checkpoint
45         $this->permission_check($this->permissionPrefix, AclItem::READ);

46         $upcomingTimeMinutes = 10;

47         $model = new $this->model;
48         $this->globalFilterTrigger($model);
49         if ($f3->exists('GET.screenmode')) {
50             $models = $model->find([
51                 "time_start >= ? or timeEnded >= ?" and deleted_on = ?",
52                 date('Y-m-d H:i:s', strtotime("-$upcomingTimeMinutes minute"))),
53                 date('Y-m-d H:i:s', strtotime("-$upcomingTimeMinutes minute")),
54                 null
55             ], ["order" => "time_start ASC"]);
56         } else {
57             $models = $model->find(["deleted_on = ?", null]);
58         }

59         if ($models === false) {
60             $models = [];
61         } else {
62             $models = $models->castAll();
63         }
64         return \View\Api::success($models);
65     }

66     public function get_item_notifications(\Base $f3)
67     {
68         $this->permission_check($this->permissionPrefix, AclItem::READ);
69         $this->permission_check($this->permissionPrefix, AclItem::READ);
70         $ujian = parent::getMentionedItem($f3);

71         $notif = new Notification();
72         $notif->has("participants", ["exam=?", $ujian->_id]);
73         $notifs = $notif->find();

74         if (!$notifs) {
75             $notifs = [];
76         }

77         return \View\Api::success(array_map(function ($data) {
78             return $data->cast(null, null, false);
79         }, (array) $notifs));
80     }

81     public function get_item_participants(\Base $f3)
82     {
83         $this->permission_check($this->permissionPrefix, AclItem::READ);
84         $ujian = parent::getMentionedItem($f3);

85         $participant = new \Model\Ujian\Participant();
86         $participant->fields(['exam'], true);
87         $participants = $participant->find(["exam = ? and deleted_on = ?", $ujian->_id, null]);

88         if (!$participants) {
89             $participants = [];
90         } else {
91             $participants = $participants->castAll();
92         }
93     }

```

```

107     return \View\Api::success($participants);
108 }
109
110 public function post_item_populate(\Base $f3)
111 {
112     // permission checkpoint
113     $this->permission_check($this->permissionPrefix, AclItem::UPDATE);
114     $ujian = parent::getMentionedItem($f3);
115
116     $logger = $ujian->getLoggerInstance();
117     $logger->notice("Exam participants getting populated...");
118     try {
119         // id from param must be exists
120         $validator = v::key("computers", v::notOptional()->each(v::mustExists("\Model\Ujian\Computer", 'id', 'deleted_on')));
121             ->key("participants", v::notOptional());
122             ->key("should_cleanup", v::boolVal(), false);
123         $validator->assert($f3->POST);
124
125         if (array_key_exists("should_cleanup", $f3->POST) && $f3->POST['should_cleanup']) {
126             $pastParticipants = (new Participant())->find(["exam = ?", $ujian->_id]);
127             foreach ($array $pastParticipants as $par) {
128                 $par->deleted_at = time();
129                 $par->save();
130             }
131         }
132
133         $computers = $f3->POST['computers'];
134         $participants = $f3->POST['participants'];
135
136         try {
137             $participants = array_map(function ($par) {
138                 $info = \Chez14\NpmParser\Solver::getInfo($par);
139                 return [$par, $info];
140             }, $participants);
141         } catch (NotParseable $e) {
142             throw new InvalidArgumentException("There's invalid participant NPM. Please recheck! ERR: " . $e->getMessage());
143         }
144
145         shuffle($participants);
146         shuffle($computers);
147
148         foreach ($participants as $parto) {
149             $selectedComp = array_pop($computers);
150             $transpiler = $f3->get('NPM_TRANSPILE');
151             $username = substr($parto[1]['enrollment_year'], 2) . $parto[1]['no_urut'];
152
153             if (array_key_exists($parto[1]['prodi_id'], $transpiler)) {
154                 $username = $transpiler[$parto[1]['prodi_id']] . $username;
155             } else {
156                 throw new InvalidArgumentException("Unknown prodi_id " . $parto[1]['prodi_id'] . " from " . $parto[0]);
157             }
158             $participant = new Participant();
159             $participant->copyfrom([
160                 "username" => $username,
161                 "npm" => $parto[0],
162                 "exam" => $ujian->_id,
163                 "computer" => $selectedComp
164             ]);
165
166             $participant->save();
167             $logger->notice("Populate finished!");
168         }
169     } catch (NestedValidationException $e) {
170         $logger->notice("Participant populate action is failed with Validation error: " + $e->getMessage());
171         throw \Helper\Ruler::transformToError($e);
172     } catch (InvalidArgumentException $e) {
173         $logger->notice("Participant populate action is failed with Bad Request error: " + $e->getMessage());
174         throw new Error("Invalid Input", $e->getMessage(), "X400", "Exception", 400);
175     }
176
177     // GENERATE THINGS.
178     return \View\Api::success($ujian);
179 }
180
181 /**
182 * Open answer slot for current exam and start the timer
183 *
184 * @param \Base $f3 FatFree instance
185 * @return void
186 */
187 public function post_item_start(\Base $f3)
188 {
189     $this->permission_check($this->permissionPrefix, AclItem::UPDATE);
190     $ujian = parent::getMentionedItem($f3);
191
192     $ujian->time_opened = time();
193     $ujian->timeEnded = strtotime("+" . $ujian->time_duration . " seconds");
194     $ujian->save();
195
196     $logger = $ujian->getLoggerInstance();
197     $logger->notice("Timer start via " . $f3->IP);
198
199     return \View\Api::success($ujian->cast());
200 }

```

```

204 /**
205  * Close answer slot for current exam and stop the timer
206 *
207 * @param \Base $f3
208 * @return void
209 */
210 public function post_item_close(\Base $f3)
211 {
212     $this->permission_check($this->permissionPrefix, AclItem::UPDATE);
213     $ujian = parent::getMentionedItem($f3);

214     $ujian->time_ended = time();
215     $ujian->save();

216     $logger = $ujian->getLoggerInstance();
217     $logger->notice("Timer (forced) stop via " . $f3->IP);

218     return \View\Api::success($ujian->cast());
219 }

220 /**
221  * Remove open/closed status for answer slot & timer on current exam.
222 *
223 * @param \Base $f3
224 * @return void
225 */
226 public function delete_item_start(\Base $f3)
227 {
228     $this->permission_check($this->permissionPrefix, AclItem::UPDATE);
229     $ujian = parent::getMentionedItem($f3);

230     $ujian->time_start = time();
231     $ujian->time_opened = null;
232     $ujian->time_ended = null;
233     $ujian->save();

234     $logger = $ujian->getLoggerInstance();
235     $logger->warn("Timer reset via " . $f3->IP);

236     return \View\Api::success($ujian->cast());
237 }

238 /**
239  * Generate deployment script for current exam
240 *
241 * @param \Base $f3
242 * @return void
243 */
244 public function get_item_script(\Base $f3)
245 {
246     $this->permission_check($this->permissionPrefix, AclItem::READ);
247     $ujian = parent::getMentionedItem($f3);
248     $filename = $ujian->lecture->lecture_code . " - " . $ujian->lecture->name;

249     if ($ujian->shift != 0) {
250         $filename .= " Shift " . $ujian->shift;
251     }

252     $filename .= ".zip";

253     $logger = $ujian->getLoggerInstance();
254     $logger->info("Deployment script download via " . $f3->IP);

255     header("X-Filename: $filename");
256     \Web::instance()->send(
257         \Service\BatGenerator::instance()->generate($ujian),
258         null,
259         0,
260         true,
261         $filename
262     );
263     \Service\BatGenerator::instance()->clean_zip();
264 }

265 /**
266  * Zip answers for current exam and return it as attachment
267 *
268 * @param \Base $f3
269 * @return void
270 */
271 public function get_item_answers(\Base $f3)
272 {
273     // zipping it all.
274     $this->permission_check($this->permissionPrefix, AclItem::READ);
275     $ujian = parent::getMentionedItem($f3);

276     $zipname = AnswerZipper::zipExam($ujian);
277     $filename = AnswerZipper::getProperFilename($ujian);

278     $filename .= ".zip";

279     $logger = $ujian->getLoggerInstance();
280     $logger->warn("Answer download request via " . $f3->IP);

281     header("X-Filename: $filename");
282     \Web::instance()->send(
283         $zipname,
284         null,
285         0,
286         true,
287         $filename
288     );
289 }

```

```

302
303     0,
304     true,
305     $filename
306   );
307   if (is_file($zipname)) {
308     unlink($zipname);
309   }
310
311 /**
312 * Move lists of participants to specific computer
313 *
314 * @param \Base $f3
315 * @return void
316 */
317 public function post_item_move(\Base $f3)
318 {
319   $this->permission_check($this->permissionPrefix, AclItem::UPDATE);
320   $ujian = parent::getMentionedItem($f3);
321
322   $logger = $ujian->getLoggerInstance();
323   $logger->notice("Move sequence requested via" . $f3->IP);
324
325   // get all position and it's destination.
326   try {
327     // id from param must be exists
328     $validator = v::key("lists", v::each(
329       v::key("participant", v::notOptional()->mustExists("\Model\Ujian\Participant", 'id', 'deleted_on'))
330       ->key("to", v::notOptional()->mustExists("\Model\Ujian\Computer", 'id'))
331     )));
332     $validator->assert($f3->POST);
333
334     $lists = $f3->POST['lists'];
335
336     $migrationList = [];
337
338     foreach ($lists as $list) {
339       $computer = new \Model\Ujian\Computer();
340       $participant = new \Model\Ujian\Participant();
341       $computer->load(["id = ?", $list['to']]);
342       $participant->load(["id = ?", $list['participant']]);
343       if ($participant->exam->id != $ujian->id) {
344         throw new \InvalidArgumentException("Invalid participant on $list[participant]");
345       }
346
347       $logger->info("[MOVE] Participant " . $participant->npm . " #" . $participant->id . " is being moved from "
348                     . $participant->computer->name . " to " . $computer->name);
349
350       $migrationList[] = [
351         "p" => $participant,
352         "c_before" => $participant->computer,
353         "c_after" => $computer,
354       ];
355       $participant->computer = $computer->id;
356       // $participant->save();
357     }
358
359     foreach ($migrationList as $mig) {
360       $mig['p']->save();
361     }
362
363     $logger->info("[MOVE] DB has been updated.");
364
365     // generate script:
366     $filename = "Migrasi " . $ujian->lecture->lecture_code . " - " . $ujian->lecture->name;
367
368     if ($ujian->shift != 0) {
369       $filename .= " Shift " . $ujian->shift;
370     }
371
372     $filename .= date("Y-m-d H:i:s") . ".zip";
373
374     $logger->info("[MOVE] Move script has been made.");
375
376     header("X-Filename: $filename");
377     \Web::instance()->send(
378       \Service\BatGenerator::instance()->generateMigration($migrationList, $ujian),
379       null,
380       0,
381       true,
382       $filename
383     );
384     \Service\BatGenerator::instance()->clean_zip();
385   } catch (NestedValidationException $e) {
386     $logger->notice("Participant move action is failed with Validation error: " + $e->getMessage());
387     throw \Helper\Ruler::transformToError($e);
388   } catch (InvalidArgumentException $e) {
389     $logger->notice("Participant move action is failed with Bad Request error: " + $e->getMessage());
390     throw new Error("Invalid Input", $e->getMessage(), "X400", "Exception", 400);
391   }
392 }
393 }
```

Listing B.21 examreport.php

1 <?php

```

3 namespace Controller\Api\Manage;
5 use Controller\CRUDBase;
6 use Helper\ExamReport as HelperExamReport;
7 use Model\System\AclItem;
8 use View\Api;
10 class ExamReport extends CRUDBase
11 {
12     protected $permissionPrefix = "manage-ujian-exam-report";
13     protected $model = "\Model\Ujian\ExamReport";
15     public function post_item_forcesend($f3)
16     {
17         $this->permission_check($this->permissionPrefix, AclItem::UPDATE);
18         $examreport = parent::getMentionedItem($f3);
19
20         $log = $examreport->exam->getLoggerInstance();
21         $log->notice("[MANUAL] Autoreport is sent to " . $examreport->tos . " with id ExamReport#" . $examreport->_id);
22
23         // push validity periode of the token
24         // just to make sure things gone right
25         $examreport->pushValidity();
26         $examreport->sent_on = time();
27         $examreport->save();
28
29         HelperExamReport::sendout($examreport);
30
31     }
32 }
33 }
```

Listing B.22 account.php

```

1 <?php
3 namespace Controller\Api\Internal;
5 use \Controller\Api;
6 use Model\System\User;
8 class Account extends Api\Base
9 {
11     public function get_me($f3)
12     {
13         parent::precheck_must_login("You need to log in before you're able to do that.");
14         $user = User::getSession(false);
15
16         return \View\Api::success(["profile" => $user->cast()]);
17     }
19     public function post_me($f3)
20     {
21         parent::precheck_must_login("You need to log in before you're able to do that.");
22         $user = User::getSession(false);
23
24         if ($f3->exists('POST.password') && (!$f3->exists('POST.old_password') || !$f3->exists('POST.password_confirm'))) {
25             throw new \Model\Error(
26                 "Unable to update profile",
27                 "Requested password change, but doesn't provide old password for confirmation or, you might not enter the
28                 password confirmation correctly.",
29                 "AA01" //TODO: Delegate Error code
30             );
31
32         if ($f3->exists('POST.password') && $f3->get('POST.password') != $f3->get('POST.password_confirm')) {
33             throw new \Model\Error(
34                 "Unable to update profile",
35                 "Your confirmation password is not identical. Please try again.",
36                 "AA02" //TODO: Delegate Error code
37             );
38
39         //password is ok. all save to go.
40         $user->copyfrom(array_intersect_key($f3->get("POST"), array_flip([
41             'username',
42             'password',
43             'email'
44         ])));
45         $user->save();
46
47         return \View\Api::success(["profile" => $user->cast()]);
48     }
49
50     public function post_login($f3)
51     {
52         //AppID Check
53         if (!$f3->exists("POST.username") || !$f3->exists("POST.password")) {
54             throw new \Model\Error("Incomplete Request", "Username and Password must be filled.", "ALP01", "");
55         }
56
57         $user = null;
58         try {
59             $user = User::login($f3->get('POST.username'), $f3->get('POST.password'));
60         } catch (\Exception $e) {
61
62     }
```

```

62         throw new \Model\Error("Bad credential", "You might've entered wrong username/password.", "ALP02", "", 400, null,
63             $e);
64     }
65 
66     return \View\Api::success(["id_token" => $user->generateToken(), "profile" => $user->cast()]);
67 }
68 /**
69 * Account creations
70 */
71 
72 public function get_index($f3)
73 {
74     parent::precheck_must_login("You need to log in before you're able to do that.");
75 
76     //add search constraint disini
77 
78     $user = new User();
79     $users = $user->find(["1"]);
80 
81     $users = array_map(function ($data) {
82         return $data->cast();
83     }, (array) $users);
84     return \View\Api::success($users);
85 }
86 
87 public function post_index($f3)
88 {
89     parent::precheck_must_login("You need to log in before you're able to do that.");
90     if (
91         !$f3->exists("POST.username") ||
92         !$f3->exists("POST.password_confirm") ||
93         !$f3->exists("POST.password") ||
94         !$f3->exists("POST.email")
95     ) {
96         throw new \Model\Error(
97             "Uncompleted field",
98             "All fields must be filled.",
99             "AA03" //TODO: Delegate Error code
100        );
101    }
102 
103    if ($f3->exists('POST.password') && $f3->get('POST.password') != $f3->get('POST.password_confirm')) {
104        throw new \Model\Error(
105            "Unable to update profile",
106            "Your confirmation password is not identical. Please try again.",
107            "AA04" //TODO: Delegate Error code
108        );
109    }
110 
111    $user = new User();
112    $user->copyFrom(array_intersect($f3->get('POST'), array_flip([
113        'email',
114        'password',
115        'username'
116    ])));
117    $user->save();
118 
119    return \View\Api::success(['profile' => $user->cast()]);
120 }
121 
122 public function delete_index($f3)
123 {
124     parent::precheck_must_login("You need to log in before you're able to do that.");
125 
126     if (!$f3->exists('GET.id')) {
127         throw new \Model\Error("Uncompleted request.", "You need to provide it's id to remove someone.", 403);
128     }
129 
130     if (in_array($f3->get('GET.id'), [1, 2])) {
131         throw new \Model\Error(
132             "Cannot delete user.",
133             "You can't delete system users. This is required.",
134             "AA05" //TODO: Delegate Error code
135         );
136     }
137 
138     $user = User::getFromSession(false);
139     if ($f3->get('GET.id') == $user->id) {
140         throw new \Model\Error(
141             "Cannot delete user.",
142             "You can't delete your self.",
143             "AA06" //TODO: Delegate Error code
144         );
145     }
146 
147     $user->load(['id=?', $f3->get("GET.id")]);
148     if ($user->dry()) {
149         throw new \Model\Error(
150             "Cannot delete user.",
151             "User is nonexistent.",
152             "AA07" //TODO: Delegate Error code
153         );
154     }
155     $user->deleted_on = time(); //soft delete them;
156     $user->save();
157     return \View\Api::success('ok');
158 }

```

```

160     public function put_index($f3)
161     {
162         parent::precheck_must_login("You need to log in before you're able to do that.");
163
164         if (!$f3->exists('GET.id'))
165             throw new \Model\Error("Uncompleted request.", "You need to provide it's id to update someone.", 403);
166     }
167
168     if (in_array($f3->get('GET.id'), [1, 2])) {
169         throw new \Model\Error(
170             "Cannot update user.",
171             "You can't update or modify system users.",
172             "AA08" //TODO: Delegate Error code
173         );
174     }
175
176     $user = new User();
177     $user->load(['id=?', $f3->get("GET.id")]);
178     if ($user->dry()) {
179         throw new \Model\Error(
180             "Cannot update user.",
181             "User is nonexistent.",
182             "AA09" //TODO: Delegate Error code
183         );
184     }
185
186     $user->copyFrom(array_intersect($f3->get('POST'), array_flip([
187         'email',
188         'password',
189         'username'
190     ])));
191     $user->save();
192
193     return \View\Api::success(['profile' => $user->cast()]);
194 }
195 }
```

Listing B.23 feed.php

```

1 <?php
2 namespace Controller\Api\Internal;
3
4 class Feed {
5     public function post_mock($f3) {
6         // TODO: Buat testing
7         // cek komputer
8         // cek exam
9         // bikinin slot jawaban 2 biji
10    }
11 }
```

Listing B.24 iplogin.php

```

1 <?php
2
3 namespace Controller\Api\Manage;
4
5 use Controller\CRUDBase;
6
7 class IPLogin extends CRUDBase
8 {
9     protected $permissionPrefix = "manage-system-iplogin";
10    protected $model = "\Model\System\IPLogin";
11 }
```

Listing B.25 lecture.php

```

1 <?php
2
3 namespace Controller\Api\Manage;
4
5 use Controller\CRUDBase;
6
7 class Lecture extends CRUDBase
8 {
9     protected $permissionPrefix = "manage-ujian-lecture";
10    protected $model = "\Model\Ujian\Lecture";
11 }
```

Listing B.26 lectureperiod.php

```

1 <?php
2
3 namespace Controller\Api\Manage;
4
5 use Controller\CRUDBase;
6
7 class LecturePeriod extends CRUDBase
8 {
9     protected $permissionPrefix = "manage-ujian-lectureperiod";
```

```

10     protected $model = "\Model\Ujian\LecturePeriod";
11 }

```

Listing B.27 location.php

```

1 <?php
3 namespace Controller\Api\Manage;
5 use Controller\CRUDBase;
7 class Location extends CRUDBase
8 {
9     protected $permissionPrefix = "manage-ujian-location";
10    protected $model = "\Model\Ujian\Location";
11 }

```

Listing B.28 notification.php

```

1 <?php
3 namespace Controller\Api\Manage;
5 use Controller\CRUDBase;
6 use InvalidArgumentException;
7 use Model\Error;
8 use Model\System\AclItem;
9 use Model\Ujian\Notification as UjianNotification;
10 use Model\Ujian\Participant;
11 use Respect\Validation\Exceptions\NestedValidationException;
12 use Respect\Validation\Validator as v;
14 class Notification extends CRUDBase
15 {
16     protected $permissionPrefix = "manage-ujian-notification";
17     protected $model = "\Model\Ujian\Notification";
19     public function delete_item($f3, $directDelete = true)
20     {
21         parent::delete_item($f3, $directDelete);
22     }
25     /**
26      * List all object in a model.
27      * May support search and paging. But not sure.
28      *
29      * @param object $f3 FatFree Object
30      * @return void
31     */
32     public function get_index($f3)
33     {
34         // permission checkpoint
35         $this->permission_check($this->permissionPrefix, AclItem::READ);
37         $model = new $this->model;
38         if (in_array("deleted_on", $model->fields())) {
39             $models = $model->find(["deleted_on = ?", null]);
40         } else {
41             $models = $model->find();
42         }
43         if ($models === false) {
44             $models = [];
45         } else {
46             $models = array_map(function ($model) {
47                 return $model->cast(null, null, false);
48             }, $models);
49         }
50         return \View\Api::success($models);
51     }
54     /**
55      * Generate massive notification for participants.
56      *
57      *
58      * Protocol design (multiple?):
59      * - Create new exam
60      * - Create participant_id - password list??
61      * - Create new notifications
62      * - Hook them to participants?
63      *
64      *
65      * Other (single):
66      * - Create new notification via API
67      * - Add participant ids.
68      *
69     */
70     public function post_mass_gen($f3)
71     {
72         // template check?
73         $this->permission_check($this->permissionPrefix, AclItem::READ);
74         try {
75             // id from param must be exists
76             $validator = v::key("lists", v::each(

```

```

77         v::key("participant", v::notOptional()->mustExists("\\Model\\Ujian\\participant", 'id', 'deleted_on'))
78             ->key("username", v::notOptional())
79             ->key("password", v::notOptional())
80     ))
81     ->key("url", v::notOptional());
82 $validator->assert($f3->POST);

85     // template is from notifications/password.html
86     $service = $f3->POST['url'];
87     $lists = $f3->POST['lists'];
88     foreach ($lists as $l) {
89         $participant = new Participant();
90         $participant->load(['id?=' . $l['participant']]);
91         $notification = new UjianNotification();
92         $notification->copyfrom([
93             "title" => "Akun untuk " . $service,
94             "type" => "credential",
95             "description" => \Template::instance()->render("notifications/password.html", 'text/html', [
96                 "notif" => [
97                     "url" => $service,
98                     "username" => $l['username'],
99                     "password" => $l['password'],
100                 ],
101             ],
102             "participants" => $participant,
103             "extras" => [
104                 "service" => $service,
105                 "username" => $l['username'],
106                 "password" => $l['password'],
107             ],
108         ]);
109         $notification->save();
110     }
111 }

113     return \View\Api::success([]);
114 } catch (NestedValidationException $e) {
115     throw \Helper\Ruler::transformToError($e);
116 } catch (InvalidArgumentException $e) {
117     throw new Error("Invalid Input", $e->getMessage(), "X400", "Exception", 400);
118 }
119 }
120 }
```

Listing B.29 participant.php

```

1 <?php
3 namespace Controller\Api\Manage;
5 use Controller\CRUDBase;
7 class Participant extends CRUDBase
8 {
9     protected $permissionPrefix = "manage-ujian-participant";
10    protected $model = "\\Model\\Ujian\\Participant";
11 }
```

Listing B.30 submission.php

```

1 <?php
3 namespace Controller\Api\Manage;
5 use Controller\CRUDBase;
7 class Submission extends CRUDBase
8 {
9     protected $permissionPrefix = "manage-ujian-submission";
10    protected $model = "\\Model\\Ujian\\Submission";
11 }
```

Listing B.31 user.php

```

1 <?php
3 namespace Controller\Api\Manage;
5 use Controller\CRUDBase;
7 class User extends CRUDBase
8 {
9     protected $permissionPrefix = "manage-system-user";
10    protected $model = "\\Model\\System\\User";
11 }
```

Listing B.32 ping.php

```
| 1 <?php
```

```

2 Namespace Controller\Api;
4 class Ping extends \Prefab {
5     public function get_index($f3){
6         return \View\Api::success([]);
7     }
8 }
```

Listing B.33 auth.php

```

1 <?php
3 namespace Controller\Api\System;
5 use Model\Error;
6 use Respect\Validation\Exceptions\NestedValidationException;
7 use \Respect\Validation\Validator as v;
9 class Auth extends \Controller\Api\Base
10 {
11     /**
12      * Login with IP
13      *
14      * @param \Base $f3
15      * @return void
16      */
17     public function post_iplogin($f3)
18     {
19         $iplogin = new \Model\System\IPLogin();
20         $iplogin->load(["ip" => ?, $f3->IP]);
21         if ($iplogin->loaded() == 0) {
22             throw new Error(
23                 "IP Login Rerequest Rejected for Obvious Reason",
24                 "IP $f3->IP is not registered to the system.",
25                 400,
26                 "Unmet conditions"
27             );
28         }
29         $token = (string)$iplogin->user->generateToken("ip", $iplogin->_id);
30         $locations = $iplogin->locations->castAll();
31
32         return \View\Api::success([
33             "id_token" => $token,
34             "profile" => $iplogin->user->cast(),
35             "locations" => $locations
36         ]);
37     }
38
39     /**
40      * Login with the normal Admin & Username thingy.
41      *
42      * @param \Base $f3
43      * @return void
44      */
45     public function post_login($f3)
46     {
47         $validation = v::key("username", v::notOptional())
48             ->key("password", v::notOptional());
49         try {
50             $validation->assert($f3->POST);
51         } catch (NestedValidationException $e) {
52             throw \Helper\Ruler::transformToError($e);
53         }
54         $user = new \Model\System\User();
55
56         try {
57             if (!$user = $user->login($f3->POST['username'], $f3->POST['password'])) {
58                 throw new Error("Login failed", "No username/password combo matching.", "400", "Validation Violation");
59             }
60         } catch (\Exception $e) {
61             throw new Error("Login failed", $e->getMessage(), "400", "Validation Violation");
62         }
63
64         return \View\Api::success([
65             "id_token" => (string) $user->generateToken(),
66             "profile" => $user->cast()
67         ]);
68     }
69 }
70 }
```

Listing B.34 user.php

```

1 <?php
3 namespace Controller\Api\System;
5 use Controller\Api\Base;
7 class User extends Base
8 {
9     public function get_me($f3)
10    {
11        parent::precheck_must_login("You need to log in before you're able to do that.");
12        $user = \Model\System\User::getFromHTTPHeader();
```

```

14     $response = ["profile" => $user->cast()];
15
16     $iplogin = \Model\System\User::getIPLoginHTTPHeader();
17     if ($iplogin !== null) {
18         $response['locations'] = $iplogin->locations->castAll();
19     }
20
21     return \View\Api::success($response);
22 }
23
24 public function post_me($f3)
25 {
26     parent::precheck_must_login("You need to log in before you're able to do that.");
27     $user = \Model\System\User::getFromHTTPHeader();
28
29     if ($f3->exists('POST.password') && (!$f3->exists('POST.old_password') || !$f3->exists('POST.password_confirm'))) {
30         throw new \Model\Error(
31             "Unable to update profile",
32             "Requested password change, but doesn't provide old password for confirmation or, you might not enter the
33                 password confirmation correctly.",
34             "AA01" //TODO: Delegate Error code
35         );
36
37     if ($f3->exists('POST.password') && $f3->get('POST.password') != $f3->get('POST.password_confirm')) {
38         throw new \Model\Error(
39             "Unable to update profile",
40             "Your confirmation password is not identical. Please try again.",
41             "AA02" //TODO: Delegate Error code
42         );
43
44     //password is ok. all save to go.
45     $user->copyfrom(array_intersect_key($f3->get("POST"), array_flip([
46         'username',
47         'password',
48         'email'
49     ])));
50     $user->save();
51
52     return \View\Api::success(["profile" => $user->cast()]);
53 }
54 }
55 }
```

Listing B.35 app.php

```

1 <?php
2 namespace Controller;
3
4 class App extends \Prefab
5 {
6     public function get_serve($f3)
7     {
8         return \View\ReactApp::render();
9     }
10 }
```

Listing B.36 crudbase.php

```

1 <?php
2 namespace Controller;
3
4 use Model\Error;
5 use InvalidArgumentException;
6 use Model\System\AclItem;
7 use Respect\Validation\Exceptions\NestedValidationException;
8 use Respect\Validation\Validator as v;
9
10 abstract class CRUDBase extends Api\Base
11 {
12
13     protected $publicPermission = \Model\System\AclItem::NONE;
14     protected $permissionPrefix = "";
15     protected $model = null;
16
17     public function globalFilterTrigger($model): void
18     {
19     }
20
21
22     /**
23      * Checks various things before we get started. If we check that this API
24      * is not for public to use, then we will need to check the user to be logged in
25      * with proper authentication.
26      *
27      * @param [type] $f3
28      * @return void
29      */
30     public function beforeroute($f3)
31     {
32         parent::beforeroute($f3);
33         v::with("\Helper\Rules");
34         if ($this->publicPermission == \Model\System\AclItem::NONE) {
```

```

35         return \Helper\Ijin::userMustPresent();
36     }
37 }
38 /**
39 * Checks the permission of current user, if no user logged in,
40 * we'll check from public permission.
41 *
42 * @param string $what ACL Name
43 * @param string $mode CRUD mode selection
44 * @return void
45 */
46 protected function permission_check($what, $mode)
47 {
48     $user = \Model\System\User::getFromHTTPHeader();
49     if (!$user && ($this->publicPermission & $mode) == 0) {
50         throw new Error("Bad Auth", "You don't have permission to do that.", "403", "Bad Permission", 403);
51     }
52 }
53 return \Helper\Ijin::mustHave($what, $mode);
54 }
55 */
56 /**
57 * List all object in a model.
58 * May support search and paging. But not sure.
59 *
60 * @param object $f3 FatFree Object
61 * @return void
62 */
63 public function get_index($f3)
64 {
65     // permission checkpoint
66     $this->permission_check($this->permissionPrefix, AclItem::READ);
67
68     $model = new $this->model;
69     $this->globalFilterTrigger($model);
70     if (in_array("deleted_on", $model->fields())) {
71         $models = $model->find(["deleted_on = ?"], null);
72     } else {
73         $models = $model->find();
74     }
75     if ($models === false) {
76         $models = [];
77     } else {
78         $models = $models->castAll();
79     }
80     return \View\Api::success($models);
81 }
82 */
83 protected function getMentionedItem($f3)
84 {
85     // id from param must be exists
86     $validator = v::key("id", v::notOptional()->mustExists($this->model, 'id', 'deleted_on'));
87     try {
88         $validator->assert($f3->PARAMS);
89     } catch (NestedValidationException $e) {
90         throw \Helper\Ruler::transformToError($e);
91     }
92     $model = new $this->model;
93     $this->globalFilterTrigger($model);
94     $model->load(["id = ?", $f3->PARAMS['id']]);
95
96     if ($model->dry()) {
97         throw new Error("Object not found", "Object cannot be found", "HTTP404", "Global Validation", 404);
98     }
99
100    return $model;
101 }
102 */
103 /**
104 * Get information of an object, with given ID.
105 *
106 * @param object $f3
107 * @return void
108 */
109 public function get_item($f3)
110 {
111     // permission checkpoint
112     $this->permission_check($this->permissionPrefix, AclItem::READ);
113
114     // id from param must be exists
115     $validator = v::key("id", v::notOptional()->mustExists($this->model, 'id', 'deleted_on'));
116     try {
117         $validator->assert($f3->PARAMS);
118     } catch (NestedValidationException $e) {
119         throw \Helper\Ruler::transformToError($e);
120     }
121     $model = new $this->model;
122     $this->globalFilterTrigger($model);
123     $model->load(["id = ?", $f3->PARAMS['id']]);
124
125     if ($model->dry()) {
126         throw new Error("Object not found", "Object cannot be found", "HTTP404", "Global Validation", 404);
127     }
128
129     return \View\Api::success($model->cast());
130 }
131 */
132 /**

```

```

134     * Create an item with given object.
135     *
136     * @param object $f3 FatFree Object
137     * @return void
138     */
139    public function post_index($f3)
140    {
141        // permission checkpoint
142        $this->permission_check($this->permissionPrefix, AclItem::CREATE);
143
144        // create new object of model
145        $obj = [];
146        try {
147            $model = new $this->model;
148            $model->_validate($f3->POST);
149            $model->copyfromwithfilter($f3->POST, true);
150            $model->save();
151            $obj = $model->cast();
152        } catch (NestedValidationException $e) {
153            throw \Helper\Ruler::transformToError($e);
154        } catch (InvalidArgumentException $e) {
155            throw new Error(
156                "Bad Request",
157                $e->getMessage(),
158                "400",
159                "Precheck Validation",
160                405
161            );
162        }
163        return \View\Api::success($obj);
164    }
165
166    /**
167     * Update an item with given ID.
168     *
169     * @param object $f3 FatFree Object
170     * @return void
171     */
172    public function put_item($f3)
173    {
174        // permission checkpoint
175        $this->permission_check($this->permissionPrefix, AclItem::UPDATE);
176
177        // create new object of model
178        $validator = v::key("id", v::notOptional()->mustExists($this->model, 'id', 'deleted_on'));
179        $obj = [];
180        try {
181            $validator->assert($f3->PARAMS);
182            $model = new $this->model;
183            $model->validate($f3->POST);
184            $this->globalFilterTrigger($model);
185            $model->load(["id=?", $f3->PARAMS['id']]);
186            if ($model->dry()) {
187                throw new InvalidArgumentException("Global validation failed");
188            }
189            $model->copyfromwithfilter(array_diff_key($f3->POST, array_reverse(['id', '_id'])), true);
190            $model->save();
191            $obj = $model->cast();
192        } catch (NestedValidationException $e) {
193            throw \Helper\Ruler::transformToError($e);
194        } catch (InvalidArgumentException $e) {
195            throw new Error(
196                "Bad Request",
197                $e->getMessage(),
198                "400",
199                "Precheck Validation",
200                405
201            );
202        }
203        return \View\Api::success($obj);
204    }
205
206    /**
207     * Delete an Item with given Object ID
208     *
209     * @param object $f3 FatFree Object
210     * @return void
211     */
212    public function delete_item($f3, $directDelete = false)
213    {
214        // permission checkpoint
215        $this->permission_check($this->permissionPrefix, AclItem::DELETE);
216
217        // id from param must be exists
218        $validator = v::key("id", v::notOptional()->mustExists($this->model, 'id', 'deleted_on'));
219        try {
220            $validator->assert($f3->PARAMS);
221        } catch (NestedValidationException $e) {
222            throw \Helper\Ruler::transformToError($e);
223        }
224        $model = new $this->model;
225        if (!in_array("deleted_on", $model->fields()) && !$directDelete) {
226            throw new Error(
227                "Method not allowed",
228                "Object cannot be deleted",
229                "405",
230                "Precheck Validation",
231                405
232            );

```

```

233     }
234
235     $this->globalFilterTrigger($model);
236     $model->load(["id = ?", $f3->PARAMS['id']]);
237     if ($model->dry()) {
238         throw new Error(
239             "Object not Found",
240             "Object cannot be deleted",
241             "404",
242             "Precheck Validation",
243             404
244         );
245     }
246     if ($directDelete) {
247         $model->erase();
248     } else {
249         $model->deleted_on = time();
250         $model->save();
251     }
252
253     return \View\Api::success($model->cast());
254 }
255 }
```

Listing B.37 autoreport.php

```

1 <?php
3 namespace CronJob;
5 use Helper\ExamReport as HelperExamReport;
6 use Model\Ujian\ExamReport;
8 class AutoReport extends \Prefab
9 {
10     public function exam($f3)
11     {
12         // fetch all exam with sent report is null.
13         $examreport = new ExamReport();
14         $examreport->has("exam", ["time_ended < ?", date("Y-m-d H:i:s", strtotime("-15 minutes"))]); // TODO: Move fine tuner
15         $exams = $examreport->find(["sent_on = ?", null]);
16         if ($exams === false) {
17             $exams = [];
18         }
19         foreach ($exams as $examr) {
20             $log = $examr->exam->getLoggerInstance();
21             $log->notice("[CRONJOB] Autoreport is sent to " . $examr->tos . " with id ExamReport#" . $examr->_id);
22
23             $examr->sent_on = time();
24             $examr->pushValidity();
25
26             HelperExamReport::sendout($examr);
27         }
28     }
29 }
```

Listing B.38 answerzipper.php

```

1 <?php
3 namespace Helper;
5 use Model\Ujian\Exam;
6 use Model\Ujian\Submission;
8 class AnswerZipper extends \Prefab
9 {
10     public static function zipExam(Exam $exam): string
11     {
12         $submission = new Submission();
13         $submission->has("answer_slot", ["deleted_on = ?", null]);
14         $submission->has("participant.exam", ["id=?", $exam->_id]);
15         $submissions = $submission->find();
16
17         if (!is_array($submissions)) {
18             $submissions = [];
19         }
20
21         // starting to packing things.
22         $zip = new \ZipArchive();
23         $zipname = \F3::get('TEMP') . DIRECTORY_SEPARATOR . $exam->exam_uniqcode . ".zip";
24         $zip->open($zipname, \ZipArchive::CREATE | \ZipArchive::OVERWRITE);
25
26         $zip->addFromString("_examInfo.json", json_encode($exam->cast()));
27
28         foreach ($submissions as $sub) {
29             $foldername = $sub->participant->username;
30             $foldername .= DIRECTORY_SEPARATOR . $sub->answer_slot->simulateFormat($sub->participant);
31
32             $zip->addFromString($foldername, file_get_contents($sub->getFullPath()));
33         }
34         $zip->close();
35
36         return $zipname;
37 }
```

```

37     }

40     public static function getProperFilename(Exam $exam, bool $withDate = true): string
41     {
42         $filename = $exam->lecture->lecture_code . " " . $exam->lecture->name;
43         if ($exam->shift) {
44             $filename .= " Shift " . $exam->shift;
45         }

47         if ($withDate) {
48             $filename .= " -- " . date("Y-m-d");
49         }
50     }
51     return $filename;
52 }
53 }
```

Listing B.39 `emailer.php`

```

1 <?php
2
3 namespace Helper;
4
5 use PHPMailer\PHPMailer\PHPMailer;
6
7 class Emailer extends \Prefab
8 {
9     /**
10      * Constructs fully configured PHPMailer instance, ready to use.
11      *
12      * @return PHPMailer
13      */
14     public static function getEmailInstance(): PHPMailer
15     {
16
17         $mailer = new PHPMailer(true);
18         $mailer->isSMTP();
19
20         $f3 = \Base::instance();
21         // setting up credential
22         $mailer->Host = $f3->get("SMTP.host");
23         $mailer->SMTPAuth = true;
24         $mailer->Username = $f3->get("SMTP.username");
25         $mailer->Password = $f3->get("SMTP.password");
26         $mailer->Port = intval($f3->get("SMTP.port"));
27
28         if ($f3->exists("SMTP.from")) {
29             $mailer->From = $f3->get("SMTP.from");
30         }
31         if ($f3->exists("SMTP.replyto")) {
32             $mailer->addReplyTo($f3->get("SMTP.replyto"));
33         }
34
35     }
36 }
37 }
```

Listing B.40 `examreport.php`

