

**PENGEMBANGAN SISTEM GRADER DI JURUSAN ILMU KOMPUTER
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG**

(Skripsi)

**Oleh
MEI RUSFANDI**



**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2019**

ABSTRACT

**development of a grader system in the mathematics and natural sciences faculty
computer science department**

by

Mei Rusfandi

Grader is a system develop to completement the need for the use of an online learning system at the Computer Science Department, Mathematics and Natural Science Faculty, Universitas Lampung. The virtual class system are used to facilities the assessment with automatic correction assignments and multiple choice tests. Regrader is used to make automatic correction for programming tasks.

Research using extreme programming method starting with the planning phase. This phase carried out to analyze the requirements of the grader system and produce use case diagrams. Design is the second phase, that is changing the results of the analysis into Unified Modelling Language (UML) diagram like activity, class, and sequence diagram. The third phase is coding, that is do coding based on the design that has been made. Coding process used a software like Sublime Text by using the CodeIgniter PHP framework. Black box testing is the method used for the fourth phase from system development using extreme programming methods. Release phase done when all needs have been met.

The development carried out in this study is the addition of user categories to admin, editor and participants as well as the addition of features to share code. Testing is done twice. Every testing gives some suggestion to the system. After repairs, and do the testing again, respondents gave the result that the system can be used and can work as needed.

Keyword : grader system, automation testing, extreme programming, black box testing.

ABSTRAK

PENGEMBANGAN SISTEM GRADER DI JURUSAN ILMU KOMPUTER FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM UNIVERSITAS LAMPUNG

Oleh

Mei Rusfandi

Grader merupakan sistem yang dikembangkan untuk melengkapi kebutuhan penggunaan sistem belajar *online* pada Jurusan Ilmu Komputer FMIPA Unila. Sistem Virtual Class (VClass) digunakan untuk mempermudah penilaian dengan koreksi otomatis tugas dan ujian *multiple choice* atau isian singkat. Sedangkan grader digunakan untuk melakukan koreksi otomatis untuk tugas pemrograman.

Penelitian menggunakan metode *Extreme Programming* dimulai dengan tahap *planning*. Tahap ini dilakukan untuk menganalisa kebutuhan sistem *grader* dan menghasilkan *use case* diagram. *Design* merupakan tahap kedua, yaitu mengubah hasil analisa menjadi bentuk diagram *Unified Modelling Language* (UML) berupa *activity*, *class* dan *sequence* diagram. Tahap ketiga merupakan tahap *coding* yaitu melakukan pengkodean berdasarkan *design* yang sudah dibuat. Proses *coding* menggunakan perangkat lunak berupa Sublime Text dengan menggunakan *framework* PHP CodeIgniter. *Black box testing* merupakan metode yang digunakan untuk tahap keempat dari pengembangan sistem menggunakan metode pengembangan *Extreme Programming*. Tahap *release* dilakukan ketika semua kebutuhan sudah terpenuhi.

Pengembangan yang dilakukan pada penelitian ini merupakan penambahan kategori pengguna menjadi admin, editor dan peserta serta penambahan fitur untuk share code. Pengujian dilakukan sebanyak dua kali. Setiap pengujian memberikan beberapa saran dan masukan ke sistem. Setelah dilakukan perbaikan, dan melakukan pengujian kembali, responden memberikan hasil bahwa sistem sudah dapat digunakan, dan dapat berjalan sesuai kebutuhan.

Kata Kunci : *grader*, *automation testing*, *extreme programming*, *black box testing*

**PENGEMBANGAN SISTEM GRADER DI JURUSAN ILMU KOMPUTER
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG**

Oleh
MEI RUSFANDI

Skripsi
Sebagai Salah Satu Syarat Untuk Mencapai Gelar
SARJANA KOMPUTER

Pada
Jurusan Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam



**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2019**

Judul Skripsi : **PENGEMBANGAN SISTEM GRADER DI
JURUSAN ILMU KOMPUTER FAKULTAS
MATEMATIKA DAN ILMU PENGETAHUAN
ALAM UNIVERSITAS LAMPUNG**

Nama Mahasiswa : **Mei Rusfandi**

Nomor Pokok Mahasiswa : 1417051090

Jurusan : Ilmu Komputer

Fakultas : Matematika dan Ilmu Pengetahuan Alam

MENYETUJUI

1. Komisi Pembimbing


Didik Kurniawan, S.Si., M.T.
NIP 19800419 200501 1 004


Ardiansyah, S.Kom., M.Kom.
NIP 19870128 201803 1 001

2. Ketua Jurusan Ilmu Komputer


Dr. Ir. Kurnia Muludi, M.S.Sc.
NIP 19640616 198902 1 001

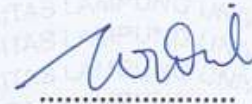
MENGESAHKAN

1. Tim Penguji

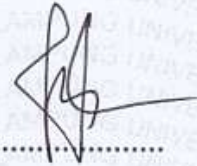
Ketua : Didik Kurniawan, S.Si., M.T.



Sekretaris : Ardiansyah, S.Kom., M.Kom.



**Penguji
Bukan Pembimbing : Febi Eka Febriansyah, S.T., M.T.**

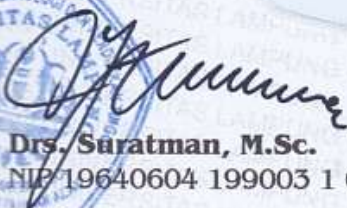


2. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam



Drs. Saratman, M.Sc.

NIP 19640604 199003 1 002



Tanggal Lulus Ujian Skripsi : 7 Februari 2019

PERNYATAAN

Saya yang bertanda tangan di bawah ini, menyatakan bahwa skripsi saya yang berjudul **“Pengembangan Sistem Grader di Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Universitas Lampung”** merupakan karya saya sendiri dan bukan karya orang lain. Semua tulisan yang tertuang di skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti skripsi saya merupakan hasil penjiplakan atau dibuat orang lain, maka saya bersedia menerima sanksi berupa pencabutan gelar yang telah saya terima.

Bandar Lampung, 28 Juni 2019



Mei Rusfandi
NPM. 1417051090

RIWAYAT HIDUP



Penulis dilahirkan di Kediri, Kecamatan Gadingrejo Kabupaten Pringsewu pada tanggal 11 Mei 1996, sebagai anak kedua dari enam bersaudara, dari bapak Slamet Miyanto dan ibu Sugi Harti.

Penulis memulai menempuh pendidikan dibangku Sekolah Dasar di SD Negeri 1 Kediri dan menyelesaikan pendidikan pada tahun 2008. Penulis melanjutkan pendidikan ke jenjang Menengah di Sekolah Menengah Pertama (SMP) Negeri 3 Gadingrejo, dan menyelesaikannya dengan waktu 3 tahun, dan lulus pada tahun 2011. Pendidikan selanjutnya ditempuh di Sekolah Menengah Atas (SMA) Negeri 1 Gadingrejo dan lulus pada tahun 2014.

Pada tahun 2014, penulis resmi menjadi mahasiswa jurusan Ilmu Komputer FMIPA Unila melalui jalur Seleksi Bersama Masuk Perguruan Tinggi Negeri (SBMPTN) dan mendapatkan beasiswa Bidikmisi. Selama menjadi mahasiswa, penulis aktif di organisasi Himpunan Mahasiswa Jurusan Ilmu Komputer (Himakom) hingga tahun 2016 sebagai anggota bidang Keilmuan. Penulis juga aktif menjadi asisten dosen dan asisten lab untuk membantu dosen dalam praktikum maupun responsi. Pada tahun 2016 penulis menjadi anggota Relawan TIK Lampung dan mendapatkan sertifikasi programmer BNSP dari lembaga LSP Inixindo. Pada tahun yang sama, penulis

melaksanakan Program Kerja Praktek (KP) di lembaga Astana Ilmu di Jakarta Timur. Masih di tahun 2016, penulis menjadi delegasi untuk mengikuti Bukalapak Programming Contest Regional Sumatera, yang diadakan oleh Bukalapak di Medan, Sumatera Utara. Pada tahun 2017, penulis melaksanakan Kuliah Kerja Nyata (KKN) di desa Panutan, Kecamatan Pagelaran, Kabupaten Pringsewu. Pada tahun 2018, penulis mendapatkan sertifikasi Intermediate Programmer dari BNSP yang diselenggarakan oleh Kominfo. Penulis juga mendapatkan beasiswa dan menyelesaikan Dicoding Android Developer Academy, untuk kategori Android Developer Expert.

PERSEMBAHAN

Saya persembahkan karya ini untuk:

Ayah dan Ibu tercinta, motivator sekaligus alasan untuk sukses. Orang yang telah membesarkan, mendidik, mendukung, mendoakan, memberikan nasehat bimbingan dan telah mengorbankan segalanya untuk penulis.

Kakak, adik dan keponakan tercinta dan seluruh keluarga besar yang selalu memberikan doa, dan semangat.

Teman-teman seperjuangan Ilmu Komputer angkatan 2014

Almamater Universitas Lampung.

MOTTO

“Cintai apa yang kamu kerjakan, maka yang kamu kerjakan akan mencintaimu dan memberikan manfaat untukmu”

(Anonim)

“Ide itu murah, aksi yang membuatnya jadi bernilai”

(Anonim)

“‘I don’t Know’ has become ‘I don’t know yet’”

(Bill Gates)

“Apa yang membuatmu tak nyaman adalah kesempatan terbesarmu untuk tumbuh”

(Bryan McGill)

SANWACANA

Puji Syukur penulis ucapkan kehadiran Allah SWT, Tuhan Yang Maha Esa, karena berkat rahmat dan hidayah-Nya skripsi ini dapat diselesaikan.

Skripsi dengan judul “Pengembangan Sistem Grader di Jurusan Ilmu Komputer FMIPA Unila” adalah salah satu syarat untuk memperoleh gelar Sarjana Ilmu Komputer di Universitas Lampung. Dalam kesempatan ini, penulis mengucapkan terima kasih kepada:

1. Kedua orang tua, bapak Slamet Miyanto dan Ibu Sugi Harti yang telah mengorbankan segalanya untuk penulis, baik tenaga, pikiran, doa, semangat, nasehat, dan segala bentuk pengorbanan supaya penulis dapat menyelesaikan tugas akhir dengan cepat.
2. Kakak Eka Puspita Rini dan suami, adik Leny Mita Sari, Andi Prabowo, Kalista Vitha S., M. Rayyan Al-Ghafary, keponakan Gladis Lailatul Z., Geovano Al-farizi yang telah memberikan doa, semangat dan keinginan untuk cepat menyelesaikan tugas akhir ini.
3. Bapak Dr. Ir. Kurnia Muludi Selaku Ketua Jurusan Ilmu Komputer FMIPA Unila dan dosen pembimbing akademik.

4. Bapak Didik Kurniawan, S.Si., M.T., selaku dosen pembimbing utama atas kesediaannya untuk memberikan bimbingan, arahan, kritik dan saran selama proses pengerjaan skripsi.
5. Bapak Ardiansyah, S.Kom., M.Kom., selaku pembimbing kedua atas kesediaannya untuk memberikan bimbingan, arahan, kritik dan saran selama proses pengerjaan skripsi.
6. Bapak Febi Eka Febriansyah, S.T., M.T., selaku penguji utama pada ujian skripsi. Terima kasih atas masukan dan saran untuk pengembangan sistem ini.
7. Bapak/Ibu dosen yang telah membimbing, mengarahkan dan memberikan pembelajaran selama kuliah.
8. Bapak/ibu staff administrasi Jurusan Ilmu Komputer FMIPA Unila yang telah membantu segala bentuk urusan administrasi.
9. Mas Ardi Novalian, Mas Zainudin, Mbak Karmilia (Lia) atas kesediaannya untuk memberikan tempat untuk melakukan penelitian.
10. Hanifatussa'diah rekan, teman, partner yang selalu menemani, membantu dan mengingatkan ketika malas mengerjakan tugas akhir.
11. Dimas Aji Sukma, Estu Ramadian W. Nagara (Liong), teman berbagi kamar, teman berbagi code, bertukar pikiran yang telah menemani penulis sejak awal kuliah hingga akhir kuliah.
12. M. Wildan Mubarak, Rifky Haryuda, Rahmat Purnama, Feri Krisnanto, Firmansyah, Fikri Muhammad, yang telah membantu penulis untuk menyelesaikan tugas akhir ini.

13. Teman seperjuangan jurusan Ilmu Komputer FMIPA Unila angkatan 2014 yang saling membantu, memberi semangat dan memberi warna selama perkuliahan.
14. Kakak-kakak tingkat dan adik tingkat yang telah membantu dan memberi dukungan selama kuliah.
15. Teman-teman KKN desa Panutan, Devi Ranita, Satria Yudistira, Fergina Tyas, Lukman Nawawi dan KKN kabupaten Pringsewu yang telah membantu, memberikan semangat dan memberikan nasehat kepada penulis.
16. Teman-teman Kompor Mleduk kelas XII IPA 4 lulusan 2014 SMAN 1 Gadingrejo yang telah memberikan dukungan kepada penulis.
17. Teman-teman kelas 9 A lulusan 2011 SMP N 3 Gadingrejo
18. Teman-teman pengajar di SMK N 8 Bandar Lampung yang telah membantu dan memberikan dukungan kepada penulis.
19. Teman-teman pengajar OSN Astana Ilmu, Science Olympiad Club (SOC) dan Erick Institute yang telah memberikan pekerjaan, pengalaman kerja dan memberikan dorongan semangat dan bantuan lainnya kepada penulis.
20. Seluruh pihak yang telah membantu proses perkuliahan hingga menyelesaikan skripsi yang tidak dapat penulis sebutkan satu persatu atas dukungan, dan doa yang telah diberikan.

Bandar Lampung, Juni 2019

Mei Rusfandi
NPM. 1417051090

DAFTAR ISI

	Halaman
DAFTAR ISI.....	xvi
DAFTAR TABEL.....	ii
DAFTAR GAMBAR	vii
DAFTAR KODE.....	xiii
I. PENDAHULUAN	1
1.1. Latar Belakang.....	1
1.2. Rumusan Masalah	3
1.3. Batasan Masalah.....	3
1.4. Tujuan	3
1.5. Manfaat.....	4
II. TINJAUAN PUSTAKA	5
2.1. Pengertian <i>Grader</i>	5
2.2. Pengertian <i>MOE Contest Environment</i>	5
2.3. <i>Framework</i>	6
2.4. <i>CodeIgniter</i>	7
1. <i>MVC (Model View Controller)</i>	8
2. <i>Model</i>	8
3. <i>View</i>	8
4. <i>Controller</i>	9

2.5.	<i>Bootstrap</i>	10
2.6.	<i>Unified Modelling Language (UML)</i>	11
2.6.1.	Diagram <i>use case (use case diagram)</i>	11
2.6.2.	Diagram aktivitas (<i>activity diagram</i>)	13
2.6.3.	Diagram sekuensial (<i>Sequence diagram</i>).....	14
2.6.4.	Diagram kelas (<i>class diagram</i>)	15
2.7.	Basis Data.....	16
2.8.	<i>Extreme Programming</i>	17
2.8.1.	<i>Planning</i>	19
2.8.2.	<i>Design</i>	19
2.8.3.	<i>Coding</i>	19
2.8.4.	<i>Testing</i>	20
2.8.5.	<i>Release</i>	20
2.9.	<i>Black Box Testing</i>	20
III.	METODE PENELITIAN	22
3.1.	Waktu dan Tempat Penelitian.....	22
3.2.	Spesifikasi <i>Software</i> dan <i>Hardware</i> yang Digunakan.....	22
3.2.1.	Spesifikasi <i>software</i>	22
3.2.2.	Spesifikasi <i>hardware</i>	23
3.3.	Tahapan Penelitian	23
3.3.1.	Tahap Identifikasi Masalah.....	24
3.3.2.	Tahap Studi Literatur	26
3.3.3.	<i>Planning</i>	27
3.3.4.	<i>Design</i>	31
3.3.5.	<i>Coding System</i>	121

3.3.6.	<i>Testing</i>	121
3.3.7.	<i>Release</i>	132
IV.	HASIL DAN PEMBAHASAN	133
4.1.	Hasil	133
4.2.	Coding.....	133
4.2.1.	Halaman <i>Register</i> Mahasiswa.....	133
4.2.2.	Halaman <i>Login</i>	136
4.2.3.	Halaman <i>Dashboard</i> Mahasiswa	137
4.2.4.	Halaman <i>Dashboard Admin / Editor</i>	138
4.2.5.	Halaman Soal / <i>Problem</i> Mahasiswa	140
4.2.6.	Halaman Soal / <i>Problem Admin / Editor</i>	141
4.2.7.	Halaman Tambah/Ubah <i>Problem</i>	142
4.2.8.	Halaman Tambah <i>Testcase</i>	144
4.2.9.	Halaman Tambah <i>Checker</i>	145
4.2.10.	Halaman <i>Submission</i> Mahasiswa	146
4.2.11.	<i>Form</i> Tambah <i>Submission</i>	148
4.2.12.	Halaman <i>Submission Admin / Editor</i>	149
4.2.13.	Halaman <i>Detail Submission</i>	151
4.2.14.	Halaman <i>Clarification</i>	152
4.2.15.	<i>Form</i> Tambah <i>Clarification</i>	154
4.2.16.	Halaman <i>Scoreboard</i> Mahasiswa.....	155
4.2.17.	Halaman <i>Scoreboard Admin / Editor</i>	157
4.2.18.	Halaman <i>User</i>	158

4.2.19.	Halaman Tambah/Ubah <i>User</i>	159
4.2.20.	Halaman <i>Category</i>	161
4.2.21.	Halaman Tambah/Ubah <i>Category</i>	162
4.2.22.	Halaman <i>Contest Admin / Editor</i>	163
4.2.23.	Halaman Tambah/Ubah <i>Contest</i>	165
4.2.24.	Halaman <i>Language</i>	166
4.2.25.	Halaman Tambah/ubah <i>Language</i>	168
4.2.26.	Halaman <i>File</i>	169
4.2.27.	Halaman Tambah <i>File</i>	170
4.2.28.	Halaman Ubah <i>Options</i>	171
4.3.	<i>Testing</i>	172
4.4.	<i>Release</i>	184
4.4.1.	Pengujian Pertama.....	185
4.4.2.	Pengujian Kedua	189
4.5.	Pembahasan.....	197
V.	SIMPULAN DAN SARAN.....	198
5.1.	Simpulan.....	198
5.2.	Saran.....	199
	DAFTAR PUSTAKA	201

DAFTAR TABEL

Tabel	Halaman
Tabel 2.1 Simbol-simbol <i>use case diagram</i> (Shalahuddin dan Rosa, 2013)	12
Tabel 2.2 simbol-simbol <i>activity diagram</i> (Shalahuddin dan Rosa, 2013).....	13
Tabel 2.3 Simbol-simbol <i>Sequence diagram</i> (Shalahuddin dan Rosa, 2013).....	14
Tabel 2.4 Simbol-simbol <i>class diagram</i> (Shalahuddin dan Rosa, 2013).....	15
Tabel 3.1 Tabel rencana pengujian pada aktor <i>admin</i>	122
Tabel 3.2 Tabel rencana pengujian pada aktor <i>editor</i>	127
Tabel 3.3 Tabel rencana pengujian pada aktor <i>contestant/mahasiswa</i>	130
Tabel 4.1 Tabel pengujian pada aktor <i>admin</i>	172
Tabel 4.2 Tabel pengujian pada aktor <i>editor</i>	178
Tabel 4.3 Tabel Pengujian pada aktor <i>contestant/mahasiswa</i>	182
Tabel 4.4 Tabel hasil pengujian pertama	185
Tabel 4.5 Tabel hasil pengujian kedua.....	189

DAFTAR GAMBAR

Gambar	Halaman
Gambar 2.1. Hubungan antara MVC di CodeIgniter (Pribadi, 2016).....	9
Gambar 2.2 <i>Extreme Programming</i> (XP) (Pressman, 2014)	18
Gambar 3.1 Alur Penelitian.....	24
Gambar 3.2 <i>Use case diagram user/contestant/mahasiswa</i>	28
Gambar 3.3 <i>Use case diagram editor/dosen/asisten dosen</i>	29
Gambar 3.4.1 <i>Use case diagram admin bagian 1</i>	30
Gambar 3.4.2 <i>Use case diagram admin bagian 2</i>	31
Gambar 3.5 <i>Activity diagram melihat contest/Kelas</i>	33
Gambar 3.6 <i>Activity diagram melihat problem/Soal</i>	34
Gambar 3.7 <i>Activity diagram melihat submission/jawaban</i>	35
Gambar 3.8 <i>Activity diagram tambah submission</i>	36
Gambar 3.9 <i>Activity diagram melihat clarification</i>	37
Gambar 3.10 <i>Activity diagram tambah clarification</i>	38
Gambar 3.11 <i>Activity diagram melihat scoreboard</i>	39
Gambar 3.12 <i>Activity diagram melihat contest/kelas</i>	40
Gambar 3.13 <i>Activity diagram mengubah contest/kelas</i>	41
Gambar 3.14 <i>Activity diagram menambah problem ke contest/kelas</i>	42

Gambar 3.15 <i>Activity diagram</i> melihat <i>problem/soal</i>	43
Gambar 3.16 <i>Activity diagram</i> menambah <i>problem/soal</i>	44
Gambar 3.17 <i>Activity diagram</i> mengubah <i>problem/soal</i>	45
Gambar 3.18 <i>Activity diagram</i> menghapus <i>problem/soal</i>	46
Gambar 3.19 <i>Activity diagram</i> menambah <i>checker</i>	47
Gambar 3.20 <i>Activity diagram</i> melihat <i>testcase</i>	48
Gambar 3.21 <i>Activity diagram</i> menambah <i>testcase</i>	49
Gambar 3.22 <i>Activity diagram</i> menghapus <i>testcase</i>	50
Gambar 3.23 <i>Activity diagram</i> export <i>testcase</i>	51
Gambar 3.24 <i>Activity diagram</i> melihat <i>clarification</i>	52
Gambar 3.25 <i>Activity diagram</i> menambah <i>clarification</i>	53
Gambar 3.26 <i>Activity diagram</i> menjawab <i>clarification</i>	54
Gambar 3.27 <i>Activity diagram</i> melihat <i>submission</i>	55
Gambar 3.28 <i>Activity diagram</i> regrade <i>submission</i>	56
Gambar 3.29 <i>Activity diagram</i> menghapus <i>submission</i>	57
Gambar 3.30 <i>Activity diagram</i> melihat <i>scoreboard</i>	58
Gambar 3.31 <i>Activity diagram</i> export <i>scoreboard</i>	59
Gambar 3.32 <i>Activity diagram</i> melihat <i>category</i>	60
Gambar 3.33 <i>Activity diagram</i> menambah <i>category</i>	61
Gambar 3.34 <i>Activity diagram</i> mengubah <i>category</i>	62
Gambar 3.35 <i>Activity diagram</i> menghapus <i>category</i>	63
Gambar 3.36 <i>Activity diagram</i> melihat <i>file</i>	64
Gambar 3.37 <i>Activity diagram</i> menambah <i>file</i>	65