```

1 <?php
2
3 namespace Helper;
4
5 use Model\Ujian\ExamReport as UjianExamReport;
6
7 class ExamReport extends \Prefab
8 {
9
10    public static function generateReportContent(UjianExamReport $examReport): string
11    {
12        $f3 = \Base::instance();
13        // generate link before sending email to leturer.
14        $link = [$f3->get("BASE_URL_PUBLIC"), "autonomus", "exam-extract", $examReport->token];
15        $link = implode('/', array_map(function ($parts) {
16            return trim($parts, "\\\\");
17        }, $link));
18
19        $f3->mset([
20            "link" => $link,
21            "exam" => $examReport->exam,
22            "examreport" => $examReport
23        ], "email.");
24
25        return \Template::instance()->render("emails/exam-finished-lecturer.html");
26    }
27
28    public static function sendout(UjianExamReport $examReport)
29    {
30        $mailer = \Helper\Emailer::getEmailInstance();
31        $mailer->isHTML();
32        foreach (explode(",", $examReport->tos) as $recipient) {
33            $mailer->addAddress($recipient);
34        }
35    }
36 }
```

```

35     $mailer->Subject = "[Oxam] Berkas ujian " . $examReport->exam->lecture->lecture_code . " - " . $examReport->exam->
36     lecture->name . " tersedia untuk diunduh";
37     $mailer->Body = self::generateReportContent($examReport);
38 }
39 }
```

Listing B.41 mustexistsexception.php

```

1 <?php
2 namespace Helper\Exceptions;
3
4 use Respect\Validation\Exceptions\ValidationException;
5
6 class MustExistsException extends ValidationException {
7     public static $defaultTemplates = [
8         self::MODE_DEFAULT => [
9             self::STANDARD => '{{name}} must exists',
10        ],
11        self::MODE_NEGATIVE => [
12            self::STANDARD => '{{name}} must not exists',
13        ]
14    ];
15 }
```

Listing B.42 ijin.php

```

1 <?php
2 namespace Helper;
3
4 use Model\System\User;
5 use Model\Error;
6 use Model\System\AclItem;
7
8 class Ijin {
9     protected static function isUserPresent() {
10         if(\Model\System\User::getFromHTTPHeader())
11             return true;
12
13         if(!\F3::instance()->exists('SYSTEM.USER'))
14             return false;
15
16         if(!\F3::instance()->get('SYSTEM.USER'))
17             return false;
18
19         return true;
20     }
21     public static function userMustPresent() {
22         if(!self::isUserPresent()) {
23             throw new Error(
24                 "Bad Auth",
25                 "You need to login to do that.",
26                 "403",
27                 "No Authless Permitted",
28                 403
29             );
30         }
31     }
32
33     protected static function checkIfUserHave($acl_codename, $level) {
34         self::userMustPresent();
35         $user = \F3::instance()->get('SYSTEM.USER');
36
37         if(!$user->acl)
38             return false;
39
40         $aclitem = new AclItem();
41         $aclitem->load(['acl=? and codename=?', $user->acl->_id, $acl_codename]);
42
43         if(!$aclitem->loaded())
44             return false;
45
46         return (($level & intval($aclitem->permission)) == $level);
47     }
48
49     public static function mustHave($acl_codename, $level) {
50         if(!self::checkIfUserHave($acl_codename, $level)){
51             throw new Error(
52                 "Bad Auth",
53                 "You don't have permission to do that. Please check if you have that kind of permission on " . $acl_codename
54                 . " with level " . $level . ".",
55                 "403",
56                 "Bad Permission",
57                 403
58             );
59         }
60     }
61 }
```

Listing B.43 preapi.php

```
| 1 <?php
```

```

3 namespace Helper;
5 class PreAPI
6 {
7     public static function doTheThing()
8     {
9         //auto transform when post/put content type is json.
10        if (in_array(strtolower(\F3::instance()->VERB), ["put", "post"])) {
11            if ($strpos(\F3::instance()->HEADERS['Content-Type'], "application/json") > -1) {
12                \F3::instance()->POST = array_merge(\F3::instance()->POST, json_decode(\F3::instance()->BODY, true));
13            }
14            if (\F3::instance()->HEADERS['Content-Type'] == "application/x-www-form-urlencoded") {
15                $res = [];
16                parse_str(\F3::instance()->BODY, $res);
17                \F3::instance()->POST = array_merge(\F3::instance()->POST, $res);
18            }
19        }
20    }
21 }

```

Listing B.44 ruler.php

```

1 <?php
2 namespace Helper;
4 use Respect\Validation\Exceptions\NestedValidationException;
5 use Model\Error;
7 class Ruler {
8     public static function transformToError(NestedValidationException $e) {
9         $messages = $e->getMessages();
11        $messages = implode(", ", $messages);
12        return new Error(
13            "Bad Request",
14            $messages,
15            "400",
16            "Param not valid", 400
17        );
18    }
19 }

```

Listing B.45 mustexists.php

```

1 <?php
3 namespace Helper\Rules;
5 use Respect\Validation\Rules\AbstractRule;
7 class MustExists extends AbstractRule
8 {
10    protected $modelName, $fieldName, $deleted_on_field;
12    public function __construct($modelName, $fieldName = 'id', $deleted_on_field = null)
13    {
14        $this->modelName = $modelName;
15        $this->fieldName = $fieldName;
16        $this->deleted_on_field = $deleted_on_field;
17    }
19    public function validate($input)
20    {
21        $model = new $this->modelName;
22        $id = $input;
23        if ($this->fieldName == null) {
24            if (is_array($input) && array_key_exists('_id', $input)) {
25                $id = $input->id;
26                $this->fieldName = "_id";
27            }
28            if (is_array($input) && array_key_exists('id', $input)) {
29                $id = $input->id;
30                $this->fieldName = "id";
31            }
32        } else {
33            if (is_array($input) && array_key_exists($this->fieldName, $input)) {
34                $id = $input->{$this->fieldName};
35            }
36        }
38        if ($this->deleted_on_field && in_array($this->deleted_on_field, $model->fields())) {
39            $model->load([$this->fieldName . ' = ? AND ' . $this->deleted_on_field . ' = ?', $id, null]);
40        } else {
41            $model->load([$this->fieldName . ' = ?', $id]);
42        }
43        return $model->loaded();
44    }
45 }

```

Listing B.46 error.php

```

1 <?php
2 namespace Model;
3
4 class Error extends \Exception{
5     private
6         $reason,
7         $error_code,
8         $title,
9         $description,
10        $exception,
11        $prevErrorModel,
12        $http_code;
13
14     public function __construct(
15         $title,
16         $description,
17         $error_code,
18         $reason = "Unknown error",
19         $http_code = 400,
20         ErrorModel $prevErrorModel = null,
21         \Throwable $exception = null) {
22
23         parent::__construct($title, $http_code, $exception);
24         $this->title = $title;
25         $this->description = $description;
26         $this->reason = $reason;
27         $this->error_code = $error_code;
28         $this->prevErrorModel = $prevErrorModel;
29         $this->exception = $exception;
30         $this->http_code = $http_code;
31     }
32
33     public function get_http_status() {
34         return $this->http_code;
35     }
36
37     public function serve_exception(){
38         return [
39             "title" => $this->title,
40             "description" => $this->description,
41             "reason" => $this->reason,
42             "error_code" => $this->error_code,
43             "error_stack" => $this->prevErrorModel->serve_exception,
44             "exception" => (array)$this->exception
45         ];
46     }
47 }

```

Listing B.47 modelbase.php

```

1 <?php
2 namespace Model;
3
4 use InvalidArgumentException;
5 use Respect\Validation\Validator as v;
6
7 class ModelBase extends \DB\Cortex
8 {
9     static $validators;
10    public function __construct($db = NULL, $table = NULL, $fluid = NULL, $ttl = 0)
11    {
12        parent::__construct($db, $table, $fluid, $ttl);
13        v::with("\Helper\Rules");
14
15        if (!self::$validators)
16            self::$validators = [
17                \DB\SQL\Schema::DT_BOOL      => v::oneOf(v::boolType(), v::intVal()->boolVal()),
18                \DB\SQL\Schema::DT_BOOLEAN   => v::oneOf(v::boolType(), v::intVal()->boolVal()),
19                \DB\SQL\Schema::DT_INT1      => v::intVal(),
20                \DB\SQL\Schema::DT_TINYINT   => v::intVal(),
21                \DB\SQL\Schema::DT_INT2      => v::intVal(),
22                \DB\SQL\Schema::DT_SMALLINT  => v::intVal(),
23                \DB\SQL\Schema::DT_INT4      => v::intVal(),
24                \DB\SQL\Schema::DT_INT      => v::intVal(),
25                \DB\SQL\Schema::DT_INT8     => v::intVal(),
26                \DB\SQL\Schema::DT_BIGINT    => v::intVal(),
27                \DB\SQL\Schema::DT_FLOAT     => v::floatVal(),
28                \DB\SQL\Schema::DT_DOUBLE    => v::floatVal(),
29                \DB\SQL\Schema::DT_DECIMAL   => v::floatVal(),
30                \DB\SQL\Schema::DT_VARCHAR128 => v::stringType(),
31                \DB\SQL\Schema::DT_VARCHAR256 => v::stringType(),
32                \DB\SQL\Schema::DT_VARCHAR512 => v::stringType(),
33                \DB\SQL\Schema::DT_TEXT      => v::stringType(),
34                \DB\SQL\Schema::DT_LONGTEXT   => v::stringType(),
35                \DB\SQL\Schema::DT_DATE      => v::oneOf(v::date("Y-m-d H:i:s"), v::intType()),
36                \DB\SQL\Schema::DT_DATETIME   => v::oneOf(v::date("Y-m-d H:i:s"), v::intType()),
37                \DB\SQL\Schema::DT_TIMESTAMP  => v::oneOf(v::date("Y-m-d H:i:s"), v::intType()),
38                \DB\SQL\Schema::DT_BLOB       => v::alwaysValid(),
39                \DB\SQL\Schema::DT_BINARY     => v::alwaysValid(),
40                self::DT_JSON => v::alwaysValid()
41            ];
42    }
43
44    public function copyfromwithfilter($data, $strict = false)
45    {
46        //filtering.
47    }

```

```

48     $allowed = [];
49     foreach ($this->fieldConf as $key => $setting) {
50         if (array_key_exists("_copyable", $setting) && $setting['_copyable']) {
51             $allowed[] = $key;
52         } else {
53             if ($strict && array_key_exists($key, $data)) {
54                 throw new InvalidArgumentException("$key should not be used.");
55             }
56         }
57     }
58
59     if ($strict && count($diff = array_diff_key($data, array_flip($allowed))) > 0) {
60         throw new InvalidArgumentException(implode(", ", array_keys($diff)) . " should not be used.");
61     }
62
63     return $this->copyfrom(array_intersect_key($data, array_flip($allowed)));
64 }
65
66 /**
67 * Validate all models for models.
68 *
69 * @param array $data
70 * @param string $mode
71 * @return true if validated, exception otherwise.
72 */
73 public function _validate($data)
74 {
75     $validator = [];
76     foreach ($this->fieldConf as $key => $setting) {
77         if (array_key_exists("nullable", $setting) && $setting['nullable']) {
78             continue;
79         }
80         if (array_key_exists("_copyable", $setting) && !$setting['_copyable']) {
81             continue;
82         }
83
84         $vali = v::notOptional();
85         if (array_key_exists("has-many", $setting)) {
86             $vali = v::arrayType()->each(v::intVal()->mustExists($setting["has-many"][0], 'id', 'deleted_on'));
87         } else if (array_key_exists($setting['type'], self::$validators)) {
88             $vali = self::$validators[$setting['type']];
89         } else {
90
91         }
92
93         $validator[] = v::key($key, $vali);
94     }
95
96     return v::allof(...$validator)->assert($data);
97 }
98 }
```

Listing B.48 acl.php

```

1 <?php
2 namespace Model\System;
3
4 class Acl extends \Model\ModelBase {
5     protected
6     $fieldConf = array(
7         'name'=>[
8             'type'=>\DB\SQL\Schema::DT_TEXT,
9             'nullable' => false,
10            'index' => false,
11            'unique' => false,
12            '_copyable' => true
13        ],
14        'items' => [
15            'has-many' => ['\Model\System\AclItem', 'acl']
16        ],
17
18        'deleted_on'=>[
19            'type'=>\DB\SQL\Schema::DT_DATETIME,
20            'nullable' => true,
21            'index' => false,
22            'unique' => false,
23        ],
24        'created_on'=>[
25            'type'=>\DB\SQL\Schema::DT_DATETIME,
26            'nullable' => true,
27            'index' => false,
28            'unique' => false,
29        ],
30        'updated_on'=>[
31            'type'=>\DB\SQL\Schema::DT_DATETIME,
32            'nullable' => true,
33            'index' => false,
34            'unique' => false,
35        ],
36    ),
37    $db = 'DB',
38    $table = 'system_acl';
39
40    public function set_deleted_on($date) {
41        return date("Y-m-d H:i:s", $date);
42    }
43 }
```

```

44     public function set_created_on($date) {
45         return date("Y-m-d H:i:s", $date);
46     }
47
48     public function set_updated_on($date) {
49         return date("Y-m-d H:i:s", $date);
50     }
51
52     public function save() {
53         if(!$this->created_on)
54             $this->created_on = time();
55         $this->updated_on = time();
56         return parent::save();
57     }
58
59     public function cast ($obj = NULL, $rel_depths = 1, $save_cast = true) {
60         $obj = parent::cast($obj, $rel_depths);
61         if(!$save_cast) {
62             return $obj;
63         } else {
64             unset($obj['items']);
65             return $obj;
66         }
67     }
68 }
```

Listing B.49 aclitem.php

```

1 <?php
2 namespace Model\System;
3
4 class AclItem extends \Model\ModelBase {
5     protected
6     $fieldConf = array(
7         'codename'=>[
8             'type'=>\DB\SQL\Schema::DT_TEXT,
9             'nullable' => false,
10            'index' => false,
11            'unique' => false,
12            '_copyable' => true
13        ],
14        'permission'=>[
15            'type'=>\DB\SQL\Schema::DT_INT,
16            'nullable' => false,
17            'index' => false,
18            'unique' => false,
19            '_copyable' => true
20        ],
21        'acl' => [
22            'belongs-to-one' => '\Model\System\Acl',
23            '_copyable' => true
24        ],
25    ),
26    $db = 'DB',
27    $table = 'system_acl_item';
28
29    const
30        NONE = 0,
31        CREATE = 1,
32        READ = 1 << 1,
33        UPDATE = 1 << 2,
34        DELETE = 1 << 3,
35        ALL = self::CREATE | self::READ | self::UPDATE | self::DELETE;
36 }
```

Listing B.50 iplogin.php

```

1 <?php
2 namespace Model\System;
3
4 class IPLLogin extends \Model\ModelBase
5 {
6     protected
7     $fieldConf = array(
8         'ip' => [
9             'type' => \DB\SQL\Schema::DT_TEXT,
10            'nullable' => false,
11            'index' => false,
12            'unique' => true,
13            '_copyable' => true
14        ],
15        'user' => [
16            'belongs-to-one' => '\Model\System\User',
17            'nullable' => false,
18            '_copyable' => true
19        ],
20        'locations' => [
21            'has-many' => ["\Model\Ujian\Location", "iplogins", "system_location_iplogin_link"],
22            '_copyable' => true
23        ],
24        'notes' => [
25            'type' => \DB\SQL\Schema::DT_TEXT,
26            'nullable' => false,
27            'index' => false,
28        ]
29    );
30 }
```

```

29         'unique' => false,
30         '_copyable' => true
31     ],
32
33     'created_on' => [
34         'type' => \DB\SQL\Schema::DT_DATETIME,
35         'nullable' => true,
36         'index' => false,
37         'unique' => false,
38     ],
39     'updated_on' => [
40         'type' => \DB\SQL\Schema::DT_DATETIME,
41         'nullable' => true,
42         'index' => false,
43         'unique' => false,
44     ],
45 ),
46 $db = 'DB',
47 $table = 'system_iplogin';
48
49 public function save()
50 {
51     if (!$this->created_on)
52         $this->created_on = time();
53     $this->updated_on = time();
54     return parent::save();
55 }
56
57 public function set_created_on($date) {
58     return date("Y-m-d H:i:s", $date);
59 }
60
61 public function set_updated_on($date) {
62     return date("Y-m-d H:i:s", $date);
63 }
64 }
```

Listing B.51 user.php

```

1 <?php
2
3 namespace Model\System;
4
5 use DateInterval;
6 use DateTimeImmutable;
7 use Lcobucci\JWT\Builder;
8 use Lcobucci\JWT\Parser;
9 use Lcobucci\JWT\Signer\Key;
10 use Lcobucci\JWT\ValidationData;
11 use Lcobucci\JWT\Signer\Keychain;
12 use Lcobucci\JWT\Signer\Rsa\Sha256;
13
14 class User extends \Model\ModelBase
15 {
16     protected
17         $fieldConf = array(
18             "username" => [
19                 'type' => \DB\SQL\Schema::DT_TEXT,
20                 '_copyable' => true
21             ],
22             "password" => [
23                 'type' => \DB\SQL\Schema::DT_TEXT,
24                 'nullable' => true,
25                 '_copyable' => true
26             ],
27             "email" => [
28                 'type' => \DB\SQL\Schema::DT_TEXT,
29                 '_copyable' => true
30             ],
31             "acl" => [
32                 'belongs-to-one' => '\Model\System\Acl',
33                 '_copyable' => true
34             ],
35
36             'deleted_on' => [
37                 'type' => \DB\SQL\Schema::DT_DATETIME,
38                 'nullable' => true,
39                 'index' => false,
40                 'unique' => false,
41             ],
42             'created_on' => [
43                 'type' => \DB\SQL\Schema::DT_DATETIME,
44                 'nullable' => true,
45                 'index' => false,
46                 'unique' => false,
47             ],
48             'updated_on' => [
49                 'type' => \DB\SQL\Schema::DT_DATETIME,
50                 'nullable' => true,
51                 'index' => false,
52                 'unique' => false,
53             ],
54         ),
55         $db = 'DB',
56         $table = 'system_user';
57
58     public const
```

```

59     E_GENERIC = "Kombinasi User dan Password tidak ditemukan, coba lagi",
60     E_LOGIN_NOT_SUPPORTED = "Login untuk tipe user ini saat ini belum di support.";
61
62
63
64     public function set_deleted_on($date)
65     {
66         return date("Y-m-d H:i:s", $date);
67     }
68
69     public function set_created_on($date)
70     {
71         return date("Y-m-d H:i:s", $date);
72     }
73
74     public function set_updated_on($date)
75     {
76         return date("Y-m-d H:i:s", $date);
77     }
78
79     public function save()
80     {
81         if (!$this->created_on)
82             $this->created_on = time();
83         $this->updated_on = time();
84         return parent::save();
85     }
86
87     public static function getFromSession($statusOnly = false)
88     {
89         $f3 = \Base::instance();
90         if (!$f3->exists('SESSION.user'))
91             return null;
92
93         if ($statusOnly) {
94             return "user";
95         }
96
97         $user = new self();
98         $user->load(['id = ?', $f3->get('SESSION.user')]);
99         return $user;
100    }
101
102    public static function getFromHTTPHeader()
103    {
104        if (!\Base::instance()->exists('HEADERS.Authorization')) {
105            return false;
106        }
107        $auth_header = \Base::instance()->get('HEADERS.Authorization');
108        $auth_header = \substr($auth_header, strlen("Bearer "));
109        $token = (new Parser())->parse($auth_header);
110
111        $data = new ValidationData(); // It will use the current time to validate (iat, nbf and exp)
112        $data->setIssuer(\Base::instance()->get('SECURITY.issuer'));
113
114
115        $signer = new Sha256();
116        $keychain = new Keychain();
117
118        if (!$token->validate($data) || !$token->verify(
119            $signer,
120            $keychain->getPublicKey("file://" . \Base::instance()->get('SECURITY.publickey_path'))
121        )) {
122            return false;
123        }
124
125        $user = new self();
126        $user->load(['id=?', $token->getClaim('uid')]);
127        if ($user->dry()) {
128            return false;
129        }
130        return $user;
131    }
132
133    public static function getIPLoginHTTPHeader()
134    {
135        if (!\Base::instance()->exists('HEADERS.Authorization')) {
136            return null;
137        }
138        $auth_header = \Base::instance()->get('HEADERS.Authorization');
139        $auth_header = \substr($auth_header, strlen("Bearer "));
140        $token = (new Parser())->parse($auth_header);
141
142        $data = new ValidationData(); // It will use the current time to validate (iat, nbf and exp)
143        $data->setIssuer(\Base::instance()->get('SECURITY.issuer'));
144
145        $signer = new Sha256();
146        $keychain = new Keychain();
147
148        if (!$token->validate($data) || empty($token->getClaim('iplid')) || !$token->verify(
149            $signer,
150            $keychain->getPublicKey("file://" . \Base::instance()->get('SECURITY.publickey_path'))
151        )) {
152            return null;
153        }
154
155        $iplogin = new IPLogin();
156        $iplogin->load(['user=? and id=?', $token->getClaim('uid'), $token->getClaim('iplid')]);
157        if ($iplogin->dry()) {

```

```

158         return null;
159     }
160     return $iplogin;
161 }
162
163 public function generateToken($as = "basic", $iploginId = null)
164 {
165     if ($this->dry()) {
166         throw new \Exception('Model is dry!');
167     }
168     $issued = new DateTimeImmutable();
169     $expiration = $issued->add(new DateInterval("PT" . \Base::instance()->get('SECURITY.expiration') . "S"));
170     $signer = new Sha256();
171     $keychain = new Key("file://" . \Base::instance()->get('SECURITY.privatekey_path'));
172     $token = (new Builder())->issuedBy(\Base::instance()->get('SECURITY.issuer'))
173         ->issuedAt($issued)
174         ->expiresAt($expiration)
175         ->withClaim('uid', $this->id)
176         ->withClaim('uid-as', $as)
177         ->withClaim('ipid', $iploginId)
178         ->getToken($signer, $keychain);
179     return $token;
180 }
181
182 public static function login($username, $password, $set_session = false)
183 {
184     $user = new self;
185     $user->load(['username=? or email=?', $username, $username]);
186     if (!$user->loaded())
187         throw new \Exception(self::E_GENERIC);
188
189     if ($user->password == null) {
190         throw new \Exception(self::E_LOGIN_NOT_SUPPORTED);
191     }
192
193     if (!$user->auth($password))
194         throw new \Exception(self::E_GENERIC);
195
196     $f3 = \Base::instance();
197
198     if ($set_session) {
199         $f3->set('SESSION.user', $user->id);
200     }
201     return $user;
202 }
203
204 public function auth($password)
205 {
206     return password_verify($password, $this->password);
207 }
208
209 public function set_password($pass)
210 {
211     return ($pass === null) ? $pass : password_hash($pass, CRYPT_BLOWFISH);
212 }
213
214 public function cast($obj = NULL, $rel_depths = 1, $save_cast = true)
215 {
216     $obj = parent::cast($obj, $rel_depths);
217     if (!$save_cast) {
218         return $obj;
219     } else {
220         unset($obj['password']);
221         return $obj;
222     }
223 }
224 }
225 }
```

Listing B.52 answerslot.php

```

1 <?php
2
3 namespace Model\Ujian;
4
5 class AnswerSlot extends \Model\ModelBase
6 {
7     protected
8         $fieldConf = array(
9             'format' => [
10                 'type' => \DB\SQL\Schema::DT_TEXT,
11                 'nullable' => false,
12                 'index' => false,
13                 'unique' => false,
14                 '_copyable' => true
15             ],
16             'exam' => [
17                 'belongs-to-one' => '\Model\Ujian\Exam',
18                 'nullable' => false,
19                 '_copyable' => true
20             ],
21             'submissions' => [
22                 'has-many' => ['\Model\Ujian\Submission', 'answer_slot'],
23                 'nullable' => true,
24                 '_copyable' => false
25             ],
26         );
27 }
```

```

27     'deleted_on' => [
28         'type' => \DB\SQL\Schema::DT_DATETIME,
29         'nullable' => true,
30         'index' => false,
31         'unique' => false,
32     ],
33     'created_on' => [
34         'type' => \DB\SQL\Schema::DT_DATETIME,
35         'nullable' => true,
36         'index' => false,
37         'unique' => false,
38     ],
39     'updated_on' => [
40         'type' => \DB\SQL\Schema::DT_DATETIME,
41         'nullable' => true,
42         'index' => false,
43         'unique' => false,
44     ],
45     $db = 'DB',
46     $table = 'ujian_answer_slot';

47     public function set_deleted_on($date)
48     {
49         return date("Y-m-d H:i:s", $date);
50     }
51
52     public function set_created_on($date)
53     {
54         return date("Y-m-d H:i:s", $date);
55     }
56
57     public function set_updated_on($date)
58     {
59         return date("Y-m-d H:i:s", $date);
60     }
61
62     public function save()
63     {
64         if (!$this->created_on)
65             $this->created_on = time();
66         $this->updated_on = time();
67         return parent::save();
68     }

69 /**
70 * Simulate answer filename format. This will enables us to check between the filenames answers.
71 *
72 * @param Participant $as_participant participant
73 * @return string final format
74 */
75 public function simulateFormat(Participant $as_participant): string
76 {
77     return \preg_replace_callback("/\%([a-zA-Z0-9\_]+)\%\mi", function ($match) use ($as_participant) {
78         if ($as_participant->{$match[1]}) {
79             return $as_participant->{$match[1]};
80         } else {
81             return "-&&-"; // means it's impossible.
82         }
83     }, $this->format);
84 }

85     public function getSubmission(Participant $participant): Submission
86     {
87         $submission = new \Model\Ujian\Submission();
88         $submission->load(["participant = ? and answer_slot = ?", $participant->_id, $this->_id]);
89         return $submission;
90     }

91 /**
92 * Submit the answer for this answer slot.
93 * This function will scramble the filename and save it into the abyss,
94 * and return a submission object for a report.
95 *
96 * @param string $filepath Filepath to the temp
97 * @param Participant $as_participant The participant that submits the answer.
98 * @return Submission Submission obejct that holds the truth behind the abyss.
99 */
100    public function submit(string $filepath, Participant $as_participant): Submission
101    {
102        // detect submission
103        $submission = new \Model\Ujian\Submission();
104        $submission->load(["participant = ? and answer_slot = ?", $as_participant->_id, $this->_id]);
105        if ($submission->dry()) {
106            $submission->copyfrom([
107                "participant" => $as_participant->_id,
108                "answer_slot" => $this->_id
109            ]);
110            $submission->save();
111        }

112        // copy filepath:
113        $fullpath = $submission->getFullPath();
114        if (is_file($fullpath)) { // remove existing file
115            unlink($fullpath);
116        }
117        copy($filepath, $fullpath);

118        $logger = $as_participant->exam->getLoggerInstance();
119
120
121
122
123
124
125

```

```

126     $logger->warn("Participant ". $as_participant->npm . " (#". $as_participant->_id . ") uploaded ". $this->
127     simulateFormat($as_participant) . " via Submission#". $submission->_id . " on ". $f3->IP);
128
129     // return the Submission
130     return $submission;
131 }
132
133 public function cast($obj = NULL, $rel_depths = 1, $save_cast = true, Participant $as_participant = null)
134 {
135     $obj = parent::cast($obj, $rel_depths);
136
137     if ($as_participant != null) {
138         if ($as_participant->loaded() == 0) {
139             throw new \Exception("Participant is dry.");
140         }
141         $obj['format'] = $this->simulateFormat($as_participant);
142     }
143
144     if (!$save_cast) {
145         return $obj;
146     } else {
147         unset($obj['submissions']); //will prevent leaking information
148         return $obj;
149     }
150 }

```

Listing B.53 computer.php

```

1 <?php
2 namespace Model\Ujian;
3
4 class Computer extends \Model\ModelBase {
5     protected
6         $fieldConf = array(
7             'name'=>[
8                 'type'=>\DB\SQL\Schema::DT_TEXT,
9                 'nullable' => false,
10                'index' => false,
11                'unique' => false,
12                '_copyable' => true
13            ],
14            'ip' => [
15                'type'=>\DB\SQL\Schema::DT_TEXT,
16                'nullable' => false,
17                'index' => false,
18                'unique' => false,
19                '_copyable' => true
20            ],
21            'reverse_dns' => [
22                'type'=>\DB\SQL\Schema::DT_TEXT,
23                'nullable' => false,
24                'index' => false,
25                'unique' => false,
26                '_copyable' => true
27            ],
28            'd_pos' => [
29                'type'=>self::DT_JSON,
30                'nullable' => false,
31                'index' => false,
32                'unique' => false,
33                '_copyable' => true
34            ],
35            'location' => [
36                'belongs-to-one'=> '\Model\Ujian\Location',
37                'nullable' => false,
38                '_copyable' => true
39            ],
40
41            'deleted_on'=>[
42                'type'=>\DB\SQL\Schema::DT_DATETIME,
43                'nullable' => true,
44                'index' => false,
45                'unique' => false,
46            ],
47            'created_on'=>[
48                'type'=>\DB\SQL\Schema::DT_DATETIME,
49                'nullable' => true,
50                'index' => false,
51                'unique' => false,
52            ],
53            'updated_on'=>[
54                'type'=>\DB\SQL\Schema::DT_DATETIME,
55                'nullable' => true,
56                'index' => false,
57                'unique' => false,
58            ],
59        ),
60        $db = 'DB',
61        $table = 'ujian_computer';
62
63    public function set_deleted_on($date) {
64        return date("Y-m-d H:i:s", $date);
65    }
66
67    public function set_created_on($date) {
68        return date("Y-m-d H:i:s", $date);

```

```

69     }
70
71     public function set_updated_on($date) {
72         return date("Y-m-d H:i:s", $date);
73     }
74
75     public function save() {
76         if(!$this->created_on)
77             $this->created_on = time();
78         $this->updated_on = time();
79         return parent::save();
80     }
81 }
```

Listing B.54 exam.php

```

1 <?php
3 namespace Model\Ujian;
5 use Monolog\Handler\StreamHandler;
6 use Monolog\Logger;
8 class Exam extends \Model\ModelBase
9 {
10     protected
11         $fieldConf = array(
12             'lecture_period' => [
13                 'belongs-to-one' => '\Model\Ujian\LecturePeriod',
14                 'nullable' => false,
15                 'index' => false,
16                 'unique' => false,
17                 '_copyable' => true
18             ],
19             'time_start' => [
20                 'type' => \DB\SQL\Schema::DT_DATETIME,
21                 'nullable' => false,
22                 'index' => false,
23                 'unique' => false,
24                 '_copyable' => true
25             ],
26             'time_ended' => [
27                 'type' => \DB\SQL\Schema::DT_DATETIME,
28                 'nullable' => true,
29                 'index' => false,
30                 'unique' => false,
31                 '_copyable' => true
32             ],
33             // WARNING: DURATION TIME SHOULD BE IN MINUTES.
34             // TODO: Convert all usage of this column from seconds to minutes.
35             'time_duration' => [
36                 'type' => \DB\SQL\Schema::DT_INT,
37                 'nullable' => false,
38                 'index' => false,
39                 'unique' => false,
40                 '_copyable' => true
41             ],
42             // WARNING: OPEN = EXAM OPENED AND PEOPLE CAN SUBMIT
43             'time_opened' => [
44                 'type' => \DB\SQL\Schema::DT_DATETIME,
45                 'nullable' => true,
46                 'index' => false,
47                 'unique' => false,
48                 '_copyable' => true
49             ],
50             'lecture' => [
51                 'belongs-to-one' => '\Model\Ujian\Lecture',
52                 'nullable' => false,
53                 '_copyable' => true
54             ],
55             'exam_uniqcode' => [
56                 'type' => \DB\SQL\Schema::DT_TEXT,
57                 'nullable' => false,
58                 'index' => false,
59                 'unique' => true,
60                 '_copyable' => false
61             ],
62             'uts' => [
63                 'type' => \DB\SQL\Schema::DT_BOOL,
64                 'default' => 1,
65                 '_copyable' => true
66             ],
67             'shift' => [
68                 'type' => \DB\SQL\Schema::DT_INT,
69                 'nullable' => true,
70                 'default' => null,
71                 '_copyable' => true
72             ],
73             'answer_slot' => [
74                 'nullable' => true,
75                 '_copyable' => false,
76                 'has-many' => ['\Model\Ujian\AnswerSlot', 'exam'],
77             ],
78             'participants' => [
79                 'nullable' => true,
80                 '_copyable' => false,
81                 'has-many' => ['\Model\Ujian\Participant', 'exam'],
82             ]
83 }
```

```

82         '_show_in-list' => false
83     ],
84     'exam_report' => [
85         'nullable' => true,
86         '_copyable' => false,
87         'has-many' => ['\Model\Ujian\ExamReport', 'exam'],
88     ],
89
90     'deleted_on' => [
91         'type' => \DB\SQL\Schema::DT_DATETIME,
92         'nullable' => true,
93         'index' => false,
94         'unique' => false,
95     ],
96     'created_on' => [
97         'type' => \DB\SQL\Schema::DT_DATETIME,
98         'nullable' => true,
99         'index' => false,
100        'unique' => false,
101    ],
102    'updated_on' => [
103        'type' => \DB\SQL\Schema::DT_DATETIME,
104        'nullable' => true,
105        'index' => false,
106        'unique' => false,
107    ],
108 ],
109 ),
110 $db = 'DB',
111 $table = 'ujian_exam';
112
113 public function __construct()
114 {
115     parent::__construct();
116
117     $this->virtual("time_left", function ($dis) {
118         if (!$dis->time_ended) {
119             return $dis->time_duration;
120         }
121
122         return max(strtotime($dis->time_ended) - time(), 0);
123     });
124 }
125
126
127 public function set_deleted_on($date)
128 {
129     return date("Y-m-d H:i:s", $date);
130 }
131
132 public function set_created_on($date)
133 {
134     return date("Y-m-d H:i:s", $date);
135 }
136
137 public function set_updated_on($date)
138 {
139     return date("Y-m-d H:i:s", $date);
140 }
141
142 public function set_time_opened($date)
143 {
144     if ($date === null) {
145         return null;
146     }
147     return date("Y-m-d H:i:s", $date);
148 }
149
150 public function set_time_ended($date)
151 {
152     if ($date === null) {
153         return null;
154     }
155     return date("Y-m-d H:i:s", $date);
156 }
157
158 public function set_time_start($date)
159 {
160     if (is_string($date)) {
161         $date = strtotime($date);
162     }
163     return date("Y-m-d H:i:s", $date);
164 }
165
166 public function cast($obj = NULL, $rel_depths = 1, $save_cast = true)
167 {
168     $obj = parent::cast($obj, $rel_depths);
169     if (!$save_cast) {
170         return $obj;
171     } else {
172         unset($obj['exam_uniqcode']);
173         return $obj;
174     }
175 }
176
177 public function save()
178 {
179     if (!$this->created_on)
180         $this->created_on = time();

```

```

181     if (!$this->exam_uniqcode)
182         $this->exam_uniqcode = bin2hex(random_bytes(16));
183     if (!$this->lecture_period) {
184         $this->lecture_period = LecturePeriod::getLatestPeriod()->id;
185     }
186     $this->updated_on = time();
187     return parent::save();
188 }
189
190 public function touch($key = 'updated_on', $timestamp = NULL)
191 {
192     parent::touch($key, $timestamp);
193 }
194
195 public function getFullPath()
196 {
197     $f3 = \Base::instance();
198
199     $securedAsset = trim($f3->get("UPLOADS"), DIRECTORY_SEPARATOR);
200     $securedAsset .= DIRECTORY_SEPARATOR . "secured_assets";
201     $securedAsset .= DIRECTORY_SEPARATOR . $this->exam_uniqcode;
202
203     if (!is_dir($securedAsset)) {
204         mkdir($securedAsset, 0777, true);
205     }
206
207     return $securedAsset;
208 }
209
210 public function getLoggerInstance($loggerName = "general"): Logger
211 {
212     $log = new Logger($loggerName);
213     $log->pushHandler(new StreamHandler($this->getFullPath() . DIRECTORY_SEPARATOR . "exam-log-essential.log", Logger::NOTICE));
214     $log->pushHandler(new StreamHandler($this->getFullPath() . DIRECTORY_SEPARATOR . "exam-log-verbose.log", Logger::INFO));
215
216     return $log;
217 }
218 }
```

Listing B.55 examreport.php

```

1 <?php
2
3 namespace Model\Ujian;
4
5 class ExamReport extends \Model\ModelBase
6 {
7     protected
8         $fieldConf = array(
9             'token' => [
10                 'type' => \DB\SQL\Schema::DT_TEXT,
11                 'nullable' => true,
12                 'index' => true,
13                 'unique' => false,
14                 '_copyable' => false
15             ],
16             'tos' => [
17                 'type' => \DB\SQL\Schema::DT_TEXT,
18                 'nullable' => false,
19                 'index' => false,
20                 'unique' => false,
21                 '_copyable' => true
22             ],
23             'exam' => [
24                 'belongs-to-one' => '\Model\Ujian\Exam',
25                 'nullable' => false,
26                 '_copyable' => true
27             ],
28
29             'sent_on' => [
30                 'type' => \DB\SQL\Schema::DT_DATETIME,
31                 'nullable' => true,
32                 'index' => false,
33                 'unique' => false,
34             ],
35             'valid_until' => [
36                 'type' => \DB\SQL\Schema::DT_DATETIME,
37                 'nullable' => false,
38                 'index' => false,
39                 'unique' => false,
40                 '_copyable' => true
41             ],
42             'created_on' => [
43                 'type' => \DB\SQL\Schema::DT_DATETIME,
44                 'nullable' => true,
45                 'index' => false,
46                 'unique' => false,
47             ],
48             'updated_on' => [
49                 'type' => \DB\SQL\Schema::DT_DATETIME,
50                 'nullable' => true,
51                 'index' => false,
52                 'unique' => false,
53             ],
54         ],
55     )
56 }
```