Gambar 3.38 <i>Activity diagram</i> menghapus <i>file</i>	66
Gambar 3.39 <i>Activity diagram</i> mengedit <i>option</i>	67
Gambar 3.40 <i>Activity diagram</i> melihat <i>languages</i>	68
Gambar 3.41 <i>Activity diagram</i> menambah <i>languages</i>	69
Gambar 3.42 <i>Activity diagram</i> mengedit <i>languages</i>	70
Gambar 3.43 <i>Activity diagram</i> menghapus <i>languages</i>	71
Gambar 3.45 <i>Activity diagram</i> menambah <i>contest/kelas</i>	72
Gambar 3.46 <i>Activity diagram</i> menghapus <i>contest/kelas</i>	73
Gambar 3.47 <i>Activity diagram</i> register mahasiswa	74
Gambar 3.48 <i>Activity diagram</i> <i>sharecode</i>	75
Gambar 3.49 <i>Class Diagram</i>	76
Gambar 3.50 <i>Sequence diagram</i> melihat <i>contest</i>	78
Gambar 3.51 <i>Sequence diagram</i> melihat <i>problem</i>	79
Gambar 3.52 <i>Sequence diagram</i> melihat <i>scoreboard</i>	80
Gambar 3.53 <i>Sequence diagram</i> melihat <i>submission</i>	81
Gambar 3.54 <i>Sequence diagram</i> menambah <i>submission</i>	82
Gambar 3.55 <i>Sequence diagram</i> melihat <i>clarification</i>	83
Gambar 3.56 <i>Sequence diagram</i> menambah <i>clarification</i>	84
Gambar 3.57 <i>Sequence diagram</i> melihat <i>contest</i>	85
Gambar 3.58 <i>Sequence diagram</i> mengubah <i>contest</i>	86
Gambar 3.59 <i>Sequence diagram</i> menambah <i>problem</i> ke <i>contest</i>	87
Gambar 3.60 <i>Sequence diagram</i> melihat <i>problem</i>	88

Gambar 3.61 <i>Sequence diagram</i> menambah <i>problem</i>	89
Gambar 3.62 <i>Sequence diagram</i> mengubah <i>problem</i>	90
Gambar 3.63 <i>Sequence diagram</i> menghapus <i>problem</i>	91
Gambar 3.64 <i>Sequence diagram</i> menambah <i>checker</i>	92
Gambar 3.65 <i>Sequence diagram</i> melihat <i>testcase</i>	93
Gambar 3.66 <i>Sequence diagram</i> menambah <i>testcase</i>	94
Gambar 3.67 <i>Sequence diagram</i> menghapus <i>testcase</i>	95
Gambar 3.68 <i>Sequence diagram</i> mengexport <i>testcase</i>	96
Gambar 3.69 <i>Sequence diagram</i> melihat <i>submission</i>	97
Gambar 3.70 <i>Sequence diagram</i> regrade <i>submission</i>	98
Gambar 3.71 <i>Sequence diagram</i> menghapus <i>submission</i>	99
Gambar 3.72 <i>Sequence diagram</i> melihat <i>clarification</i>	100
Gambar 3.73 <i>Sequence diagram</i> menambah <i>clarification</i>	101
Gambar 3.74 <i>Sequence diagram</i> menjawab <i>clarification</i>	102
Gambar 3.75 <i>Sequence diagram</i> melihat <i>scoreboard</i>	103
Gambar 3.76 <i>Sequence diagram</i> mengexport <i>scoreboard</i>	104
Gambar 3.77 <i>Sequence diagram</i> menambah <i>contest</i>	105
Gambar 3.78 <i>Sequence diagram</i> menghapus <i>contest</i>	106
Gambar 3.79 <i>Sequence diagram</i> melihat <i>languages</i>	107
Gambar 3.80 <i>Sequence diagram</i> menambah <i>languages</i>	108
Gambar 3.81 <i>Sequence diagram</i> mengubah <i>languages</i>	109
Gambar 3.82 <i>Sequence diagram</i> menghapus <i>languages</i>	110
Gambar 3.83 <i>Sequence diagram</i> melihat <i>file</i>	111

Gambar 3.84 <i>Sequence diagram</i> menambah <i>file</i>	112
Gambar 3.85 <i>Sequence diagram</i> menghapus <i>file</i>	113
Gambar 3.86 <i>Sequence diagram</i> melihat <i>category</i>	114
Gambar 3.87 <i>Sequence diagram</i> menambah <i>category</i>	115
Gambar 3.88 <i>Sequence diagram</i> mengubah <i>category</i>	116
Gambar 3.89 <i>Sequence diagram</i> menghapus <i>category</i>	117
Gambar 3.90 <i>Sequence diagram</i> mengubah <i>option</i>	118
Gambar 4.91 <i>Sequence diagram</i> register mahasiswa	119
Gambar 4.92 <i>Sequence diagram</i> <i>sharecode</i>	120
Gambar 4.1 Halaman <i>register</i> mahasiswa.....	135
Gambar 4.2 Halaman <i>login</i>	136
Gambar 4.3 Halaman <i>dashboard</i> mahasiswa.....	137
Gambar 4.4 Halaman <i>dashboard</i> admin/editor.....	139
Gambar 4.5 Halaman <i>soal/problem</i> mahasiswa	140
Gambar 4.6 Halaman <i>soal/problem editor/admin</i>	141
Gambar 4.7 Halaman <i>tambah/ubah problem</i>	143
Gambar 4.8 Halaman <i>tambah testcase</i>	144
Gambar 4.9 Halaman <i>tambah checker</i>	145
Gambar 4.10 Halaman <i>submission</i> mahasiswa	147
Gambar 4.11 <i>Form</i> <i>tambah submission</i>	148
Gambar 4.12 Halaman <i>submission</i> admin/editor	150
Gambar 4.13 Halaman <i>detail submission</i>	151

Gambar 4.14 Halaman <i>clarification</i>	153
Gambar 4.15 <i>Form</i> tambah <i>clarification</i>	154
Gambar 4.16 Halaman <i>scoreboard</i> mahasiswa.....	156
Gambar 4.17 Halaman <i>Scoreboard Admin/Editor</i>	157
Gambar 4.18 Halaman <i>user</i>	158
Gambar 4.19 Halaman tambah/ubah <i>user</i>	160
Gambar 4.20 Halaman <i>category</i>	161
Gambar 4.21 Halaman tambah/ubah <i>category</i>	162
Gambar 4.22 Halaman <i>contest admin/editor</i>	164
Gambar 4.23 Halaman tambah/ubah <i>contest</i>	165
Gambar 4.24 Halaman <i>language</i>	167
Gambar 4.25 Halaman tambah/ubah <i>language</i>	168
Gambar 4.26 Halaman <i>file</i>	169
Gambar 4.27 Halaman tambah <i>file</i>	170
Gambar 4.28 Halaman ubah <i>options</i>	171
Gambar 4.29. <i>Use case diagram</i> mahasiswa sebelum pengujian	190
Gambar 4.30 <i>Use case diagram</i> mahasiswa setelah pengujian.....	191
Gambar 4.31 <i>Activity diagram register</i>	192
Gambar 4.33 <i>Activity diagram sharecode</i>	195
Gambar 4.34 <i>Sequence diagram sharecode</i>	196

DAFTAR KODE

Kode	Halaman
Kode 4.1 Potongan kode program <i>register</i> mahasiswa.....	135
Kode 4.2 Potongan kode program halaman <i>login</i>	136
Kode 4.3 Potongan kode program halaman <i>dashboard</i> mahasiswa	138
Kode 4.4 Potongan kode program halaman <i>dashboard admin/editor</i>	139
Kode 4.5 Potongan kode program halaman soal/ <i>problem</i> mahasiswa.....	140
Kode 4.6 Potongan kode program halaman soal/ <i>problem editor/admin</i>	142
Kode 4.7 Potongan kode program halaman tambah/ubah <i>problem</i>	143
Kode 4.8 Potongan program halaman tambah <i>testcase</i>	144
Kode 4.9 Potongan program halaman tambah <i>checker</i>	146
Kode 4.10 Potongan kode program halaman <i>submission</i> mahasiswa.....	147
Kode 4.11 Potongan program <i>form</i> tambah <i>submission</i>	149
Kode 4.12 Potongan kode program <i>submission editor/admin</i>	150
Kode 4.13 Potongan kode program halaman <i>detail submission</i>	152
Kode 4.14 Potongan kode program halaman <i>clarification</i>	153
Kode 4.15 Potongan kode program tambah <i>clarification</i>	155
Kode 4.16 Potongan kode program halaman <i>scoreboard</i> mahasiswa	156
Kode 4.17 Potongan kode program <i>scoreboard admin/editor</i>	158
Kode 4.18 Potongan kode program halaman <i>user</i>	159

Kode 4.19 Potongan program tambah/ubah <i>user</i>	160
Kode 4.20 Potongan kode program halaman <i>category</i>	162
Kode 4.21 Potongan kode program halaman tambah/ubah <i>category</i>	163
Kode 4.22 Potongan kode program <i>contest admin/editor</i>	164
Kode 4.23 Potongan kode program halaman tambah/ubah <i>contest</i>	166
Kode 4.24 Potongan kode program halaman <i>language</i>	167
Kode 4.25 Potongan kode program halaman tambah/ubah <i>language</i>	168
Kode 4.26 Potongan kode program halaman <i>file</i>	169
Kode 4.27 Potongan kode program tambah <i>file</i>	170
Kode 4.28 Potongan kode program halaman ubah <i>options</i>	172
Kode 4.29 Kode program <i>detail submission</i> sebelum perubahan	186
Kode 4.30 Kode program <i>detail submission</i> setelah perubahan	187
Kode 4.31 Kode program <i>form submission</i> sebelum perubahan	188
Kode 4.32 Kode program <i>form submission</i> setelah perubahan	188
Kode 4.33 Kode program <i>sharecode</i>	194
Kode 4.34 Kode program <i>register user</i>	194

I. PENDAHULUAN

1.1. Latar Belakang

Grader merupakan sistem *grading* atau sistem yang digunakan untuk melakukan pengujian algoritma pemrograman dalam Bahasa Pemrograman Pascal, Java, C/C++. *Grader* digunakan oleh Tim Olimpiade Komputer Indonesia (TOKI) untuk melatih siswa-siswi SMA/ sederajat untuk dapat mengikuti perlombaan Olimpiade Sains Nasional (OSN) Bidang Komputer. *Grader* dibangun mengikuti aturan *International Olympiad in Informatics (IOI)* dan *Association for Computing Machinery – International Collegiate Programming Contest (ACM-ICPC)*.

Sistem ini telah dikembangkan oleh alumni Olimpiade Sains Nasional (OSN) dan menjadi aplikasi sumber terbuka yang dapat di-unduh pada laman <https://github.com/fushar/regrader>. Saat ini *regrader* sudah mencapai versi 2.2.2, namun aplikasi ini masih belum sempurna dan harus dikembangkan lagi. Sistem ini dibangun untuk dapat digunakan oleh dosen, dan asisten dosen untuk memeriksa tugas, atau ujian mahasiswa jurusan Ilmu Komputer FMIPA Unila. Sistem ini digunakan dengan membantu menyelesaikan beberapa masalah yang harus diselesaikan dengan menggunakan Bahasa Pemrograman Pascal, C, C++ dan Java. Sistem juga dapat dikembangkan untuk dapat melakukan *compile* dan *running*

program untuk semua bahasa pemrograman. Sistem akan melakukan pengujian secara otomatis dengan terlebih dahulu *admin* atau *editor* memberikan beberapa *test case* untuk masing-masing soal/masalah yang diberikan.

Sistem ini dikembangkan karena sistem atau aplikasi pembelajaran yang sudah ada seperti situs Kuliah *Online* (Kulon) untuk dosen dan mahasiswa dan situs Belajar *Online* (BeOn) untuk asisten dosen dan mahasiswa di Jurusan Ilmu Komputer Unila tidak mampu melakukan koreksi secara otomatis untuk tugas pemrograman, sehingga dosen dan asisten dosen harus melakukan koreksi untuk tugas pemrograman mahasiswa. Sistem ini dapat digunakan oleh dosen dan asisten dosen untuk dapat memajemen kelas/*contest* dan dapat memajemen nilai mahasiswa. Sistem dapat melakukan koreksi otomatis untuk tugas pemrograman. Sistem juga dapat digunakan untuk melatih kemampuan pemrograman (*coding*) sesuai dengan bahasa pemrograman yang diinginkan.

Sistem ini dibuat untuk dapat meningkatkan kualitas dan kemampuan membuat program mahasiswa dan membantu dosen dan asisten dosen dalam mengelola tugas dan nilainya. Nantinya sistem dapat dikembangkan sebagai kelas *online* yang digunakan oleh dosen dan asisten dosen. Manfaat lain dari sistem ini adalah dapat digunakan sebagai sistem untuk lomba pemrograman yang diadakan oleh Himpunan Mahasiswa Jurusan Ilmu Komputer (Himakom) pada acara Pekan Raya Jurusan (PRJ).

1.2. Rumusan Masalah

Permasalahan yang akan diangkat dalam tugas akhir ini adalah

1. Sistem pembelajaran *online* yang ada di Jurusan Ilmu Komputer FMIPA Unila tidak mampu melakukan koreksi otomatis untuk tugas pemrograman.
2. Manajemen nilai tugas dan ujian pada sistem yang ada masih kurang.

1.3. Batasan Masalah

Batasan masalah dalam penelitian ini dibutuhkan agar pembahasan tidak melebar dari topik yang telah dijabarkan pada rumusan masalah, antara lain.

1. Sistem yang dibuat digunakan untuk melakukan *auto correction/auto testing* untuk semua bahasa pemrograman.
2. Sistem dikembangkan dengan *Framework Codeigniter*.

1.4. Tujuan

Aplikasi *grading* dibuat, dengan tujuan sebagai berikut.

1. Membantu dosen dan asisten dosen untuk melakukan koreksi tugas mahasiswa secara otomatis.
2. Melakukan manajemen nilai tugas dan ujian mahasiswa dengan lebih mudah.

1.5. Manfaat

Manfaat dari penelitian ini yaitu

1. Mengimplementasikan ilmu yang sudah didapatkan selama kuliah di Jurusan Ilmu Komputer FMIPA Unila.
2. Memberikan pelatihan kepada mahasiswa untuk persiapan menghadapi lomba pemrograman.
3. Membantu Himakom untuk melaksanakan lomba pemrograman pada acara Pekan Raya Jurusan (PRJ).
4. Sebagai referensi untuk penelitian selanjutnya.

II. TINJAUAN PUSTAKA

2.1. Pengertian *Grader*

Regrader atau *Grader* adalah sistem yang digunakan untuk kontes pemrograman yang memiliki skenario seperti pada perlombaan *International Collegiate Programming Contest (ICPC)* yang diadakan oleh *Association for Computing Machinery (ACM)* (Regrader, 2015).

2.2. Pengertian *MOE Contest Environment*

MOE Contest Environment (dulu *MO-Eval*) adalah sebuah sistem untuk menyelenggarakan kompetisi pemrograman yang serupa dengan semangat pada *International Olympiad in Informatics (IOI)*. Peserta menyelesaikan tugas pemrograman, mengumpulkan kode program (*source code*) solusi mahasiswa, yang kemudian diuji secara otomatis pada satu set tes masukan. MOE dibangun dengan cara yang *modular*, sehingga mudah untuk menyesuaikannya dengan spesifikasi kontes tertentu, jenis tugas lainnya, atau bahasa pemrograman lainnya.

MOE masih dalam pengembangan dan versi terbaru dapat diunduh di [git://git.ucw.cz/moe.git](https://git.ucw.cz/moe.git). MOE digunakan oleh beberapa kontes, yaitu

1. CZECH Olympiad in programming sejak 2002
2. CPSPC (Czech-Polish-Slovak Preparation Camp) dalam beberapa tahun
3. CEOI 2007
4. IOI sejak 2009
5. CodEX

Bahasa pemrograman yang dapat dikerjakan menggunakan MOE adalah Pascal, C dan C++, C#, dan Haskell. Penambahan bahasa pemrograman baru akan lebih mudah (MOE *Contest Environment*, 2015).

2.3. Framework

Framework untuk *website* atau sering disebut *Web Application Framework* atau *web framework* adalah suatu kumpulan kode berupa pustaka (*library*) dan alat (*tool*) yang dipadukan sedemikian rupa menjadi satu kerangka kerja (*framework*) guna memudahkan dan mempercepat proses pengembangan aplikasi web (Raharjo, 2015).

Framework sebagaimana arti dalam Bahasa Indonesia yaitu kerangka kerja dapat diartikan sebagai kumpulan dari *library* (*class*) yang dapat diturunkan, atau dapat langsung dipakai fungsinya oleh modul-modul atau fungsi yang akan kita kembangkan (Septian, 2011).

2.4. *CodeIgniter*

CodeIgniter adalah *framework web* untuk bahasa pemrograman PHP, yang dibuat oleh Rick Ellis pada tahun 2006, penemu dan pendiri EllisLab (www.ellislab.com). Ellislab adalah suatu tim kerja yang berdiri pada tahun 2002 dan bergerak di bidang pembuatan *software* dan *tool* untuk para pengembang web. Sejak tahun 2014 sampai sekarang, Ellislab telah menyerahkan hak kepemilikan *CodeIgniter* ke *British Columbia Institute of Technology (BCIT)* untuk proses pengembangan lebih lanjut. Saat ini stus resmi dari *CodeIgniter* telah berubah dari www.ellislab.com ke www.codeigniter.com (Raharjo, 2015).

Tujuan dari pembuatan *Framework Codeigniter* ini menurut *user* manualnya adalah untuk menghasilkan *framework* yang akan dapat digunakan untuk pengembangan proyek pembuatan *website* dengan cara *coding* manual, dengan menyediakan banyak pustaka yang dibutuhkan dalam pembuatan *website*, dengan antarmuka yang sederhana dan logika terstruktur untuk mengakses pustaka yang dibutuhkan. *Codeigniter* memungkinkan pengguna untuk focus pada pembuatan *website* dengan meminimalkan pembuatan kode (Daqiqil, 2011).

CodeIgniter adalah aplikasi *open source* yang berupa *framework* dengan *model MVC* (*Model, View, Controller*) untuk membangun *website* dinamis. Dengan menggunakan PHP *CodeIgniter* akan memudahkan *developer* untuk membuat aplikasi web dengan cepat dan mudah dibandingkan dengan membuat dari awal (Septian, 2011).

1. MVC (Model View Controller)

MVC adalah sebuah pendekatan perangkat lunak yang memisahkan aplikasi logika dari presentasi. Ini meminimalkan *script* dari halaman-halaman web sejak *script* presentasi (*HTML, CSS, JavaScript, dsb*) dipisahkan dari *PHP scripting*, istilah umum yang *familiar* adalah menghindari terjadinya *spaghetti code* (Pribadi, 2016).

2. Model

Model merepresentasikan struktur data yang dibangun. Umumnya kelas *model* berisi fungsi-fungsi yang membantu *developer* untuk mengelola, memasukkan, dan mengubah informasi dalam *database* (Pribadi, 2016).

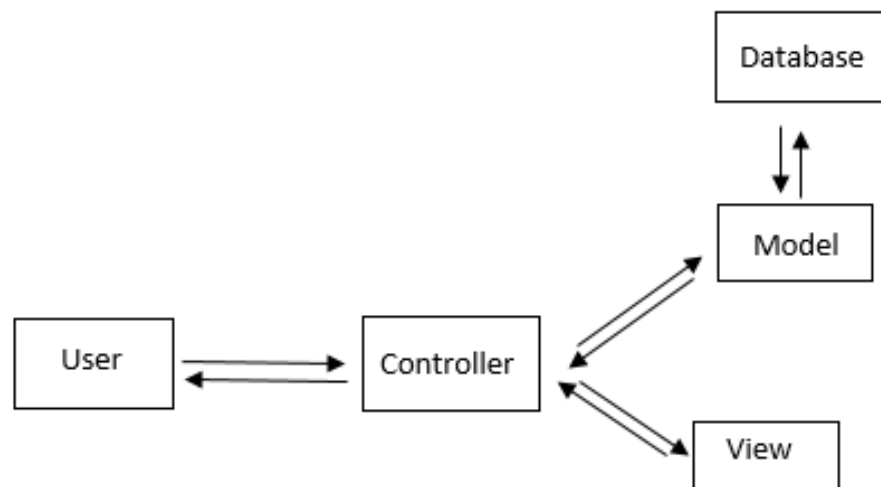
3. View

View adalah informasi yang disajikan untuk *user*, berupa tampilan atau *user interface*. *View* umumnya adalah tampilan sebuah halaman web itu sendiri, tetapi dalam *CodeIgniter*, *view* dapat juga menjadi bagian-bagian atau penggalan-penggalan halaman seperti *header* atau *footer*. *View* dapat juga sebagai halaman RSS, atau tipe-tipe halaman lainnya (Pribadi, 2016).

4. Controller

Controller bertugas sebagai penghubung antara *Model*, *View*, dan beberapa *resource* lainnya yang dibutuhkan untuk memproses *HTTP request* untuk meng-*generate* sebuah halaman web (Priadi, 2016).