```

55     ),
56     $db = 'DB',
57     $table = 'ujian_exam_report';
58
59     public function set_sent_on($date)
60     {
61         return date("Y-m-d H:i:s", $date);
62     }
63
64     public function set_created_on($date)
65     {
66         return date("Y-m-d H:i:s", $date);
67     }
68
69     public function set_updated_on($date)
70     {
71         return date("Y-m-d H:i:s", $date);
72     }
73
74     public function set_valid_until($date)
75     {
76         return date("Y-m-d H:i:s", $date);
77     }
78
79     /**
80      * Push validity period of the token
81      *
82      * @param integer $addSeconds Push validity duration in seconds
83      * @param bool $autoSave (self explanatory)
84      * @return void
85      */
86     public function pushValidity(int $addSeconds = 43200, bool $autoSave = true): void
87     {
88         $this->valid_until = max((time() + $addSeconds), $this->valid_until);
89         if ($autoSave) {
90             $this->save();
91         }
92     }
93
94     public function save()
95     {
96         if (!$this->created_on) {
97             $this->created_on = time();
98         }
99
100        if (!$this->token) {
101            $this->token = bin2hex(random_bytes(16));
102        }
103
104        if (!$this->valid_until) {
105            if ($this->exam) {
106                $this->valid_until = $this->exam->time_ended;
107            } else {
108                $this->valid_until = time() + 43200; // add 12 hour from creation
109            }
110        }
111
112        $this->updated_on = time();
113        return parent::save();
114    }
115 }

```

Listing B.56 lecture.php

```

1 <?php
2 namespace Model\Ujian;
3
4 class Lecture extends \Model\ModelBase {
5     protected
6         $fieldConf = array(
7             'name' => [
8                 'type' =>\DB\SQL\Schema::DT_TEXT,
9                 'nullable' => false,
10                'index' => false,
11                'unique' => false,
12                '_copyable' => true
13            ],
14            'lecture_code' => [
15                'type' =>\DB\SQL\Schema::DT_TEXT,
16                'nullable' => false,
17                'index' => false,
18                'unique' => false,
19                '_copyable' => true
20            ],
21            'deleted_on' => [
22                'type' =>\DB\SQL\Schema::DT_DATETIME,
23                'nullable' => true,
24                'index' => false,
25                'unique' => false,
26            ],
27            'created_on' => [
28                'type' =>\DB\SQL\Schema::DT_DATETIME,
29                'nullable' => true,
30                'index' => false,
31                'unique' => false,
32            ],
33        ],

```

```

34     'updated_on' => [
35         'type' => \DB\SQL\Schema::DT_DATETIME,
36         'nullable' => true,
37         'index' => false,
38         'unique' => false,
39     ],
40 ),
41 $db = 'DB',
42 $table = 'ujian_lecture';

44     public function set_deleted_on($date) {
45         return date("Y-m-d H:i:s", $date);
46     }

48     public function set_created_on($date) {
49         return date("Y-m-d H:i:s", $date);
50     }

52     public function set_updated_on($date) {
53         return date("Y-m-d H:i:s", $date);
54     }

56     public function save() {
57         if (!$this->created_on)
58             $this->created_on = time();
59         $this->updated_on = time();
60         return parent::save();
61     }
62 }

```

Listing B.57 lectureperiod.php

```

1  <?php
3  namespace Model\Ujian;

5  class LecturePeriod extends \Model\ModelBase
6  {
7      protected
8          $fieldConf = array(
9              'period_code' => [
10                  'type' => \DB\SQL\Schema::DT_TEXT,
11                  'nullable' => false,
12                  'index' => false,
13                  'unique' => false,
14                  '_copyable' => true
15             ],
16
17             'deleted_on' => [
18                 'type' => \DB\SQL\Schema::DT_DATETIME,
19                 'nullable' => true,
20                 'index' => false,
21                 'unique' => false,
22             ],
23             'created_on' => [
24                 'type' => \DB\SQL\Schema::DT_DATETIME,
25                 'nullable' => true,
26                 'index' => false,
27                 'unique' => false,
28             ],
29             'updated_on' => [
30                 'type' => \DB\SQL\Schema::DT_DATETIME,
31                 'nullable' => true,
32                 'index' => false,
33                 'unique' => false,
34             ],
35         ),
36         $db = 'DB',
37         $table = 'ujian_lecture_period';

39         public function set_deleted_on($date)
40         {
41             return date("Y-m-d H:i:s", $date);
42         }

44         public function set_created_on($date)
45         {
46             return date("Y-m-d H:i:s", $date);
47         }

49         public function set_updated_on($date)
50         {
51             return date("Y-m-d H:i:s", $date);
52         }

54         public function save()
55         {
56             if (!$this->created_on)
57                 $this->created_on = time();
58             $this->updated_on = time();
59             return parent::save();
60         }

62         public static function getLatestPeriod()
63         {
64             $periodeCode = date("Y");
65             $bulan = date("m");

```

```

67     if ($bulan < 6) {
68         // genap
69         $periodeCode =date("Y2", strtotime("-1 year"));
70     } else if ($bulan < 7) {
71         // SP
72         $periodeCode .= "3";
73     } else {
74         // ganjil
75         $periodeCode .= "1";
76     }
77
78     $object = new self();
79     $object->load(["period_code = ?", $periodeCode]);
80     if ($object->dry()) {
81         $object->copyfrom([
82             "period_code" => $periodeCode
83         ]);
84         $object->save();
85     }
86
87     return $object;
88 }
89

```

Listing B.58 location.php

```

1  <?php
2
3  namespace Model\Ujian;
4
5  class Location extends \Model\ModelBase
6  {
7      protected
8          $fieldConf = array(
9              'room_name' => [
10                  'type' => \DB\SQL\Schema::DT_TEXT,
11                  'nullable' => false,
12                  'index' => false,
13                  'unique' => false,
14                  '_copyable' => true
15              ],
16              'name_alias' => [
17                  'type' => \DB\SQL\Schema::DT_TEXT,
18                  'nullable' => false,
19                  'index' => false,
20                  'unique' => false,
21                  '_copyable' => true
22              ],
23              'computers' => [
24                  'has-many' => ["\Model\Ujian\Computer", "location"],
25                  '_copyable' => false
26              ],
27              'iplogins' => [
28                  'has-many' => ["\Model\System\IPLogin", "locations", "system_location_iplogin_link"],
29                  '_copyable' => false
30              ],
31
32              'deleted_on' => [
33                  'type' => \DB\SQL\Schema::DT_DATETIME,
34                  'nullable' => true,
35                  'index' => false,
36                  'unique' => false,
37                  '_copyable' => false
38              ],
39              'created_on' => [
40                  'type' => \DB\SQL\Schema::DT_DATETIME,
41                  'nullable' => false,
42                  'index' => false,
43                  'unique' => false,
44                  '_copyable' => false
45              ],
46              'updated_on' => [
47                  'type' => \DB\SQL\Schema::DT_DATETIME,
48                  'nullable' => false,
49                  'index' => false,
50                  'unique' => false,
51                  '_copyable' => false
52              ],
53          ),
54          $db = 'DB',
55          $table = 'ujian_location';
56
57      public function set_deleted_on($date)
58      {
59          return date("Y-m-d H:i:s", $date);
60      }
61
62      public function set_created_on($date)
63      {
64          return date("Y-m-d H:i:s", $date);
65      }
66
67      public function set_updated_on($date)
68      {
69          return date("Y-m-d H:i:s", $date);
70      }
71

```

```

72     public function save()
73     {
74         if (!$this->created_on) {
75             $this->created_on = time();
76         }
77         $this->updated_on = time();
78         return parent::save();
79     }
80 }

```

Listing B.59 notification.php

```

1  <?php
3  namespace Model\Ujian;
5  class Notification extends \Model\ModelBase
6  {
8      protected
9          $fieldConf = array(
10             'participants' => [
11                 'has-many' => ["\Model\Ujian\Participant", "notifications", "ujian_notif_aud"],
12                 '_copyable' => true
13             ],
15             'title' => [
16                 'type' => \DB\SQL\Schema::DT_TEXT,
17                 'nullable' => false,
18                 'index' => false,
19                 'unique' => false,
20                 '_copyable' => true
21             ],
22             'type' => [
23                 'type' => \DB\SQL\Schema::DT_VARCHAR256,
24                 'nullable' => true,
25                 'index' => false,
26                 'unique' => false,
27                 '_copyable' => true
28             ],
29             'description' => [
30                 'type' => \DB\SQL\Schema::DT_TEXT,
31                 'nullable' => false,
32                 'index' => false,
33                 'unique' => false,
34                 '_copyable' => true
35             ],
36             'extras' => [
37                 'type' => \DB\Cortex::DT_JSON,
38                 'nullable' => true,
39                 'index' => false,
40                 'unique' => false,
41                 '_copyable' => true
42             ],
44             'created_on' => [
45                 'type' => \DB\SQL\Schema::DT_DATETIME,
46                 'nullable' => false,
47                 'index' => false,
48                 'unique' => false,
49                 '_copyable' => false
50             ],
51             'updated_on' => [
52                 'type' => \DB\SQL\Schema::DT_DATETIME,
53                 'nullable' => false,
54                 'index' => false,
55                 'unique' => false,
56                 '_copyable' => false
57             ],
58         ),
59         $db = 'DB',
60         $table = 'ujian_notification';
62     public function set_created_on($date)
63     {
64         return date("Y-m-d H:i:s", $date);
65     }
67     public function set_updated_on($date)
68     {
69         return date("Y-m-d H:i:s", $date);
70     }
72     public function save()
73     {
74         if (!$this->created_on) {
75             $this->created_on = time();
76         }
77         $this->updated_on = time();
78         return parent::save();
79     }
81     public function cast($obj = NULL, $rel_depths = 1, $secureCast = true)
82     {
83         if (!$obj) {
84             $obj = $this;

```

```

85     }
86
87     $casted = parent::cast($obj);
88
89     if ($secureCast) {
90         $casted = array_diff_key($casted, array_flip(["participants"]));
91     }
92
93     return $casted;
94 }
95 }
```

Listing B.60 participant.php

```

1 <?php
2
3 namespace Model\Ujian;
4
5 class Participant extends \Model\ModelBase
6 {
7     protected
8         $fieldConf = array(
9             'username' => [
10                 'type' => \DB\SQL\Schema::DT_TEXT,
11                 'nullable' => false,
12                 'index' => false,
13                 'unique' => false,
14                 '_copyable' => true
15             ],
16             'notifications' => [
17                 'has-many' => ["\Model\Ujian\Notification", "participants", "ujian_notif_aud"],
18                 '_copyable' => false
19             ],
20             'npm' => [
21                 'type' => \DB\SQL\Schema::DT_TEXT,
22                 'nullable' => false,
23                 'index' => false,
24                 'unique' => false,
25                 '_copyable' => true
26             ],
27             'exam' => [
28                 'belongs-to-one' => '\Model\Ujian\Exam',
29                 '_copyable' => true
30             ],
31             'computer' => [
32                 'belongs-to-one' => '\Model\Ujian\Computer',
33                 '_copyable' => true
34             ],
35
36             'deleted_on' => [
37                 'type' => \DB\SQL\Schema::DT_DATETIME,
38                 'nullable' => true,
39                 'index' => false,
40                 'unique' => false,
41             ],
42             'created_on' => [
43                 'type' => \DB\SQL\Schema::DT_DATETIME,
44                 'nullable' => true,
45                 'index' => false,
46                 'unique' => false,
47             ],
48             'updated_on' => [
49                 'type' => \DB\SQL\Schema::DT_DATETIME,
50                 'nullable' => true,
51                 'index' => false,
52                 'unique' => false,
53             ],
54         ),
55         $db = 'DB',
56         $table = 'ujian_participant';
57
58     public function __construct()
59     {
60         parent::__construct();
61
62         parent::virtual('xxxxx', function ($x) {
63             return substr($x->username, 1);
64         });
65
66         parent::virtual('is_upcoming', function ($x) {
67             // is_upcoming is checking if the time is not ended (yet).
68             // since upon timer start, time_ended filled, so we can just check
69             // if it's null or not. Yeah!
70             return !($x->time_ended);
71         });
72
73         parent::virtual('display_name', function ($sumu) {
74             if (!F3::get('dev_setting.query_ldap')) {
75                 return null;
76                 $username = $sumu->username;
77                 $prov = F3::get('LDAP.provider');
78                 $account = $prov->search()->users()->findBy('sAMAccountName', $username);
79                 if (!$account)
80                     return $username;
81                 return mb_convert_case($account->getFirstName() . " " . $account->getLastName(), MB_CASE_TITLE);
82             });
83         });
84     }
85 }
```

```

85     public function set_deleted_on($date)
86     {
87         return date("Y-m-d H:i:s", $date);
88     }
89
90     public function set_created_on($date)
91     {
92         return date("Y-m-d H:i:s", $date);
93     }
94
95     public function set_updated_on($date)
96     {
97         return date("Y-m-d H:i:s", $date);
98     }
99
100    public function save()
101    {
102        if (!$this->created_on)
103            $this->created_on = time();
104        $this->updated_on = time();
105        return parent::save();
106    }
107
108 /**
109 * Fetch active examination with given IP. IP will be detected automatically
110 * from SERVER variable.
111 *
112 * @param boolean $upcomingToo Should we fetch the upcoming
113 * @param string $ip IP address to check on, default to autodetect.
114 * @param integer $upcomingTimeMinutes How far the upcoming exam should be fetched
115 * @return Participant|null Participant
116 */
117 public static function getActiveParticipant($upcoming = false, $ip = null, $upcomingTimeMinutes = 10)
118 {
119     $f3 = \Base::instance();
120
121     if ($ip == null) {
122         $ip = $f3->get('IP');
123     }
124
125     $computer = new \Model\Ujian\Computer();
126     $computer->load(['ip = ?', $ip]);
127     if ($computer->loaded() == 0) {
128         throw new \Model\Error(
129             "This computer is not registered for exam.",
130             "This computer's IP ($ip) is not listed in computer that used for examination.",
131             "EI01",
132             "unregistered",
133             403
134         );
135     }
136
137     $participant = new self();
138     if ($upcoming) {
139         $participant->has('exam', [
140             "(deleted_on = ? AND timeEnded = ? AND timeOpened = ?) AND (timeStart <= ? AND timeStart >= ?)",
141             null,
142             null,
143             null,
144             date('Y-m-d H:i:s', strtotime(("+$upcomingTimeMinutes minute"))),
145             date('Y-m-d H:i:s', strtotime(("{$upcomingTimeMinutes} minute")))
146         ]);
147     } else {
148         $participant->has('exam', [
149             "(deleted_on = ? AND timeEnded != ?) AND (timeOpened <= ? AND timeEnded >= ?)",
150             null,
151             null,
152             date('Y-m-d H:i:s'),
153             date('Y-m-d H:i:s'),
154         ]);
155     }
156
157     $participant->load(['computer = ?', $computer->id], [
158         "order" => "time_start asc"
159     ]);
160
161     if ($participant->loaded()) {
162         return $participant;
163     }
164
165     return null;
166 }
167
168 /**
169 * Get any active participant. Will return either active, upcoming, or null.
170 *
171 * @param string $ip IP of the computer to be queried default to autodetect.
172 * @param integer $upcomingTimeMinutes How far the upcoming exam should be fetched?
173 * @return Participant|null
174 */
175 public static function getAnyParticipant($ip = null, $upcomingTimeMinutes = 10)
176 {
177     $activePar = self::getActiveParticipant(false, $ip, $upcomingTimeMinutes);
178     if ($activePar) {
179         return $activePar;
180     }
181
182     return self::getActiveParticipant(true, $ip, $upcomingTimeMinutes);

```

```

183     }
184 }
```

Listing B.61 submission.php

```

1 <?php
3 namespace Model\Ujian;
5 /**
6  * @property Participant $participant
7  * @property AnswerSlot $answer_slot
8  * @property string $stored_filename
9 */
10 class Submission extends \Model\ModelBase
11 {
12     protected
13         $fieldConf = array(
14             'participant' => [
15                 'belongs-to-one' => '\Model\Ujian\Participant',
16                 '_copyable' => true
17             ],
18             'answer_slot' => [
19                 'belongs-to-one' => '\Model\Ujian\AnswerSlot',
20                 '_copyable' => true
21             ],
22             'stored_filename' => [
23                 'type' => \DB\SQL\Schema::DT_TEXT,
24                 'nullable' => false,
25                 'index' => false,
26                 'unique' => false,
27                 '_copyable' => true
28             ],
29             'deleted_on' => [
30                 'type' => \DB\SQL\Schema::DT_DATETIME,
31                 'nullable' => true,
32                 'index' => false,
33                 'unique' => false,
34             ],
35             'created_on' => [
36                 'type' => \DB\SQL\Schema::DT_DATETIME,
37                 'nullable' => true,
38                 'index' => false,
39                 'unique' => false,
40             ],
41             'updated_on' => [
42                 'type' => \DB\SQL\Schema::DT_DATETIME,
43                 'nullable' => true,
44                 'index' => false,
45                 'unique' => false,
46             ],
47         ),
48         $db = 'DB',
49         $table = 'ujian_submission';
50
51     public function set_deleted_on($date)
52     {
53         return date("Y-m-d H:i:s", $date);
54     }
55
56     public function set_created_on($date)
57     {
58         return date("Y-m-d H:i:s", $date);
59     }
60
61     public function set_updated_on($date)
62     {
63         return date("Y-m-d H:i:s", $date);
64     }
65
66     public function cast($obj = NULL, $rel_depths = 1, $save_cast = true)
67     {
68         $obj = parent::cast($obj, $rel_depths);
69         if (!$save_cast) {
70             return $obj;
71         } else {
72             unset($obj['stored_filename']);
73             return $obj;
74         }
75     }
76 }
77
78     public function save()
79     {
80         if (!$this->created_on)
81             $this->created_on = time();
82         if (!$this->stored_filename)
83             $this->stored_filename = bin2hex(random_bytes(16));
84         $this->updated_on = time();
85         return parent::save();
86     }
87
88     public function getFullPath()
89     {
90         return $this->participant->exam->getFullPath() . DIRECTORY_SEPARATOR . $this->stored_filename;
91     }

```

```

93     public function touch($key = "updated_on", $timestamp = null)
94     {
95         parent::touch($key, $timestamp);
96     }
97 }
```

Listing B.62 formatter.php

```

1 <?php
2 namespace Output;
3
4 class formatter extends \Prefab{
5     public function format_success($data){
6         return [
7             "status" => true,
8             "data" => $data,
9             "error" => null
10        ];
11    }
12
13    public function format_error(\Model\Error $data){
14        return [
15            "status" => false,
16            "data" => null,
17            "error" => $data->serve_exception()
18        ];
19    }
20 }
```

Listing B.63 json.php

```

1 <?php
2
3 namespace Output;
4
5 class JSON extends \Prefab
6 {
7     public function serve($data): string
8     {
9         $f3 = \F3::instance();
10        header('Content-type: application/json');
11        $formattedData = json_encode(
12            $data,
13            ($f3->exists("GET.json.pretty_print")) ? JSON_PRETTY_PRINT : 0
14        );
15
16        echo $formattedData;
17        return $formattedData;
18    }
19 }
```

Listing B.64 plain.php

```

1 <?php
2
3 namespace Output;
4
5 class Plain extends \Prefab
6 {
7     public function serve($data): string
8     {
9         $f3 = \F3::instance();
10        header('Content-type: text/plain');
11        $formattedData = "Unsupported API Result type. Please use defined result type, or contact administrator.";
12        echo $formattedData;
13        return $formattedData;
14    }
15 }
```

Listing B.65 xml.php

```

1 <?php
2
3 namespace Output;
4
5 use Spatie\ArrayToXml\ArrayToXml;
6
7 class XML extends \Prefab
8 {
9
10    public $xmlns = "https://git.christianto.net/f3-api/xmlns/generic-v1";
11    public $rootElementName = 'Result';
12
13    public function serve($data): string
14    {
15        $f3 = \F3::instance();
16        header('Content-type: application/xml');
17
18        $data_filtered = [
19            '_attributes' => [

```

```

20         'api:status' => $data['status'] ? "true" : "false"
21     ],
22     'api:data' => $data['data'],
23     'api:error' => $data['error']
24   ];
25
26   $formattedData = ArrayToXml::convert($data_filtered, [
27     'rootElementName' => $this->rootElementName,
28     '_attributes' => [
29       'xmlns:api' => $this->xmllns
30     ]
31   ]);
32
33   echo $formattedData;
34   return $formattedData;
35 }
36 }
```

Listing B.66 batgenerator.php

```

1 <?php
2
3 namespace Service;
4
5 class BatGenerator extends \Prefab
6 {
7   protected
8   //$/zip,
9   //$/ujian,
10  $zipname;
11
12  public function generate(\Model\Ujian\Exam $ujian)
13  {
14    $zip = new \ZipArchive();
15    $zipname = $this->tempFileGenerator(".zip");
16    //$/this->zip = $zip;
17    $ujian->participants->orderBy('posisi asc');
18    //$/this->ujian = $ujian;
19
20    \F3::set("data.ujian", $ujian);
21    $zip->open($zipname, \ZipArchive::CREATE | \ZipArchive::OVERWRITE);
22    $zip->addFromString('01-mkdir.bat', \View\Bat::render("lab-mkdir.bat", "01-mkdir.log"));
23    $zip->addFromString('02-copy.bat', \View\Bat::render("lab-copy.bat", "02-copy.log"));
24    $zip->addFromString('03-takeown.bat', \View\Bat::render("lab-takeown.bat", "03-takeown.log"));
25    $zip->addFromString('WARNING-special--takeown-full-ujian.bat', \View\Bat::render("special-ujian-fulltakeown.bat", "04
26      -special-takeown.log"));
27    $zip->close();
28
29    return $zipname;
30  }
31
32  public function generateMigration(array $migrationLists, $exam)
33  {
34    $zip = new \ZipArchive();
35    $zipname = $this->tempFileGenerator(".zip");
36    $this->zipname = $zipname;
37
38    \F3::set("data.migrations", $migrationLists);
39    $zip->open($zipname, \ZipArchive::CREATE | \ZipArchive::OVERWRITE);
40    $zip->addFromString('00-migrate-folders.bat', \View\Bat::render("lab-migrate.bat"));
41
42    // Limit exam lists.
43    $participants = $exam->participants;
44    $exam->participants = array_map(function ($d) {
45      return $d['p'];
46    }, $migrationLists);
47    \F3::set("data.ujian", $exam);
48    $zip->addFromString('01-mkdir.bat', \View\Bat::render("lab-mkdir.bat", "01-mkdir.log"));
49    $zip->addFromString('02-copy.bat', \View\Bat::render("lab-copy.bat", "02-copy.log"));
50    $zip->close();
51
52    $exam->participants = $participants;
53    return $zipname;
54  }
55
56  public function clean_zip()
57  {
58    \unlink($this->zipname);
59  }
60
61  public function tempFileGenerator($prefix = ".tmp", $length = 10, $chars = '
62  ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz')
63  {
64    $temp = "";
65    while ($length-- > 0) {
66      $temp .= $chars[\rand(0, \strlen($chars))];
67    }
68    return \F3::get('TEMP') . '/' . $temp . $prefix;
69  }
70 }
```

Listing B.67 devmode.html

```
| 1 <h1>It... somekind of works</h1>
```

```
2 <p>You're in development mode. Please check out some of your project settings or read the README files to get started and get rid of this message.</p>
```

Listing B.68 exam-finished-lecturer.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Exam Finished</title>
7
8     <style>
9       @import url('https://fonts.googleapis.com/css2?family=Roboto&display=swap');
10
11    body {
12      font-family: 'Roboto', sans-serif;
13    }
14  </style>
15 </head>
16
17 <body>
18   <p>Yth. Bapak/Ibu dosen,</p>
19   <p>
20     Berkas ujian yang telah di laksanakan di laboratorium untuk mata kuliah
21     <b>{{@email.exam->lecture->lecture_code}} - {{@email.exam->lecture->name}}</b> sudah tersedia untuk diunduh.
22     <br />
23     Karena alasan keamanan, berkas hanya dapat diunduh via link yang dikirimkan pada email ini. Mohon segera di
24     akses sebelum tautan kadaluarsa.
25   </p>
26   <p>
27     <a href="{{@email.link}}>{{@email.link}}</a>
28     <br />
29     <small>
30       Tidak dapat diklik? Salin-tempel link tersebut pada peramban.
31     </small>
32   </p>
33   <p>
34     Link tersebut akan aktif hingga <b>{{strftime("%C", strtotime(@email.examreport->valid_until))}}</b>.
35     Untuk informasi lebih lanjut, anda dapat menghubungi Tim Administrator.
36     <br />
37     Terima kasih.
38   </p>
39   <p>
40     Salam,<br />
41     Sistem Oxam
42   </p>
43   <hr />
44   <p>
45     <small>
46       Email ini dikirimkan secara otomatis oleh sistem <a href="https://gitlab.com/ftis-admin/oxam">Oxam</a>.
47       Tautan tertera pada email bersifat sensitif dan harus diperlakukan seperti kata sandi.
48     </small>
49   </p>
50 </body>
51 </html>
```

Listing B.69 layout.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <meta http-equiv="X-UA-Compatible" content="IE=edge">
6     <title>{{isset(@page.title)?@page.title:"}}</title>
7     <meta name="viewport" content="width=device-width, initial-scale=1">
8   </head>
9   <body>
10     <include href="{{@page.location}}" if="{{isset(@page.location)}}"/>
11   </body>
12 </html>
```

Listing B.70 password.html

```
1 Informasi akun untuk masuk ke <code>{{@notif.url}}</code> kamu adalah<br />
2 username: <span data-type="username">{{@notif.username}}</span><br />
3 Password: <span data-type="password">{{@notif.password}}</span><br />
4 <br />
5 Selamat ujian!
```

Listing B.71 lab-copy.bat

```
1 @echo off
2 echo OXAM Copy Soal creation batch
3 echo Logfile will be collected to {{@data.logfile}}.
4 echo
5 echo
6 <f3:repeat group="{{@data.ujian.participants}}" value="{{@pos}}">
```

```

7 echo =====
8 echo Working on {{@pos->computer->name}}
9 echo =====
10 echo ===== >> {{@data.logfile}}
11 echo Working on {{@pos->computer->name}} >> {{@data.logfile}}
12 echo ===== >> {{@data.logfile}}
13 echo copying...
14 echo copying... >> {{@data.logfile}}
15 copy "C:\Users\Administrator\Desktop\FILE UJIAN OXAM (WEB)" "\\{{@pos->ip}}\D$\ujian\{{@pos.username}} - {{@data.ujian.lecture->name}}" 2>> {{@data.logfile}}

17 </f3:repeat>
18 echo
19 echo
20 echo Finished.

```

Listing B.72 lab-migrate.bat

```

1 @echo off
2 echo OXAM Migration Tool
3 echo Logfile will be collected to {{@data.logfile}}.
4 echo
5 echo
6 <f3:repeat group="{{@data.migrations}}" value="{{@pos}}">
7 echo =====
8 echo Migrating participant {{@pos.p->name}} on {{@pos.c_before->name}} to {{@pos.c_after->name}}
9 echo =====
10 echo ===== >> {{@data.logfile}}
11 echo Migrating participant {{@pos.p->name}} on {{@pos.c_before->name}} to {{@pos.c_after->name}} >> {{@data.logfile}}
12 echo ===== >> {{@data.logfile}}
13 echo copying...
14 echo copying... >> {{@data.logfile}}
15 copy "\\{{@pos.c_before->ip}}\D$\ujian\{{@pos.p->username}} - {{@data.ujian.lecture->name}}" "\\{{@pos.c_after->ip}}\D$\ujian\{{@pos.p->username}} - {{@data.ujian.lecture->name}}" 2>> {{@data.logfile}}

17 echo Updating permissions...
18 echo Updating permissions... >> {{@data.logfile}}
19 icacls "\\{{@pos.c_after->ip}}\D$\ujian\{{@pos.p->username}} - {{@data.ujian.lecture->name}}" /inheritance:d /T>> {{@data.logfile}}
20 icacls "\\{{@pos.c_after->ip}}\D$\ujian\{{@pos.p->username}} - {{@data.ujian.lecture->name}}" /T /grant ftis\administrator:(OI)(CI)F ftis\administrator:(OI)(CI)(RX,W,DC) >> {{@data.logfile}}
21 icacls "\\{{@pos.c_after->ip}}\D$\ujian\{{@pos.p->username}} - {{@data.ujian.lecture->name}}" /T /grant "ftis\{{@pos.p->username}}:(OI)(CI)(RX,W,DC)" /T >> {{@data.logfile}}
22 icacls "\\{{@pos.c_after->ip}}\D$\ujian\{{@pos.p->username}} - {{@data.ujian.lecture->name}}" /T /remove:g "Authenticated Users" >> {{@data.logfile}}
23 icacls "\\{{@pos.c_after->ip}}\D$\ujian\{{@pos.p->username}} - {{@data.ujian.lecture->name}}" /T /remove:g "Users" >> {{@data.logfile}}
24 </f3:repeat>
25 echo
26 echo
27 echo Finished.

```

Listing B.73 lab-mkdir.bat

```

1 @echo off
2 echo OXAM Dir creation batch
3 echo Logfile will be collected to {{@data.logfile}}.
4 echo
5 echo
6 <f3:repeat group="{{@data.ujian.participants}}" value="{{@pos}}">
7 echo =====
8 echo Working on {{@pos->computer->name}}
9 echo =====
10 echo ===== >> {{@data.logfile}}
11 echo Working on {{@pos->computer->name}} >> {{@data.logfile}}
12 echo ===== >> {{@data.logfile}}
13 echo icacls-ing...
14 echo icacls-ing... >> {{@data.logfile}}
15 echo rd-ing...
16 echo rd-ing... >> {{@data.logfile}}
17 rd /S /Q "\\{{@pos->computer->ip}}\D$\ujian\{{@pos.username}} - {{@data.ujian.lecture->name}}" 2>> {{@data.logfile}}
18 echo mkdir-ing...
19 echo mkdir-ing... >> {{@data.logfile}}
20 mkdir "\\{{@pos->computer->ip}}\D$\ujian\{{@pos.username}} - {{@data.ujian.lecture->name}}" 2>> {{@data.logfile}}
21 echo takeown-ing...
22 echo takeown-ing... >> {{@data.logfile}}
23 takeown /A /F "\\{{@pos->computer->ip}}\D$\ujian\{{@pos.username}} - {{@data.ujian.lecture->name}}"
24 echo icacls-ing...
25 echo icacls-ing... >> {{@data.logfile}}
26 icacls "\\{{@pos->computer->ip}}\D$\ujian\{{@pos.username}} - {{@data.ujian.lecture->name}}" /inheritance:d /T>> {{@data.logfile}}
27 icacls "\\{{@pos->computer->ip}}\D$\ujian\{{@pos.username}} - {{@data.ujian.lecture->name}}" /T /grant ftis\administrator:(OI)(CI)F ftis\administrator:(OI)(CI)(RX,W,DC) >> {{@data.logfile}}
28 icacls "\\{{@pos->computer->ip}}\D$\ujian\{{@pos.username}} - {{@data.ujian.lecture->name}}" /T /grant "ftis\{{@pos.username}}:(OI)(CI)(RX,W,DC)" /T >> {{@data.logfile}}
29 icacls "\\{{@pos->computer->ip}}\D$\ujian\{{@pos.username}} - {{@data.ujian.lecture->name}}" /T /remove:g "Authenticated Users" >> {{@data.logfile}}
30 icacls "\\{{@pos->computer->ip}}\D$\ujian\{{@pos.username}} - {{@data.ujian.lecture->name}}" /T /remove:g "Users" >> {{@data.logfile}}
31 </f3:repeat>
32 echo
33 echo
34 echo Finished.

```

Listing B.74 lab-takeown.bat

```

1 @echo off
2 echo OXAM Takeowner creation batch
3 echo Logfile will be collected to {{@data.logfile}}.
4 echo
5 echo
6 <f3:repeat group="{{@data.ujian.participants}}" value="{{@pos}}">
7 echo =====
8 echo Working on {{@pos.computer.name}}
9 echo =====
10 echo ===== >> {{@data.logfile}}
11 echo Working on {{@pos.computer.name}} >> {{@data.logfile}}
12 echo ===== >> {{@data.logfile}}
13 echo takeowning...
14 echo takeowning... >> {{@data.logfile}}

15 icacls "\{{@pos->computer->ip}\}\$ujian\{{@pos->username}} - {{@data.ujian->lecture->name}}" /T /deny ftis\Students:(OI)(CI
   ) (RX,W,DC) /remove:g "ftis\{{@pos.username}}" /T 2>> {{@data.logfile}}

16 </f3:repeat>
17 echo
18 echo
19 echo Finished.

```

Listing B.75 special-ujian-fulltakeown.bat

```

1 @echo off
2 echo OXAM special "ujian" folder takeowner
3 echo Logfile will be collected to {{@data.logfile}}.
4 echo
5 echo
6 <f3:repeat group="{{@data.ujian.participants}}" value="{{@pos}}">
7 echo =====
8 echo Working on {{@pos.posisi}}
9 echo =====
10 echo ===== >> {{@data.logfile}}
11 echo Working on {{@pos.posisi}} >> {{@data.logfile}}
12 echo ===== >> {{@data.logfile}}
13 takeown /A /F "\{{@pos.computer->ip}}\$ujian" >> {{@data.logfile}}
14 icacls "\{{@pos.computer->ip}}\$ujian" /inheritance:d /T >> {{@data.logfile}}
15 icacls "\{{@pos.computer->ip}}\$ujian" /T /remove:g "Authenticated Users" >> {{@data.logfile}}
16 icacls "\{{@pos.computer->ip}}\$ujian" /grant Everyone:RX ftis\administrator:F >> {{@data.logfile}}
17 </f3:repeat>

```

Listing B.76 api.php

```

1 <?php
3 namespace View;
5 class Api
6 {
7     public static function error($data, $http_code = 400, $desc = null): string
8     {
9         if (!($data instanceof \Model\Error)) {
10             $data = new \Model\Error(
11                 $data,
12                 $desc,
13                 "HTTP" . $http_code,
14                 null,
15                 $http_code,
16                 null,
17                 null
18             );
19         }
20         return self::serve(\Output\Formatter::instance()->format_error($data));
21     }
23     public static function success($data): string
24     {
25         return self::serve(\Output\Formatter::instance()->format_success($data));
26     }
28     protected static function serve($data): string
29     {
30         $f3 = \F3::instance();
31         if (strtolower($f3->PARAMS['extension']) == 'json' || $f3->PARAMS['extension'] == '') {
32             return \Output\JSON::instance()->serve($data);
33         } else if (strtolower($f3->PARAMS['extension']) == 'xml') {
34             return \Output\XML::instance()->serve($data);
35         } else {
36             return \Output\Plain::instance()->serve($data);
37         }
38     }
39 }

```

Listing B.77 bat.php

```

1 <?php
2 namespace View;

```

```

4 class Bat {
5     public static function render($ba_name, $log_file = "batlog.log") {
6         \F3::mset([
7             "data.logfile" => $log_file,
8         ]);
9         return \Template::instance()->render("_bat/" . $ba_name);
10    }
11 }

```

Listing B.78 reactapp.php

```

1 <?php
2 namespace View;
3
4 class ReactApp {
5     public static function render() {
6         echo \Template::instance()->render("index.html");
7     }
8 }

```

Listing B.79 template.php

```

1 <?php
2 namespace View;
3
4 class Template {
5     public static function render($page_location, $page_title = "Untitled Document") {
6         // \F3::instance()->set("page.title", $page_title);
7         // \F3::instance()->set("page.location", $page_location);
8         \F3::mset([
9             "page.title" => $page_title,
10            "page.location" => $page_location
11        ]);
12        echo \Template::instance()->render("layout.html");
13    }
14 }

```

Listing B.80 docker-compose.dev.yml

```

1 version: '3'
2 services:
3   web:
4     build:
5       context: .
6     ports:
7       - "80:80"
8     volumes:
9       - ./:/var/www/html
10    links:
11      - mysql
12    env_file:
13      - ./env.development
14    environment:
15      APACHE_RUN_USER: '#1000'
16  mysql:
17    image: mysql:5.7
18    command: --default-authentication-plugin=mysql_native_password
19    volumes:
20      - ./mysql-db:/var/lib/mysql
21    env_file:
22      - ./env.development
23  phpmyadmin:
24    image: phpmyadmin/phpmyadmin:4.8
25    ports:
26      - "8000:80"
27    links:
28      - mysql:db
29    env_file:
30      - ./env.development

```

Listing B.81 Dockerfile

```

1 # Change the image version if you want to test it on other php version.
2 # versions are listed on https://hub.docker.com/_/php?tab=tags
3 # always use the --apache tags. It have apache to serve the thing.
4 FROM php:7.3.8-apache

5 RUN apt-get update && apt-get install -y --fix-missing \
6     apt-utils \
7     gnupg

8 RUN echo "deb http://packages.dotdeb.org jessie all" >> /etc/apt/sources.list
9 RUN echo "deb-src http://packages.dotdeb.org jessie all" >> /etc/apt/sources.list
10 RUN curl -sS --insecure https://www.dotdeb.org/dotdeb.gpg | apt-key add -

11 RUN apt-get update && apt-get install -y libfreetype6-dev libjpeg62-turbo-dev libpng-dev \
12     libzip-dev \
13     libssl-dev libldap2-dev libicu-dev locales locales-all

14 WORKDIR /var/www/html

```

```

20 RUN curl -sS https://getcomposer.org/installer | php -- --install-dir=/usr/bin/ --filename=composer

23 # Installing required extensions for the PHP.
24 RUN docker-php-ext-configure intl
25 RUN docker-php-ext-install pdo pdo_mysql mysqli zip ldap intl

27 # Configuring apache and PHP (for development)
28 ENV APACHE_DOCUMENT_ROOT /var/www/html/public_html
29 RUN mv "$PHP_INI_DIR/php.ini-development" "$PHP_INI_DIR/php.ini"
30 RUN sed -ri -e 's!/var/www/html!${APACHE_DOCUMENT_ROOT}!g' /etc/apache2/sites-available/*.conf
31 RUN sed -ri -e 's!/var/www/!${APACHE_DOCUMENT_ROOT}!g' /etc/apache2/apache2.conf /etc/apache2/conf-available/*.conf

33 ENV LANG "id_ID.UTF-8"
34 ENV LANGUAGE "id_ID.UTF-8"
35 ENV LC_ALL "id_ID.UTF-8"
36 ENV LC_CTYPE "id_ID.UTF-8"
37 ENV LC_MESSAGES "id_ID.UTF-8"
38 RUN locale-gen "id_ID.UTF-8"
39 # RUN dpkg-reconfigure locales

42 # Add rewrite engine for APACHE
43 RUN a2enmod rewrite

```

Listing B.82 phunit.xml

```

1 <phpunit bootstrap="tests/autoload.php">
2   <testsuites>
3     <testsuite name="deployment">
4       <directory>tests/system-wide</directory>
5     </testsuite>
6   </testsuites>
7 </phpunit>

```

Listing B.83 router.php

```

1 <?php
2 // taken from https://www.sitepoint.com/taking-advantage-of-phps-built-in-server/
3 if ($php_sapi_name() == "cli-server") {
4   $extensions = array("json", "xml");
5   $path = parse_url($_SERVER["REQUEST_URI"], PHP_URL_PATH);
6   $ext = pathinfo($path, PATHINFO_EXTENSION);
7   if (!in_array($ext, $extensions)) {
8     return false;
9   }
10  $_SERVER['SCRIPT_NAME'] = 'index.php';
11  include "public_html/index.php";
12 }

```

Listing B.84 authenticationTest.php

```

1 <?php
3 use Model\System\User;
4 use Model\System\Acl;
5 use PHPUnit\Framework\TestCase;
7 class AuthenticationTest extends TestCase
8 {
10   /**
11    * User used during the testing periode
12   *
13   * @var \Model\System\User
14   */
15   protected $user = null;
17   /**
18    * Create new user during the test.
19   *
20   * @return void
21   */
22   protected function setUp(): void
23   {
24     // test if user able to do login
25     $acl = new Acl();
26     $acl->load(["id=?", 1]);
27     $user = new User();
28     $user->copyfrom([
29       "username" => "testsuite",
30       "password" => "testsuite",
31       "email" => "testsuite@testingground.labftis.net",
32       "acl" => $acl
33     ]);
34     $user->save();
36     $this->user = $user;
37   }
39   /**

```

```

40     * Remove the testser
41     *
42     * @return void
43     */
44     protected function tearDown(): void
45     {
46         $this->user->erase();
47     }
48
49     public function testLoginShouldFailOnWrongPassword()
50     {
51         // login test:
52         $this->expectException(Exception::class);
53         $user = User::login("testsuite", "defenitely-wrong-password");
54     }
55
56     public function testLoginShouldFailOnWrongUsername()
57     {
58         // login test:
59         $this->expectException(Exception::class);
60         $user = User::login("unknown-username", "testsuite");
61     }
62
63     public function testLoginShouldBeOkOnCorrectCredential() {
64         $user = User::login("testsuite", "testsuite");
65         $this->assertNotNull($user);
66     }
67 }
```

Listing B.85 examinationTest.php

```

1  <?php
2
3  use Model\Ujian\Computer;
4  use Model\Ujian\Exam;
5  use Model\Ujian\Lecture;
6  use Model\Ujian\LecturePeriod;
7  use Model\Ujian\Location;
8  use Model\Ujian\Participant;
9  use PHPUnit\Framework\TestCase;
10
11 class ExaminationTest extends TestCase
12 {
13
14     protected
15     /**
16      * @var Lecture Lecture object
17      */
18     $lecture = null,
19
20     /**
21      * @var LecturePeriod Lecture periode object
22      */
23     $lecturePeriode = null,
24
25     /**
26      * @var Location
27      */
28     $location = null,
29
30     /**
31      * @var Computer
32      */
33     $computer = null;
34
35     protected function setUp(): void
36     {
37         // create a lecture periode and new lecture just for test case.
38         $lecture = new Lecture();
39         $lecture->copyfrom([
40             "name" => "Rescue Exam",
41             "lecture_code" => "HXH98001",
42         ]);
43         $lecture->save();
44         $this->lecture = $lecture;
45
46         $lecturePeriode = LecturePeriod::getLatestPeriod(); // auto generated.
47         $this->lecturePeriode = $lecturePeriode;
48
49         // adding location for computer to reside
50         $location = new Location();
51         $location->copyfrom([
52             "room_name" => "Zoldyck's Family",
53             "name_alias" => "zoldyck"
54         ]);
55         $location->save();
56         $this->location = $location;
57
58         // add computer for self.
59         $computer = new Computer();
60         $computer->copyfrom([
61             "name" => "Testing Gate",
62             "ip" => "127.14.0.1",
63             "reverse_dns" => "localhost",
64             "d_pos" => json_encode([]),
65             "location" => $location->_id,
66         ]);
67         $computer->save();
68         $this->computer = $computer;
69 }
```

```

67     }
68
69     protected function tearDown(): void
70     {
71         // check if lecture periode is used in examination
72         $exam = new Exam();
73         $exam->load(["lecture_period = ?", $this->lecturePeriode->_id]);
74         if ($exam->loaded() == 0) {
75             // delete them as nothing is using them
76             $this->lecturePeriode->erase();
77         }
78
79         // remove lecture periode and lecture that are used on test case.
80         $this->lecture->erase();
81
82         $this->computer->erase();
83         $this->location->erase();
84     }
85
86
87     /**
88      * @test
89      * @testdox Activated exam should be able to be obtained
90      *
91      * Exam should return an examination object when exam is active, within
92      * timeframe and queried.
93      */
94     public function ActiveExamfunction()
95     {
96         $exam = new Exam();
97         $exam->copyfrom([
98             "lecture_period" => $this->lecturePeriode->_id,
99             "time_start" => strtotime("-1 hour"),
100            "time_duration" => 3600 * 2, // a day
101            "lecture" => $this->lecture->_id,
102            "time_opened" => strtotime("-1 hour"),
103            "timeEnded" => strtotime("4" . (3600 * 2) . " seconds"),
104            "uts" => 0,
105            "shift" => null, // no shift!
106        ]);
107        $exam->save();
108
109        $participant = new Participant();
110        $participant->copyfrom([
111            "username" => "gon",
112            "npm" => "109824803287",
113            "exam" => $exam->_id,
114            "computer" => $this->computer->_id
115        ]);
116        $participant->save();
117        // var_dump($participant->cast());
118
119        $participantTest = Participant::getActiveParticipant(true, $this->computer->ip);
120        $this->assertNull($participantTest, "Shouldn't be able to get exam data with upcoming flag active");
121
122        $participantTest = Participant::getActiveParticipant(false, $this->computer->ip);
123        // die(\Base::instance()->DB->log());
124        $this->assertIsObject($participantTest);
125        $this->assertEquals($participant->_id, $participantTest->_id, "Able to fetch exam with explicitly removed upcoming
126        flag");
127
128        // delete exam and participant
129        $exam->erase();
130        $participant->erase();
131    }
132
133    /**
134     * @test
135     * @testdox Upcoming exam should be able to obtained with `upcoming` parameter only.
136     */
137    public function UpcomingExamfunction()
138    {
139        $exam = new Exam();
140        $exam->copyfrom([
141            "lecture_period" => $this->lecturePeriode->_id,
142            "time_start" => strtotime("+5 minutes"),
143            "time_opened" => null,
144            "timeEnded" => null,
145            "time_duration" => 3600 * 2, // a day
146            "lecture" => $this->lecture->_id,
147            "uts" => 0,
148            "shift" => null, // no shift!
149        ]);
150        $exam->save();
151
152        $participant = new Participant();
153        $participant->copyfrom([
154            "username" => "gon",
155            "npm" => "109824803287",
156            "exam" => $exam->_id,
157            "computer" => $this->computer->_id
158        ]);
159        $participant->save();
160
161        $participantTest = Participant::getActiveParticipant(true, $this->computer->ip);
162        $this->assertIsObject($participantTest);
163        // $this->assertEquals(1, count($participantTest), "Able to get upcoming examination data");
164        $this->assertEquals($participant->_id, $participantTest->_id, "Exam data is same");
165    }
166

```

```

166     $participantTest = Participant::getActiveParticipant(false, $this->computer->ip);
167     $this->assertNull($participantTest);
168     // $this->assertEquals(0, count($participantTest), "Able to explicitly remove upcoming exam data");
169
170     // delete exam and participant
171     $exam->erase();
172     $participant->erase();
173 }
174
175 /**
176 * @test
177 * @testdox Closed exam should not appear on result.
178 */
179 public function PostExamfunction()
180 {
181     $exam = new Exam();
182     $exam->copyfrom([
183         "lecture_period" => $this->lecturePeriode->_id,
184         "time_start" => strtotime("-1 day"),
185         "time_duration" => 3600 * 20, // a day
186         "time_opened" => strtotime("-1 day"),
187         "timeEnded" => strtotime("+".(3600 * 20)." seconds", strtotime("-1 day")),
188         "lecture" => $this->lecture->_id,
189         "uts" => 0,
190         "shift" => null, // no shift!
191     ]);
192     $exam->save();
193
194     $participant = new Participant();
195     $participant->copyfrom([
196         "username" => "gon",
197         "npm" => "109824803287",
198         "exam" => $exam->_id,
199         "computer" => $this->computer->_id
200     ]);
201     $participant->save();
202
203     $participantTest = Participant::getActiveParticipant(true, $this->computer->ip);
204     $this->assertNull($participantTest);
205     // $this->assertEquals(0, count($participantTest), "Able to get upcoming examination data");
206
207     $participantTest = Participant::getActiveParticipant(false, $this->computer->ip);
208     $this->assertNull($participantTest);
209     // $this->assertEquals(0, count($participantTest), "Able to explicitly remove upcoming exam data");
210
211     // delete exam and participant
212     $exam->erase();
213     $participant->erase();
214 }
215 }
```

Listing B.86 examinationWithShiftTest.php

```

1 <?php
2
3 use Model\Ujian\Computer;
4 use Model\Ujian\Exam;
5 use Model\Ujian\Lecture;
6 use Model\Ujian\LecturePeriod;
7 use Model\Ujian\Location;
8 use Model\Ujian\Participant;
9 use PHPUnit\Framework\TestCase;
10
11 class ExaminationWithShiftTest extends TestCase
12 {
13
14     protected static
15     /**
16      * @var Lecture Lecture object
17      */
18     $lecture = null,
19
20     /**
21      * @var LecturePeriod Lecture periode object
22      */
23     $lecturePeriode = null,
24
25     /**
26      * @var Location
27      */
28     $location = null,
29
30     /**
31      * @var Computer
32      */
33     $computer = null;
34
35
36     public static function setUpBeforeClass(): void
37     {
38         // create a lecture periode and new lecture just for test case.
39         $lecture = new Lecture();
40         $lecture->copyfrom([
41             "name" => "Rescue Exam",
42             "lecture_code" => "HXH98001",
43         ]);
44         $lecture->save();
45         self::$lecture = $lecture;
46     }
47 }
```

```

45     $lecturePeriode = LecturePeriod::getLatestPeriod(); // auto generated.
46     self::$lecturePeriode = $lecturePeriode;
47
48     // adding location for computer to reside
49     $location = new Location();
50     $location->copyfrom([
51         "room_name" => "Zoldyck's Family",
52         "name_alias" => "zoldyck"
53     ]);
54     $location->save();
55     self::$location = $location;
56
57     // add computer for self.
58     $computer = new Computer();
59     $computer->copyfrom([
60         "name" => "Testing Gate",
61         "ip" => "127.14.0.1",
62         "reverse_dns" => "localhost",
63         "d_pos" => json_encode([]),
64         "location" => $location->_id,
65     ]);
66     $computer->save();
67     self::$computer = $computer;
68 }
69
70 public static function tearDownAfterClass(): void
71 {
72     // check if lecture periode is used in examination
73     $exam = new Exam();
74     $exam->load(["lecture_period = ?", self::$lecturePeriode->_id]);
75     if ($exam->loaded() == 0) {
76         // delete them as nothing is using them
77         self::$lecturePeriode->erase();
78     }
79
80     // remove lecture periode and lecture that are used on test case.
81     $exams = $exam->find(["lecture=?", self::$lecture->_id]);
82     if ($exams === false) {
83         $exams = [];
84     }
85     foreach ($exams as $e) {
86         $participant = new Participant();
87         $participants = $participant->find(["exam = ?", $e->_id]);
88         foreach ($participants as $par) {
89             $par->erase();
90         }
91         $e->erase();
92     }
93     self::$lecture->erase();
94
95     self::$computer->erase();
96     self::$location->erase();
97 }
98
99 /**
100 * Generate Exam
101 *
102 * @return Exam
103 */
104 protected function generateBareExam($baseDate = "-10 minute")
105 {
106     $exam = new Exam();
107     $exam->copyfrom([
108         "lecture_period" => self::$lecturePeriode->_id,
109         "time_start" => strtotime($baseDate),
110         "time_duration" => 3600 * 3,
111         "lecture" => self::$lecture->_id,
112         "uts" => 0,
113         "shift" => null, // no shift!
114     ]);
115     $exam->save();
116
117     if (!self::$computer) {
118         throw new Exception("COMPUTER UNSET!");
119     }
120
121     $participant = new Participant();
122     $participant->copyfrom([
123         "username" => "gon",
124         "npm" => "109824803287",
125         "exam" => $exam->_id,
126         "computer" => self::$computer->_id
127     ]);
128     $participant->save();
129
130     return $exam;
131 }
132
133 /**
134 * Clean following examination
135 *
136 * @param Exam $exam
137 * @return void
138 */
139 protected function cleanGeneratedExam($exam)
140 {
141     $participant = new Participant();
142     $participants = $participant->find(["exam = ?", $exam->_id]);

```

```

143     foreach ($participants as $par) {
144         $par->erase();
145     }
146     $exam->erase();
147 }
148
149     protected function cleanAllExam()
150     {
151         $exam = new Exam();
152         $exams = $exam->find(["lecture=?", self::$lecture->_id]);
153         if ($exams === false) {
154             $exams = [];
155         }
156         foreach ($exams as $e) {
157             $participant = new Participant();
158             $participants = $participant->find(["exam = ?", $e->_id]);
159             foreach ($participants as $par) {
160                 $par->erase();
161             }
162             $e->erase();
163         }
164     }
165
166 /**
167 * Test data will consist of following arrays (AS CALLBACK!)
168 * - 0 => Shifts
169 * - 1 => Expected exam order
170 *   - 0 => upcoming
171 *   - 1 => active
172 *
173 * @return array datas
174 */
175 public function examProvider(): array
176 {
177     $testData = [];
178     // both are upcoming =====
179     $testData[] = function () {
180         // first shift
181         $exam1 = $this->generateBareExam(); //normally upcoming
182         $exam1->shift = 1;
183         $exam1->save();
184
185         // second shift
186         $exam2 = $this->generateBareExam(); //normally upcoming
187         $exam2->time_start = strtotime("+" . $exam1->time_duration . " seconds", strtotime($exam1->time_start));
188         $exam2->shift = 2;
189         $exam2->save();
190
191         return [
192             [$exam1, $exam2],
193             [
194                 $exam1,
195                 null
196             ]
197         ];
198     };
199
200     // 1 is active, 1 is upcoming =====
201     $testData[] = function () {
202         // first shift
203         $exam1 = $this->generateBareExam("-175 minutes"); //normally upcoming
204         $exam1->shift = 1;
205         $exam1->time_ended = strtotime("+" . $exam1->time_duration . " seconds", strtotime($exam1->time_start));
206         $exam1->time_opened = $exam1->time_start; // open the time
207         $exam1->save();
208
209         // second shift
210         $exam2 = $this->generateBareExam("-5 minutes"); //normally upcoming
211         $exam2->shift = 2;
212         $exam2->save();
213
214         return [
215             [$exam1, $exam2],
216             [
217                 $exam2,
218                 $exam1
219             ]
220         ];
221     };
222
223     // 1 is closed, 1 is upcoming =====
224     $testData[] = function () {
225         // first shift
226         $exam1 = $this->generateBareExam("-180 minutes"); //normally upcoming
227         $exam1->shift = 1;
228         $exam1->time_opened = $exam1->time_start;
229         $exam1->time_ended = $exam1->time_start; // close the time
230         $exam1->save();
231
232         // second shift
233         $exam2 = $this->generateBareExam(); //normally upcoming
234         $exam2->shift = 2;
235         $exam2->save();
236
237         return [
238             [$exam1, $exam2],
239             [
240                 $exam2,
241                 null
242             ]
243         ];
244     };
245 }
```

```

242         ]
243     ];
244 }
245
246 // 1 is closed, 1 is active =====
247 $testData[] = function () {
248     // first shift
249     $exam1 = $this->generateBareExam("-180 minutes"); //normally upcoming
250     $exam1->shift = 1;
251     $exam1->time_opened = $exam1->time_start;
252     $exam1->time_ended = $exam1->time_start; // close the time
253     $exam1->save();
254
255     // second shift
256     $exam2 = $this->generateBareExam(); //normally upcoming
257     $exam2->shift = 2;
258     $exam2->time_opened = $exam2->time_start;
259     $exam2->time_ended = strtotime("+ " . $exam2->time_duration . " seconds");
260     $exam2->save();
261
262     return [
263         [$exam1, $exam2],
264         [
265             null,
266             $exam2
267         ]
268     ];
269 }
270
271 // both are closed. =====
272 $testData[] = function () {
273     // first shift
274     $exam1 = $this->generateBareExam("-6 hours"); //normally upcoming
275     $exam1->shift = 1;
276     $exam1->time_opened = $exam1->time_start;
277     $exam1->time_ended = $exam1->time_start; // close the time
278     $exam1->save();
279
280     // second shift
281     $exam2 = $this->generateBareExam("-3 hours"); //normally upcoming
282     $exam2->time_start = strtotime("+ " . $exam1->time_duration . " seconds", strtotime($exam1->time_start));
283     $exam2->shift = 2;
284     $exam2->time_opened = $exam2->time_start;
285     $exam2->time_ended = $exam2->time_start; // close the time
286     $exam2->save();
287
288     return [
289         [$exam1, $exam2],
290         [
291             null,
292             null
293         ]
294     ];
295 }
296
297
298     return array_map(function ($data) {
299         return [$data];
300     }, $testData);
301 }
302
303 /**
304 * @test
305 * @testdox Exam shift test
306 *
307 * @dataProvider examProvider
308 * @skipTest
309 */
310 public function SuperExamTestfunction($dataCallback)
311 {
312     $this->cleanAllExam();
313     $ testcase = $dataCallback();
314
315     $participantUpcoming = Participant::getActiveParticipant(true, self::$computer->ip);
316     $participantActive = Participant::getActiveParticipant(false, self::$computer->ip);
317
318     $expectedUpcoming = $testcase[1][0];
319     $expectedActive = $testcase[1][1];
320
321     // detect first
322     if ($expectedUpcoming === null) {
323         $this->assertNull($participantUpcoming);
324     } else {
325         $this->assertNotNull($participantUpcoming);
326         $this->assertEquals($expectedUpcoming->_id, $participantUpcoming->_id, "Upcoming exam match");
327     }
328
329     if ($expectedActive === null) {
330         $this->assertNull($participantActive);
331     } else {
332         $this->assertNotNull($participantActive);
333         $this->assertEquals($expectedActive->_id, $participantActive->_id, "Active exam match");
334     }
335
336     // clearance.
337     foreach ($testcase[0] as $exam) {
338         $this->cleanGeneratedExam($exam);
339     }
340 }

```

```
341     }
342 }
```

Listing B.87 config-overrides.js

```
1 const { override, addBabelPlugins } = require('customize-cra');

3 module.exports = override(
4   ...addBabelPlugins(['root-import', {
5     rootPathPrefix: '~',
6     rootPathSuffix: 'src',
7   }])
8 )
```

Listing B.88 apicall.js

```
1 import axios from "axios";

3 const apicall = axios.create({
4   baseURL: '/api/v1'
5 });

7 apicall.interceptors.request.use(function (config) {
8   if (window.localStorage.getItem('auth-token')) {
9     config.headers['Authorization'] = 'Bearer ' + window.localStorage.getItem('auth-token');
10  } else {
11    if (config.headers.hasOwnProperty('Authorization')){
12      delete config.headers['Authorization']
13    }
14  }
15  return config;
16 }, function (error) {
17   return Promise.reject(error);
18 });

21 function setAuth(IDToken) {
22   window.localStorage.setItem('auth-token', IDToken);
23   apicall.defaults.headers.common['Authorization'] = 'Bearer ' + IDToken;
24 }
25 function clearAuth() {
26   window.localStorage.removeItem('auth-token');
27   delete apicall.defaults.headers.common['Authorization'];
28 }

30 export { apicall as axios, setAuth, clearAuth };
```

Listing B.89 App.js

```
1 import React, { Component } from 'react';
2 import { BrowserRouter, Route, Switch } from "react-router-dom";
3 import MainIndex from '~/pages/index/Index';
4 import AdminController from '~/pages/admin/Admin';
5 import ExamController from '~/pages/exam/Exam';
6 import { library } from '@fortawesome/fontawesome-svg-core'

8 import ExamStore from '~/store/examStore';
9 import { Provider } from "mobx-react";
10 import { faMehRollingEyes, faSurprise, faMeh, faGrinTears, faDizzy, faLemon, faUserCircle, faBell } from '@fortawesome/free-regular-svg-icons'
11 import { faCloudUploadAlt } from "@fortawesome/free-solid-svg-icons";
12 import AdminStore from '~/store/adminStore';
13 import EntityStore from '~/store/entityStore';
14 import AutonomusIndex from '~/pages/autonomus';
15 library.add(faMehRollingEyes, faSurprise, faMeh, faGrinTears, faDizzy, faLemon, faUserCircle, faBell, faCloudUploadAlt);

17 class App extends Component {
18   render() {
19     return (
20       <Provider examStore={new ExamStore()} adminStore={new AdminStore()} entityStore={new EntityStore()}>
21         <BrowserRouter>
22           <Switch>
23             <Route exact path="/" component={MainIndex} />
24             <Route path="/exam" component={ExamController} />
25             <Route path="/admin" component={AdminController} />
26             <Route path="/autonomus" component={AutonomusIndex} />
27           </Switch>
28         </BrowserRouter>
29       </Provider>
30     );
31   }
32 }

34 export default App;
```

Listing B.90 App.test.js

```
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import App from './App';
```

```

5 it('renders without crashing', () => {
6   const div = document.createElement('div');
7   ReactDOM.render(<App />, div);
8   ReactDOM.unmountComponentAtNode(div);
9 });

```

Listing B.91 AdminNavbar.js

```

1 import React, { Component } from 'react';
2 import { observer, inject } from 'mobx-react';
3
4 import { NavLink as Link, withRouter } from 'react-router-dom';
5
6 import {
7   Collapse,
8   Navbar,
9   NavbarToggler,
10  NavbarBrand,
11  Nav,
12  NavItem,
13  NavLink,
14  UncontrolledDropdown,
15  DropdownToggle,
16  DropdownMenu,
17  DropdownItem,
18  Container
19 } from 'reactstrap';
20
21 class AdminNavbar extends Component {
22   state = {
23     isOpen: false
24   }
25
26   toggle() {
27     const { isOpen } = this.state;
28     this.setState({ isOpen: !isOpen });
29   }
30
31   componentDidMount() {
32     const { adminStore, history } = this.props;
33     adminStore.fetchProfile().catch(e => {
34       history.push('/admin/account/login');
35     })
36   }
37
38   handleLogout() {
39     const { adminStore, history } = this.props;
40     adminStore.userLogout().then(() => {
41       history.push('/admin/account/login');
42     });
43   }
44
45   render() {
46     const { adminStore } = this.props;
47     const { isOpen } = this.state;
48
49
50     return (
51       <div>
52         <Navbar color="dark" dark expand="md">
53           <Container>
54             <NavbarBrand href="/admin" to="/admin" tag={Link}>Oxam</NavbarBrand>
55             <NavbarToggler onClick={() => this.toggle()} />
56             <Collapse isOpen={isOpen} navbar>
57               <Nav className="mr-auto" navbar>
58                 <NavItem>
59                   <NavLink to="/admin/exam/" tag={Link}>Ujian</NavLink>
60                 </NavItem>
61                 <UncontrolledDropdown nav inNavbar>
62                   <DropdownToggle nav caret>Exam Params</DropdownToggle>
63                   <DropdownMenu right>
64                     <DropdownItem tag={Link} to="/admin/manage/lectures">Lectures</DropdownItem>
65                     <DropdownItem tag={Link} to="/admin/manage/lectureperiods">Lecture Periods</DropdownItem>
66                     <DropdownItem tag={Link} to="/admin/manage/computers">Computers</DropdownItem>
67                     <DropdownItem tag={Link} to="/admin/manage/locations">Location</DropdownItem>
68                   </DropdownMenu>
69                 </UncontrolledDropdown>
70                 <UncontrolledDropdown nav inNavbar>
71                   <DropdownToggle nav caret>
72                     Oxam Params
73                   </DropdownToggle>
74                   <DropdownMenu right>
75                     <DropdownItem tag={Link} to="/admin/manage/acls">Accls</DropdownItem>
76                     <DropdownItem tag={Link} to="/admin/manage/admins">Admins</DropdownItem>
77                     <DropdownItem tag={Link} to="/admin/manage/iplogins">IP Login</DropdownItem>
78                   </DropdownMenu>
79                 </UncontrolledDropdown>
80               </Nav>
81             <Nav navbar>
82               <UncontrolledDropdown nav inNavbar>
83                 <DropdownToggle nav caret>
84                   {@adminStore?.user?.username || "Reg"}
85                 </DropdownToggle>
86                 <DropdownMenu right>
87                   <DropdownItem tag={Link} to={"/admin/manage/admins/" + adminStore?.user?.id}>Account Setting</DropdownItem>

```

```

88             <DropdownItem onClick={this.handleLogout.bind(this)}>Logout</DropdownItem>
89         </DropdownMenu>
90     </UncontrolledDropdown>
91   </Nav>
92   </Collapse>
93 </Container>
94 </Navbar>
95 </div>
96 );
97 }
98 }

100 export default inject("adminStore")(
101   withRouter(observer(AdminNavbar))
102 );

```

Listing B.92 Buildinfo.js

```

1 import React from 'react';
2 import { observer } from 'mobx-react';
3
4 import moment from 'moment';
5 import 'moment/locale/id';
6
7 moment.locale("id");
8
9 const Buildinfo = () => {
10   return (
11     <>
12       <b>0xam v5.0.</b>: Into the abyss | Built on {moment(process.env.REACT_APP_BUILD_DATE || Date.now()).format("LLL")},
13       Commit: {process.env.REACT_APP_BUILD_COMMIT || "dev"}
14     </>
15   )
16 Buildinfo.propTypes = {
17 }
18
19 export default (observer(Buildinfo));

```

Listing B.93 computer-container.js

```

1 import React from 'react';
2 import { Col, Row } from 'reactstrap'
3 import Computer from './computer'
4 import { observer } from 'mobx-react';
5
6 function ComputersContainer({ computers = [], editable = true, onComputerClick = () => {} }) {
7
8   // PROCESSOR
9   let rows = {};
10  computers.map(comp => {
11    if (!rows[comp.d_pos.y]) {
12      rows[comp.d_pos.y] = [];
13    }
14    rows[comp.d_pos.y].push(comp);
15    return rows;
16  })
17
18  // reurut:
19  let comp = Object.values(rows).sort((a, b) => (a[0].d_pos.y - b[0].d_pos.y)).map((el) => el.sort((a, b) => (a.d_pos.x - b.d_pos.x)));
20  return (
21    <>
22      {comp.map((el, i) => <Row key={i}>
23        {el.map((com, c, arr) => {
24          let offset = 0;
25
26          if (c === 0 && com.d_pos.x > 1) {
27            offset = com.d_pos.x - 1;
28          } else if (c > 0 && com.d_pos.x - arr[c - 1].d_pos.x !== 1) {
29            offset = com.d_pos.x - arr[c - 1].d_pos.x - 1;
30          }
31
32          return <Col xs={{ size: 2, offset: (offset * 2) }} key={c}>
33            <Computer
34              setting={com}
35              selectable={editable}
36              selected={!com.selected}
37              onClick={() => onComputerClick(com)}
38            />
39          </Col>
40        }))
41      </Row>)
42    </>
43  )
44 }
45
46 export default observer(ComputersContainer);

```

Listing B.94 computer.js

```
| 1 import React from 'react'
```

```

2 import { createUseStyles } from 'react-jss'
3 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
4 import { faDesktop } from '@fortawesome/free-solid-svg-icons'
5 import { observer } from 'mobx-react'

7 const useStyle = createUseStyles({
8   root: ({ selected, selectable }) => ({
9     display: "flex",
10    alignItems: "center",
12    color: selected ? "var(--light)" : "var(--dark)",
13    backgroundColor: (!selected) ? null : "var(--primary)",
15    borderColor: (selected ? "var(--gray)" : "var(--dark))",
16    borderStyle: "solid",
17    borderWidth: 0,
19    marginTop: 10,
20    marginBottom: 10,
22    cursor: selectable ? "pointer" : "unset",
24    "&:hover": {
25      backgroundColor: "var(--gray)",
26    },
28    "& .icon": {
29      padding: "0.3rem"
30    },
31    "& .content": {
32      flexGrow: 1,
33      padding: "0.3rem 1rem",
35      "& .computer-name": {
36        }
37    },
38  },
39  orientLeft: {
40    paddingLeft: 3,
41    borderLeftWidth: "3px !important"
42  },
43  orientRight: {
44    paddingRight: 3,
45    borderRightWidth: "3px !important",
46    flexDirection: "row-reverse",
47    textAlign: "right"
48  },
49 })
51 function Computer({ setting, selected = false, onClick = () => {} , selectable = false }) {
53   const styles = useStyle({
54     selected: selected,
55     selectable: selectable
56   });
58   let orientClass = styles.orientLeft;
59   if ((setting.d_pos || {}).orient === "r") {
60     orientClass = styles.orientRight
61   }
63   return (
64     <div className={`${styles.root} ${orientClass}`}>
65       <div className="icon"><FontAwesomeIcon icon={faDesktop} /></div>
66       <div className="content">
67         <span className="computer-name text-monospace">{setting.name || "Computer"}</span>
68       </div>
69     </div>
70   )
71 }
73 export default observer(Computer);

```

Listing B.95 countdown.js

```

1 import React, { useState, useEffect } from 'react'
3 function CountDown({ secondsLeft, onEnded = () => {} }) {
5   const [internalTimer, setInternalTimer] = useState(secondsLeft)
6   useEffect(() => {
7     setInternalTimer(secondsLeft);
8     return () => {};
9   }, [secondsLeft])
12   useEffect(() => {
13     let timer = null;
14     let actualTime = internalTimer;
16     timer = setInterval(() => {
17       setInternalTimer(Math.max(actualTime - 1, 0));
18       actualTime = Math.max(actualTime - 1, 0);
19       if (actualTime === 0) {
20         onEnded();
21       }
22     }, 1000);

```

```

24     return () => {
25       clearInterval(timer);
26     };
27   // eslint-disable-next-line
28 }, []);
29
30 let formatters = [
31   Math.floor(internalTimer / 3600),
32   Math.floor((internalTimer % 3600) / 60),
33   (internalTimer % 60),
34 ];
35
36 return (
37   <span>
38     {formatters.map(e => String(e).padStart(2, 0)).join(":")}
39   </span>
40 )
41 }
42 export default CountDown

```

Listing B.96 date-time-picker.js

```

1 import React, { useState, useEffect } from 'react'
2 import { Row, Col, Input, InputGroup } from 'reactstrap'
3 import { DayPickerSingleDateController } from "react-dates";
4 import 'react-dates/initialize';
5 import 'react-dates/lib/css/_datepicker.css';
6 import './datepicker-override.scss';
7 import moment from "moment";
8 import { When } from 'react-if';
9 import TimeKeeper from 'react-timekeeper';

11 function DateTimePicker({ defaultValue = new Date(), onChange = () => null, disabled = false }) {
12   const [selectedDate, setSelectedDate] = useState(moment(defaultValue))
13   const [dpOpen, setDpOpen] = useState(false);
14   const [tpOpen, setTpOpen] = useState(false);

16   function handleDateChange(day) {
17     setSelectedDate(day);
18     setDpOpen(false);
19   }

21   function handleTimeChange(time) {
22     let m = moment(selectedDate);
23     m.hour(time.hour);
24     m.minutes(time.minute);
25     setSelectedDate(m);
26     setTpOpen(true);
27   }

29   useEffect(() => {
30     onChange(selectedDate);
31     return () => { };
32   }, [selectedDate, onChange])

34   return (
35     <Row>
36       <Col>
37         <InputGroup>
38           <Input className="bg-white" type="text" disabled={disabled} value={selectedDate.format("YYYY-MM-DD")}
39             onClick={() => { setDpOpen(true); setTpOpen(false) }} onChange={() => null} />
40           <Input className="bg-white" type="text" disabled={disabled} readOnly value={selectedDate.format("HH:mm")}
41             onClick={() => { setTpOpen(true); setDpOpen(false) }} onChange={() => null} />
42         </InputGroup>
43         <When condition={dpOpen}>
44           <DayPickerSingleDateController
45             date={selectedDate}
46             numberOfWorkdays={2}
47             onDateChange={handleDateChange}
48             onOutsideClick={() => setDpOpen(false)} />
49         </When>
50         <When condition={tpOpen}>
51           <TimeKeeper hour24Mode
52             time={selectedDate.format("HH:mm")}
53             onDoneClick={handleTimeChange}
54             switchToMinuteOnHourSelect
55             closeOnMinuteSelect />
56         </When>
57       </Col>
58     </Row>
59   )
60 export default DateTimePicker

```

Listing B.97 datepicker-override.scss

```

1 .DayPicker {
2   position: absolute;
3   z-index: 500;
4 }
5
6 .react-timekeeper {

```

```

7     position: absolute !important;
8     z-index: 500;
9 }

```

Listing B.98 Navbar.js

```

1 import React, { Component } from 'react';
2 import PropTypes from 'prop-types';
3 import { observer } from 'mobx-react';
4 import Navbar from 'reactstrap/lib/Navbar';
5 import Nav from 'reactstrap/lib/Nav';
6 import NavItem from 'reactstrap/lib/NavItem';
7 import NavLink from 'reactstrap/lib/NavLink';
8 import NavbarBrand from 'reactstrap/lib/NavbarBrand';
9 import { FontAwesomeIcon } from "@fortawesome/react-fontawesome";
10 import { Link } from "react-router-dom";
11 import { UncontrolledDropdown, DropdownToggle, DropdownMenu, DropdownItem } from 'reactstrap';

13 class NavbarExam extends Component {
14   render() {
15     const { type, participant, onNotifShowRequested = () => {} } = this.props;

17   const { notifications = [] } = participant;
18   return (
19     <Navbar color={({type === "inprogress") ? "info" : "warning"}) light={type !== "inprogress"} dark={type === "inprogress"} expand="xs">
20       <NavbarBrand href="/exam" to="/exam" tag={Link}>LabIF UNPAR</NavbarBrand>
21       <Nav className="ml-auto" navbar>
22         <UncontrolledDropdown nav inNavbar>
23           <DropdownToggle nav caret>
24             <FontAwesomeIcon icon={['far', 'bell']} />
25           </DropdownToggle>
26           <DropdownMenu right>
27             {(notifications || []).map((item) => <DropdownItem key={"notification" + item._id} onClick={() => onNotifShowRequested(item)}>
28               {item.title}
29             </DropdownItem>
30           )}
31           </DropdownMenu>
32         </UncontrolledDropdown>

34       <NavItem className="bg-dark px-2 ml-3" style={{ borderRadius: "2rem" }}>
35         <NavLink className="text-light"><FontAwesomeIcon icon={['far', 'user-circle']} /> {participant.username}</NavLink>
36       </NavItem>
37     </Nav>
38   </Navbar>
39 )
40 }
41 }
42 NavbarExam.propTypes = {
43   type: PropTypes.oneOf(["inprogress", "upcoming"]),
44   participant: PropTypes.object.isRequired
45 }

47 NavbarExam.defaultProps = {
48   type: "inprogress"
49 }
50 export default (observer(NavbarExam));

```

Listing B.99 Helmet.js

```

1 import React from 'react';
2 import { Helmet } from "react-helmet";
3 import { helmet } from "~/config";

5 const WrappedHelmet = (props) => {
6   return (
7     <Helmet {...helmet} {...props}/>
8   )
9 }

11 export default WrappedHelmet;

```

Listing B.100 use-store.js

```

1 import React from 'react';
2 import { MobXProviderContext } from 'mobx-react'

4 // eslint-disable-next-line
5 import AdminStore from '~/store/adminStore';
6 // eslint-disable-next-line
7 import ExamStore from '~/store/examStore';
8 // eslint-disable-next-line
9 import EntityStore from '~/store/entityStore';

11 function useStores() {
12   return React.useContext(MobXProviderContext)
13 }

15 export default useStores;

```

```
17 /**
18  * Use Exam Store
19  * @return {ExamStore} exam store
20 */
21 export function useExamStore() {
22   const { examStore } = useStores();
23   return examStore;
24 }

26 /**
27  * Use AdminStore
28  * @return {AdminStore} admin store
29 */
30 export function useAdminStore() {
31   const { adminStore } = useStores();
32   return adminStore;
33 }

35 /**
36  * Use EntityStore
37  * @return {EntityStore} entity store
38 */
39 export function useEntityStore() {
40   const { entityStore } = useStores();
41   return entityStore;
42 }
```

Listing B.101 config.js

```
1 export let helmet = {  
2   titleTemplate: "%s - Oxam v5",  
3   defer: false  
4 }
```

Listing B.102 index.js

```
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.scss';
4 import App from './app/App';
5 import * as serviceWorker from './serviceWorker';

7 ReactDOM.render(<App />, document.getElementById('root'));

9 // If you want your app to work offline and load faster, you can change
10 // unregister() to register() below. Note this comes with some pitfalls.
11 // Learn more about service workers: http://bit.ly/CRA-PWA
12 serviceWorker.unregister();
```

Listing B.103 index.scss

```
1 /*!
2 *
3 *      [ ]      /--| | |
4 *      ---| |-- _---| | | |
5 *      / --| | -\ / -\ \ / | --| |
6 *      ( _| | | | _// / | | | |
7 *      \---| | -| \---/---| | |
8 *
9 * Crafted by Chez14
10 *
11 * Oxam is a part of Informatika UNPAR's Family.
12 * This project uses Bootstrap to fuel the front end.
13 */
14 @import "~opensans-npm-webfont/style.css";
15 $font-family-sans-serif: "Open Sans", -apple-system, BlinkMacSystemFont, Segoe UI, Roboto, Helvetica Neue, Arial, sans-serif, Apple Color Emoji, Segoe UI Emoji, Segoe UI Symbol;
16 @import "~bootstrap/scss/functions";
17 @import "~bootstrap/scss/variables";
18 @import "~bootstrap/scss/mixins";
19 $headings-margin-bottom: ($spacer /1.7 );
20 $headings-font-weight: 100;
21 $headings-line-height: 1.2;
22 @import "~bootstrap/scss/root";
23 @import "~bootstrap/scss/reboot";
24 @import "~bootstrap/scss/type";
25 @import "~bootstrap/scss/images";
26 @import "~bootstrap/scss/code";
27 @import "~bootstrap/scss/grid";
28 @import "~bootstrap/scss/tables";
29 @import "~bootstrap/scss/forms";
30 @import "~bootstrap/scss/buttons";
31 @import "~bootstrap/scss/transitions";
32 @import "~bootstrap/scss/dropdown";
33 @import "~bootstrap/scss/button-group";
34 @import "~bootstrap/scss/input-group";
35 @import "~bootstrap/scss/custom-forms";
36 @import "~bootstrap/scss/nav";
37 @import "~bootstrap/scss/navbar";
38 @import "~bootstrap/scss/card";
39 @import "~bootstrap/scss/breadcrumb";
40 @import "~bootstrap/scss/pagination";
```

```

41   @import "~bootstrap/scss/badge";
42   @import "~bootstrap/scss/jumbotron";
43   @import "~bootstrap/scss/alert";
44   @import "~bootstrap/scss/progress";
45   @import "~bootstrap/scss/media";
46   @import "~bootstrap/scss/list-group";
47   @import "~bootstrap/scss/close";
48   @import "~bootstrap/scss/toasts";
49   @import "~bootstrap/scss/modal";
50   @import "~bootstrap/scss/tooltip";
51   @import "~bootstrap/scss/popover";
52   @import "~bootstrap/scss/carousel";
53   @import "~bootstrap/scss/spinners";
54   @import "~bootstrap/scss/utilities";
55   @import "~bootstrap/scss/print";