CodeIgniter adalah *framework* PHP yang memakai sistem *Model View Controller* (*MVC*). Jadi secara garis besar ada tiga komponen utama, yaitu *Model*, *View* dan *Controller*. Gambaran sederhana tentang *MVC* yang diterapkan *CodeIgniter* dan bagaimana hubungan antara ketiganya disajikan pada Gambar 2.1.



Gambar 2.1. Hubungan antara MVC di CodeIgniter (Priadi, 2016)

Keterangan Gambar 2.1.

1. *User* melakukan *request* suatu halaman kepada aplikasi
2. *Request* diterima oleh *Controller*

3. Untuk memproses data, *Controller* memanggil *Model*. Jika diperlukan *query* ke *database*, maka juga dilakukan di *Model*. Hasil *query* dan pengolahan data di *Model* dikembalikan ke *Controller*.
4. Untuk menampilkan data yang dihasilkan *Model*, *Controller* memanggil *View*, mengolah data di *View* dan mengembalikan hasilnya ke *User*.

2.5. Bootstrap

Bootstrap adalah sebuah *framework* CSS. *Bootstrap* digunakan untuk mendesain sebuah *website*. *Bootstrap* memiliki tampilan yang *responsive* dan memiliki banyak kelebihan dari pada menggunakan CSS standar (Dyazin, 2015).

Bootstrap menurut Alatas (2013) merupakan *framework* ataupun *tools* untuk membuat aplikasi web ataupun situs web *responsive* secara cepat, mudah dan gratis. *Bootstrap* terdiri dari CSS dan HTML untuk menghasilkan *Grid*, *Layout*, *Typography*, *Table*, *Form*, *Navigation*, dan lain-lain. Di dalam *Bootstrap* juga sudah terdapat *jQuery plugins* untuk menghasilkan komponen UI yang cantik seperti *Transitions*, *Modal*, *Dropdown*, *Scrollspy*, *Tooltip*, *Tab*, *Popover*, *Alert*, *Button*, *Carousel* dan lain-lain. Dengan bantuan *Bootstrap*, kita bisa membuat *responsive website* dengan cepat dan mudah dan dapat berjalan sempurna pada *browser-browser* populer seperti *Chrome*, *Firefox*, *Safari*, *Opera* dan *Internet Explorer*.

Bootstrap adalah *framework front-end* yang intuitif dan *powerful* untuk pengembangan aplikasi web yang lebih cepat dan mudah. *Bootstrap* menggunakan HTML, CSS dan *Javascript*. *Bootstrap* dikembangkan oleh Mark Otto dan Jacob

Thornton dari *Twitter*. *Framework* ini diluncurkan sebagai produk *open source* pada Agustus 2011 di *Github*. *Bootstrap* memiliki fitur-fitur komponen *interface* yang bagus seperti *Typography, Forms, Buttons, Tables, Navigations, Dropdowns, Alerts, Modals, Tabs, Accordion, Carousel* dan lain sebagainya (Enterprise, 2016).

2.6. Unified Modelling Language (UML)

Menurut Shalahudin dan Rosa (2013) UML merupakan bahasa *visual* untuk *pemodelan* dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. UML muncul karena adanya kebutuhan *pemodelan* visual untuk menspesifikasikan, menggambarkan, membangun dan mendokumentasi dari sistem perangkat lunak. Menurut Sholiq (2006) ada beberapa diagram yang disediakan dalam UML antara lain.

2.6.1. Diagram use case (use case diagram)

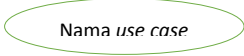




Menurut Whitten (2007), diagram *use case* adalah diagram yang menggambarkan interaksi antara sistem dengan sistem eksternal dan pengguna. Dengan kata lain, secara grafis menggambarkan siapa yang akan menggunakan sistem dan dengan cara apa pengguna mengharapkan untuk berinteraksi dengan sistem.

Diagram *use case* menyajikan interaksi antara *use case* dan aktor. Dimana, aktor dapat berupa orang, peralatan, atau sistem lain yang berinteraksi dengan sistem yang dibangun. *Use case* menggambarkan fungsionalitas sistem atau persyaratan-

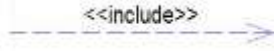
persyaratan yang harus dipenuhi sistem dari pandangan pemakai. *Use case* adalah bagian tingkat tinggi dari fungsionalitas yang disediakan oleh sistem. Dengan kata lain, *use case* menggambarkan bagaimana seseorang menggunakan sistem (Sholiq, 2006).

Simbol-simbol pada *use case diagram* dapat dilihat pada Tabel 2.1.

Tabel 2.1 Simbol-simbol *use case diagram* (Shalahuddin dan Rosa, 2013)

Simbol	Deskripsi
<p><i>Use case</i></p> 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit dan aktor.
<p>Aktor / <i>actor</i></p> 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi.
<p>Asosiasi / <i>association</i></p> 	Komunikasi antar aktor dan <i>use case</i> yang berpartisipasi.
<p>Ekstensi / <i>extend</i> <></p> 	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambah dapat berdiri sendiri walau tanpa <i>use case</i> tambahan.
<p>Generalisasi / <i>generalization</i></p> 	Hubungan generalisasi dan spesialisasi antara dua buah <i>use case</i> yang mana fungsi yang satu lebih umum dari yang lainnya.

Tabel 2.1 Simbol-simbol *use case diagram* (Lanjutan)

<p>Menggunakan <i>include</i></p> 	<p>Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan memerlukan <i>use case</i> ini untuk menjalankan fungsinya.</p>
---	--



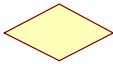
2.6.2. Diagram aktivitas (*activity diagram*)

Menurut Whitten (2007) *activity diagram* adalah sebuah diagram yang dapat digunakan untuk menggambarkan secara grafis aliran proses bisnis, langkah-langkah sebuah *use case* atau logika *behavior* (metode) objek.



Diagram aktivitas menggambarkan aliran fungsional sistem. Dapat digunakan untuk menggambarkan aliran kejadian dalam *use case*. Diagram aktivitas adalah cara lainnya untuk memodelkan aliran kejadian (Sholiq, 2006).

Simbol-simbol pada *activity diagram* dapat dilihat pada Tabel 2.2.

Tabel 2.2 simbol-simbol *activity diagram* (Shalahuddin dan Rosa, 2013)

Simbol	Keterangan
<p>Status awal</p> 	<p>Status awal aktivitas pada sebuah diagram aktivitas memiliki sebuah status awal.</p>
<p>Aktivitas</p> 	<p>Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.</p>
<p>Percabangan / <i>join</i></p> 	<p>Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.</p>

Tabel 2.2 Simbol-simbol *activity diagram* (Lanjutan)


Penggabungan / <i>join</i> 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.

2.6.3. Diagram sekuensial (*Sequence diagram*)




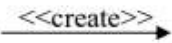
Sequence diagram menggambarkan perilaku objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek (Shalahuddin dan Rosa, 2013).

Simbol-simbol pada *Sequence diagram* dapat dilihat pada Tabel 2.3

Tabel 2.3 Simbol-simbol *Sequence diagram* (Shalahuddin dan Rosa, 2013)

Simbol	Keterangan
Aktor 	Orang, proses atau sistem yang lain yang berinteraksi dengan sistem <i>informasi</i> yang akan dibuat di sistem itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tetapi aktor belum tentu menggunakan orang; biasanya dinyatakan menggunakan kata benda di awal <i>frase</i> nama aktor

Tabel 2.3 Simbol-simbol *sequence diagram* (Lanjutan)

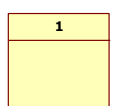

Garis hidup / <i>Life line</i> 	Menyatakan hidup suatu objek
Objek 	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan
Pesan tipe <i>create</i> 	Objek yang lain, arah panah mengarah pada objek yang di buat

2.6.4. Diagram kelas (*class diagram*)






Diagram kelas menunjukkan interaksi antar kelas dalam sistem. Sebuah kelas pada diagram kelas dibuat untuk setiap tipe obyek pada diagram sekuensial atau diagram kolaborasi (Sholiq, 2006).

Simbol-simbol pada *class diagram* dapat dilihat pada Tabel 2.4.

Tabel 2.4 Simbol-simbol *class diagram* (Shalahuddin dan Rosa, 2013)

Simbol	Keterangan
Kelas 	Kelas pada struktur sistem.
Antarmuka / interface 	Sama dengan konsep interface dalam pemrograman berorientasi objek.

Tabel 2.4 Simbol-simbol *class diagram* (Lanjutan)

Asosiasi / association 	Relasi antarkelas dengan makna umum, asosiasi biasanya disertai dengan multiplicity.
Asosiasi berarah / directed association 	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan multiplicity.
Generalisasi 	Relasi antarkelas dengan makna generalisasi spesialisasi (umum khusus).
Kebergantungan / dependency 	Relasi antarkelas dengan makna kebergantungan antarkelas
Agregasi / aggregation 	Relasi antarkelas dengan makna semua-bagian.

2.7. Basis Data

Basis data merupakan suatu kumpulan data yang berhubungan secara logis dengan deskripsi data tersebut, yang dirancang untuk memenuhi informasi yang dibutuhkan oleh suatu organisasi. Basis data merupakan tempat penyimpanan data yang besar, dimana dapat digunakan oleh banyak pengguna. Seluruh item basis data tidak lagi dimiliki oleh suatu departemen, melainkan menjadi sumber daya perusahaan yang dapat digunakan bersama (Indrajani, 2014).

Basis data adalah kumpulan *informasi* yang disimpan di dalam komputer secara sistematis sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh *informasi* dari basis data tersebut. Perangkat lunak yang digunakan untuk mengelola dan memanggil *query* basis data disebut dengan sistem manajemen basis data (*database management system / DBMS*) (Subagia, 2016).

2.8. *Extreme Programming*

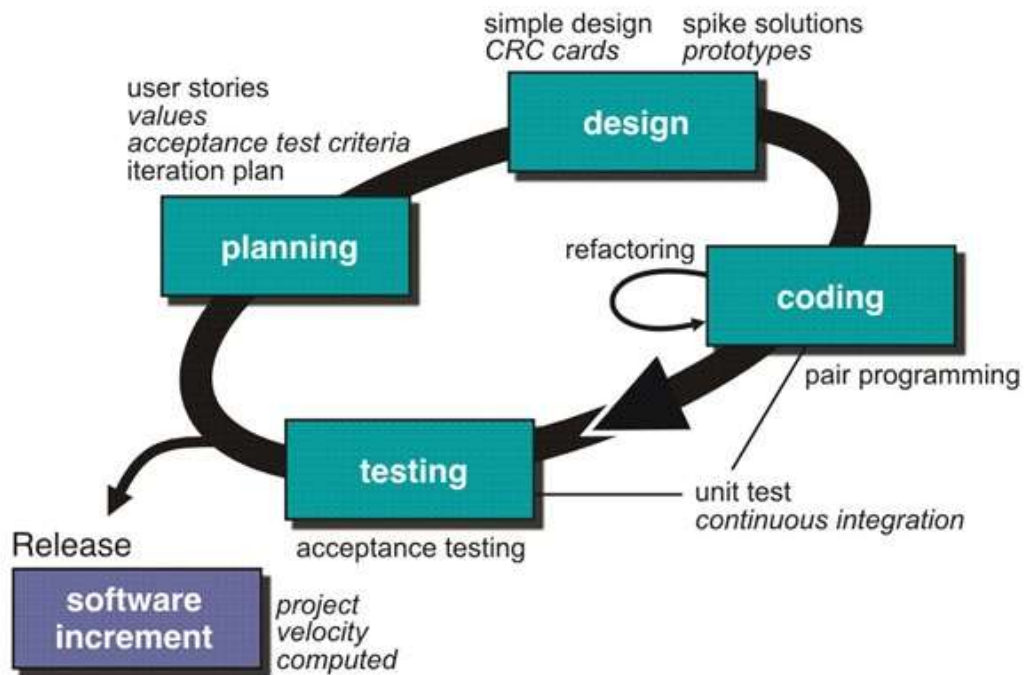
Menurut Widodo dan Massus Subekti (2006), *Extreme Programming (XP)* adalah metode pengembangan perangkat lunak yang ringan dan termasuk salah satu *agile methods* yang dipelopori oleh Kent Beck, Ron Jeffries, dan Ward Cunningham. XP merupakan *agile methods* yang paling banyak digunakan dan menjadi sebuah pendekatan yang sangat terkenal. Sasaran XP adalah tim yang dibentuk berukuran kecil sampai *medium* saja, tidak perlu menggunakan sebuah tim yang besar. Hal ini dimaksudkan untuk menghadapi *requirements* yang tidak jelas maupun terjadinya perubahan-perubahan *requirements* yang sangat cepat.