57   .h-100vh {
58     height: 100vh;
59   }

```

Listing B.104 Login.js

```

1 import React, { Component } from 'react';
2 import Card from "reactstrap/lib/Card";
3 import CardBody from "reactstrap/lib/CardBody";
4 import Container from "reactstrap/lib/Container";
5 import Row from "reactstrap/lib/Row";
6 import Col from "reactstrap/lib/Col";
7 import Form from "reactstrap/lib/Form";
8 import FormGroup from "reactstrap/lib/FormGroup";
9 import Label from "reactstrap/lib/Label";
10 import Input from "reactstrap/lib/Input";
11 import Button from "reactstrap/lib/Button";
12 import FormText from "reactstrap/lib/FormText";
13 import Alert from "reactstrap/lib/Alert";

15 import { Link } from "react-router-dom";
16 import Buildinfo from './components/buildinfo/Buildinfo';
17 import { withRouter } from "react-router";
18 import { inject, observer } from "mobx-react";

20 class Login extends Component {

22   state = {
23     username: "",
24     password: "",
25     alert: null
26   }

28   submitHelper(e) {
29     e.preventDefault();
30     const { adminStore, history } = this.props;
31     const { username, password } = this.state;
32     adminStore.tryLogin(username, password).then(e => {
33       history.push("/admin/");
34     }).catch(e => {
35       if (e.response) {
36         this.setState({ alert: "Error: " + e.response.data.error.title + ": " + e.response.data.error.description });
37       } else {
38         this.setState({ alert: "Something happened, umm... idk what. Please check your connection and console.logs." });
39       }
40     })
41   }

43   handleIPLogin(e) {
44     e.preventDefault();
45     const { adminStore, history } = this.props;
46     adminStore.tryIPLogin().then(e => {
47       history.push("/admin/screen");
48     }).catch(e => {
49       if (e.response) {
50         this.setState({ alert: "Error: " + e.response.data.error.title + ": " + e.response.data.error.description });
51       } else {
52         this.setState({ alert: "Something happened, umm... idk what. Please check your connection and console.logs." });
53       }
54     })
55   }

57   render() {
58     const { alert } = this.state;
59
60     return (
61       <>
62         <Container>
63           <Row className="h-100vh align-items-center justify-content-center">
64             <Col xs={12} md={8} lg={6}>
65               <Card>
66                 <CardBody>
67                   <h3>Oam Admin Login</h3>
68                   <Alert color="danger" isOpen={alert !== null} toggle={e => this.setState({ alert: null })}>
69                     {alert}
70                   </Alert>
71                   <Form onSubmit={(e) => this.submitHelper(e)}>
72                     <FormGroup>
73                       <Label>Admin username</Label>
74                       <Input type="text" placeholder="admin, maybe." onChange={e => this.setState({ username: e.target.value })} />

```

```

75     </FormGroup>
76     <FormGroup>
77         <Label>Password</Label>
78         <Input type="password" placeholder="Our secret words, it's only between us." onChange={e =>
79             this.setState({ password: e.target.value })} />
80         <FormText color="muted">
81             Have some difficulties, try our <Link to="/admin/account/recover">Forgot Password</Link> tool here.
82         </FormText>
83     </FormGroup>
84     <FormGroup className="text-right">
85         <Button color="secondary" type="button" onClick={this.handleIPLogin.bind(this)}>Or Login with IP</
86             Button> &nbsp;
87         <Button color="primary">Login</Button>
88     </FormGroup>
89 </CardBody>
90 <Row>
91     <Col>
92         <p><small><Buildinfo /></small>
93     </p>
94 </Row>
95 </Container>
96 </>
97 )
98 }
99 }

101 export default inject("adminStore")(
102     withRouter(
103         observer(Login)
104     );

```

Listing B.105 Admin.js

```

1 import React from 'react';
2 import { observer } from 'mobx-react';

4 import { Route, Switch } from "react-router-dom";
5 import Login from './account/login/Login';
6 import Exam from './exam/Index';
7 import Index from './index/Index';
8 import Manage from './manage/Manage';
9 import ScreenIndex from './screen';

11 function Admin({ match }) {
12     return (
13         <>
14             <Switch>
15                 <Route exact path={match.path} component={Index} />
16                 <Route path={match.path + "/account/login"} component={Login} />
17                 <Route path={match.path + "/exam"} component={Exam} />
18                 <Route path={match.path + "/screen"} component={ScreenIndex} />
19                 <Route path={match.path + "/manage"} component={Manage} />
20             </Switch>
21         </>
22     )
23 }
25 export default (observer(Admin));

```

Listing B.106 create.js

```

1 import React, { useState } from 'react'
2 import { Container, Row, Col } from 'reactstrap'
3 import Stepper from "react-stepper-horizontal";
4 import { When } from 'react-if';
5 import Step1 from './step-1';
6 import Step2 from './step-2';
7 import Step3 from './step-3';
8 import Step4 from './step-4';

11 let steps = [
12     { title: "Exam Details" },
13     { title: "Seat Plotting" },
14     { title: "Confirmation" },
15     { title: "Finish" },
16 ];
17 function ExamCreate() {
18     const [examDetails, setExamDetails] = useState({});
19     const [currentStep, setCurrentStep] = useState(0);

22     function handleStep1Confirmation(data) {
23         if (data.peserta.length === 0) {
24             return alert("Peserta kosong!");
25         }
26         setExamDetails({ ...examDetails, ...data });
27         setCurrentStep(currentStep + 1);
28     }
29     function handleStep2Confirmation(data) {
30         setExamDetails({ ...examDetails, ...{ computers: data } });
31         setCurrentStep(currentStep + 1);
32     }

```

```

34     function handleStep3Confirmation(data) {
35         setExamDetails({ ...examDetails, ...{ exam: data } });
36         setCurrentStep(currentStep + 1);
37     }
38     return (
39         <div>
40             <Container>
41                 <Row>
42                     <Col xs="12">
43                         <h2 className="my-3">Examination Plotter</h2>
44
45                         <Stepper
46                             steps={steps.map((step, i) => ({ ...step, onClick: (e) => setCurrentStep(i) }))}
47                             activeStep={currentStep}
48                             activeColor="var(--primary)"
49                             completeBarColor="var(--info)"
50                             completeColor="var(--info)">
51                     />
52                 </Col>
53             </Row>
54             <Row>
55                 <Col>
56                     <When condition={currentStep === 0}><Step1 onRequestNext={handleStep1Confirmation} examDetails={examDetails} /></When>
57                     <When condition={currentStep === 1}><Step2 onRequestNext={handleStep2Confirmation} examDetails={examDetails} /></When>
58                     <When condition={currentStep === 2}><Step3 onRequestNext={handleStep3Confirmation} examDetails={examDetails} /></When>
59                     <When condition={currentStep === 3}><Step4 examDetails={examDetails} /></When>
60                 </Col>
61             </Row>
62         </Container>
63     )
64 }
65 }

66 export default ExamCreate

```

Listing B.107 LocationComputerStore.js

```

1 import { decorate, observable } from "mobx";
3 class ComputerLocationStore {
4     locationComputer = [];
5 }
7 decorate(ComputerLocationStore, {
8     locationComputer: observable
9 })
11 export default ComputerLocationStore;

```

Listing B.108 step-1.js

```

1 import React, { useState, useEffect } from 'react'
2 import { observer } from 'mobx-react'
3 import { Form, FormGroup, Input, Row, Col, Label, Alert, Button, ButtonGroup, InputGroup, InputGroupAddon } from 'reactstrap';
4 import Datepicker from '~components/date-time-picker/date-time-picker';
5 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
6 import { faChevronRight } from '@fortawesome/free-solid-svg-icons';
7 import { axios } from '~apicall';
8 import moment from "moment";
9 import "moment/locale/id";

11 moment.locale('id');

13 const quickPick = [
14     120,
15     110,
16     105
17 ]

19 function Step1({ examDetails, onRequestNext }) {
21     const [lectures, setLectures] = useState([]);
23     const [tipeUjian, setTipeUjian] = useState(1)
24     const [mataKuliah, setMataKuliah] = useState(-1)
25     const [shift, setShift] = useState(0)
26     const [timeStart, setTimeStart] = useState(moment())
27     const [duration, setDuration] = useState(7200)
28     const [peserta, setPeserta] = useState("")

30     useEffect(() => {
31         if (examDetails.uts) {
32             setTipeUjian(examDetails.uts);
33         }
35         if (examDetails.lecture) {
36             setMataKuliah(examDetails.lecture)
37         }
39         if (examDetails.shift) {

```

```

40         setShift(examDetails.shift)
41     }
42
43     if (examDetails.timeStart) {
44       setTimeStart(moment(examDetails.timeStart))
45     }
46
47     if (examDetails.duration) {
48       setDuration(examDetails.duration)
49     }
50
51     if (examDetails.peserta) {
52       setPeserta((examDetails.peserta || []).join("\n"));
53     }
54
55     return () => { };
56   }, [examDetails]);
57
58   useEffect(() => {
59     axios.get("manage/lecture").then(data => {
60       setLectures(data.data.data);
61     })
62     return () => { };
63   }, [])
64
65   function handleSubmission(e) {
66     e.preventDefault();
67     let examDetails = {
68       uts: tipeUjian,
69       lecture: mataKuliah,
70       // eslint-disable-next-line
71       lecture_obj: lectures.filter(e => e._id == mataKuliah)[0],
72       shift: shift,
73       timeStart: timeStart.format("YYYY-MM-DD HH:mm:ss"),
74       duration: duration,
75
76       peserta: peserta
77         .split(/\n/)
78         // clean up
79         .map(e => e.replace(/\s+/, ""))
80         .filter(e => e.length > 0)
81     };
82
83     onRequestNext(examDetails);
84   }
85
86   function handlePesertaFileUpload(e) {
87     let file = (e.target.files || [])[0];
88     if (file) {
89       let reader = new FileReader();
90       reader.onload = (event) => {
91         setPeserta(event.target.result);
92       }
93       reader.readAsText(file);
94     }
95   }
96
97   return (
98     <div className="py-5">
99       <Form className="w-100" onSubmit={handleSubmission}>
100         <Row>
101           <Col md={2}>
102             <FormGroup>
103               <Label>Tipe Ujian</Label>
104               <Input type="select" name="tipe_ujian" value={tipeUjian} onChange={e => setTipeUjian(
105                 e.target.value)}>
106                 <option value={1}>UTS</option>
107                 <option value={0}>UAS</option>
108               </Input>
109             </FormGroup>
110           <Col>
111             <FormGroup>
112               <Label>Mata Kuliah</Label>
113               <Input type="select" name="mata_kuliah" disabled={!lectures} value={mataKuliah} onChange={e =>
114                 setMataKuliah(e.target.value)}>
115                 <option value={-1} disabled>-- {lectures.length > 0 ? "Pilih Mata Kuliah" : "Memuat..."} --</
116                 option>
117                 {lectures.map(data => <option value={data._id} key={data._id}>{data.lecture_code} - {
118                   data.name}</option>)}
119               </Input>
120             </FormGroup>
121           </Col>
122         </Row>
123         <Row>
124           <Col>
125             <FormGroup>
126               <Label>Shift Ujian</Label>
127               <Input type="select" name="shift" value={shift} onChange={e => setShift(e.target.value)}>
128                 <option value={0}>Tidak ada shift</option>
129                 <option value={1}>Shift 1</option>
130                 <option value={2}>Shift 2</option>
131               </Input>
132             </FormGroup>
133           <Col>
134             <FormGroup>
135               <Label>Mulai pada</Label>
136               <DatePicker time={timeStart} onChange={setTimeStart} />

```

```

135         </FormGroup>
136     </Col>
137     <Col>
138         <FormGroup>
139             <Label>Selama</Label>
140             <InputGroup>
141                 <Input type="number" name="duration" value={duration / 60} onChange={e => setDuration(
142                     e.target.value * 60)} />
143                     <InputGroupAddon addonType="append">Mnt</InputGroupAddon>
144             </InputGroup>
145             <p className="m-0">Quick Pick:</p>
146             <ButtonGroup>
147                 {quickPick.map((el, i) => <Button onClick={e => setDuration(el * 60)} key={i}>{el} Mnt</
148                     Button>)}
149             </ButtonGroup>
150         </Col>
151     </Row>
152     <Row>
153         <Col md={8}>
154             <FormGroup>
155                 <Label>Peserta</Label>
156                 <Input type="textArea" rows={15} value={peserta} onChange={(e) => setPeserta(e.target.value)} />
157             </FormGroup>
158             <FormGroup>
159                 <Label>atau, Unggah Berkas Daftar Peserta?</Label>
160                 <Input type="file" accept="text/plain" onChange={handlePesertaFileUpload} />
161             </FormGroup>
162         </Col>
163         <Col>
164             <Alert color="info">
165                 <h3>Peserta</h3>
166                 <p>
167                     NPM standar 1955 (20xx7 yyyy) dan NPM standar 2018 (6 xx01yyy ) dapat digunakan. Aplikasi
168                     akan otomatis melakukan abstraksi terhadap absensi tersebut dan
169                     melakukan transformasi ke username standar lab.
170                 </p>
171                 <p>
172                     Informasi lebih lanjut, Anda dapat merujuk pada berkas <a href="https://gitlab.com/ftis-admin
173                         /oxam/blob/master/backend%2Fapp%2Fcontroller%2Fapi%2Fmanage%2Fexam.php" target="_blank"
174                         rel="noopener noreferrer">controller/api/manage/exam.php</a>.
175                 </p>
176             </Alert>
177             <p>
178                 Total Peserta: {(peserta.split(/\n/) || []).filter(e => e.length > 0).length} jiwa.
179             </p>
180         </Col>
181     </Row>
182     <Row>
183         <Col>
184             <div className="text-right mt-5">
185                 <Button color="primary" size="lg" type="submit">Seat Plotting <FontAwesomeIcon icon={
186                     faChevronRight} /></Button>
187             </div>
188         </Col>
189     </Row>
190   </Form>
191 }
192
193 export default observer(Step1);

```

Listing B.109 step-2.js

```

1 import React, { useState, useEffect } from 'react'
2 import ComputersContainer from './components/computers/computer-container'
3 import { axios } from '/apicall'
4 import { Nav, NavItem, NavLink, TabContent, TabPane, Row, Col, Badge, Button } from 'reactstrap'
5 import classNames from 'classnames';
6 import { observer } from 'mobx-react';
7 import LocationComputerStore from './LocationComputerStore';
8 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
9 import { faChevronRight } from '@fortawesome/free-solid-svg-icons';
10 let locationComputer = new LocationComputerStore();

12 function Step2({ examDetails, onRequestNext }) {
13     const [selectedLocations, setSelectedLocations] = useState(0)

15     useEffect(() => {
16         axios.get("manage/location").then(data => {
17             locationComputer.locationComputer = (data.data.data);
18         })
19         return () => {};
20     }, []);

22     useEffect(() => {
23         locationComputer.locationComputer.map(loc =>
24             loc.computers.map(comp => {
25                 if ((examDetails.computers || []).includes(comp._id)) {
26                     comp.selected = true;
27                 }
28                 return null;
29             })
30         )
31     })
32     return () => {};

```

```

32     }, [examDetails])
33
34     function handleRequestNext() {
35         if (selectedComputerCounts < (examDetails.peserta || []).length) {
36             return;
37         }
38
39         // get all those selected computers across the rooms.
40         let selectedComputers = [];
41         locationComputer.locationComputer.map((loc) => loc.computers.map(comp => {
42             if (comp.selected) {
43                 selectedComputers.push(comp._id);
44             }
45         })
46     );
47
48     onRequestNext(selectedComputers);
49 }
50
51     let selectedComputerCounts = 0;
52     locationComputer.locationComputer.map((loc) => loc.computers.map(comp => {
53         if (comp.selected) {
54             selectedComputerCounts++;
55         }
56     })
57 );
58
59     return (<>
60         <div className="my-5">
61             <Nav tabs>
62                 {locationComputer.locationComputer.map((loc, i) => <NavItem key={loc._id}>
63                     <NavLink
64                         className={classNames({ active: selectedLocations === i })}
65                         onClick={() => { setSelectedLocations(i); }}
66                         {loc.room_name} <Badge color={(loc.computers || []).filter(e => !e.selected).length > 0 ? "primary"
67                             : "secondary"}>{loc.name_alias}</Badge>
68                     </NavLink>
69                 </NavItem>
70             </Nav>
71             <TabContent activeTab={selectedLocations}>
72                 {locationComputer.locationComputer.map((loc, i) => <TabPane tabId={i} key={i}>
73                     <div className="p-4">
74                         <ComputersContainer computers={(loc.computers) || []} onComputerClick={(com) => (com.selected = !(com.selected || false))} />
75                     </div>
76                 </TabPane>)
77             </TabContent>
78             <div className="my-3">
79                 <p>
80                     Terdapat {(examDetails.peserta || []).length} peserta dan terpilih <b>{selectedComputerCounts}</b> komputer</p>
81                     b> ({Math.round(selectedComputerCounts / (examDetails.peserta || []).length * 100)}%).
82                 </div>
83             <div>
84                 <Row>
85                     <Col>
86                         <div className="text-right mb-5">
87                             <Button color="primary"
88                                 size="lg"
89                                 type="submit"
90                                 onClick={handleRequestNext}
91                                 disabled={selectedComputerCounts < (examDetails.peserta || []).length}>
92                                 Next <FontAwesomeIcon icon={faChevronRight} />
93                             </Button>
94                         </div>
95                     </Col>
96                 </Row>
97             </div>
98         )
99     }
100 }
101
102 export default observer(Step2);

```

Listing B.110 step-3.js

```

1 import React, { useState } from 'react'
2 import { Row, Col, Button, Badge, Alert } from 'reactstrap'
3 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
4 import { faWalking } from '@fortawesome/free-solid-svg-icons'

6 import moment from "moment";
7 import "moment/locale/id";
8 import { axios } from '~/apicall';
9 import { If, Then, Else, When } from 'react-if';

11 moment.locale('id');

13 function Step3({ examDetails, onRequestNext }) {
14
15     const [isLoading, setIsLoading] = useState(false);
16     const [status, setStatus] = useState("");
17     const [errorMsg, setErrorMsg] = useState(null);

19     function handleCreation() {
20         setIsLoading(true)

```

```

21     setStatus("Spinning up new exam...");
22     axios.post("manage/exam", {
23       lecture: examDetails.lecture,
24       uts: examDetails.uts,
25       time_duration: examDetails.duration,
26       shift: examDetails.shift,
27       time_start: examDetails.timeStart
28     }).then((response) => {
29       setStatus("Populating area...");
30       return axios.post("/manage/exam/" + response.data.data._id + "/populate", {
31         computers: examDetails.computers,
32         participants: examDetails.peserta,
33       }).then(promi => {
34         return Promise.resolve({ exam: response, populate: promi });
35       }).catch((resp) => {
36         setStatus("Error happened, reverting things...");
37         return axios.delete("manage/exam/" + response.data.data._id).then(() => {
38           Promise.reject(resp);
39         })
40       });
41     }).then((response) => {
42       onRequestNext(response.exam.data.data);
43     }).catch((resp) => {
44       setIsLoading(false);

45       if (resp.response.data && resp.response.data.error) {
46         setErrorMsg(<>
47           <h5>{resp.response.data.error.title} [{resp.response.data.error.error_code}]</h5>
48           <p>{resp.response.data.error.description}</p>
49         </>)
50       } else {
51         console.error(resp);
52         console.log(JSON.stringify(resp));
53         setErrorMsg(<>
54           <h5>Error happened</h5>
55           <p>{resp.message}</p>
56           <p><small>Because it's a browser(/network) related error, the error has been emitted to the console.</small></p>
57         </>)
58       }
59     });
60   });

61   return (
62     <div className="my-4">
63       <h3>Konfirmasi</h3>
64       <Row>
65         <Col>
66           <p className="h2">
67             <Badge color={examDetails.uts ? "info" : "success"}>{examDetails.uts ? "UTS" : "UAS"}</Badge>
68             <span className="ml-3">{(examDetails.lecture_obj || {}).name || "Nama Matakuliah di sini"}</span>
69           </p>
70           <p className="lead">{examDetails.shift ? "Shift " + examDetails.shift : "Tanpa Shift"} - {(
71             examDetails.peserta || []).length} Peserta</p>
72           <p>Dilaksanakan pada ruang (ruangan). Akan dimulai pada <b>{moment(examDetails.timeStart).format('LLLL')}</b>
73             selama <b>{examDetails.duration / 3600}</b> jam</p>
74           <When condition={!errorMsg}>
75             <Alert color="danger">{errorMsg}</Alert>
76           </When>
77         </Col>
78       </Row>
79       <Row>
80         <Col>
81           <div className="text-center my-5 pt-5">
82             <p>Untuk konfirmasi pembuatan ujian, tekan tombol di bawah ini.</p>
83             <Button color="warning" size="lg" onClick={handleCreation} disabled={isLoading}>
84               <If condition={isLoading}>
85                 <Then>{status}</Then>
86                 <Else>Create Exam <FontAwesomeIcon icon={faWalking} /></Else>
87               </If>
88             </Button>
89           </div>
90         </Col>
91       </Row>
92     </div>
93   )
94 }

95 export default Step3

```

Listing B.111 step-4.js

```

1 import React from 'react'
2 import { Button } from 'reactstrap'
3 import { Link } from 'react-router-dom'

5 function Step4({ examDetails }) {
6   return (
7     <div className="my-5">
8       <h3>Finish...</h3>
9       <p>Examination has been created and participants has been plotted to designated seat.</p>
10      <div className="mt-4">
11        <Button tag={Link} to={`/admin/exam/${examDetails.exam._id}/detail`}>See Exam</Button>
12      </div>
13    </div>
14  )
15}

```

```

16 }
18 export default Step4

```

Listing B.112 autoreport-child.js

```

1 import React from 'react'
2 import { faPaperPlane, faTrash } from '@fortawesome/free-solid-svg-icons'
3 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
4 import { Button } from 'reactstrap'

6 function AutoreportChild({ examReport = {}, onDeleteRequest = () => { }, onSendRequest = () => { } }) {
7     return (
8         <>
9             <tr>
10                 <td>{examReport.tos}</td>
11                 <td>{examReport.valid_until}</td>
12                 <td>{examReport.sent_on || "Belum"}</td>
13                 <td>
14                     <Button color="info" onClick={() => onSendRequest(examReport)}><FontAwesomeIcon icon={faPaperPlane} /></Button>
15                     <Button color="danger" onClick={() => onDeleteRequest(examReport)}><FontAwesomeIcon icon={faTrash} /></Button>
16                 </td>
17             </tr>
18         </>
19     )
20 }
21 }

23 export default AutoreportChild

```

Listing B.113 autoreport.js

```

1 import React, { useState } from 'react'
2 import { Row, Col, Button, Table, Alert, Modal, ModalHeader, ModalBody, Form, ModalFooter, FormText, FormGroup, Label, Input, Spinner } from 'reactstrap'
3 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
4 import { faPlus } from '@fortawesome/free-solid-svg-icons'
5 import AutoreportChild from './autoreport-child'
6 import { useAdminStore } from '~/components/use-store'
7 import { axios } from '~/apicall'
8 import moment from "moment"
9 import { Else, If, Then } from 'react-if'

11 function ExamDetailAutoreport({ exam }) {
12     const adminStore = useAdminStore();

14     const [showAddExamReportModal, setShowAddExamReportModal] = useState(false)
15     const [dataToDelete, setDataToDelete] = useState(undefined)
16     const [dataToSend, setDataToSend] = useState(undefined)
17     const [sending, setSending] = useState(false)

19     const [emailInput, setEmailInput] = useState("")

22     function refreshExamReport() {
23         adminStore.examFetch(exam._id)
24     }

26     function handleAutoReportCreation(e) {
27         e.preventDefault();
28         let validuntil = moment().add('7', 'days').format("YYYY-MM-DD HH:mm:ss");
29         axios.post("manage/examreport", {
30             tos: emailInput,
31             valid_until: validuntil,
32             exam: exam._id
33         }).then(response => {
34             setShowAddExamReportModal(false);
35             refreshExamReport();
36         })
37     }

39     function handleDeleteRequest() {
40         axios.delete("manage/examreport/" + dataToDelete._id).then((d) => {
41             setDataToDelete(undefined);
42             refreshExamReport();
43         })
44     }

46     function handleSendRequest() {
47         setSending(true);
48         axios.post("manage/examreport/" + dataToSend._id + "/forcesend").then((d) => {
49             setDataToSend(undefined);
50             setSending(false);
51             refreshExamReport();
52         })
53     }

56     return (
57         <div>
58             <Row>
59                 <Col>
60                     <h3>Autoreport</h3>

```

```

61         </Col>
62     <Col>
63         <div className="text-right">
64             <Button className="mr-2" color="primary" onClick={() => setShowAddExamReportModal(true)} > <
65                 FontAwesomeIcon icon={faPlus} /></Button>
66         </div>
67     </Row>
68     <Row>
69         <Col>
70             <Alert color="secondary">
71                 Autoreport akan berjalan via Cronjob sesuai dengan konfigurasi yang telah dilakukan.
72                 Sebuah email akan dikirimkan ke alamat yang telah ditentukan secara otomatis setelah ujian selesai.
73             </Alert>
74             <Table>
75                 <thead>
76                     <tr>
77                         <th>Email</th>
78                         <th>Kadaluarsa</th>
79                         <th>Terkirim</th>
80                         <th></th>
81                     </tr>
82                 </thead>
83                 <tbody>
84                     {(exam.exam_report || []).map((data, i) => <AutoreportChild key={i} examReport={data}
85                         onDeleteRequest={setDeleteRequest} onSendRequest={setDataToSend} />)}
86                 </tbody>
87             </Table>
88         </Col>
89     </Row>
90     <Modal isOpen={showAddExamReportModal} backdrop>
91         <Form onSubmit={handleAutoReportCreation}>
92             <ModalHeader>Tambah Autoreport</ModalHeader>
93             <ModalBody>
94                 <FormGroup>
95                     <Label>Email</Label>
96                     <Input type="text" value={emailInput} onChange={e => setEmailInput(e.target.value)} />
97                     <FormText>Pisahkan dengan koma untuk email lebih dari satu.</FormText>
98                 </FormGroup>
99             </ModalBody>
100            <ModalFooter>
101                <Button color="secondary" onClick={() => setShowAddExamReportModal(false)}>Batal</Button>
102                <Button color="primary">Buat</Button>
103            </ModalFooter>
104        </Form>
105    </Modal>
106
107    <Modal isOpen={!dataToDelete} backdrop>
108        <ModalHeader>Konfirmasi penghapusan</ModalHeader>
109        <ModalBody>
110            Apakah anda yakin ingin menghapus autoreport untuk email <code>{(dataToDelete || {}).tos}</code>?
111        </ModalBody>
112        <ModalFooter>
113            <Button color="primary" onClick={() => setDataToDelete(undefined)}>Batalkan</Button>
114            <Button color="secondary" onClick={handleDeleteRequest}>Hapus</Button>
115        </ModalFooter>
116    </Modal>
117
118    <Modal isOpen={!dataToSend} backdrop>
119        <ModalHeader>Konfirmasi Pengiriman</ModalHeader>
120        <ModalBody>
121            Apakah anda yakin ingin mengirim jawaban ke email <code>{(dataToSend || {}).tos}</code>?
122        </ModalBody>
123        <ModalFooter>
124            <Button color="secondary" onClick={() => setDataToSend(undefined)} disabled={sending}>Batalkan</Button>
125            <Button color="info" onClick={handleSendRequest} disabled={sending}>Kirim</Button>
126            <If condition={sending}>
127                <Then>
128                    <Spinner animation="border" size="sm" role="status" aria-hidden="true" />
129                </Then>
130                <Else>
131                    Kirimkan
132                </Else>
133            </If>
134
135        </ModalFooter>
136    </Modal>
137 </div>
138 }
139 }
140
141 export default ExamDetailAutoreport

```

Listing B.114 Detail.js

```

1 import { faBriefcase, faDesktop, faFileCode, faTable } from '@fortawesome/free-solid-svg-icons';
2 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
3 import fileDownload from 'js-file-download';
4 import { observer } from 'mobx-react';
5 import React, { useEffect, useState } from 'react';
6 import Countdown from 'react-countdown';
7 import { Else, If, Then, When } from 'react-if';
8 import { Link } from "reactstrap/lib";
9 import Badge from "reactstrap/lib/Badge";
10 import Button from "reactstrap/lib/Button";
11 import Col from "reactstrap/lib/Col";

```

```

12 import Container from "reactstrap/lib/Container";
13 import Row from "reactstrap/lib/Row";
14 import Table from "reactstrap/lib/Table";
15 import { axios } from '~/apicall';
16 import { useAdminStore } from '~/components/use-store';
17 import ExamDetailAutoreport from './autoreport/autoreport';
18 import AdminExamMigrator from './migrator/migrator';
19 import ExamNotificationLister from './notifications/lister';
20 import SlotJawaban from './slot-jawaban';

23 function Detail({ match }) {
24   const adminStore = useAdminStore();
25   const { exam } = adminStore;
26   // console.log("exams", exam);

28   const [showOpenMigratorTool, setShowOpenMigratorTool] = useState(false)

30   useEffect(() => {
31     adminStore.selectedExam = match.params.id;
32     adminStore.examFetch(match.params.id);
33     return () => {
34   };
35 }, [adminStore, match])

37   function handleScriptDownload() {
38     axios.get(`manage/exam/${match.params.id}/script`, {
39       responseType: 'blob'
40     }).then((data) => {
41       fileDownload(data.data, data.headers['x-filename']);
42     });
43 }

45   return (<>
46     <div className="bg-success text-white py-5 mb-3">
47       <If condition={!exam._id && !adminStore.isLoading}>
48         <Then>
49           <Container>
50             <Row className="my-3 align-items-center">
51               <Col>
52                 <h5>{exam.lecture.lecture_code} <When condition={exam.shift !== 0}><Badge color="light">SHIFT-{exam.shift}</Badge></When></h5>
53                 <h2 className="font-weight-bold">{exam.lecture.name}</h2>
54               </Col>
55               <Col>
56                 <div className="text-right">
57                   <p className="h3">
58                     <If condition={exam?.time_opened && exam?.time_left > 0}>
59                       <Then>
60                         <b><Countdown date={Date.now() + (exam.time_left * 1000)} className="d-flex mx-3" autoStart={!exam.time_opened}>/><br />
61                         <Badge color="danger">ON-GOING</Badge>
62                       </Then>
63                     <Else>
64                       <If condition={exam?.time_left === 0}>
65                         <Then>
66                           <Badge color="light">FINISHED</Badge>
67                         </Then>
68                       <Else>
69                         <Badge color="warning">STANDBY</Badge>
70                       </Else>
71                     </If>
72                   </p>
73                 </div>
74               </Col>
75             </Row>
76           </Container>
77         </Then>
78       <Else>
79         <If condition={!adminStore.isLoading}>
80           <Then>
81             <Container>
82               <h3>Not found</h3>
83             </Container>
84           </Then>
85         <Else>
86           <Container>
87             <h3>Loading...</h3>
88           </Container>
89         </Else>
90       </If>
91     </Else>
92   </If>
93 </div>
94 <Container>
95   <Row>
96     <Col>
97       <h6>Tools</h6>
98       <Button className="mr-3" tag={Link} to={`/admin/exam/${match.params.id}/participant/signature`}><FontAwesomeIcon icon={faTable}> Daftar Hadir (Tanda Tangan)</Button>
99       <Button className="mr-3" tag={Link} to={`/admin/exam/${match.params.id}/participant/door`}><FontAwesomeIcon icon={faTable}> Daftar Hadir (Pintu)</Button>
100      <Button className="mr-3" tag={Link} to={`/admin/exam/${match.params.id}/screen`}><FontAwesomeIcon icon={faDesktop}> Layar Projektor</Button>
101      <Button className="mr-3" color="info" onClick={handleScriptDownload}><FontAwesomeIcon icon={faFileCode}> Download Script</Button>
102      <Button className="mr-3" color="warning" onClick={() => setShowOpenMigratorTool(true)}><FontAwesomeIcon icon={faFileCode}> Download Script</Button>
103    </Col>
104  </Row>

```

```

        faBriefcase} /> Pindah Peserta</Button>
106    /* Migrator Tool */
107    <AdminExamMigrator isOpen={showOpenMigratorTool} participants={exam.participants || []} onUpdated={() =>
108      adminStore.examFetch(match.params.id) onCloseRequested={() => setShowOpenMigratorTool(false)} />
109    </Col>
110  </Row>
111  <Row>
112    <Col>
113      <div className="my-4">
114        <h3>Informasi Ujian <Button color="warning" tag={Link} to={"/admin/manage/exams/" + match.params.id + "?referral=" +
115          " + window.location.pathname}>Ubah</Button></h3>
116        <Row>
117          <Col>
118            <Table>
119              <tbody>
120                <tr>
121                  <th>Mata Kuliah</th>
122                  <td>{exam.lecture.name} ({exam.lecture.lecture_code})</td>
123                </tr>
124                <tr>
125                  <th>Tahun Ajaran</th>
126                  <td>{exam.lecture_period.period_code}</td>
127                </tr>
128                <tr>
129                  <th>Mulai - Berakhir</th>
130                  <td>{exam.time_opened} - {exam.time_ended}</td>
131                </tr>
132                <tr>
133                  <th>Waktu Mulai Pengumpulan</th>
134                  <td>{exam.time_start}</td>
135                </tr>
136                <tr>
137                  <th>Durasi</th>
138                  <td>{exam.time_duration / 60} Menit</td>
139                </tr>
140                <tr>
141                  <th>Uts/Uas</th>
142                  <td>{exam.uts ? "UTS" : "UAS"}</td>
143                </tr>
144            </tbody>
145          </Table>
146        <Col>
147          <Table>
148            <tbody>
149              <tr>
150                <th>Jumlah Peserta</th>
151                <td>{(exam.participants || []).length} Orang</td>
152              </tr>
153              <tr>
154                <th>Dibuat pada</th>
155                <td>{exam.created_on}</td>
156              </tr>
157              <tr>
158                <th>Terakhir diperbaharui</th>
159                <td>{exam.updated_on}</td>
160              </tr>
161            </tbody>
162          </Table>
163        </Col>
164      </div>
165    </Col>
166  </Row>
167  <Row>
168    <Col xs={6}>
169      <SlotJawaban exam={exam} />
170    </Col>
171    <Col xs={6}>
172      <ExamNotificationLister exam={exam} />
173    </Col>
174    <Col xs={6}>
175      <ExamDetailAutoreport exam={exam} />
176    </Col>
177  </Row>
178 </Container>
179 </>
180 )
181 }

183 export default observer(Detail);

```

Listing B.115 LocationComputerStore.js

```

1 import { decorate, observable } from "mobx";
3 class ComputerLocationStore {
4   locationComputer = [];
5 }
7 decorate(ComputerLocationStore, {
8   locationComputer: observable
9 })
11 export default ComputerLocationStore;

```

Listing B.116 migrator.js

```

1 import React, { useState, useEffect } from 'react'
2 import { Button, Modal, ModalHeader, ModalBody, ModalFooter, FormGroup, Label, Input, Table, Nav, NavItem, TabContent,
3         TabPane, Badge, Col, Row, NavLink } from 'reactstrap'
4 import { If, Then, Else } from 'react-if'
5 import LocationComputerStore from "./LocationComputerStore";
6 import ComputersContainer from '~/components/computers/computer-container'
7 import { axios } from '~/apicall';
8 import { observer } from 'mobx-react';
9 import classNames from 'classnames';
10 import { faPlus } from '@fortawesome/free-solid-svg-icons';
11 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
12 import fileDownload from 'js-file-download';
13 import { useAdminStore } from '~/components/use-store';

14 let locationComputer = new LocationComputerStore();

16 function AdminExamMigrator({ isOpen = false, onUpdated = () => {} , participants = [], onCloseRequested = () => {} }) {
17     const [showAddDialog, setShowAddDialog] = useState(false)
18     const [selectedParticipant, setSelectedParticipant] = useState({})
19     const [participantFinder, setParticipantFinder] = useState("")
20     const [finalMigrtationScript, setFinalMigrtationScript] = useState(undefined);
21     const [scriptDownloaded, setScriptDownloaded] = useState(false)

24     const [migrationCanidates, setMigrationCanidates] = useState([]);
26     const [computerList, setComputerList] = useState({})

28     const adminStore = useAdminStore();

30     let participantFiltered = participants.filter((par) =>
31         participantFinder.length === 0 ||
32         (par.username || "").includes(participantFinder) ||
33         (par.name || "").includes(participantFinder) ||
34         (par.npm || "").includes(participantFinder) ||
35         ((par.computer || {}).name || "").includes(participantFinder)
36     );
38     const [selectedLocations, setSelectedLocations] = useState(0)

40     useEffect(() => {
41         axios.get("manage/location").then(data => {
42             locationComputer.locationComputer = (data.data.data);

44             let komp = {};
45             data.data.data.forEach(element => {
46                 element.computers.forEach(comp => {
47                     komp[comp._id] = comp;
48                 })
49             });
50             setComputerList(komp);
51         })
52         return () => {};
53     }, []);

55     function handleAddMigrator(comp) {
56         setMigrationCanidates([...migrationCanidates, {
57             participant: selectedParticipant,
58             to: comp._id
59         }]);
61         setSelectedParticipant({})
62         setParticipantFinder("")
63         setShowAddDialog(false)
64     }

66     function handleScriptGenerationRequest() {
67         axios.post("manage/exam/" + adminStore.exam._id + "/move", {
68             lists: migrationCanidates.map((data) => ({ participant: data.participant._id, to: data.to }))
69         }, {
70             responseType: 'blob'
71         }).then((response) => {
72             setShowAddDialog(false)
73             setFinalMigrtationScript({
74                 data: response.data,
75                 filename: response.headers['x-filename']
76             })
77             setMigrationCanidates([])
78         })
79     }

81     function handleScriptDownloadSimulate() {
82         setScriptDownloaded(true);
83         fileDownload(finalMigrtationScript.data, finalMigrtationScript.filename);
84     }
85     return (
86         <>
87             <Modal isOpen={isOpen} size="lg" backdrop>
88                 <ModalHeader>Migrator</ModalHeader>
89                 <ModalBody>
90                     <Row>
91                         <Col>
92                             <Button onClick={() => setShowAddDialog(true)}><FontAwesomeIcon icon={faPlus} /> Tambah</Button>
93                         </Col>
94                     </Row>
95                     <Table bordered striped>
96                         <thead>
```

```

97          <tr>
98              <th>Peserta</th>
99              <th>Komputer Asal</th>
100             <th>Komputer Tujuan</th>
101         </tr>
102     </thead>
103     <tbody>
104         {migrationCanidates.map((data, i) => <tr key={"migration-" + i}>
105             <td>{data.participant.name} - {data.participant.npm}</td>
106             <td>{(computerList[data.participant.computer] || {}).name}</td>
107             <td>{(computerList[data.to] || {}).name}</td>
108         </tr>)
109     </tbody>
110 </Table>
111 </ModalBody>
112 <ModalFooter>
113     <Button color="secondary" onClick={() => {
114         setMigrationCanidates([]);
115         onCloseRequested()
116     }}>Batal</Button>
117     <Button color="primary" onClick={handleScriptGenerationRequest}>Pindah dan Buat Script</Button>
118 </ModalFooter>
119 </Modal>
120 <Modal isOpen={showAddDialog} size="xl">
121     <ModalHeader>Tambahkan Peserta</ModalHeader>
122     <ModalBody>
123         <If condition={!selectedParticipant || !selectedParticipant._id}>
124             <Then>
125                 <FormGroup>
126                     <Label>Cari Peserta</Label>
127                     <Input type="text" value={participantFinder} onChange={e => setParticipantFinder(e.target.value)} />
128                 </FormGroup>
129                 <Table bordered striped>
130                     <thead>
131                         <tr>
132                             <th>NPM</th>
133                             <th>Nama</th>
134                             <th>Komputer</th>
135                             <th></th>
136                         </tr>
137                     </thead>
138                     <tbody>
139                         {participantFiltered.map((par) => <tr key={"participant-" + par._id}>
140                             <td>{par.npm}</td>
141                             <td>{par.name}</td>
142                             <td>{(computerList[par.computer] || {}).name}</td>
143                             <td><Button color="info" onClick={() => setSelectedParticipant(par)}>Pilih</Button></td>
144                         </tr>)}
145                     </tbody>
146                 </Table>
147             </Then>
148         <Else>
149             <p>{selectedParticipant.name} ({selectedParticipant.npm}) dari Komputer {{ selectedParticipant.computer || {}}.name} akan dipindah ke komputer: </p>
150             <Nav tabs>
151                 {locationComputer.locationComputer.map((loc, i) => <NavItem key={loc._id}>
152                     <NavLink
153                         className={classNames({ active: selectedLocations === i })}
154                         onClick={() => { setSelectedLocations(i); }}
155                     >
156                         {loc.room_name} <Badge color={({loc.computers || []}.filter(e => !e.selected).length > 0 ? "primary" : "secondary"}>{loc.name_alias}</Badge>
157                     </NavLink>
158                 </NavItem>)
159             </Nav>
160             <TabContent activeTab={selectedLocations}>
161                 {locationComputer.locationComputer.map((loc, i) => <TabPane tabId={i} key={i}>
162                     <div className="p-4">
163                         <ComputersContainer computers={({loc.computers || []})} onComputerClick={(com) => handleAddMigrator(com)} />
164                     </div>
165                 </TabPane>)
166             </TabContent>
167         </Else>
168     </If>
169 </ModalBody>
170 <ModalFooter>
171     <Button color="secondary" onClick={() => {
172         setShowAddDialog(false)
173         setSelectedParticipant({})
174         setParticipantFinder("")
175     }}>Batalkan</Button>
176 </ModalFooter>
177 </Modal>
178 <Modal isOpen={!finalMigrtationScript}>
179     <ModalHeader>Unduh Script</ModalHeader>
180     <ModalBody>
181         <p>Mohon pastikan anda mengunduh script ini karna perpindahan telah dilakukan pada database dan perubahan dibuat permanen.</p>
182         <div className="text-center"><Button color="info" onClick={handleScriptDownloadSimulate}>Unduh Script</Button></div>
183     </ModalBody>
184     <ModalFooter>
185         <Button color="warning" disabled={!scriptDownloaded} onClick={() => {
186             setScriptDownloaded(false)
187             setFinalMigrtationScript(undefined)
188             onCloseRequested()
189         }}>Unduh</Button>
190     </ModalFooter>
191 </Modal>

```

```

189             })>Tutup</Button>
190         </ModalFooter>
191     </Modal>
192   </>
193 }
194 }

195 export default observer(AdminExamMigrator)

```

Listing B.117 minipanel.js

```

1 import { faHistory, faStop, faStopwatch } from '@fortawesome/free-solid-svg-icons';
2 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
3 import { toJS } from 'mobx';
4 import { observer } from 'mobx-react';
5 import React, { useEffect } from 'react';
6 import { Else, If, Then, When } from 'react-if';
7 import { Badge, Button, ButtonGroup, Col, Container, Row, Table } from 'reactstrap';
8 import { axios } from '~/apicall';
9 import CountDown from '~/components/countdown/countdown';
10 import { useAdminStore } from '~/components/use-store';

11 function MinipanelControlButton({ handleStart, handleStop, handleReset, exam }) {
12   return (<ButtonGroup className="btn-block">
13     <If condition={exam.time_left === null}>
14       <Then>
15         <Button className="mr-2" color="primary" onClick={handleStart}><FontAwesomeIcon icon={faStopwatch} /> Start</
16           Button>
17         </Then>
18       <Else>
19         <When condition={exam.time_left !== 0}>
20           <Button className="mr-2" color="warning" onClick={handleStop}><FontAwesomeIcon icon={faStop} /> Stop</
21             Button>
22           </When>
23           <Button className="mr-2" color="danger" onClick={handleReset}><FontAwesomeIcon icon={faHistory} /> Reset</
24             Button>
25         </Else>
26       </If>
27     </ButtonGroup>;
28 }

29 function LoaderComponent() {
30   return <Container className="text-center">
31     <h3>Loading...</h3>
32   </Container>
33 }

34 function ExamMinipanel({ match }) {
35   const adminStore = useAdminStore();
36   const { exam } = adminStore;

37   useEffect(() => {
38     adminStore.selectedExam = match.params.id;
39     adminStore.examFetch(match.params.id);
40     return () => {
41     };
42   }, [adminStore, match]);

43   function handleStart() {
44     axios.post("manage/exam/" + match.params.id + "/start").then(() => {
45       adminStore.examFetch(match.params.id);
46     })
47   }

48   function handleStop() {
49     axios.post("manage/exam/" + match.params.id + "/close").then(() => {
50       adminStore.examFetch(match.params.id);
51     })
52   }

53   function handleReset() {
54     axios.delete("manage/exam/" + match.params.id + "/start").then(() => {
55       adminStore.examFetch(match.params.id);
56     })
57   }

58   if (adminStore.isLoading) {
59     return <LoaderComponent />
60   }

61   return (
62     <Container>
63       <Row className="mt-4 justify-content-center">
64         <Col xs={12} md={5}>
65           <h4 className="text-center">Minipanel - {exam.lecture.lecture_code} <When condition={exam.shift !== 0}>*<
66             br /><Badge color="dark">SHIFT-{exam.shift}</Badge></When></h4>
67           <section className="my-5">
68             <div className="h2 text-center">
69               <CountDown secondsLeft={toJS(exam.time_left)} />
70             </div>
71             <MinipanelControlButton handleReset={handleReset} handleStart={handleStart} handleStop={handleStop} exam={exam} />
72           </section>
73         </Col>
74       </Row>
75     </Container>
76   )
77 }

78 export default ExamMinipanel;

```

```

83         <Table>
84             <tbody>
85                 <tr>
86                     <th>Mata Kuliah</th>
87                     <td>{exam.lecture.name} ({exam.lecture.lecture_code})</td>
88                 </tr>
89                 <tr>
90                     <th>Tahun Ajaran</th>
91                     <td>{exam.lecture_period.period_code}</td>
92                 </tr>
93                 <tr>
94                     <th>Mulai - Berakhir</th>
95                     <td>{exam.time_opened} - {exam.time_ended}</td>
96                 </tr>
97                 <tr>
98                     <th>Waktu Mulai Pengumpulan</th>
99                     <td>{exam.time_start || "Belum di buka"}</td>
100                </tr>
101                <tr>
102                    <th>Durasi</th>
103                    <td>{exam.time_duration / 60} Menit</td>
104                </tr>
105                <tr>
106                    <th>Uts/Uas</th>
107                    <td>{exam.uts ? "UTS" : "UAS"}</td>
108                </tr>
109                <tr>
110                    <th>Jumlah Peserta</th>
111                    <td>{(exam.participants || []).length} Orang</td>
112                </tr>
113                <tr>
114                    <th>Dibuat pada</th>
115                    <td>{exam.created_on}</td>
116                </tr>
117                <tr>
118                    <th>Terakhir diperbaharui</th>
119                    <td>{exam.updated_on}</td>
120                </tr>
121            </tbody>
122        </Table>
123
124        <section className="my-4">
125            <MinipanelControlButton handleReset={handleReset} handleStart={handleStart} handleStop={handleStop}
126            exam={exam} />
127        </section>
128    </Col>
129  </Row>
130</Container>
131}
132
133 export default observer(ExamMinipanel)

```

Listing B.118 audience-picker.js

```

1 import React, { useState, useEffect } from 'react'
2 import { Modal, ModalHeader, ModalBody, Table, Input, Button, ModalFooter } from 'reactstrap'
4 function NotificationAudiencePicker({ participants = [], onSelected = () => {} , selectedAudiences = [] , onCanceled = () => {} }) {
6     const [_selectedAudiences, set_SelectedAudiences] = useState(selectedAudiences)
8     useEffect(() => {
9         set_SelectedAudiences(selectedAudiences)
10        return () => {}
11    }, [selectedAudiences])
13    function handleSelectAll() {
14        set_SelectedAudiences(participants.map((data) => data._id));
15    }
17    function handleCheckChange(e, participant) {
18        if (e.target.checked) {
19            set_SelectedAudiences([..._selectedAudiences, participant._id])
20        } else {
21            set_SelectedAudiences([..._selectedAudiences.filter((data) => data !== participant._id)]);
22        }
23    }
25    return (
26        <>
27            <Modal isOpen backdrop="static">
28                <ModalHeader>Pilih Penemima Notifikasi</ModalHeader>
29                <ModalBody>
30                    <div>
31                        <Button onClick={handleSelectAll}>Pilih semua</Button>
32                    </div>
33                    <Table>
34                        <thead>
35                            <tr>
36                                <th></th>
37                                <th>Nama & Username</th>
38                                <th>NPM</th>
39                            </tr>
40                        </thead>

```

```

41             <tbody>
42                 {participants.map((participant) => <tr key={"participant-" + participant._id}>
43                     <td><input type="checkbox" checked={_selectedAudiences.includes(participant._id)} onChange={()=> handleCheckChange(e, participant)} /></td>
44                     <td>{participant.name} - <code>{participant.username}</code></td>
45                     <td>{participant.npm}</td>
46                 </tr>)}
47             </tbody>
48         </Table>
49     </ModalBody>
50     <ModalFooter>
51         <Button onClick={onCanceled}>Batal</Button>
52         <Button onClick={() => onSelected(_selectedAudiences)}>Selesai</Button>
53     </ModalFooter>
54 </Modal>
55 )
56 )
57 }
58
59 export default NotificationAudiencePicker

```

Listing B.119 lister-child.js

```

1 import React, { useState } from 'react'
2 import { Modal, ModalHeader, ModalBody, ModalFooter, Button, Form, FormGroup, Label, Input } from 'reactstrap'
3 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
4 import { faPen, faTrash } from '@fortawesome/free-solid-svg-icons'
5 import { axios } from '~/apicall'
6 import 'suneditor/dist/css/suneditor.min.css';
7 import SunEditor, { buttonList } from 'suneditor-react';
8 import NotificationAudiencePicker from './audience-picker'
9 import { useAdminStore } from '~/components/use-store'
10 import { When } from 'react-if'
11 import { observer } from 'mobx-react'

14 function NotificationListerChild({ notif = {}, onDelete = () => { }, onEdited = () => { } }) {
15     const [deleteRequested, setDeleteRequested] = useState(false)

17     const [notification, setNotification] = useState(notif); // used for editors
18     const [editOpen, setEditOpen] = useState(false);
19     const [audiencePickerShow, setAudiencePickerShow] = useState(false)

21     const adminStore = useAdminStore();
22     const exam = adminStore.exam;

24     function handleDeleteRequest() {
25         axios.put("manage/notification/" + notif._id, { description: notif.description, title: notif.title, participants: [] })
26             .then(() => {
27                 return axios.delete("manage/notification/" + notif._id + "")
28             }).then((resp) => {
29                 setDeleteRequested(false);
30                 onDelete(notif);
31             })
32     }

33     function handleEditRequest() {
34         let notifFiltered = { ...notification };
35         delete notifFiltered['_id'];
36         delete notifFiltered['created_on'];
37         delete notifFiltered['updated_on'];
38         delete notifFiltered['deleted_on'];

40         axios.put("manage/notification/" + notif._id, notifFiltered).then((resp) => {
41             onEdited(resp.data.data);
42             setEditOpen(false);
43         })
44     }

46     return (
47         <>
48             <tr>
49                 <td>
50                     {notif.title}
51                 </td>
52                 <td style={{ textAlign: "right" }}>
53                     <Button className="ml-2" onClick={() => setEditOpen(true)}><FontAwesomeIcon icon={faPen} /></Button>
54                     <Button className="ml-2" color="danger" onClick={() => setDeleteRequested(true)}><FontAwesomeIcon icon={faTrash} /></Button>
55                 </td>
56             </tr>
57         <Modal isOpen={deleteRequested} backdrop="static">
58             <ModalHeader>Konfirmasi</ModalHeader>
59             <ModalBody>
60                 <p>Apakah anda yakin untuk menghapus notifikasi ini?</p>
61                 <div className="p-4 bg-dark text-light">
62                     <div dangerouslySetInnerHTML={{ __html: notif.description }} />
63                 </div>
64             </ModalBody>
65             <ModalFooter>
66                 <Button color="secondary" onClick={handleDeleteRequest}>Hapus</Button>
67                 <Button color="primary" onClick={() => setDeleteRequested(false)}>Batal</Button>
68             </ModalFooter>
69         </Modal>
70
71         <Modal isOpen={editOpen} size="lg" backdrop="static">
72             <ModalHeader>Ubah Notifikasi</ModalHeader>

```

```

73         <ModalBody>
74             <Form>
75                 <FormGroup>
76                     <Label>Judul</Label>
77                     <Input type="text" defaultValue={notification.title} onChange={(e) => setNotification({ ...notification, title: e.target.value })} />
78                 </FormGroup>
79                 <FormGroup>
80                     <Label>Penerima Notifikasi</Label>
81                     <Input plaintext value={notification.participants.length + " orang, (Klik untuk mengubah)"}
82                         onClick={() => setAudiencePickerShow(true)} onChange={() => { }} />
83                 </FormGroup>
84                 <FormGroup>
85                     <Label>
86                         Deskripsi
87                     </Label>
88                     <SunEditor setOptions={{ buttonList: buttonList.formatting, height: 200 }} setContents={ notification.description } onChange={(val) => setNotification({ ...notification, description: val })} />
89                 </FormGroup>
90             </ModalBody>
91             <ModalFooter>
92                 <Button color="secondary" onClick={() => {
93                     setNotification(notif); // reset
94                     setEditOpen(false);
95                 }}>Batal</Button>
96                 <Button color="primary" onClick={handleEditRequest}>Simpan</Button>
97             </ModalFooter>
98         </Modal>
99
100        /* Modal pendukung */
101        <When condition={audiencePickerShow}>
102            <NotificationAudiencePicker participants={exam.participants} selectedAudiences={notification.participants}
103                onSelected={(val) => {
104                    setNotification({ ...notification, participants: val })
105                    setAudiencePickerShow(false)
106                }} onCanceled={() => setAudiencePickerShow(false)} />
107        </When>
108    </>
109 }
110
111 export default observer(NotificationListerChild)

```

Listing B.120 lister.js

```

1 import React, { useState, useEffect, useCallback } from 'react'
2 import { Row, Col, Button, Table, Modal, ModalBody, ModalFooter, ModalHeader, FormGroup, Label, Input, FormText, Form } from
3     'reactstrap'
4 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
5 import { faPlus } from '@fortawesome/free-solid-svg-icons'
6 import { axios } from './apicall';
7 import NotificationListerChild from './lister-child';
8 import 'suneditor/dist/css/suneditor.min.css';
9 import SunEditor, { buttonList } from 'suneditor-react';
10 import NotificationAudiencePicker from './audience-picker'
11 import { When } from 'react-if';

12 function ExamNotificationLister({ exam }) {
13
14     const [notifications, setNotifications] = useState([]);
15     const [showNewNotificationsModal, setShowNewNotificationsModal] = useState(false)
16     const [showPasswordNotifModal, setShowPasswordNotifModal] = useState(false)
17     const [showCustomNotifModal, setShowCustomNotifModal] = useState(false)
18
19     const [nniServiceList, setNniServiceList] = useState("");
20     const [nniPasswordList, setNniPasswordList] = useState("");
21
22     const [customNotificationParticipants, setCustomNotificationParticipants] = useState([])
23     const [customNotificationTitle, setCustomNotificationTitle] = useState(undefined)
24     const [customNotificationDescription, setCustomNotificationDescription] = useState(undefined)
25
26     const [audiencePickerShow, setAudiencePickerShow] = useState(false)
27
28     const [participantLister, setParticipantLister] = useState({});
29     useEffect(() => {
30         let parti = {};
31         (exam.participants || []).forEach((par) => {
32             parti[par.npm] = par;
33         });
34
35         setParticipantLister(partici);
36         return () => {}
37     }, [exam])
38
39     let refreshNotifications = useCallback(
40         (exam) => {
41             axios.get("manage/exam/" + exam._id + "/notifications").then((response) => {
42                 setNotifications(response.data.data.map(data => {
43                     data.participants = data.participants.map((e) => e._id) // just need to list their id.
44                     return data;
45                 }));
46             })
47         },
48     [], [])
49

```

```

51     useEffect(() => {
52       refreshNotifications(exam);
53     }
54   ), [exam, refreshNotifications])
55
56
57   // transform passwordlist
58   let transformedPasslist = nniPasswordList.split("\n").map((line) => line.split("|"));
59
60   function handleGenMass(e) {
61     e.preventDefault();
62     // handle shit
63     axios.post("manage/notification/mass_gen", {
64       lists: transformedPasslist.map((line) => {
65         let [parti, uname = "", pass = ""] = line;
66
67         if (!participantLister.hasOwnProperty(parti)) {
68           throw new Error("NPM tidak valid " + parti)
69         }
70
71         return {
72           participant: participantLister[parti]...id,
73           username: uname,
74           password: pass
75         }
76       }),
77       url: nniServiceList
78     }).then((response) => {
79       //done.
80       setShowPasswordNotifModal(false);
81       refreshNotifications(exam);
82     }).catch(alert);
83   }
84
85   function handleCustomNotifCreateRequest(e) {
86     e.preventDefault();
87     if (customNotificationParticipants.length === 0) {
88       return alert("Penerima notifikasi belum di pilih.");
89     }
90     axios.post("manage/notification", {
91       title: customNotificationTitle,
92       description: customNotificationDescription,
93       type: "custom",
94       participants: customNotificationParticipants
95     }).then((response) => {
96       setShowCustomNotifModal(false);
97       refreshNotifications(exam);
98     })
99   }
100
101   return (
102     <div>
103       <Row>
104         <Col>
105           <h3>Notifikasi</h3>
106         </Col>
107         <Col>
108           <div className="text-right">
109             <Button className="mr-2" color="primary" onClick={() => setShowNewNotificationsModal(true)}><
110               FontAwesomeIcon icon={faPlus} /></Button>
111           </div>
112         </Col>
113       </Row>
114       <Row>
115         <Col>
116           <Table>
117             <tbody>
118               {notifications.map((notif) => <NotificationListerChild key={"notif-" + notif._id} notif={notif}>
119                 <small>On Deleted={() => refreshNotifications(exam)} On Edited={() => refreshNotifications(exam)} />
120               </NotificationListerChild>)
121             </tbody>
122           </Table>
123         </Col>
124       </Row>
125
126       <Modal isOpen={showNewNotificationsModal}>
127         <ModalHeader>Pilih jenis notifikasi yang ingin diberikan:</ModalHeader>
128         <ModalBody>
129           <Button block onClick={() => {
130             setShowNewNotificationsModal(false);
131             setShowPasswordNotifModal(true);
132           }}>
133             <b>Password</b><br />
134             <small>Gunakan jenis ini untuk menyebarkan informasi mengenai password untuk tiap peserta.</small>
135           </Button>
136           <Button block onClick={() => {
137             setShowNewNotificationsModal(false);
138             setShowCustomNotifModal(true);
139           }}>
140             <b>Lainnya</b><br />
141             <small>Gunakan jenis ini untuk memberikan informasi pada peserta.</small>
142           </Button>
143         </ModalBody>
144         <ModalFooter>
145           <Button onClick={() => setShowNewNotificationsModal(false)}>Batal</Button>
146         </ModalFooter>
147       </Modal>
148
149     /* Modal buat tambah notifikasi */
150   )

```

```

146     <Modal isOpen={showPasswordNotifModal} size="xl">
147         <Form onSubmit={handleGenMass}>
148             <ModalHeader>Notifikasi Password Akun</ModalHeader>
149             <ModalBody>
150                 <Row>
151                     <Col>
152                         <FormGroup>
153                             <Label>Service/Url</Label>
154                             <Input name="service" placeholder="Contoh: judgeujian.ftis.unpar" onChange={(e) =>
155                                 setNniServiceList(e.target.value)} value={nniServiceList} required />
156                         </FormGroup>
157                         <FormGroup>
158                             <Label>Daftar Password</Label>
159                             <Input type="textare" name="passwords" onChange={(e) => setNniPasswordList(
160                                 e.target.value)} value={nniPasswordList} required />
161                             <FormText color="muted">Bentuk daftar password adalah <code>npm | username service | password</code>.</FormText>
162                         </FormGroup>
163                     </Col>
164                     <Col>
165                         <Table bordered striped>
166                             <thead>
167                                 <tr>
168                                     <th>NPM - Username</th>
169                                     <th>Service Username</th>
170                                     <th>Service Password</th>
171                             </tr>
172                         </thead>
173                         <tbody>
174                             {transformedPasslist.map((data) => {
175                                 let [npm = "", uname = "", pass = ""] = data;
176
177                                 if (!participantLister.hasOwnProperty(npm)) {
178                                     return <tr>
179                                         <td colSpan={3} className="text-danger"><i>{npm} tidak ditemukan.</i></td>
180                                     </tr>
181
182                                 return <tr>
183                                     <td>{npm} - {participantLister[npm].username}</td>
184                                     <td>{uname}</td>
185                                     <td>{pass}</td>
186                                 </tr>
187                             ))}
188                         </tbody>
189                     </Table>
190                 </Row>
191             <ModalBody>
192                 <Button onClick={() => setShowPasswordNotifModal(false)}>Batal</Button>
193                 <Button type="submit">Sebarkan</Button>
194             </ModalBody>
195         </Form>
196     </Modal>
197
198     /* Add modal for special users */
199     <Modal isOpen={showCustomNotifModal} size="lg" backdrop>
200         <Form onSubmit={handleCustomNotifCreateRequest}>
201             <ModalHeader>Tambah Notifikasi Kustom</ModalHeader>
202             <ModalBody>
203                 <FormGroup>
204                     <Label>Judul</Label>
205                     <Input type="text" defaultValue={customNotificationTitle} onChange={(e) =>
206                         setCustomNotificationTitle(e.target.value)} />
207                 </FormGroup>
208                 <FormGroup>
209                     <Label>Penerima Notifikasi</Label>
210                     <Input plaintext value={customNotificationParticipants.length + " orang, (klik untuk mengubah)"}
211                         onClick={() => setAudiencePickerShow(true)} onChange={() => { }} />
212                 </FormGroup>
213                 <FormGroup>
214                     <Label> Deskripsi
215                     </Label>
216                     <SunEditor setOptions={{ buttonList: buttonList.formatting, height: 200 }} setContents={customNotificationDescription} onChange={(val) => setCustomNotificationDescription(val)} />
217                 </FormGroup>
218             </ModalBody>
219             <ModalFooter>
220                 <Button type="button" color="secondary" onClick={() => setShowCustomNotifModal(false)}>Batal</Button>
221                 <Button type="submit" color="primary">Buat</Button>
222             </ModalFooter>
223         </Form>
224     </Modal>
225
226     /* Modal pendukung */
227     <When condition={audiencePickerShow}>
228         <NotificationAudiencePicker participants={exam.participants} selectedAudiences={customNotificationParticipants} onSelected={(val) => {
229             setCustomNotificationParticipants(val)
230             setAudiencePickerShow(false)
231         }} onCanceled={() => setAudiencePickerShow(false)} />
232     </When>
233   </div>
234 )
235 }

```

```
| 237 export default ExamNotificationLister
```

Listing B.121 slot-jawaban-child.js

```

1 import { faCheck, faPen, faTimes, faTrash } from '@fortawesome/free-solid-svg-icons'
2 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
3 import React, { useState } from 'react'
4 import { Else, If, Then } from 'react-if'
5 import { Button, Input, Modal, ModalBody, ModalFooter } from 'reactstrap'
6 import { axios } from '~/apicall'
7 import { useAdminStore } from '~/components/use-store'

8 function SlotJawabanChild({ answer_slot }) {
9
10    const [openDeleteConfirm, setOpenDeleteConfirm] = useState(false)
11    const [letEdit, setLetEdit] = useState(false);
12
13    const [inputFormat, setInputFormat] = useState(answer_slot.format)
14
15    const adminStore = useAdminStore();
16
17    function handleConfirmDelete() {
18        axios.delete(`manage/answerslot/${answer_slot._id}`).then(() => {
19            adminStore.examFetch(adminStore.selectedExam);
20        })
21    }
22
23    function handleSlotUpdate() {
24        axios.put(`manage/answerslot/${answer_slot._id}`, {
25            format: inputFormat
26        }).then(() => {
27            adminStore.examFetch(adminStore.selectedExam);
28            setLetEdit(false);
29        })
30    }
31
32    return (<>
33        <tr>
34            <If condition={!letEdit}>
35                <Then>
36                    <td>
37                        <p className="font-monospace font-weight-bold">
38                            {answer_slot.format}
39                        </p>
40                    <td className="text-right">
41                        <Button className="ml-2" color="secondary" onClick={() => setLetEdit(true)}><FontAwesomeIcon icon={faPen} /></Button>
42                        <Button className="ml-2" color="danger" onClick={() => setOpenDeleteConfirm(true)}><FontAwesomeIcon icon={faTrash} /></Button>
43                    </td>
44                </Then>
45                <Else>
46                    <td>
47                        <Input type="text" defaultValue={inputFormat} onChange={(e) => setInputFormat(e.target.value)} />
48                    </td>
49                    <td>
50                        <Button className="ml-2" color="primary" onClick={handleSlotUpdate}><FontAwesomeIcon icon={faCheck} /></Button>
51                        <Button className="ml-2" color="danger" onClick={() => {
52                            setLetEdit(false)
53                        }}><FontAwesomeIcon icon={faTimes} /></Button>
54                    </td>
55                </Else>
56            </If>
57        </tr>
58        <Modal isOpen={openDeleteConfirm} toggle={() => setOpenDeleteConfirm(!openDeleteConfirm)} backdrop=>
59            <ModalBody>
60                <p> Are you sure to delete <span className="font-monospace">{answer_slot.format}</span>?
61            </ModalBody>
62            <ModalFooter>
63                <Button color="secondary" onClick={() => setOpenDeleteConfirm(false)}>Cancel</Button>
64                <Button color="danger" onClick={handleConfirmDelete}><FontAwesomeIcon icon={faTrash} /> Delete</Button>
65            </ModalFooter>
66        </Modal>
67    </>
68    )
69 }
70
71
72 }

73 export default SlotJawabanChild

```

Listing B.122 slot-jawaban.js

```

1 import React, { useState } from 'react'
2 import { Row, Col, Button, Table, Modal, ModalBody, ModalHeader, ModalFooter, InputGroup, Input, InputGroupAddon } from 'reactstrap'
3 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
4 import { faPlus, faFileDownload } from '@fortawesome/free-solid-svg-icons'
5 import { axios } from '~/apicall'
6 import fileDownload from 'js-file-download'
7 import { useAdminStore } from '~/components/use-store'
8 import SlotJawabanChild from './slot-jawaban-child'

9 function SlotJawaban({ exam }) {

```

```

11  const [openSlotAdder, setOpenSlotAdder] = useState(false)
12  const [slotAdding, setSlotAdding] = useState(false)
14  const [inputLeft, setInputLeft] = useState(undefined)
15  const [inputRight, setInputRight] = useState(undefined)
17  const adminStore = useAdminStore();
19  function handleAnswerBundleDownload() {
20    axios.get("manage/exam/" + exam._id + "/answers", {
21      responseType: "blob"
22    }).then((resp) => {
23      fileDownload(resp.data, resp.headers['x-filename']);
24    })
25  }
27  function handleAnswerSlotInsert() {
28    setSlotAdding(true)
29    axios.post("manage/answerslot", {
30      exam: exam._id,
31      format: inputLeft + "%xxxyy%" + inputRight
32    }).then(() => {
33      adminStore.examFetch(exam._id);
34    }).finally(() => {
35      setSlotAdding(false)
36      setOpenSlotAdder(false);
37      setInputLeft(undefined);
38      setInputRight(undefined);
39    })
40  }
41  return (
42    <div>
43      <Row>
44        <Col>
45          <h3>Slot jawaban</h3>
46        </Col>
47        <Col>
48          <div className="text-right">
49            <Button className="mr-2" color="success" onClick={handleAnswerBundleDownload}><FontAwesomeIcon icon={faFileDownload} /> Zip Answers</Button>
50            <Button className="mr-2" color="primary" onClick={() => setOpenSlotAdder(true)}><FontAwesomeIcon icon={faPlus} /></Button>
51          </div>
52        </Col>
53      </Row>
54      <Table>
55        <tbody>
56          {(exam.answer_slot || []).map((ans, id) =>
57            <SlotJawabanChild answer_slot={ans} key={id} />
58          )}
59        </tbody>
60      </Table>
61      <Modal size="md" isOpen={openSlotAdder}>
62        <ModalHeader toggle={() => setOpenSlotAdder(false)}>Tambah Slot</ModalHeader>
63        <ModalBody>
64          <p>Masukkan format jawaban:</p>
65          <InputGroup>
66            <Input type="text" value={inputLeft} onChange={(e) => setInputLeft(e.target.value)} disabled={slotAdding} />
67            <InputGroupAddon addonType="append">xxxyy</InputGroupAddon>
68            <Input type="text" value={inputRight} onChange={(e) => setInputRight(e.target.value)} disabled={slotAdding} />
69          </InputGroup>
70        </ModalBody>
71        <ModalFooter>
72          <Button color="secondary" onClick={() => setOpenSlotAdder(false)} disabled={slotAdding}>Batal</Button>
73          <Button color="primary" onClick={handleAnswerSlotInsert} disabled={slotAdding}>Tambah</Button>
74        </ModalFooter>
75      </Modal>
76    </div>
77  )
78}
79
80}
81
82 export default SlotJawaban

```

Listing B.123 Exam.js

```

1 import React, { useEffect, useState } from 'react';
2 import { observer } from 'mobx-react';
3 import Container from "reactstrap/lib/Container";
4 import Row from "reactstrap/lib/Row";
5 import Col from "reactstrap/lib/Col";
6 import Button from "reactstrap/lib/Button";
7 import Badge from "reactstrap/lib/Badge";
8 import Table from "reactstrap/lib/Table";
9 import { If, Then, Else } from 'react-if';
10 import { Link } from "react-router-dom";
11 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
12 import { faTrash } from '@fortawesome/free-solid-svg-icons';
13 import moment from "moment";
14 import { Modal, ModalHeader, ModalBody, ModalFooter } from 'reactstrap';
15 import { useAdminStore } from '~/components/use-store';

17 function Exam() {
18  const adminStore = useAdminStore();

```

```

19  useEffect(() => {
20    adminStore.examFetch();
21    console.log("Fetching examination...");
22    return () => { };
23  }, [adminStore])
24
25  const [prepareDelete, setPrepareDelete] = useState({})
26
27  function handleDelete() {
28    adminStore.examDelete(prepareDelete._id).then(() => {
29      setPrepareDelete({});
30    })
31  }
32
33  return (
34    <Container>
35      <Row className="mt-4">
36          <Col>
37              <h1>Ujian</h1>
38          </Col>
39          <Col>
40              <div className="text-right">
41                  <Button color="primary" size="lg" tag={Link} to="/admin/exam/new">Create New</Button>
42              </div>
43          </Col>
44      </Row>
45      <Row>
46          <Col>
47              <Table striped>
48                  <thead>
49                      <tr>
50                          <th>Lecture Info</th>
51                          <th>Shift</th>
52                          <th colSpan={2}>Duration (Start-End)</th>
53                          <th></th>
54                  </tr>
55              </thead>
56              <tbody>
57                  {adminStore.exams.map((exam, i) =>
58                      <tr key={i}>
59                          <td>
60                              <b>{(exam.lecture || {}).name} ({(exam.lecture || {}).lecture_code})</b> <br />
61                              {(exam.lecture_period || {}).period_code} <Badge color={exam.uts ? "info" : "success"}>{exam.uts ? "UTS" : "UAS"}</Badge>
62                          </td>
63                          <td>
64                              <p className="h3">
65                                  <If condition={exam.shift === null}>
66                                      <Then></Then>
67                                      <Else>{exam.shift}</Else>
68                                  </If>
69                              </p>
70                          </td>
71                          <td>
72                              {(exam.time_start) - (exam.time_ended)}
73                          </td>
74                          <td>
75                              Durasi: {(exam.time_duration / 60} Menit<br />
76                              Dimulai pada: {(exam.time_opened || "(belum di mulai")}<br />
77                          </td>
78                          <td className="text-right">
79                              <Button className="mx-2 d-none d-md-block" color="warning" tag={Link} to={`/admin/exam/${exam._id}/detail`}>Lihat</Button>
80                              <Button className="mx-2 d-block d-md-none" color="warning" tag={Link} to={`/admin/exam/${exam._id}/minipanel`}>Lihat</Button>
81                              <Button className="mx-2" color="danger" onClick={() => setPrepareDelete(exam)}>Hapus</Button>
82                          </td>
83                      </tr>
84                  )
85              </tbody>
86          </Table>
87      </Col>
88  </Row>
89
90  <Modal isOpen={!!(prepareDelete._id)} toggle={() => setPrepareDelete({})} backdrop>
91      <ModalHeader toggle={() => setPrepareDelete({})}>Konfirmasi Hapus</ModalHeader>
92      <ModalBody>
93          <h4>Are you sure to delete this exam?</h4>
94          <h3>#{prepareDelete._id} <Badge color={prepareDelete.uts ? "info" : "success"}>{prepareDelete.uts ? "UTS" : "UAS"}</Badge> {prepareDelete.lecture?.name} ({(prepareDelete.lecture || {}).lecture_code})</h3>
95          <p className="lead"> {(prepareDelete.shift === 0) ? "Shiftless" : ("Shift" + prepareDelete.shift)} | {moment(
96              prepareDelete.time_start).format("LLL")}) | {(prepareDelete.duration / 60} Menit.</p>
97      </ModalBody>
98      <ModalFooter>
99          <Button color="secondary" onClick={() => setPrepareDelete({})}>Cancel</Button>{' '}
100         <Button color="danger" onClick={handleDelete}><FontAwesomeIcon icon={faTrash} /> Hapus</Button>
101     </ModalFooter>
102  </Modal>
103  </Container >
104 }
105
106 export default observer(Exam)

```

Listing B.124 Index.js

```
| 1 import React from 'react';
```

```

2 import { observer } from 'mobx-react';
3 import { Route, Switch, Redirect } from "react-router-dom";
4 import Exam from './Exam';
5 import Detail from './detail/Detail';
6 import AdminNavbar from '~/components/admin/navbar/AdminNavbar';
7 import Participants from './participants/Participants';
8 import ExamCreate from './create/create';
9 import ScreenTime from './screentime/screen';
10 import ExamMinipanel from './detail/minipanel/minipanel';

12 function Index({ match }) {
13   return (
14     <>
15       <AdminNavbar />
16       <Switch>
17         <Route exact path={match.path} component={Exam} />
18         <Route path={`${match.path} /new`} component={ExamCreate} />
19         <Route path={`${match.path} /:id/minipanel`} component={ExamMinipanel} />
20         <Route path={`${match.path} /:id/detail`} component={Detail} />
21         <Route path={`${match.path} /:id/participant`} component={Participants} />
22         <Route path={`${match.path} /:id/screen`} component={ScreenTime} />
23         <Redirect push to={`${match.path} /`} />
24       </Switch>
25     </>
26   )
27 }

29 export default observer(Index);

```

Listing B.125 PaperHeader.js

```

1 import React from 'react';
2 import Container from "reactstrap/lib/Container";
3 import Row from "reactstrap/lib/Row";
4 import Col from "reactstrap/lib/Col";
5
6 import UnparLogo from "~/assets/unpar-wb-01.svg";
7 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
8 import { faChevronLeft } from '@fortawesome/free-solid-svg-icons';
9 import { Link } from 'react-router-dom';
10
11 function PaperHeader({ backLink }) {
12   return (
13     <header className="my-5 d-none d-print-block">
14       <Container>
15         <Row className="justify-content-center align-items-center">
16           <Col xs={1}>
17             <div className="text-right">
18               <img src={UnparLogo} alt="unpar-logo" style={{ height: "4rem", width: "auto" }} />
19             </div>
20           </Col>
21           <Col md={5}>
22             <b>Universitas Katolik Parahyangan</b>
23             <p className="lead m-0">Fakultas Teknologi Informasi dan Sains</p>
24           </Col>
25         </Row>
26       </Container>
27     </header>
28     <header className="my-5 d-print-none">
29       <Container>
30         <Row>
31           <Col>
32             <Link to={backLink}><FontAwesomeIcon icon={faChevronLeft} /> Back to Exam Details</Link>
33           </Col>
34         </Row>
35       </Container>
36     </header>
37   )
38 }
39
40
41 export default PaperHeader;

```

Listing B.126 Participants.js

```

1 import React, { useEffect, useState } from 'react'
2 import { useRouteMatch } from 'react-router-dom'
3 import { Container, Row, Col, Badge, Table } from 'reactstrap';
4 import { useAdminStore } from '~/components/use-store';
5 import PaperHeader from './PaperHeader';
6 import { observer } from 'mobx-react';
7 import { axios } from '~/apicall';
8 import moment from "moment";
9 import { When, If, Then, Else } from 'react-if';
10 import { faDesktop } from '@fortawesome/free-solid-svg-icons';
11 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
12 import { createUseStyles } from 'react-jss';
13
14 const useStyles = createUseStyles({
15   root: {
16     "&>section": {
17       pageBreakBefore: "always"
18     },
19     "&>section:first-child": {

```