XP sebagai sebuah metode yang dinamis diperlihatkan dalam empat *values* yang dimilikinya dan keempatnya merupakan dasar-dasar yang diperlukan dalam XP. Kent Beck menyatakan bahwa tujuan jangka pendek individu sering berbenturan dengan tujuan sosial jangka panjang. Karena itu dibuatlah empat *values* yang menjadi aturan, hukuman, dan juga penghargaan. Keempat *values* tersebut adalah sebagai berikut.

1. Komunikasi (*Communication*)
2. Kesederhanaan (*Simplicity*)

3. Umpan Balik (*Feedback*)
4. Keberanian (*Courage*)

Dalam proses pengembangan perangkat lunak menggunakan metode *Extreme Programming* terdiri dari beberapa tahapan. Tahapan-tahapan pengembangan perangkat lunak menggunakan metode *Extreme Programming* ditunjukkan pada Gambar 2.2.



Gambar 2.2 *Extreme Programming* (XP) (Pressman, 2014)

Menurut Pressman (2014) ada empat kerangka kerja yang terdapat dalam metode *Extreme Programming* (XP) yaitu sebagai berikut.

2.8.1. Planning

Planning activities atau yang biasa disebut *planning game*, dimulai dengan *team* mengumpulkan segala sesuatu kebutuhan yang diperlukan dalam pengembangan aplikasi. Setelah mengumpulkan kebutuhan, *team* mulai melakukan penyusunan *user stories* untuk menggambarkan output apa yang diperlukan, fitur, dan kegunaan.

2.8.2. Design

Proses desain pada XP menerapkan prinsip *Keep It Simple (KIS)*. Desain yang sederhana biasanya lebih disukai dibandingkan dengan desain yang kompleks dan desain tersebut harus mengikuti *stories* yang telah dibuat sebelumnya.

2.8.3. Coding

Tahap *coding* merupakan tahapan mengubah desain yang sudah dibuat ke dalam bentuk program. Dalam hal ini, kode program dibuat dengan menggunakan bahasa pemrograman PHP dan menggunakan *framework* Codeigniter. Pada tahap ini, juga terdapat proses *refactoring*. *Refactoring* merupakan proses mengubah sistem perangkat lunak dimana struktur *code* berubah dan menjadi sederhana namun hasil akhir yang dihasilkan sama. Setelah penulis menyelesaikan tahap *planning* dan *design* penulis melakukan pengujian pada setiap unit yang terdapat pada *stories* terlebih dahulu sebelum memulai pengkodean. Kemudian penulis dapat fokus kembali di dalam pengkodean.

2.8.4. Testing

Pada tahap ini *acceptance tests* atau *customer tests* melakukan pengujian terhadap program yang telah selesai. Tahap ini dilakukan untuk mengetahui apakah semua kebutuhan sistem sudah terpenuhi atau belum. Jika setelah *testing* tidak ditemukan kesalahan atau kekurangan, maka bisa dilanjutkan ke proses *release*. Namun, jika masih ada kebutuhan yang belum terpenuhi, maka proses diulang mulai dari proses *planning*, *design*, *coding*, dan melakukan *testing* kembali.

2.8.5. Release

Release merupakan tahapan terakhir yang dilakukan pada metode *Extreme Programming*. *Release* dilakukan jika sudah tidak ada lagi *requirements* atau kebutuhan yang belum terpenuhi. Jika proses setelah proses *testing* terdapat kekurangan, maka akan kembali ke tahap *planning*, *design*, *coding* dan *testing* kembali. Proses ini dilakukan sebanyak yang diperlukan selama kebutuhan yang diminta belum terpenuhi.

2.9. Black Box Testing

Pengujian menggunakan sekumpulan aktifitas validasi, dengan pendekatan *black box testing*. Menurut Shalahuddin dan Rosa (2011), *black box testing* adalah menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan,

dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian kotak hitam dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji yang dibuat untuk melakukan pengujian *black box testing* harus dibuat dengan kasus benar dan kasus salah.

Menurut Pressman (2014), *black box testing* juga disebut pengujian tingkah laku, memusat pada kebutuhan fungsional perangkat lunak. Teknik pengujian *black box* memungkinkan memperoleh serangkaian kondisi masukan yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program. Beberapa jenis kesalahan yang dapat diidentifikasi adalah fungsi tidak benar atau hilang, kesalahan antarmuka, kesalahan pada struktur data (pengaksesan basisdata), kesalahan *performansi*, kesalahan inisialisasi dan akhir program.

III. METODE PENELITIAN

3.1. Waktu dan Tempat Penelitian

Penelitian ini dilakukan pada semester genap Tahun Ajaran 2017/2018 yang bertempat di Ruang Baca Gedung MIPA Terpadu Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung.

3.2. Spesifikasi *Software* dan *Hardware* yang Digunakan

Pengembangan sistem *grader* ini menggunakan satu unit laptop dengan spesifikasi sebagai berikut.

3.2.1. Spesifikasi *software*

Spesifikasi *software* yang digunakan dalam pengembangan sistem ini adalah sebagai berikut .

1. Sistem Operasi Linux Ubuntu 16.04 64 bit
2. Apache *web server*
3. MySQL sebagai *database* manajemen sistem
4. Sublime *Text 3*
5. Google Chrome

3.2.2. Spesifikasi *hardware*

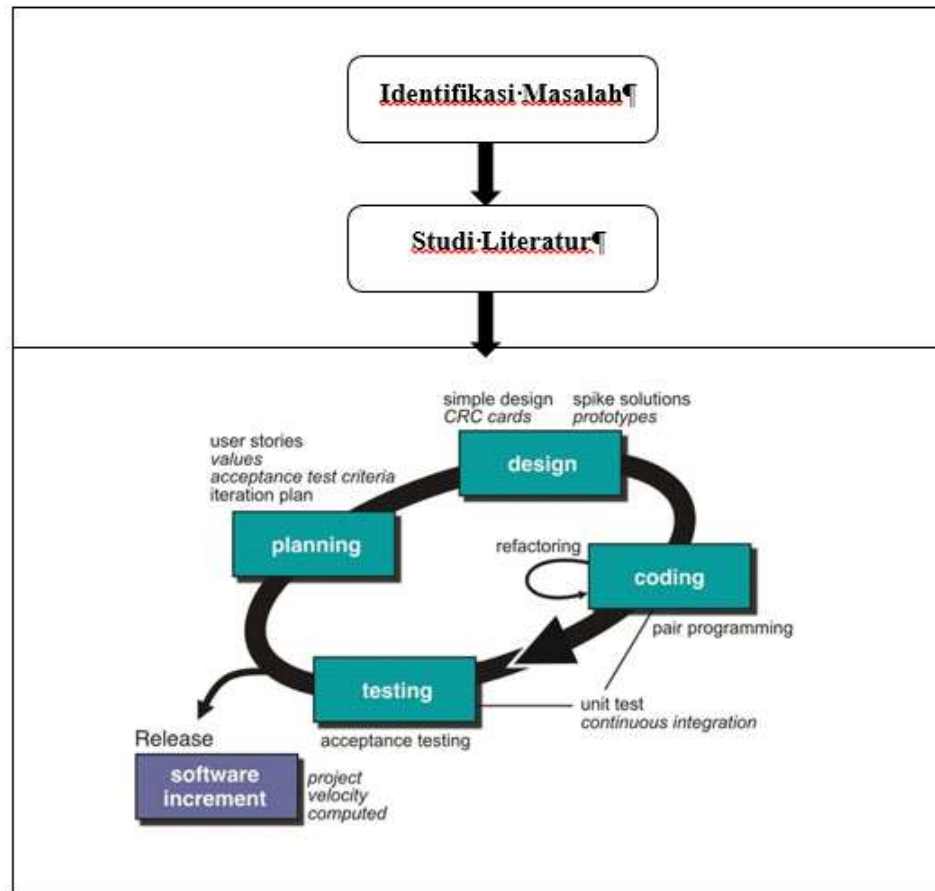
Spesifikasi *hardware* yang digunakan dalam pengembangan sistem ini adalah sebagai berikut.

1. Laptop tipe Toshiba Satellite L-40A
2. Processor Intel(R) Core(TM) i3-3227U CPU @ 1.90GHz, 1901 MHz, 2 Core(s), 4 Logical Processors
3. Memory RAM DDR3 4GB
4. Kapasitas HDD 500GB
5. Display Intel(R) HD Graphics 4000

3.3. Tahapan Penelitian

Tahapan penelitian merupakan suatu bentuk tahap penelitian yang dapat digunakan sebagai pendekatan dalam memecahkan masalah. Tahapan penelitian ini menggunakan pendekatan ilmiah dan memperlihatkan hubungan antar *variable* dalam proses analisisnya. Penelitian ini dibuat dengan menggunakan metode *Extreme Programming (XP)* yang didalamnya terdapat empat macam tahapan untuk penyelesaian masalah yaitu *planning*, *design*, *coding*, dan *testing*.

Alur penelitian yang dilakukan pada penelitian ini didasarkan pada diagram alir metodologi penelitian yang terdapat pada Gambar 3.1.



Gambar 3.1 Alur Penelitian

3.3.1. Tahap Identifikasi Masalah

Langkah awal dari penelitian ini dimulai dengan melakukan kegiatan identifikasi masalah. Tahap ini merupakan tahap paling dasar untuk melakukan identifikasi dan melakukan analisis terhadap permasalahan-permasalahan yang ada, baik proses pembuatan, perbaikan, atau penambahan proses bisnis yang sudah ada.

1. Deskripsi Sistem

Sistem *grader* adalah sebuah pengembangan dari sistem yang sudah digunakan oleh Tim Olimpiade Komputer Indonesia (TOKI) untuk melatih siswa-siswa untuk mempersiapkan diri mengikuti lomba Olimpiade Sains Nasional (OSN). Sistem yang sudah ada memiliki 2 kategori *user*, yaitu *admin* dan *contestant*. Sedangkan pada pengembangan sistem ini akan diberikan tambahan *user* berupa *editor* yang dikhususkan untuk mengelola *contest* atau kelas. Penambahan fitur yang lain adalah menambahkan fitur *export* untuk *scoreboard* agar dapat disimpan dalam bentuk *file excel* (ekstensi .xls/xlsx).