```

20     pageBreakBefore: "avoid"
21   },
22
23   "& table td": {
24     verticalAlign: "middle"
25   },
26   "& table tr > *": {
27     borderColor: "black !important",
28     borderWidth: "3px !important"
29   },
30   "& .table-striped tbody tr:nth-of-type(odd)": {
31     backgroundColor: "rgba(0,0,0,0.3) !important"
32   }
33 }
34 )))

35 function Participants() {
36   const path = useRouteMatch("/admin/exam/:id/participant/:type");
37   const adminStore = useAdminStore();

38   // eslint-disable-next-line
39   const [participantByRoom, setParticipantByRoom] = useState([]);
40   // eslint-disable-next-line
41   const [isLoading, setIsLoading] = useState(true);
42   const [locations, setLocations] = useState({})

43   useEffect(() => {
44     adminStore.selectedExam = path.params.id;
45     adminStore.examFetch(path.params.id);

46     axios.get("manage/exam/" + path.params.id + "/participants").then(resp => {
47       let participantByRoom = {};
48       resp.data.data.forEach(el => {
49         if (!participantByRoom.hasOwnProperty(el.computer.location)) {
50           participantByRoom[el.computer.location] = {
51             location: el.computer.location,
52             participants: []
53           }
54         }
55         participantByRoom[el.computer.location].participants.push(el)
56       })
57       setParticipantByRoom(Object.values(participantByRoom));
58       // setParticipants(resp.data.data);
59       // setIsLoading(false);
60     });

61     axios.get("manage/location").then(resp => {
62       let loka = {};
63       resp.data.data.forEach(el => {
64         loka[el._id] = { ...el, ...{ computers: null } };
65       })
66       setLocations(loka);
67     })
68     return () => { };
69   }, [adminStore, path.params.id]);

70   const styles = useStyle();
71   return (
72     <div className={styles.root}>
73       {participantByRoom.map(({ location, participants }, id) => <section key={id}>
74         <PaperHeader backLink={"/admin/exam/" + path.params.id + "/detail"} />
75         <Container>
76           <Row className="justify-content-center">
77             <Col xs={12}>
78               <div className="text-center mb-3">
79                 <h4>Ujian {adminStore.exam.uts ? "Tengah" : "Akhir"} Semester</h4>
80                 <h3><small>({adminStore.exam.lecture.lecture_code}) - {adminStore.exam.lecture.name}</small></h3>
81                 <p className="lead">
82                   <When condition={adminStore.exam.shift !== 0}>
83                     <Badge color={adminStore.exam.uts ? "info" : "success"}>Shift {adminStore.exam.shift}</Badge> {' '}
84                   </When>
85                   {moment(adminStore.exam.time_start).format("LLL")}
86                 </p>
87               </div>
88             </Col>
89           </Row>
90         <Row>
91           <Col>
92             <p className="text-center lead"> Ruangan {{locations[location] || {}}.room_name || "#" + location}</p>
93             <If condition={path.params.type === "signature"}>
94               <Then>
95                 <Table bordered>
96                   <thead>
97                     <tr>
98                       <th width="5%">No</th>
99                       <th width="10%">Seat</th>
100                      <th width="15%">NPM</th>
101                      <th width="30%">Name</th>
102                      <th width="20%">Signature</th>
103                    </tr>
104                  </thead>
105                  <tbody>
106                    {participants.sort((a, b) => a.computer.name.localeCompare(b.computer.name)).map((par, i) => {
107                      return <tr key={i}>
108                        <td>{i + 1}</td>
109                        <td><FontAwesomeIcon icon={faDesktop} /> {par.computer.name}</td>
110                        <td className="text-monospace">{par.npm}</td>
111                        <td>{par.display_name}</td>
112                      </tr>
113                    )}
114                  </tbody>
115                </Table>
116              </Then>
117            </If>
118          </Col>
119        </Row>
120      </Container>
121    </div>
122  )
123
```

```

119             </tr>
120         )}
121     </tbody>
122   </Table>
123 </Then>
124 <Else>
125   <div style={{ columnCount: 2 }}>
126     <Table striped>
127       <thead>
128         <tr className="text-center">
129           <th>Seat</th>
130           <th>NPM</th>
131         </tr>
132       </thead>
133       <tbody>
134         {participants.sort((a, b) => a.username.localeCompare(b.username)).map((par, i) => {
135           return <tr key={i}>
136             <td className="text-center"><FontAwesomeIcon icon={faDesktop} /> {par.computer.name}</td>
137             <td className="text-center text-monospace">{par.npm}</td>
138           </tr>
139         ))}
140       </tbody>
141     </Table>
142   </div>
143 </Else>
144 </If>
145 </Col>
146 </Row>
147 </Container>
148 </section>
149 );
150 </div>
151 }

154 export default observer(Participants);

```

Listing B.127 screen.js

```

1 import { faHistory, faStop, faStopwatch } from '@fortawesome/free-solid-svg-icons';
2 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
3 import { observer } from 'mobx-react';
4 import React, { useEffect, useState } from 'react';
5 import Countdown from 'react-countdown';
6 import { Else, If, Then } from 'react-if';
7 import { useRouteMatch } from 'react-router-dom';
8 import { Badge, Button, Col, Container, Row } from 'reactstrap';
9 import { axios } from './apicall';
10 import { useAdminStore } from './components/use-store';

12 function ScreenTime() {
13   const match = useRouteMatch("/admin/exam/:id/screen");
14   const adminStore = useAdminStore();
15   // eslint-disable-next-line
16   const [waitForTime, setWaitRefreshTime] = useState(120)
17   useEffect(() => {
18     if (adminStore.selectedExam !== match.params.id) {
19       adminStore.selectedExam = match.params.id;
20       adminStore.examFetch(match.params.id);
21     }
22
23     let timer = setInterval(() => {
24       adminStore.examFetch(match.params.id);
25     }, waitForTime * 1000);
26
27     return () => {
28       clearInterval(timer);
29     };
30   }, [adminStore, match.params.id, waitForTime]);

32 // useEffect(() => {
33 //   if (adminStore.exam.time_left < 300 && waitForTime !== 30) {
34 //     setWaitRefreshTime(30);
35 //   } else if (adminStore.exam.time_left === 0 || adminStore.exam.time_left === null) {
36 //     setWaitRefreshTime(120);
37 //   }
38 //   return () => {};
39 // }, [adminStore, waitForTime])

41 function handleStart() {
42   axios.post("manage/exam/" + match.params.id + "/start").then(() => {
43     adminStore.examFetch(match.params.id);
44   })
45 }

47 function handleStop() {
48   axios.post("manage/exam/" + match.params.id + "/close").then(() => {
49     adminStore.examFetch(match.params.id);
50   })
51 }

53 function handleReset() {
54   axios.delete("manage/exam/" + match.params.id + "/start").then(() => {
55     adminStore.examFetch(match.params.id);
56   })
57 }

```

```

59   const { exam } = adminStore;
60   return (
61     <div>
62       <Container>
63         <Row style={{ minHeight: "calc(100vh - 64px)" }} className="align-items-center">
64           <Col>
65             <div>
66               <div className="d-flex align-items-center">
67                 <h4 className="display-4 flex-grow-1">
68                   <Badge color={exam.uts ? "info" : "success"}>{exam.uts ? "UTS" : "UAS"}</Badge>{' '}
69                   {(exam.lecture.lecture_code){(exam.shift !== null && exam.shift !== 0) ? " SHIFT-" +
70                     exam.shift : ""}}
71                 </h4>
72                 <h4 className="font-weight-bold">
73                   {(exam.time_duration / 60)} Menit
74                 </h4>
75                 <h1 className="display-3">{exam.lecture.name}</h1>
76               </div>
77             <div>
78               <If condition={!exam.time_opened && exam.time_left >= 1}>
79                 <Then>
80                   <p className="display-1 text-center font-weight-bold my-2 py-3" style={{ fontSize: "10rem" }}>
81                     <Countdown date={Date.now() + exam.time_left * 1000} onComplete={() =>
82                       adminStore.examFetch(match.params.id)} autoStart={!exam.time_opened} />
83                   </p>
84                 </Then>
85                 <Else>
86                   <If condition={exam.time_left >= 1}>
87                     <Then>
88                       <p className="display-2 font-weight-bold text-center bg-primary text-light my-2 py-3">READY</p>
89                     </Then>
90                     <Else>
91                       <p className="display-2 font-weight-bold text-center bg-dark text-light my-2 py-3">FINISH</p>
92                     </Else>
93                   </If>
94                 </Else>
95               </div>
96               <div className="mt-5">
97                 <pControl/>
98                 <div>
99                   <If condition={!exam.time_opened}>
100                     <Then>
101                       <Button className="mr-2" color="primary" onClick={handleStart}><FontAwesomeIcon icon={faStopwatch} /> Start Timer</Button>
102                     <p className="mt-3">
103                       Untuk memulai dan membuka tempat pengumpulan, anda dapat menekan <b>Start Timer</b>.
104                   </Then>
105                 <Else>
106                   <If condition={exam.time_left === 0}>
107                     <Then>
108                       <Button className="mr-2" color="danger" onClick={handleReset}><FontAwesomeIcon icon={faHistory} /> Reset Timer</Button>
109                     </Then>
110                   <Else>
111                     <Button className="mr-2" color="warning" onClick={handleStop}><FontAwesomeIcon icon={faStop} /> Stop Timer</Button>
112                   </Else>
113                   <Button className="mr-2" color="danger" onClick={handleReset}><FontAwesomeIcon icon={faHistory} /> Reset Timer</Button>
114                 <p className="mt-3">
115                   <b>Penting:</b> Dengan menutup/mereset timer, anda akan menutup tempat pengumpulan peserta ujian.
116                 </p>
117               </Else>
118             </If>
119           </Else>
120         </div>
121       </Col>
122     </Row>
123   </Container>
124 </div>
125 )
126 }
127 }
128 }

130 export default observer(ScreenTime)

```

Listing B.128 Index.js

```

1 import React, { Component } from 'react';
2 import { Else, If, Then } from 'react-if';
3 import { Redirect, withRouter } from "react-router-dom";
4 import { observer, inject } from "mobx-react"
5 import Mainindex from './Mainindex';
6 import Nologin from './Nologin';

8 class Index extends Component {
9   render() {
10     const { adminStore, match } = this.props;

```

```

12     return (
13       <>
14         <If condition={!adminStore.user}>
15           <Then>
16             <Nologin />
17           </Then>
18           <Else>
19             <Mainindex />
20             <Redirect path={match.path} exact to="/admin/exam/" />
21           </Else>
22         </If>
23       </>
24     )
25   }
26 }

28 export default inject("adminStore")(
29   withRouter(
30     observer(Index)
31   )
32 );

```

Listing B.129 Mainindex.js

```

1 import React, { Component } from 'react';
2 import { observer } from 'mobx-react';
3 import AdminNavbar from './components/admin/navbar/AdminNavbar';

5 class Mainindex extends Component {
6   render() {
7     return (
8       <>
9         <AdminNavbar />
10        </>
11      )
12    }
13  }
14 Mainindex.propTypes = {
15  }

17 export default (observer(Mainindex));

```

Listing B.130 Nologin.js

```

1 import React from 'react';
3 import Container from "reactstrap/lib/Container";
4 import Row from "reactstrap/lib/Row";
5 import Col from "reactstrap/lib/Col";
6 import Spinner from "reactstrap/lib/Spinner";

8 const Nologin = () => {
9   return (
10     <div className="bg-light">
11       <Container>
12         <Row className="h-100vh align-items-center">
13           <Col className="text-center">
14             <Spinner/>
15             <p className="lead">Sedang menghubungi server...</p>
16           </Col>
17         </Row>
18       </Container>
19     </div>
20   )
21 }

23 export default Nologin;

```

Listing B.131 delete.js

```

1 import React from 'react';
2 import { Button, Modal, ModalBody, ModalFooter, ModalHeader, Table } from 'reactstrap';
3 import { useEntityStore } from './components/use-store';

5 function EntityDelete({ onDeleteSucceed = () => {} , onDeleteCanceled = () => {} , entityRules, item = null }) {
6   let entityStore = useEntityStore()
7   let fields = entityRules.fields;

9   function handleDeleteRequest() {
10     entityStore.deleteItem(entityRules.apiPath, item?._id).then(() => onDeleteSucceed());
11   }

13   return (
14     <Modal isOpen={!item} toggle={() => onDeleteCanceled}>
15       <ModalHeader>
16         Delete Confirmation
17       </ModalHeader>
18       <ModalBody>
19         <Table bordered style={{ columns: 2 }}>
20           <tbody>
21             {Object.keys(fields).map((el) => {
22               let content = item?.[el] || "";
23             })
24           <tr>
25             <td>{el}</td>
26             <td>{content}</td>
27           </tr>
28         </tbody>
29       </ModalBody>
30     </Modal>
31   )
32 }

```

```

23         if (fields[el].type === "link") {
24             content = item?.[el]?.[fields[el]]?.link_label;
25         } else if (fields[el].type === "links") {
26             content = item?.[el]?.map(item => item[fields[el]]?.link_label).join(", ");
27         } else if (fields[el].type === "json") {
28             content = <pre class="pre">{JSON.stringify(item?.[el], "", 2)}</pre>
29         }
30     return <tr key={el}>
31         <th>{fields[el].name || el}</th>
32         <td>{(content)}</td>
33     </tr>
34     })
35   </tbody>
36 </Table>
37 <hr />
38 <p>Are you sure to delete this {entityRules.name}?</p>
39 </ModalBody>
40 <ModalFooter>
41     <Button type="button" color="danger" onClick={handleDeleteRequest}>Delete</Button>
42     <Button type="button" color="secondary" onClick={() => onDeleteCanceled()}>Cancel</Button>
43 </ModalFooter>
44 </Modal>
45 )
46 }

48 export default EntityDeleteor

```

Listing B.132 editor.js

```

1 import { faChevronLeft, faPlus } from '@fortawesome/free-solid-svg-icons';
2 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
3 import { inject, observer } from 'mobx-react';
4 import React, { Component } from 'react';
5 import { Else, If, Then, When } from 'react-if';
6 import { Link, withRouter } from 'react-router-dom';
7 import { Alert, Button, Col, Container, Form, FormGroup, Input, Label, Row } from 'reactstrap';
8 import Datepicker from '~/components/date-time-picker/date-time-picker';
9 import EntityDeleteor from './delete';
10 import qs from 'query-string'

13 class EntityEditor extends Component {

15     state = {
16         field: [],
17         message: [],
18         itemToDelete: undefined,
19         returnPath: undefined
20     }

22     componentDidMount() {
23         const { match, entityStore, entityRules, globalEntityRules } = this.props;
24         entityStore.selectedEntity = entityRules.apiPath;

26         // FETCH all required shits.
27         let fields = entityRules.fields;
28         Object.keys(fields).forEach(el => {
29             let theField = fields[el];

31             if (theField.type === "link" || theField.type === "links") {
32                 entityStore.fetch(globalEntityRules[theField.table].apiPath, globalEntityRules[theField.table].apiPath);
33             }
34         });

37         if (match.params.id !== "new") {
38             entityStore.selectedEntityId = match.params.id;
39             entityStore.fetchItem(entityRules.apiPath, match.params.id).then((data) => {
40                 let finalizedFields = {};
41                 Object.keys(fields).forEach(el => {
42                     finalizedFields[el] = entityStore.item[el];
43                     if (fields[el].type === "json") {
44                         finalizedFields[el] = JSON.stringify(finalizedFields[el], null, 2);
45                     }
46                 });
47                 this.setState({ field: finalizedFields })
48             })
49         } else {
50             // set the loading false since we don't need to load anything when
51             // it's new.
52             entityStore.isLoading = false;
53             // reset the entity store
54             entityStore.selectedEntityId = undefined;
55             this.setState({ field: {} });
56         }
57     }

59     let returnPath = "/admin/manage/" + entityRules.entityName;
60     if (window.location.search) {
61         let parsedQs = qs.parse(window.location.search);
62         returnPath = parsedQs?.referral || returnPath;
63     }
64     this.setState({ returnPath: returnPath });
65 }

67     componentDidUpdate(prevProps, prevState) {
68         const { match } = this.props;

```



```

160     let entityLink = globalEntityRules[theFieldDefinition.table];
161
162     theComponent = <Input
163         type="select"
164         multiple
165         value={field[el]?.map((obj) => obj?._id || obj)}
166         disabled={!theFieldDefinition?.allow?.update || false} || entityStore.isLoading}
167         onChange={e => this.setState({
168             field: {
169                 ...field,
170                 [el]: Array.from(e.target.selectedOptions).map(e => e.value)
171             }
172         })}
173
174         {entityStore.getEntityItems(entityLink.apiPath)?.map((item) => <option value={item._id} key={item._id}>
175             {item[theFieldDefinition.link_label]}
176             </option>)}
```

</Input>
177 } else if (theFieldDefinition.type === "json") {
178 // try parse:
179 let parseAble = false;
180 try {
181 JSON.parse(field[el]);
182 parseAble = true;
183 } catch (e) {
184
185 }
186 theComponent = <>
187 <Input
188 type="textarea"
189 rows={10}
190 invalid={!parseAble}
191 defaultValue={field[el]}
192 disabled={!theFieldDefinition?.allow?.update || false} ||
193 entityStore.isLoading}
194 onChange={e => this.setState({ field: { ...field, [el]: e.target.value } })} />
195
196 </>
197 } else if (theFieldDefinition.type === "datetime") {
198 theComponent = <>
199 <DatePicker
200 defaultValue={field[el]}
201 disabled={!theFieldDefinition?.allow?.update || false} ||
202 entityStore.isLoading}
203 onChange={e => this.setState({ field: { ...field, [el]: e } })} />
204 </>
205 } else {
206 theComponent = <Input
207 type={theFieldDefinition.type}
208 defaultValue={field[el]}
209 disabled={!theFieldDefinition?.allow?.update || false} || entityStore.isLoading}
210 onChange={e => this.setState({ field: { ...field, [el]: e.target.value } })} />
211
212 return <FormGroup key={"form-" + id}>
213 <Label>{fields[el].name || el}</Label>
214 {theComponent}
215 </FormGroup>
216 }
217
218 <FormGroup className="pt-4">
219 <Button type="submit" color="primary" size="lg">Save</Button>
220 <When condition={match.params.id !== "new"}>
221 <Button type="button" color="danger" onClick={(e) => this.handleDeleteRequest(e)} className="ml-4">Delete</Button>
222 </When>
223 </FormGroup>
224 </Col>
225 </Row>
226 </Container>
227 <EntityDelete
228 entityRules={entityRules}
229 onDeleteCanceled={() => this.setState({ itemToDelete: undefined })}
230 onDeleteSucceed={(e) => this.handleDeleteSucceed(e)}
231 item={itemToDelete}
232 />
233 </div >
234
235 </div >
236
237 }
238
239 handleDeleteSucceed() {
240 const { history, entityStore, entityName, entityRules } = this.props;
241 this.setState({ itemToDelete: undefined })
242 entityStore.fetch(entityRules.apiPath)
243 history.push("/admin/manage/" + entityName)
244 }
245
246 handleDeleteRequest() {
247 const { entityStore } = this.props;
248 this.setState({ itemToDelete: entityStore.item })
249 }
250
251
252 export default withRouter(inject("entityStore")(observer(EntityEditor)));

Listing B.133 handler.js

```

1 import React from 'react';
2 import { Route, Switch, useRouteMatch } from 'react-router-dom';
3 import EntityEditor from './editor';
4 import EntityIndex from './index';

6 function EntityHandler({ globalEntityRules, entityRules }) {
7   const match = useRouteMatch("/admin/manage/:entity");
8   return (
9     <Switch>
10       <Route path={match.url} exact><EntityIndex globalEntityRules={globalEntityRules} entityRules={entityRules} /></Route>
11       <Route path={match.url + "/:id"}><EntityEditor globalEntityRules={globalEntityRules} entityRules={entityRules} entityName={match.params.entity} /></Route>
12     </Switch>
13   )
14 }

16 export default EntityHandler

```

Listing B.134 index.js

```

1 import { faPen, faPlus, faTrash } from '@fortawesome/free-solid-svg-icons';
2 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
3 import { observer } from 'mobx-react';
4 import React, { useCallback, useEffect, useState } from 'react';
5 import { Link, useRouteMatch } from 'react-router-dom';
6 import { Button, Col, Container, Row, Table } from 'reactstrap';
7 import { useEntityStore } from './components/use-store';
8 import EntityDeleteor from './delete';

10 function EntityIndex({ entityRules }) {
11   const match = useRouteMatch("admin/manage/:entity");
12   const entityStore = useEntityStore();
13   const [itemToDelete, setItemToDelete] = useState(undefined)

15   const refetchItemData = useCallback(
16     () => {
17       entityStore.fetch(entityRules.apiPath);
18     },
19     [entityStore, entityRules],
20   )

22   function handleDeleteRequest(item) {
23     setItemToDelete(item);
24   }

26   function handleDeleteSuccess() {
27     refetchItemData();
28     setItemToDelete(undefined);
29   }

31   //refetch items
32   useEffect(() => {
33     entityStore.selectedEntity = entityRules.apiPath;
34     refetchItemData();
35     return () => {};
36   }, [entityStore, entityRules, refetchItemData])

39   let tableHeader = entityRules.list_display.map((e) => (e.includes(':') ? e.split(":") : e));

41   return (
42     <div className="my-5">
43       <Container>
44         <Row>
45           <Col xs={12} md={8}>
46             <h2 className="text-monospace"><span className="text-muted">Manage</span>{entityRules.name.replace(/(\W|_)/g, " ")}}<span className="text-info">index</span>(</h2>
47           </Col>
48           <Col xs={12} md={4}>
49             <div className="text-right">
50               <Button className="mx-2" color="success" tag={Link} to={match.url + "/new"}><FontAwesomeIcon icon={faPlus} /> Create New</Button>
51             </div>
52           </Col>
53         </Row>
54         <Row>
55           <Col xs={12}>
56             <div className="my-4">
57               <Table responsive striped>
58                 <thead>
59                   <tr>
60                     {tableHeader.map(el => {
61                       let headerKey = Array.isArray(el) ? el.join(":") : el;
62                       return <th key={el}>
63                         {entityRules.fields[headerKey]?.name || headerKey}
64                       </th>
65                     })}
66                   <th></th>
67                 </tr>
68               </thead>
69               <tbody>
70                 {entityStore.items.map(data => <tr key={"data-" + data._id}>
71                   {tableHeader.map(el => {
72                     let display = undefined;
73                   })
74                 })
75               </tr>}
76             </tbody>
77           </div>
78         </Col>
79       </Row>
80     </Container>
81   )
82 }

```

```

73     if (Array.isArray(el)) {
74         display = data[el[0]]?.hasOwnProperty(el[1]) ? data[el[0]][el[1]] : data[el
75             [0]];
76     } else {
77         display = (typeof (data[el]) === "object" ? JSON.stringify(data[el]) : data[
78             el]);
79     }
80     return <td key={el + "-" + data._id}>
81         {display}
82     </td>
83   </td>
84   <div className="text-right">
85     <Button color="secondary" className="mx-2" tag={Link} to={match.url + "/" +
86       data._id}>
87       <FontAwesomeIcon icon={faPen} />
88     </Button>
89     <Button color="danger" className="mx-2" onClick={() => handleDeleteRequest(
90       data)}><FontAwesomeIcon icon={faTrash} /></Button>
91   </div>
92 </td>
93 </Table>
94 </div>
95 </Row>
96 </Container>
97 <EntityDeletor
98   entityRules={entityRules}
99   onDeleteCanceled={() => setItemToDelete(undefined)}
100  onDeleteSucceed={handleDeleteSuccess}
101  item={itemToDelete}>
102 </>
103 </div>
104 )
105 }
106 <export default observer(EntityIndex)>
```

Listing B.135 Manage.js

```

1 import React from 'react'
2 import { Route, Switch } from "react-router-dom"
3 import { entityRules } from './rules';
4 import AdminNavbar from '~/components/admin/navbar/AdminNavbar';
5 import EntityHandler from './base/handler';

8 function Manage({ match }) {
9   return (
10     <>
11       <AdminNavbar />
12       <Switch>
13         <Route path={match.path + "/lectures"}><EntityHandler globalEntityRules={entityRules} entityRules={
14           entityRules.lecture} /></Route>
15         <Route path={match.path + "/lectureperiods"}><EntityHandler globalEntityRules={entityRules} entityRules={
16           entityRules.lectureperiod} /></Route>
17         <Route path={match.path + "/locations"}><EntityHandler globalEntityRules={entityRules} entityRules={
18           entityRules.locations} /></Route>
19         <Route path={match.path + "/admins"}><EntityHandler globalEntityRules={entityRules} entityRules={
20           entityRules.admins} /></Route>
21         <Route path={match.path + "/iplogins"}><EntityHandler globalEntityRules={entityRules} entityRules={

22           entityRules.iplogins} /></Route>
23         <Route path={match.path + "/acls"}><EntityHandler globalEntityRules={entityRules} entityRules={

24           entityRules.acls} /></Route>
25         <Route path={match.path + "/computers"}><EntityHandler globalEntityRules={entityRules} entityRules={

26           entityRules.computers} /></Route>
27         <Route path={match.path + "/exams"}><EntityHandler globalEntityRules={entityRules} entityRules={

28           entityRules.exams} /></Route>
29       </Switch>
30     </>
31   )
32 }
33 <export default Manage>
```

Listing B.136 rules.js

```

1 export let entityRules = {
2   exams: {
3     name: "Exam",
4     apiPath: "manage/exam",
5     fields: {
6       _id: { type: "number", name: "id", note: "", allow: { create: false, update: false } },
7       lecture: { type: "link", name: "Lecture", table: "lecture", link_label: "name", note: "", allow: { create: true,
8         update: true } },
9       lecture_period: { type: "link", name: "Lecture Period", table: "lectureperiods", link_label: "period_code", note:
10         "", allow: { create: true, update: true } },
11       time_start: { type: "text", name: "Exam Start", note: "", allow: { create: true, update: true } },
12       time_duration: { type: "number", name: "Exam Duration", note: "", allow: { create: true, update: true } },
13       time_opened: { type: "text", name: "Timer Open", note: "", allow: { create: false, update: false } },
14       time_ended: { type: "text", name: "Timer End", note: "", allow: { create: false, update: false } },
15       uts: { type: "text", name: "UTS", note: "", allow: { create: true, update: true } },
16       shift: { type: "text", name: "SHIFT", note: "", allow: { create: true, update: true } },
```

```

15     },
16     list_display: ["_id", "lecture:name", "lecture_period:period_code", "time_start", "uts", "shift", "updated_on"]
17   },
18   lectureperiods: {
19     name: "Lecture Periodes",
20     apiPath: "manage/lectureperiod",
21     fields: {
22       _id: { type: "number", name: "id", note: "", allow: { create: false, update: false } },
23       period_code: { type: "text", name: "Period Code", allow: { create: true, update: true } }
24     },
25     list_display: ["_id", "name", "lecture_code", "updated_on"]
26   },
27   lecture: {
28     name: "Lecture",
29     apiPath: "manage/lecture",
30     fields: {
31       _id: { type: "number", name: "id", note: "", allow: { create: false, update: false } },
32       name: { type: "text", name: "Name", note: "", allow: { create: true, update: true } },
33       lecture_code: { type: "text", name: "Lecture Code", allow: { create: true, update: true } }
34     },
35     list_display: ["_id", "name", "lecture_code", "updated_on"]
36   },
37   iplogins: {
38     name: "IP Login",
39     apiPath: "manage/iplogin",
40     fields: {
41       _id: { type: "number", name: "id", note: "", allow: { create: false, update: false } },
42       ip: { type: "text", name: "IP", allow: { create: true, update: true } },
43       user: { type: "link", name: "User", table: "admins", link_label: "username", allow: { create: true, update: true } },
44       notes: { type: "text", name: "Notes", allow: { create: true, update: true } },
45       locations: { type: "links", name: "Linked Location(s)", table: "locations", link_label: "room_name", allow: { create: true, update: true } },
46     },
47     list_display: ["_id", "ip", "user:username", "notes"]
48   },
49   lectureperiod: {
50     name: "Lecture Period",
51     apiPath: "manage/lectureperiod",
52     fields: {
53       _id: { type: "number", name: "id", note: "", allow: { create: false, update: false } },
54       period_code: { type: "text", allow: { create: true, update: true } },
55     },
56     list_display: ["_id", "period_code", "updated_on"]
57   },
58   computers: {
59     name: "Computers",
60     apiPath: "manage/computer",
61     fields: {
62       _id: { type: "number", name: "id", note: "", allow: { create: false, update: false } },
63       name: { type: "text", allow: { create: true, update: true } },
64       ip: { type: "text", allow: { create: true, update: true } },
65       reverse_dns: { type: "text", allow: { create: true, update: true } },
66       d_pos: { type: "json", allow: { create: true, update: true } },
67       location: { type: "link", table: "locations", link_label: "room_name", allow: { create: true, update: true } },
68     },
69     list_display: ["_id", "name", "ip", "location:room_name", "updated_on"]
70   },
71   locations: {
72     name: "Location",
73     apiPath: "manage/location",
74     fields: {
75       _id: { type: "number", name: "id", note: "", allow: { create: false, update: false } },
76       room_name: { type: "text", allow: { create: true, update: true } },
77       name_alias: { type: "text", allow: { create: true, update: true } },
78     },
79     list_display: ["_id", "room_name", "name_alias", "updated_on"]
80   },
81   acls: {
82     name: "Access Control List",
83     apiPath: "manage/acl",
84     fields: {
85       _id: { type: "number", name: "id", note: "", allow: { create: false, update: false } },
86       name: { type: "text", allow: { create: true, update: true } },
87     },
88     list_display: ["_id", "name"]
89   },
90   admins: {
91     name: "Administrator",
92     apiPath: "manage/user",
93     fields: {
94       _id: { type: "number", name: "id", note: "", allow: { create: false, update: false } },
95       username: { type: "text", allow: { create: true, update: true } },
96       password: { type: "password", allow: { create: true, update: true } },
97       email: { type: "text", allow: { create: true, update: true } },
98       acl: { type: "link", table: "acls", link_label: "name", allow: { create: true, update: true } },
99     },
100    list_display: ["_id", "username", "email", "acl:name", "updated_on"],
101  },
102}

```

Listing B.137 TimerItem.js

```

1 import { faCalendarPlus, faHistory, faPlay, faStop } from '@fortawesome/free-solid-svg-icons'
2 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
3 import PropTypes from 'prop-types'
4 import React, { Component } from 'react'

```

```

5 import Countdown from 'react-countdown'
6 import { Badge, Button, ButtonGroup, Card, CardBody, CardHeader, UncontrolledTooltip } from 'reactstrap'
7
8 export class TimerItem extends Component {
9     static propTypes = {
10         multipleMode: PropTypes.bool,
11         exam: PropTypes.object.isRequired,
12         onPlayClicked: PropTypes.func,
13         onStopClicked: PropTypes.func,
14         onResetClicked: PropTypes.func,
15         onOvertimeClicked: PropTypes.func,
16     }
17
18     props = {
19         multipleMode: false,
20         onPlayClicked: () => { },
21         onStopClicked: () => { },
22         onResetClicked: () => { },
23         onOvertimeClicked: () => { },
24     }
25
26
27     render() {
28         const { exam, onPlayClicked, onStopClicked, onResetClicked, onOvertimeClicked } = this.props;
29         let lectureCode = (<b>{exam?.lecture?.lecture_code}</b>);
30         if (exam?.shift) {
31             lectureCode = <>{lectureCode} &nbsp; SHIFT-{exam.shift}</>;
32         }
33
34         return (
35             <Card>
36                 <CardHeader className={"h4 text-light bg-" + (exam?.uts ? "info" : "success")}>
37                     <div className="d-flex">
38                         <div className="d-flex flex-grow-1">
39                             <Badge color="light">{exam?.uts ? "UTS" : "UAS"}</Badge> &nbsp; {lectureCode}
40                         </div>
41                         <div className="d-flex font-weight-bold">
42                             {/* TODO: Standarisasi ke detik */}
43                             {exam.time_duration / 60} Menit
44                         </div>
45                     </div>
46                 </CardHeader>
47                 <CardBody>
48                     <h2 className="display-4 text-center">{exam.lecture?.name}</h2>
49                     <div className="display-2 font-weight-bold text-center d-flex align-items-center justify-content-center">
50                         <Countdown date={Date.now() + (exam.time_left * 1000)} className="d-flex mx-3" autoStart={!exam?.time_opened} />
51                         <Button color="warning" id={"overtimeBtn-" + exam._id} onClick={(event) => onOvertimeClicked(event, exam)}><FontAwesomeIcon icon={faCalendarPlus} className="d-flex" /></Button>
52                         <UncontrolledTooltip placement="right" target={"overtimeBtn-" + exam._id}>Tambah Overtime/Waktu</UncontrolledTooltip>
53                     </div>
54                     <div className="mt-4">
55                         <div>
56                             <ButtonGroup className="mx-3 my-2">
57                                 <Button onClick={(event) => onPlayClicked(event, exam)} color="success" disabled={!exam?.time_opened}>
58                                     <FontAwesomeIcon icon={faPlay} />
59                                 </Button>
60                                 <Button onClick={(event) => onStopClicked(event, exam)} color="danger" disabled={!exam?.time_opened}>
61                                     <FontAwesomeIcon icon={faStop} />
62                                 </Button>
63                                 <Button onClick={(event) => onResetClicked(event, exam)} color="secondary" disabled={!exam?.time_opened}>
64                                     <FontAwesomeIcon icon={faHistory} />
65                                 </Button>
66                             </ButtonGroup>
67                         </div>
68                     </div>
69                 </CardBody>
70             </Card>
71         )
72     }
73 }
74
75 export default TimerItem

```

Listing B.138 index.js

```

1 import React, { Component } from 'react';
2 import { Redirect, Route, Switch, withRouter } from 'react-router-dom';
3 import TimerList from './timer-list';
4 import TimerSpecific from './timer-specific';
5
6 export class ScreenIndex extends Component {
7
8     _componentDidCatch(error, errorInfo) {
9         // TODO: do something when error occurred.
10    }
11
12    render() {
13        const { match } = this.props;
14        return (
15            <Switch>
16                <Route exact path={match.path} component={TimerList} />
17                <Route path={match.path + "/:id"} component={TimerSpecific} />

```

```

18         <Redirect push to={match.path + "/" } />
19     )
20   }
21 }
22 }

24 export default withRouter(ScreenIndex);

```

Listing B.139 timer-list.js

```

1 import classnames from "classnames";
2 import { parse } from "date-fns";
3 import { inject, observer } from 'mobx-react';
4 import React, { Component } from 'react';
5 import { Link } from "react-router-dom";
6 import { Button, Col, Container, Form, Input, Label, Modal, ModalBody, ModalFooter, ModalHeader, Nav, NavItem, NavLink, Row, Spinner, TabContent, TabPane } from 'reactstrap';
7 import { axios } from "~/apicall";
8 import ComputersContainer from './components/computers/computer-container';
9 import TimerItem from './components/TimerItem';

10 export class TimerList extends Component {

11   state = {
12     activeTab: 'seatmap',
13     overtimeTarget: undefined,
14     overtimeAmount: undefined
15   }
16
17   updateTimer = undefined;

18   componentWillMount() {
19     if (this.updateTimer) {
20       clearInterval(this.updateTimer);
21     }
22   }

23   componentDidMount() {
24     const { adminStore } = this.props;
25     console.log(adminStore._exams);
26     if (adminStore.isLoading) {
27       adminStore.examFetchScreenMode().then(() => {
28         adminStore.armedExam.forEach(exam => {
29           if (exam.time_left > 0) {
30             this.setState({ activeTab: "timer" });
31           }
32         });
33       });
34     }
35     adminStore.fetchProfile();
36   }

37   // Setup timer.
38   if (this.updateTimer) {
39     clearInterval(this.updateTimer);
40   }
41   this.updateTimer = setInterval(() => {
42     adminStore.examFetchScreenMode(false);
43   }, 10000);
44 }

45   handleTimerPlayClick(exam) {
46     const { adminStore } = this.props;
47     axios.post("manage/exam/" + exam._id + "/start").then(() => {
48       adminStore.examFetchScreenMode(exam._id);
49     })
50   }

51   handleTimerStopClick(exam) {
52     const { adminStore } = this.props;
53     axios.post("manage/exam/" + exam._id + "/close").then(() => {
54       adminStore.examFetchScreenMode(exam._id);
55     })
56   }

57   handleTimerResetClick(exam) {
58     const { adminStore } = this.props;
59     axios.delete("manage/exam/" + exam._id + "/start").then(() => {
60       adminStore.examFetchScreenMode(exam._id);
61     })
62   }

63   handleTimerOvertimeClick(exam) {
64     this.setState({ overtimeTarget: exam });
65   }

66   handleTimerOvertimeAction(e) {
67     const { adminStore } = this.props;
68     e.preventDefault();
69     const { overtimeAmount, overtimeTarget } = this.state;
70     let sourceTime = parse(overtimeTarget.time_ended, "yyyy-MM-dd HH:mm:ss", new Date());
71
72     // TODO: check overtimeAmount.
73     sourceTime.setTime(sourceTime.getTime() + overtimeAmount * 60000);
74
75     console.log("updated time: ", sourceTime);
76     axios.put("manage/exam/" + overtimeTarget._id, {
77       time_ended: sourceTime.toISOString()
78     })
79   }

```

```

87         ...overtimeTarget,
88         _id: undefined,
89         answer_slot: undefined,
90         participants: undefined,
91         exam_report: undefined,
92         lecture: undefined,
93         lecture_period: undefined,
94         deleted_on: undefined,
95         updated_on: undefined,
96         created_on: undefined,
97         time_left: undefined,
98         time_ended: sourceTime.getTime()/1000
99     }).then(() => {
100     adminStore.examFetchScreenMode(false);
101     this.setState({ overtimeAmount: 0, overtimeTarget: undefined });
102 })
103 }