2. Kebutuhan Fungsional

Kebutuhan fungsional sistem adalah sebagai berikut.

- Membuat *contest*/kelas,
- Membuat soal/*problem* yang disertai dengan *testcase*,
- Menambah *user*,
- Melakukan *compile* dan *running* program secara otomatis,
- Menambah *submission* jawaban
- Membuat *clarification*
- Menambah bahasa pemrograman

3. Kebutuhan Non-fungsional

Kebutuhan non-fungsional sistem *grader* adalah:

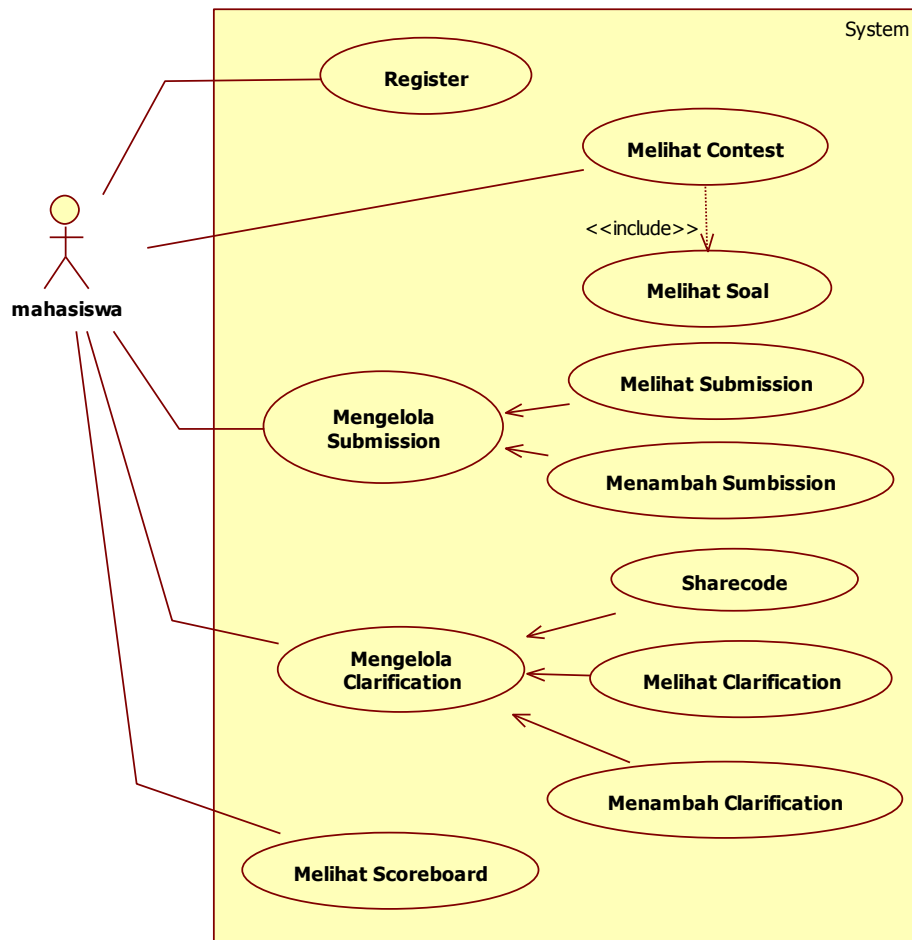
- Keamanan akses pengguna (*Username* dan *Password*),
- *User Interface* yang mudah dipahami.
- Proses *regrade*

3.3.2. Tahap Studi Literatur

Tahap kedua pada bagian pertama dari metodologi penelitian ini adalah studi literatur. Pada tahap ini dilakukan proses pencarian *informasi* yang dibutuhkan untuk sistem mulai dari awal hingga akhir sehingga dapat mengetahui apa saja kebutuhan sistem. Hasil yang diperoleh dari tahap ini berupa *informasi* yang dapat berupa teori-teori dan dapat berupa hasil penelitian yang sudah dilakukan mengenai permasalahan yang akan diteliti. Studi literatur dilakukan dengan mengumpulkan *informasi-informasi* mengenai *grader*, *CodeIgniter*, *Bootstrap*, dan *MOE Contest Environment* baik dari buku maupun dari berbagai jurnal ilmiah yang telah membahas masalah yang telah dijelaskan pada BAB I.

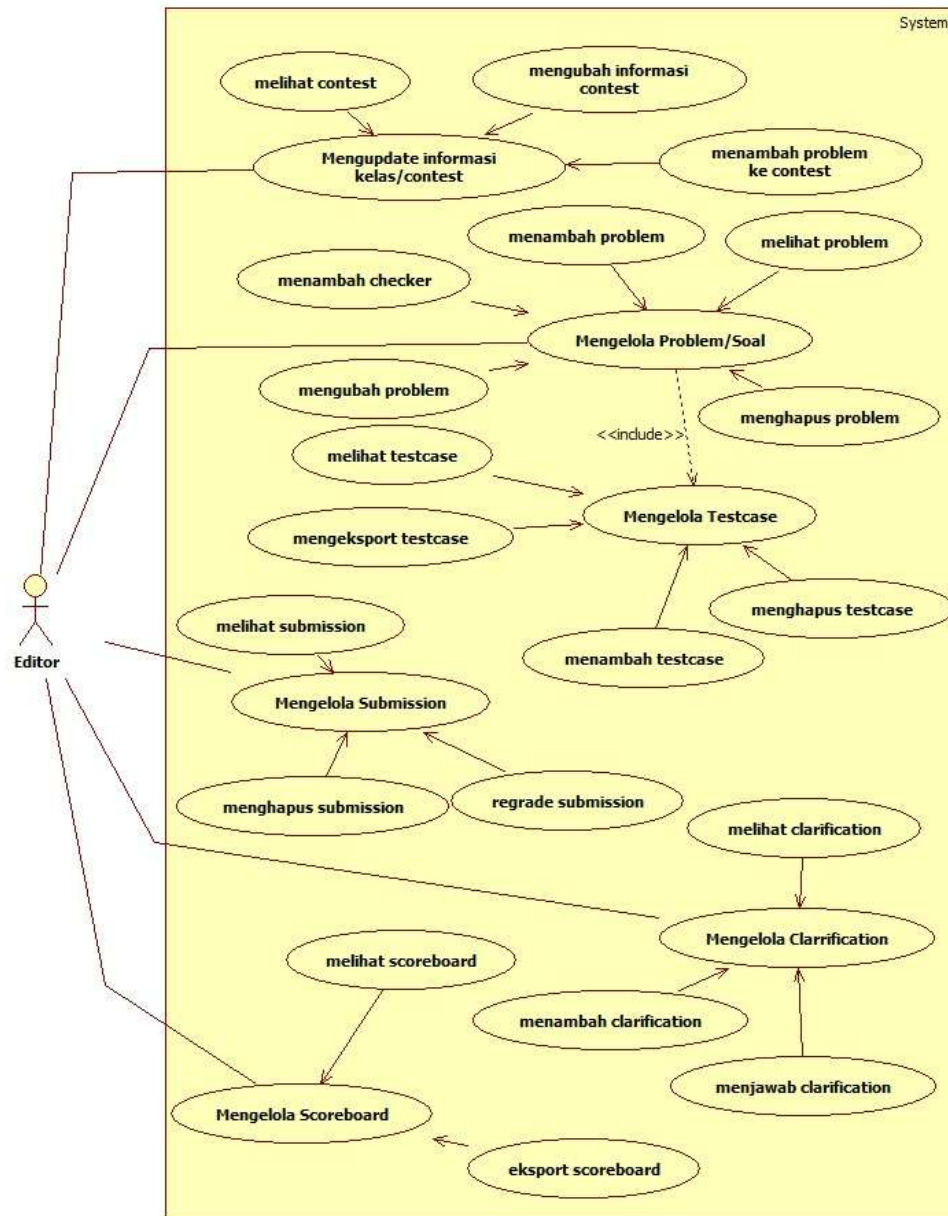
3.3.3. *Planning*

Planning merupakan salah satu tahapan dari perancangan sistem untuk mengumpulkan segala kebutuhan yang diperlukan dalam pengembangan sistem. Kebutuhan tersebut diperlukan untuk menggambarkan bagaimana sistem akan berjalan, menggambarkan *output* apa yang diperlukan, fitur, dan kegunaan sistem. Pada proses *planning*, digunakan *Unified Modelling Language* (UML) diagram dalam bentuk *use case* untuk menggambarkan hubungan antara aktor dengan sistem. *Use Case* diagram digunakan untuk menggambarkan sistem dari sudut pandang pengguna, sehingga *use case* lebih menitik beratkan pada fungsionalitas sistem. Pada sistem ini, terdapat tiga kategori pengguna, yaitu *contestant/user/mahasiswa*, *editor/dosen/asisten* dosen dan *admin*. *User/contestant* dapat melakukan 6 interaksi di dalam sistem. Interaksi tersebut adalah melakukan pendaftaran akun *contestant/mahasiswa*, melihat *contest/kelas*, melihat *problem/soal*, mengelola *submission/jawaban*, mengelola *clarification/pertanyaan*, dan dapat melihat *scoreboard*. *Use case* diagram *user/contestant* dapat dilihat pada Gambar 3.2.