106 render() {
107     const { adminStore = {} } = this.props;
108     const { armedExam } = adminStore;
109     const { activeTab, overtimeTarget, overtimeAmount } = this.state;
110
111     if (adminStore.isLoading) {
112         return this.renderLoaderScreen();
113     }
114
115     if (adminStore.loginType !== "ip") {
116         return this.renderErrorScreen("This login method doesn't support this function, please login with IPLogin to get
117         started.", (
118             <Button color="primary" tag={Link} to="/admin/account/login">Logout and Login</Button>
119         ))
120     }
121
122     if (adminStore.boundedLocations?.length === 0) {
123         return this.renderErrorScreen("This IP is not connected with any rooms. Please contact administrator.")
124     }
125
126     return (
127         <Container>
128             <Row className="h-100vh align-items-center justify-content-center text-center">
129                 <Col>
130                     <Nav tabs>
131                         <NavItem>
132                             <NavLink
133                                 className={classnames({ active: activeTab === 'seatmap' })}
134                                 onClick={() => this.setState({ activeTab: 'seatmap' })}
135                             >
136                             SeatMap
137                         </NavLink>
138                         <NavItem>
139                             <NavLink
140                                 className={classnames({ active: activeTab === 'timer' })}
141                                 onClick={() => this.setState({ activeTab: 'timer' })}
142                             >
143                             Timer
144                         </NavLink>
145                     </NavItem>
146                 </Nav>
147                 <TabContent activeTab={activeTab}>
148                     <TabPane tabId="seatmap">
149                         {adminStore.boundedLocations.map((loca) => <div key={loca._id}>
150                             <h3 className="my-3 bg-success text-light">{loca?.room_name}</h3>
151                             <ComputersContainer computers={loca.computers} editable={false} />
152                         </div>)}
153                     </TabPane>
154                     <TabPane tabId="timer">
155                         {armedExam.map((waiter, i) => <TimerItem
156                             key={i}
157                             exam={waiter}
158                             onPlayClicked={() => this.handleTimerPlayClick(waiter)}
159                             onStopClicked={() => this.handleTimerStopClick(waiter)}
160                             onResetClicked={() => this.handleTimerResetClick(waiter)}
161                             onOvertimeClicked={() => this.handleTimerOvertimeClick(waiter)}
162                         />)
163                     </TabPane>
164                 </TabContent>
165             </Col>
166         </Row>
167
168         <Modal isOpen={!overtimeTarget}>
169             <ModalHeader>Tambah Waktu</ModalHeader>
170             <Form onSubmit={(e) => this.handleTimerOvertimeAction(e)}>
171                 <ModalBody>
172                     <p>
173                         Tambah waktu untuk ujian:
174                         <b> {overtimeTarget?.lecture?.lecture_code} {overtimeTarget?.shift ? (" SHIFT-" + overtimeTarget?
175                         .shift) : ""}</b>
176                     </p>
177                     <Label>
178                         Jumlah menit yang ingin ditambahkan:
179                     </Label>
180                     <Input name="overtimeMin" id="overtimeMin" onChange={(e) => this.setState({ overtimeAmount:
181                         e.currentTarget.value })} value={overtimeAmount} />
182                 </ModalBody>
183             <ModalFooter>

```

```

183             <Button onClick={() => this.setState({ overtimeTarget: undefined })} type="button" color="secondary">Batal</Button>
184             <Button type="submit" color="warning">Tambah</Button>
185         </ModalFooter>
186     </Modal>
187 </Container>
188 )
189 }
190 }

192 renderLoaderScreen() {
193     return (<Container>
194         <Row className="h-100vh align-items-center justify-content-center text-center">
195             <Col>
196                 <Spinner />
197                 <p className="lead">Mengontak Server...</p>
198             </Col>
199         </Row>
200     </Container>
201 }

203 renderErrorScreen(message, children) {
204     return (<Container>
205         <Row className="h-100vh align-items-center justify-content-center text-center">
206             <Col>
207                 <h3>Whoops,</h3>
208                 <p className="lead">{message}</p>
209                 <div>{children}</div>
210             </Col>
211         </Row>
212     </Container>
213 }
214 }

216 export default (inject("adminStore")(observer(TimerList)))

```

Listing B.140 timer-specific.js

```

1 import React, { useEffect, useState } from 'react';
2 import { Col, Container, Row, Spinner } from 'reactstrap';
3 import { axios } from '~/apicall';
4 import TimerItem from './components/TimerItem';

7 export function TimerSpecLoader() {
8     return (
9         <Container>
10            <Row className="h-100vh align-items-center justify-content-center text-center">
11                <Col>
12                    <Spinner />
13                    <p className="lead">Mengontak Server...</p>
14                </Col>
15            </Row>
16        </Container>
17    )
18 }

21 function TimerSpecific({ match }) {
22     const [exam, setExam] = useState(undefined)

24     useEffect(() => {
25         // load exam info
26         axios.get(`manage/exam/${match.params.id}`).then((response) => {
27             setExam(response.data.data);
28         });
29         return () => { }
30     }, [match])

32     if (!exam) {
33         return (<TimerSpecLoader />)
34     }

36     return (
37         <Container>
38             <Row className="h-100vh align-items-center justify-content-center text-center">
39                 <Col>
40                     <TimerItem exam={exam} />
41                 </Col>
42             </Row>
43         </Container>
44     )
45 }

47 export default TimerSpecific

```

Listing B.141 navbar.js

```

1 import React, { useState } from 'react'
2 import { Link } from 'react-router-dom'
3 import { Collapse, Container, Nav, Navbar, NavbarBrand, NavbarToggler, NavItem, NavLink } from 'reactstrap'

5 function AutonomusNavbar({ defaultHome = "#" }) {
6     const [openMenuToogle, setopenMenuToogle] = useState(false)
7     return (

```

```

8      <>
9          <Navbar color="dark" dark expand="md">
10             <Container>
11                 <NavbarBrand href={defaultHome} to={defaultHome} tag={Link}>Oxam <small>(autnomus-mode)</small></
12                     NavbarBrand>
13                     <NavbarToggler onClick={() => setopenMenuToogle(!openMenuToogle)} />
14                     <Collapse isOpen={openMenuToogle} navbar>
15                         <Nav className="ml-auto" navbar>
16                             <NavItem>
17                                 <NavLink to="/admin" tag={Link}>Admin</NavLink>
18                             </NavItem>
19                         </Nav>
20                     </Collapse>
21                 </Container>
22             </Navbar>
23         )>
24     )
25
26 export default AutonomusNavbar

```

Listing B.142 exam-info.js

```

1 import React from 'react'
2 import { Col, Row, Table } from 'reactstrap'

4 function ExamExtractorExamInfo({ exam }) {
5     return (
6         <>
7             <Row>
8                 <Col>
9                     <Table bordered striped>
10                        <tbody>
11                            <tr>
12                                <th>Mata Kuliah</th>
13                                <td>{exam?.lecture?.name} ({exam?.lecture?.lecture_code})</td>
14                            </tr>
15                            <tr>
16                                <th>Tahun Ajaran</th>
17                                <td>{exam?.lecture_period?.period_code}</td>
18                            </tr>
19                            <tr>
20                                <th>Mulai - Berakhir</th>
21                                <td>{exam?.time_opened} - {exam?.time_ended}</td>
22                            </tr>
23                            <tr>
24                                <th>Waktu Mulai Pengumpulan</th>
25                                <td>{exam?.time_start || "Belum di buka"}</td>
26                            </tr>
27                            <tr>
28                                <th>Durasi</th>
29                                <td>{exam?.time_duration / 60} Menit</td>
30                            </tr>
31                            <tr>
32                                <th>Uts/Uas</th>
33                                <td>{exam?.uts ? "UTS" : "UAS"}</td>
34                            </tr>
35                        </tbody>
36                    </Table>
37                </Col>
38                <Col>
39                    <Table bordered striped>
40                        <tbody>
41                            <tr>
42                                <th>Jumlah Peserta</th>
43                                <td>{(exam?.participants || []).length} Orang</td>
44                            </tr>
45                            <tr>
46                                <th>Dibuat pada</th>
47                                <td>{exam?.created_on}</td>
48                            </tr>
49                            <tr>
50                                <th>Terakhir diperbaharui</th>
51                                <td>{exam?.updated_on}</td>
52                            </tr>
53                        </tbody>
54                    </Table>
55                </Col>
56            </Row>
57        )>
58    )
59
60 export default ExamExtractorExamInfo

```

Listing B.143 index.js

```

1 import { faFileDownload } from '@fortawesome/free-solid-svg-icons';
2 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
3 import React, { useEffect, useState } from 'react';
4 import { Else, If, Then, When } from 'react-if';
5 import { Alert, Button, Col, Container, Row, Spinner } from 'reactstrap';
6 import { axios } from '~/apicall';
7 import Buildinfo from '~/components/buildinfo/Buildinfo';
8 import AutonomusNavbar from '../components/navbar';

```

```

9 import ExamExtractorExamInfo from './exam-info';
10
11 function ExamExtractIndex({ match }) {
12   const [isLoading, setIsLoading] = useState(true)
13   const [examInfo, setExamInfo] = useState(undefined)
14   const [hasError, setHasError] = useState(undefined)
15
16   useEffect(() => {
17     axios.post("autonomus/examdownload", { token: match.params.token }).then((resp) => {
18       if (resp.data.status) {
19         setExamInfo(resp.data.data);
20       } else {
21         setHasError({ title: resp.error.title, description: resp.error.description })
22       }
23     }).catch(error => {
24       if (error.response === undefined) {
25         setHasError({ title: "Network Error", description: "There's error with the network. Please recheck your connection." })
26       return;
27     } else {
28       let errorData = error.response.data;
29       setHasError({ title: errorData.error.title, description: errorData.error.description })
30       return;
31     }
32   }).finally(() => {
33     setIsLoading(false);
34   });
35
36   return () => { }
37 }, [match])
38
39 return (
40   <>
41     <AutonomusNavbar />
42     <Container className="my-5" style={{ minHeight: "70vh" }}>
43       <Row>
44         <Col>
45           <h3>Exam Answer Downloader</h3>
46         </Col>
47       </Row>
48       <When condition={isLoading}>
49         <Alert color="dark" className="text-center py-4">
50           <Spinner />
51           <p className="pt-4">Mengontak Server...</p>
52         </Alert>
53       </When>
54       <If condition={!hasError}>
55         <Then>
56           <Row>
57             <Col xs={12} md={8}>
58               <Alert color="danger">
59                 <b>{hasError?.title}</b> <br />
60                 {hasError?.description}
61               </Alert>
62             </Col>
63           </Row>
64         </Then>
65       <Else>
66         <ExamExtractorExamInfo exam={examInfo?.exam || {}} />
67
68         <div className="my-4">
69           <Button href={examInfo?.downloadToken} color="success" size="lg" target="_blank">
70             <FontAwesomeIcon icon={faFileDownload} /> Unduh Jawaban
71           </Button>
72         </div>
73       </Else>
74     </If>
75   </Container>
76   <Container className="my-5">
77     <footer className="text-muted">
78       <Row>
79         <Col>
80           <Buildinfo />
81         </Col>
82       </Row>
83     </footer>
84   </Container>
85 </>
86 )
87 }
88
89 export default ExamExtractIndex

```

Listing B.144 index.js

```

1 import React from 'react'
2 import { Route, Switch } from 'react-router-dom'
3 import ExamExtractIndex from './exam-extract'
4
5 function AutonomusIndex({ match }) {
6   return (
7     <Switch>
8       <Route path={match.path + "/exam-extract/:token"} component={ExamExtractIndex} />
9     </Switch>
10   )
11 }

```

```
| 13 export default AutonomusIndex
```

Listing B.145 Exam.js

```

1 import { inject, observer } from 'mobx-react';
2 import React, { Component } from 'react';
3 import { Route, Switch, withRouter } from "react-router-dom";
4 import Col from "reactstrap/lib/Col";
5 import Container from "reactstrap/lib/Container";
6 import Row from "reactstrap/lib/Row";
7 import Spinner from "reactstrap/lib/Spinner";
8 import ExamIndex from './index/Index';

11 export class Exam extends Component {
13   refreshTime = 120;
14   refreshTimer = undefined;

16   componentWillUnmount() {
17     clearInterval(this.refreshTimer)
18   }

20   async componentDidMount() {
21     const { examStore } = this.props;

23     // Forcing the examination to be fetched first before continuing to arm the
24     // refresh timer.
25     if (examStore.isFetchingExam) {
26       await examStore.fetchExamInfo()
27     }

29     let refreshTime = this.determineRefreshTime(examStore)
30     this.armRefreshTimer(examStore, refreshTime);
31   }

33   armRefreshTimer(examStore, refreshTime) {
34     console.info("determining next tick will be ", refreshTime);
35     this.refreshTimer = setInterval(async () => {
36       await examStore.fetchExamInfo(false);

38       // Check refresh time change. If it DOES change, clear the interval and
39       // set a new one.
40       // this might be buggy tough. Not recommended, but its currently the only
41       // solution that works.
42       let detectRefreshTimeChange = this.determineRefreshTime(examStore);

45       if (refreshTime !== detectRefreshTimeChange) {
46         clearInterval(this.refreshTimer);
47         this.armRefreshTimer(examStore, detectRefreshTimeChange);
48       }
49     }, refreshTime * 1000)
50   }

52   determineRefreshTime(examStore) {
53     // For upcoming exam, we'll update every 10 seconds.
54     if (examStore.participant?.is_upcoming) {
55       return 10;
56     }

58     // for current active exam, we'll update every 30 seconds
59     if (examStore.exam?.time_left <= 120) {
60       return 30;
61     }

63     // By default, refresh for every 2 minutes
64     return 120;
65   }

67   render() {
68     const { examStore, match } = this.props;

70     //show loader when exam is loading or in fetching situation.
71     if (examStore.isFetchingExam) {
72       document.title = "Memuat info ujian...";
73       return (
74         <Container>
75           <Row className="h-100vh align-items-center justify-content-center text-center">
76             <Col>
77               <Spinner />
78               <p className="lead">Mengontak Server...</p>
79             </Col>
80           </Row>
81         </Container>
82       )
83     }

86     return (
87       <Switch>
88         <Route exact path={match.path} component={ExamIndex} />
89       </Switch>
90     )
91   }
92 }
```

```
| 94 export default withRouter(inject("examStore")(observer(Index)))
```

Listing B.146 Index.js

```

1 import { inject, observer } from "mobx-react"
2 import React, { Component } from 'react'
3 import { Else, If, Then } from 'react-if'
4 import Card from "reactstrap/lib/Card"
5 import CardBody from "reactstrap/lib/CardBody"
6 import Col from "reactstrap/lib/Col"
7 import Container from "reactstrap/lib/Container"
8 import Row from "reactstrap/lib/Row"
9 import Buildinfo from "~/components/buildinfo/Buildinfo"
10 import ExamNav from "~/components/exam/navbar/Navbar"
11 import NoExam from './NoExam'
12 import NotifModal from './NotifModal'
13 import Submitter from './Submitter'
14 import Upcoming from './Upcoming'

18 class Index extends Component {
19   state = {
20     notification: undefined
21   }

23   render() {
24     const { examStore } = this.props;
25     const { exam, participant = {} } = examStore;
27
28     const { notification } = this.state;
29
30     let ExamHandler = (!exam?.time_opened) ? Upcoming : Submitter;
31
32     // update document title
33     if (!exam) {
34       document.title = "Tidak ada ujian berlangsung."
35     } else {
36       document.title = [participant.username, "-", exam.lecture?.lecture_code, "Ujian"].join(" ");
37     }
38
39     return (
40       <NotifModal notification={notification} onCloseRequested={() => this.setState({ notification: undefined })} />
41       <div className="bg-light">
42         <Container>
43           <Row className="h-100vh align-items-center justify-content-center">
44             <Col md={8}>
45               <Card className="overflow-hidden">
46
47                 <If condition={!exam}>
48                   <Then>
49                     <NoExam />
50                   </Then>
51                 <Else>
52                   <ExamNav type={!exam?.time_opened ? "upcoming" : "inprogress"} participant={participant}
53                     onNotifyShowRequested={(e) => this.setState({ notification: e })} />
54                     <CardBody>
55                       <ExamHandler participant={participant} />
56                     </CardBody>
57                   </Else>
58                 </If>
59
60                 </Card>
61                 <p className="text-center text-secondary">
62                   <small><Buildinfo /></small>
63                 </p>
64               </Col>
65             </Row>
66           </Container>
67         </div>
68       </>
69     )
70   }

72 export default inject('examStore')(observer(Index));

```

Listing B.147 NoExam.js

```

1 import { FontAwesomeIcon } from "@fortawesome/react-fontawesome";
2 import React from 'react';
3 import CardBody from "reactstrap/lib/CardBody";

6 const displayFaces = [
7   'meh-rolling-eyes',
8   'surprise',
9   'meh',
10  'grin-tears',
11  'dizzy',
12  'lemon'
13]

15 let emotnya = Math.floor(Math.random() * displayFaces.length);

```

```

16 const NoExam = () => {
17   return (
18     <CardBody>
19       <div className="p-4">
20         <FontAwesomeIcon icon={['far', displayFaces[emotnya]]} size="3x" className="mb-5 text-muted" />
21         <h3>Tidak ada ujian yang sedang berjalan.</h3>
22         <p>Mohon kontak administrator, jika anda yakin ini sebuah kesalahan.</p>
23       </div>
24     </CardBody>
25   )
26 }
27 NoExam.propTypes = {
28 }

30 export default NoExam;

```

Listing B.148 NotifModal.js

```

1 import React from 'react';
2 import { Button, Modal, ModalBody, ModalFooter, ModalHeader } from 'reactstrap';
3
4 function NotifModal({ notification, onCloseRequested = () => {} }) {
5   return (
6     <Modal isOpen={!notification} backdrop>
7       <ModalHeader>{(notification || {}).title}</ModalHeader>
8       <ModalBody>
9         <div dangerouslySetInnerHTML={{ __html: (notification || {}).description }} />
10      </ModalBody>
11      <ModalFooter>
12        <Button color="secondary" onClick={onCloseRequested}>Tutup</Button>
13      </ModalFooter>
14    </Modal>
15  )
16 }
18 export default NotifModal

```

Listing B.149 lotter.js

```

1 import React, { useCallback, useState, useEffect } from 'react'
2 import { Progress, Modal, ModalBody, ModalFooter, Button, UncontrolledTooltip } from 'reactstrap'
3 import { If, Then, Else } from 'react-if'
4 import { useDropzone } from 'react-dropzone'
5 import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
6 import { createUseStyles } from 'react-jss'
7 import { axios } from './apicall'
8 import { faCloudDownloadAlt, faSync } from '@fortawesome/free-solid-svg-icons'
9 import fileDownload from 'js-file-download'

11 const useStyles = createUseStyles(({{
12   root: {
13     cursor: "pointer"
14   }
15 }}))

17 const reaction = [
18   "~-~",
19   "( - )~",
20   " ^ ^ #cats",
21   " ^ #bears",
22   " - gib",
23   "( ) #cats",
24   "OwO",
25   "( ) "
26 ]

28 let emotnya = reaction[Math.floor(Math.random() * reaction.length)];

30 function Lotter({ answer_slot }) {
31   const [uploading, setUploading] = useState(0);
32   const [progress, setProgress] = useState(false);
33   const [fetchingSubmission, setFetchingSubmission] = useState(true);
34   const [submission, setSubmission] = useState(null);

36   const [capturedMismatchFile, setCapturedMismatchFile] = useState(null)

38   const [errorModal, setErrorModal] = useState(null)

40   // Real upload callback.
41   const triggerUpload = useCallback((acceptedFile) => {
42     setProgress(true);
43     let data = new FormData()
44     data.append('file', acceptedFile, answer_slot.format)
45     data.append('answer_slot', answer_slot._id)

47     axios.post("exam/submission/submit", data, {
48       onUploadProgress: (progressEvent) => {
49         setUploading(progressEvent.loaded / progressEvent.total);
50       }
51     }).then((resp) => {
52       setProgress(false);
53       setSubmission(resp.data.data);
54       // TODO: Pesan sukses
55     }).catch((resp) => {
56       setProgress(false);

```

```

57     if (resp.response.data && resp.response.data.error) {
58         setErrorModal(<>
59             <h5>{resp.response.data.error.title} [{resp.response.data.error.error_code}]</h5>
60             <p>{resp.response.data.error.description}</p>
61         </>)
62     } else {
63         console.error(resp);
64         console.log(JSON.stringify(resp));
65         setErrorModal(<>
66             <h5>Error happened</h5>
67             <p>{resp.message}</p>
68             <p><small>Because it's a browser(/network) related error, the error has been emitted to the console.</small></p>
69         </>)
70     })
71 }, [answer_slot])
72
73 // drop handler
74 const onDrop = useCallback(acceptedFiles => {
75     if (acceptedFiles[0].name !== answer_slot.format) {
76         setCapturedMismatchFile(acceptedFiles[0]);
77     } else {
78         triggerUpload(acceptedFiles[0]);
79     }
80 }, [answer_slot, triggerUpload])
81 const { getRootProps, getInputProps, isDragActive } = useDropzone({ onDrop })
82 const styles = useStyles();
83
84 // hooks for fetching submission
85 useEffect(() => {
86     setFetchingSubmission(true);
87     axios.get("exam/submission/submit", { params: { answer_slot: answer_slot._id } }).then(data => {
88         setFetchingSubmission(false);
89         setSubmission(data.data.data);
90     }).catch((err) => {
91         setFetchingSubmission(false);
92         setSubmission(null);
93         console.error("ERROR HAPPENED:", err);
94     })
95     return () => {};
96 }, [answer_slot])
97
98 // handler for downloading stuffs.
99 function handleDownload() {
100     axios.get("exam/submission/submit", {
101         params: {
102             answer_slot: answer_slot._id,
103             force_download: true
104         },
105         responseType: 'blob'
106     }).then((resp) => {
107         fileDownload(resp.data, answer_slot.format);
108     }).catch(err => {
109         console.log(err);
110     })
111 }
112
113
114
115 return (
116     <tr>
117         <td className="align-middle">
118             <Button
119                 color={!submission ? "secondary" : "success"}
120                 id={"download-button-" + answer_slot._id}
121                 onClick={handleDownload}
122                 disabled={fetchingSubmission || !submission}
123             >
124                 <FontAwesomeIcon icon={fetchingSubmission ? faSync : faCloudDownloadAlt} />
125             </Button>
126             <UncontrolledTooltip placement="bottom" target={"download-button-" + answer_slot._id}>Unduh Berkas</UncontrolledTooltip>
127         </td>
128         <td className="align-middle">
129             <p className="font-monospace font-weight-bold m-0" + ((!fetchingSubmission && !submission) ? " text-danger" : "")>{answer_slot.format}</p>
130     </td>
131     <If condition={progress}>
132         <Then>
133             <td className="align-middle">
134                 <Progress animated={progress} value={uploading} max={1}>{(uploading * 100 + " %")}</Progress>
135             </td>
136         </Then>
137         <Else>
138             <td {...getRootProps({ className: (styles.root + " text-right align-middle") })}>
139                 <input {...getInputProps({ multiple: false })} />
140                 <If condition={isDragActive}>
141                     <Then>
142                         <span className="text-primary font-weight-bold"><FontAwesomeIcon icon="cloud-upload-alt" /> {emotnya}</span>
143                     </Then>
144                     <Else>
145                         <div>
146                             <FontAwesomeIcon icon="cloud-upload-alt" /> Upload dengan <b>klik di sini</b> atau<br />
147                             drag berkasnya ke sini
148                         </div>
149                     </Else>
150                 </If>
151             </td>
152         </Else>
153     </If>
154 
```

```

151         </Else>
152     </If>
153     <Modal isOpen={!capturedMismatchFile} backdrop>
154         <ModalBody>
155             <p>Anda mengupload file dengan format penamaan yang berbeda.</p>
156             <p>Anda mengupload <code>{(capturedMismatchFile || {}).name}</code>, sedangkan file yang diinginkan
157                 adalah <code>{answer_slot.format}</code></p>
158             <p>
159                 Apakah anda ingin tetap menguploadnya (namanya bakal digantiin sama kita kok), atau batalkan upload?
160             </p>
161         </ModalBody>
162         <ModalFooter>
163             <Button color="secondary" onClick={() => {
164                 triggerUpload(capturedMismatchFile);
165                 setCapturedMismatchFile(null)
166             }}>Ganti Nama dan Upload</Button>
167             <Button color="warning" onClick={() => setCapturedMismatchFile(null)}>Batalkan</Button>
168         </ModalFooter>
169     </Modal>
170     <Modal isOpen={!errorModal} backdrop>
171         <ModalBody>{errorModal}</ModalBody>
172         <ModalFooter><Button color="primary" onClick={() => setErrorModal(null)}>Ok</Button></ModalFooter>
173     </Modal>
174   </tr >
175 }
177 export default Lotter

```

Listing B.150 Submitter.js

```

1 import { observer } from 'mobx-react';
2 import PropTypes from 'prop-types';
3 import React from 'react';
4 import Countdown from 'react-countdown';
5 import { When } from 'react-if';
6 import { Badge } from 'reactstrap';
7 import Col from "reactstrap/lib/Col";
8 import Row from "reactstrap/lib/Row";
9 import Table from "reactstrap/lib/Table";
10 import { useExamStore } from './components/use-store';
11 import Lotter from './submit/lotter';

15 function Submitter({ participant = {} }) {
16     const { exam = {} } = participant;
17
18     const examStore = useExamStore();
19     return (
20         <
21             <Row className="align-items-baseline">
22                 <Col xs={{ size: 4, order: 2 }}>
23                     <p className="lead m-0 text-right"><Countdown date={Date.now() + exam.time_left * 1000} onComplete={() =>
24                         examStore.fetchExamInfo(false)} /></p>
25                     <p className="m-0 text-right">{participant.computer?.name}@{participant.computer?.location?.room_name}</p>
26                 </Col>
27                 <Col xs={{ size: 8, order: 1 }}>
28                     <h4>{exam.uts ? "UTS" : "UAS"} {exam.lecture?.lecture_code}
29                     <When condition={exam.shift !== 0}>
30                         {` ${Badge color={exam.uts ? "info" : "success"}>S-{exam.shift}}</Badge>
31                     </When>
32                     <p>{exam.lecture?.name}</p>
33                 </Col>
34             </Row>
35             <Row>
36                 <Col>
37                     <h5>Pengumpulan:</h5>
38                     <Table striped>
39                         <tbody>
40                             {exam.answer_slot?.map((ans, i) => <Lotter answer_slot={ans} exam={exam} key={i} />)}
41                         </tbody>
42                     </Table>
43                 </Col>
44             </Row>
45         </>
46     )
47 }
48 Submitter.propTypes = {
49     participant: PropTypes.object.isRequired
50 }
53 export default (observer(Submitter));

```

Listing B.151 Upcoming.js

```

1 import { observer } from 'mobx-react';
2 import PropTypes from 'prop-types';
3 import React from 'react';
4 import { When } from 'react-if';
5 import Table from "reactstrap/lib/Table";

```

```

8 const Upcoming = ({ participant }) => {
9   const { exam = {}, computer = {}, username } = participant;
10  return (
11    <>
12      <div className="p-4">
13        <h3>{exam.uts ? "Ujian Tengah Semester" : "Ujian Akhir Semester"}{exam.shift ? " Shift " + exam.shift : ""}</h3>
14        <p className="lead">Hai <b className="text-info">{username}</b>, berikut detil ujianmu sesi ini:</p>
15        <Table>
16          <tbody>
17            <tr>
18              <th>Mata Kuliah</th>
19              <td>
20                {exam.lecture?.lecture_code} <br />
21                <span className="text-muted">{exam.lecture?.name}</span>
22              </td>
23            </tr>
24            <When condition={exam.shift !== null && exam.shift !== 0}>
25              <tr>
26                <th>Shift</th>
27                <td>
28                  {exam.shift}
29                </td>
30              </tr>
31            </When>
32            <tr>
33              <th>Durasi</th>
34              <td>
35                {exam.time_duration / 60} Menit
36              </td>
37            </tr>
38            <tr>
39              <th>Nomor Kursi / Komputer</th>
40              <td>
41                {computer.name}
42              </td>
43            </tr>
44          </tbody>
45        </Table>
46
47        <p>
48          Ujian dapat dimulai saat dosen pengawas telah menekan tombol mulai pada komputer dosen.
49          Kontak dosen pengawas jika terdapat masalah atau kesalahan informasi ujian.
50        </p>
51      </div>
52    </>
53  )
54 }
55 Upcoming.propTypes = {
56   participant: PropTypes.object.isRequired
57 }
58 export default observer(Upcoming);

```

Listing B.152 Index.js

```

1 import React, { useEffect } from 'react';
2 import Spinner from "reactstrap/lib/Spinner";
3 import Container from "reactstrap/lib/Container";
4 import Row from "reactstrap/lib/Row";
5 import Col from "reactstrap/lib/Col";
6
7 import { observer } from 'mobx-react';
8 import { useHistory } from "react-router-dom";
9 import useStores from './components/use-store';
10
11 function Index() {
12   const history = useHistory();
13   const { examStore } = useStores();
14
15   useEffect(() => {
16     examStore.fetchExamInfo().then(() => {
17       history.replace('/exam');
18     }).catch((e) => {
19       history.replace('/admin');
20     });
21
22   return () => { };
23 }, [history, examStore])
24 return (
25   <div className="bg-light">
26     <Container>
27       <Row className="h-100vh align-items-center">
28         <Col className="text-center">
29           <Spinner />
30           <p className="lead">Sedang menghubungi server...</p>
31         </Col>
32       </Row>
33     </Container>
34   </div>
35 )
36
37 export default observer(Index);

```

Listing B.153 adminStore.js

```

1 import { action, computed, decorate, observable } from "mobx";
2 import { axios, clearAuth, setAuth } from "./apicall";

5 class AdminStore {
6     user = {}

8     defaultExam = {
9         "lecture_period": {
10             "period_code": null,
11             "deleted_on": null,
12             "created_on": null,
13             "updated_on": null,
14             "_id": 0
15         },
16         "time_start": null,
17         "time-ended": null,
18         "time_duration": 0,
19         "time_opened": null,
20         "lecture": {
21             "name": null,
22             "lecture_code": null,
23             "deleted_on": null,
24             "created_on": null,
25             "updated_on": null,
26             "_id": 4
27         },
28         "uts": true,
29         "shift": 0,
30         "deleted_on": null,
31         "created_on": null,
32         "updated_on": null,
33         "answer_slot": null,
34         "_id": 10
35     }
36     _exams = {};
37
38     courses = [];
39     ujianType = {};
40
41     selectedExam = null;
42
43     isLoading = true;
44
45     loginType = undefined; // Possible values: undefined || basic || ip
46
47     boundedLocations = [];
48
49     get exams() {
50         return Object.values(this._exams)
51             .sort((a, b) => b?.time_start.localeCompare(a?.time_start));
52     }
53
54     get armedExam() {
55         return Object.values(this._exams)
56             // .filter(ex) => (ex.time_left > 0 || !ex.time_opened))
57             .sort((a, b) => b?.time_start.localeCompare(a?.time_start));
58     }
59
60     get exam() {
61         if (this.selectedExam == null) {
62             return this.defaultExam;
63         } else {
64             return { ...this.defaultExam, ...this._exams[this.selectedExam] };
65         }
66     }
67
68     examFetch(id) {
69         this.isLoading = true;
70         if (id) {
71             return axios.get("manage/exam/" + id).then(resp => {
72                 this._exams[id] = resp.data.data;
73                 this.isLoading = false;
74                 return Promise.resolve(resp);
75             })
76         }
77         return axios.get("manage/exam").then(resp => {
78             let exam = {};
79             resp.data.data.forEach(element => {
80                 exam[element._id] = element;
81             });
82             this._exams = exam;
83             this.isLoading = false;
84             return Promise.resolve(resp);
85         })
86     }
87
88     examFetchScreenMode(shouldShowLoad = true) {
89         this.isLoading = true && shouldShowLoad;
90         return axios.get("manage/exam?screenmode=").then(resp => {
91             let exam = {};
92             resp.data.data.forEach(element => {
93                 exam[element._id] = element;
94             });
95             this._exams = exam;
96             this.isLoading = false;
97             return Promise.resolve(resp);
98         })
99     }

```

```

100     }
101
102     examDelete(id) {
103         this.isLoading = true;
104         return axios.delete("manage/exam/" + id).then(resp => {
105             delete this._exams[id];
106             this.isLoading = false;
107             return Promise.resolve(resp);
108         })
109     }
110
111     fetchParticipantFromExam(id) {
112         return axios.get(`manage/exam/${id}/participants`);
113     }
114
115     fetchProfile() {
116         return axios.get("system/user/me").then(e => {
117             this.user = e.data.data.profile;
118             this.loginType = "basic";
119             this.boundedLocations = [];
120             if (e.data.data.locations) {
121                 this.loginType = "ip";
122                 this.boundedLocations = e.data.data.locations;
123             }
124             return Promise.resolve(e);
125         })
126     }
127
128     tryLogin(username, password) {
129         return axios.post("system/auth/login", {
130             username: username,
131             password: password
132         }).then(e => {
133             if (e.status === 200) {
134                 this.loginType = "basic";
135                 this.boundedLocations = [];
136                 setAuth(e.data.data.id_token);
137             }
138             return Promise.resolve(e);
139         });
140     }
141
142     tryIPLogin() {
143         return axios.post("system/auth/iplogin").then(e => {
144             if (e.status === 200) {
145                 this.loginType = "ip";
146                 this.boundedLocations = e.data.locations;
147                 setAuth(e.data.data.id_token);
148             }
149             return Promise.resolve(e);
150         });
151     }
152
153     userLogout() {
154         clearAuth();
155         return Promise.resolve();
156     }
157 }
158
159 decorate(AdminStore, {
160     user: observable,
161     _exams: observable,
162     selectedExam: observable,
163     loginType: observable,
164     boundedLocations: observable,
165     exam: computed,
166     exams: computed,
167     armedExam: computed,
168     examFetch: action,
169     examDelete: action,
170     tryLogin: action,
171     tryIPLogin: action,
172     isLoading: observable
173 })
174
175
176 export default AdminStore;

```

Listing B.154 entityStore.js

```

1 import { decorate, observable, action, computed, } from "mobx";
2
3 import { axios } from "~/apicall";
4
5 class EntityStore {
6
7     defaultEntityValues = {}
8     selectedEntity = null
9     selectedEntityId = null
10
11     _items = {}
12
13     isLoading = true
14     error = {}
15
16     /**
17      * Update stuffs

```

```

18     * @param {string} url
19     */
20    fetch(url, entityName) {
21      let selectedEntity = entityName || this.selectedEntity;
22      this.isLoading = true;
23      return axios.get(url).then(response => {
24        if (response.data.status) {
25          let item = {};
26          response.data.data.forEach(element => {
27            item[element._id] = element
28          });
29          this._items[selectedEntity] = item;
30        } else {
31          throw new Error(response.data.error);
32        }
33        this.isLoading = false;
34        return Promise.resolve()
35      }).catch((err) => {
36        this.isLoading = false;
37        return Promise.reject(err);
38      })
39    }
40
41    createItem(url, values, entityName) {
42      let selectedEntity = entityName || this.selectedEntity;
43      this.isLoading = true;
44      return axios.post(url, values).then(response => {
45        if (response.data.status) {
46          if (!this._items[selectedEntity]) {
47            this._items[selectedEntity] = {}
48          }
49
50          this._items[selectedEntity][response.data.data._id] = response.data.data;
51        } else {
52          throw new Error(response.data.error);
53        }
54        this.isLoading = false;
55        return Promise.resolve(response.data.data)
56      }).catch((err) => {
57        this.isLoading = false;
58        return Promise.reject(err);
59      })
60    }
61
62    fetchItem(url, id, entityName) {
63      let selectedEntity = entityName || this.selectedEntity;
64      this.isLoading = true;
65      return axios.get(url + "/" + id).then(response => {
66        if (response.data.status) {
67          if (!this._items[selectedEntity]) {
68            this._items[selectedEntity] = {};
69          }
70          this._items[selectedEntity][id] = response.data.data;
71        } else {
72          throw new Error(response.data.error);
73        }
74        this.isLoading = false;
75        return Promise.resolve(response.data.data)
76      }).catch((err) => {
77        this.isLoading = false;
78        return Promise.reject(err);
79      })
80    }
81
82    updateItem(url, id, values, entityName) {
83      let selectedEntity = entityName || this.selectedEntity;
84      this.isLoading = true;
85      return axios.put(url + "/" + id, values).then(response => {
86        if (response.data.status) {
87          if (!this._items[selectedEntity]) {
88            this._items[selectedEntity] = {};
89          }
90          this._items[selectedEntity][id] = response.data.data;
91        } else {
92          throw new Error(response.data.error);
93        }
94        this.isLoading = false;
95        return Promise.resolve()
96      }).catch((err) => {
97        this.isLoading = false;
98        return Promise.reject(err);
99      })
100    }
101
102   deleteItem(url, id, entityName) {
103     let selectedEntity = entityName || this.selectedEntity;
104     this.isLoading = true;
105     return axios.delete(url + "/" + id).then(response => {
106       if (response.data.status) {
107         delete this._items[selectedEntity][id];
108       } else {
109         throw new Error(response.data.error);
110       }
111       this.isLoading = false;
112       return Promise.resolve()
113     }).catch((err) => {
114       this.isLoading = false;
115       return Promise.reject(err);
116     })
117   }

```

```

117     }
118
119     get items() {
120       if (this._items[this.selectedEntity]) {
121         return Object.values(this._items[this.selectedEntity])
122       }
123       return [];
124     }
125
126     getEntityItems(entityName) {
127       let selectedEntity = entityName || this.selectedEntity;
128
129       if (this._items[selectedEntity]) {
130         return Object.values(this._items[selectedEntity])
131       }
132       return [];
133     }
134
135     get item() {
136       if (this._items[this.selectedEntity]) {
137         return { ...{...this.defaultEntityValues[this.selectedEntity] || {}}, ...{...this._items[this.selectedEntity][this.selectedEntityId]} };
138       }
139       return { ...{...this.defaultEntityValues[this.selectedEntity] || {}} }
140     }
141
142     getEntityItem(id, entityName) {
143       let selectedEntity = entityName || this.selectedEntity;
144
145       if (this._items[selectedEntity]) {
146         return { ...{...this.defaultEntityValues[selectedEntity] || {}}, ...{...this._items[selectedEntity][id]} };
147       }
148       return { ...{...this.defaultEntityValues[selectedEntity] || {}} }
149     }
150   }
151
152   decorate(EntityStore, {
153     _items: observable,
154     items: computed,
155     item: computed,
156     getEntityItems: observable,
157     getEntityItem: observable,
158
159     isLoading: observable,
160     error: observable,
161
162     fetch: action,
163     fetchItem: action,
164     updateItem: action,
165     deleteItem: action,
166     createItem: action,
167     selectedEntity: observable,
168     selectedEntityId: observable,
169     defaultEntityValues: observable
170   })
171 })
172
173
174 export default EntityStore;

```

Listing B.155 examStore.js

```

1 import { action, decorate, observable } from "mobx";
2 import { axios } from "~/apicall";
3
4 class ExamStore {
5   isFetchingExam = true
6
7   notification = [];
8
9   exam = {};
10  participant = {};
11
12  notificationAdd(notif) {
13    this.notification.push(notif);
14  }
15
16  notificationRemove(id) {
17    this.notification = this.notification.map((el, index) => index === id ? null : el).filter(el => !el);
18  }
19
20  fetchExamInfo(showLoading) {
21    this.isFetchingExam = (showLoading && true);
22    return axios.get('exam/info').then((response) => {
23      this.isFetchingExam = false;
24      if (!response.data.status) {
25        return Promise.reject({ err: "Not Ok, something not ok.", ref: response.data });
26      }
27      this.participant = response.data.data;
28      this.exam = this.participant?.exam;
29      return Promise.resolve();
30    });
31  }
32}
33
34 decorate(ExamStore, {
35

```

```
36     participant: observable,  
37     exam: observable,  
38     isFetchingExam: observable,  
39     fetchExamInfo: action,  
40   });  
  
43 export default ExamStore;
```