Gambar 3.2 Use case diagram user/contestant/mahasiswa

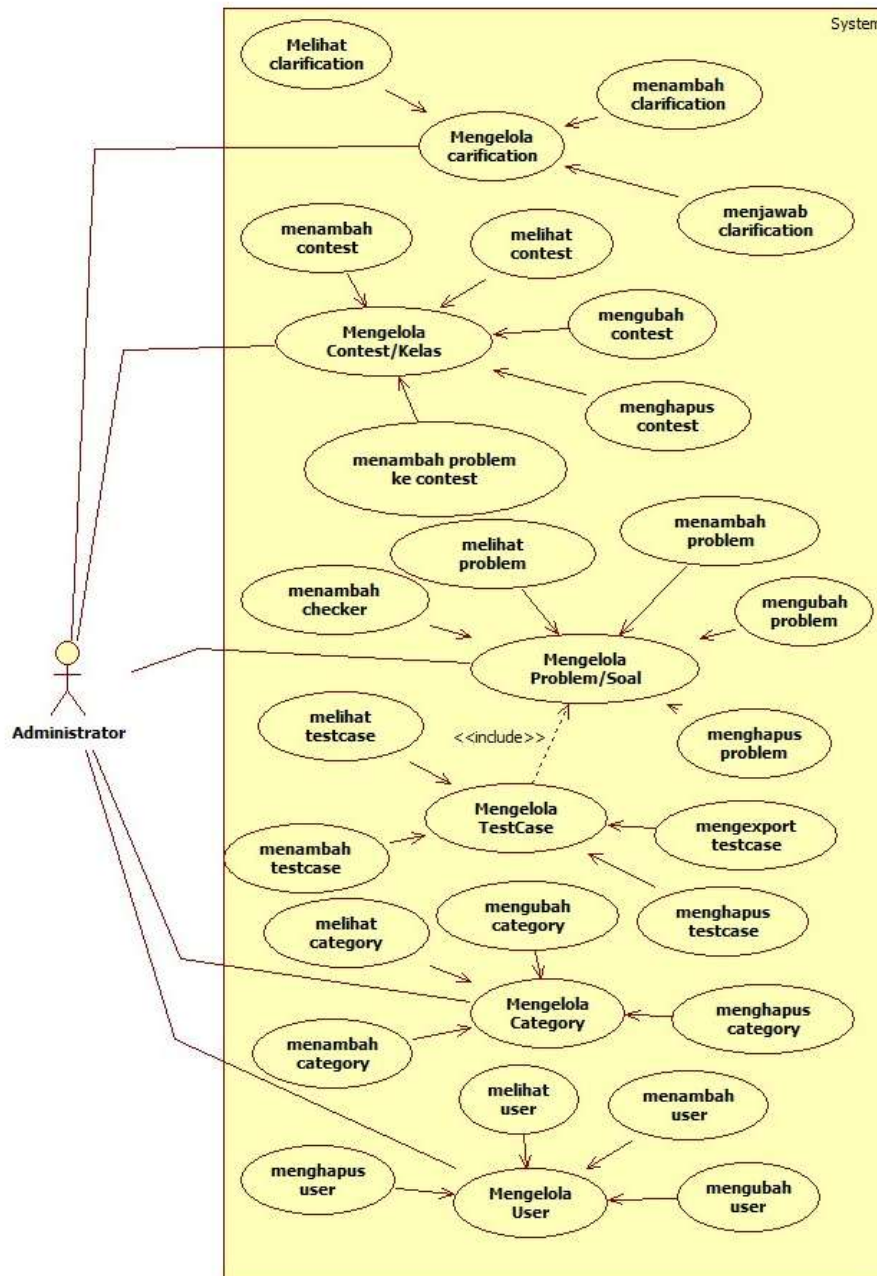
Editor/dosen/asisten dosen dapat melakukan 20 interaksi yang dikelola dalam 6 *usecase* dasar di dalam sistem. Interaksi yang dapat dilakukan oleh *editor* adalah mengubah informasi kelas/*contest*, mengelola *problem/soal*, mengelola *testcase*, mengelola *submission/jawaban*, mengelola *clarification/pertanyaan*, dan mengelola *scoreboard*. Use case diagram *editor/dosen/asisten dosen* dapat dilihat pada Gambar 3.3.



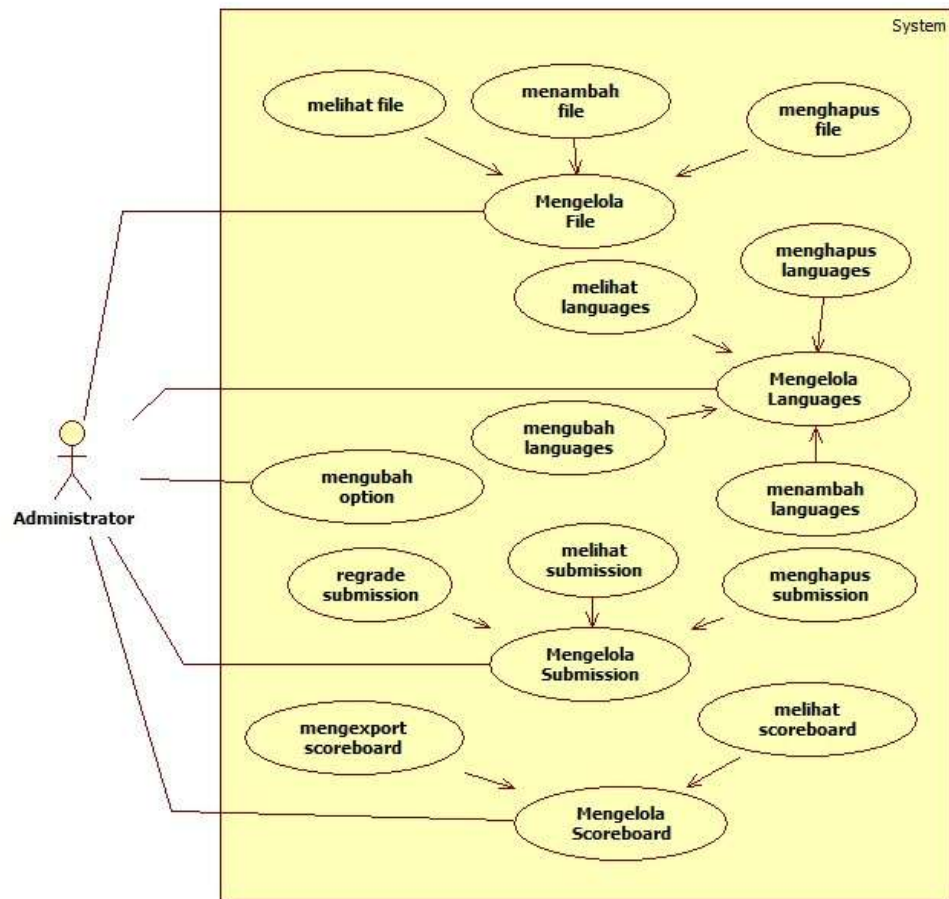
Gambar 3.3 Use case diagram editor/dosen/asisten dosen

Admin dapat melakukan 11 interaksi dasar dan 39 interaksi di dalam sistem. Interaksi yang dapat dilakukan oleh *admin* adalah mengelola *contest*/kelas, mengelola *problem*/soal, mengelola *testcase*, mengelola *languages*, mengelola

category, mengelola *user*, mengelola *file*, mengelola *option/informasi dashboard*, mengelola *submission*, mengelola *clarification*, mengelola *scoreboard*. Use case diagram yang dapat dilakukan oleh *admin* dapat dilihat pada Gambar 3.4.



Gambar 3.4.1 Use case diagram admin bagian 1



Gambar 3.4.2 Use case diagram admin bagian 2

3.3.4. Design

Pada tahap ini dilakukan perancangan desain sistem. Perancangan desain sistem menggunakan diagram *Unified Modelling Language (UML)* dengan menggunakan diagram aktivitas, diagram sekuen, dan kelas diagram. Diagram UML yang dibuat dalam penelitian ini adalah sebagai berikut.

3.3.4.1. Activity Diagram

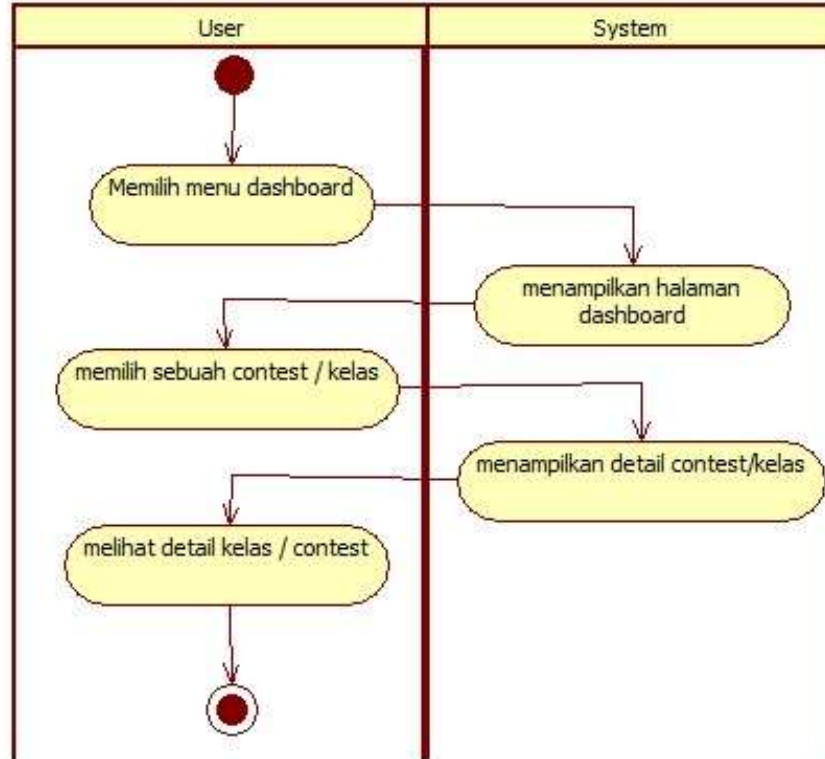
Activity diagram digunakan untuk menggambarkan serangkaian aliran dari aktivitas. *Activity diagram* juga berguna untuk mendeskripsikan aktivitas-aktivitas yang dibentuk dalam satu operasi, sehingga dapat melakukan aktivitas lainnya. Pada sistem ini, *activity diagram* dibagi menjadi 3 kategori pengguna. Ketiga pengguna tersebut adalah sebagai berikut.

1. User atau mahasiswa

Activity diagram user terdiri dari 8 *activity* yaitu sebagai berikut.

A. Activity melihat *contest*/kelas

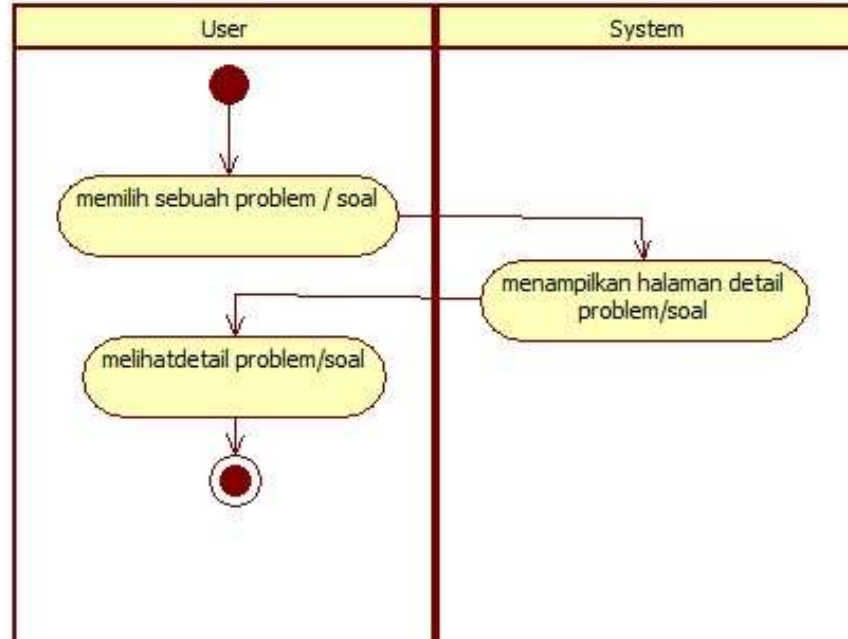
Activity diagram melihat *contest*/kelas dimulai dengan memilih menu *dashboard*, lalu sistem akan menampilkan halaman utama berupa daftar *contest*/kelas yang bisa *user*/mahasiswa ikuti. Pilih sebuah kelas/*contest*, selanjutnya sistem akan menampilkan *detail* dari kelas/*contest* tersebut. *Activity diagram* melihat *contest*/kelas disajikan pada Gambar 3.5.



Gambar 3.5 Activity diagram melihat contest/Kelas

B. Activity melihat problem/soal

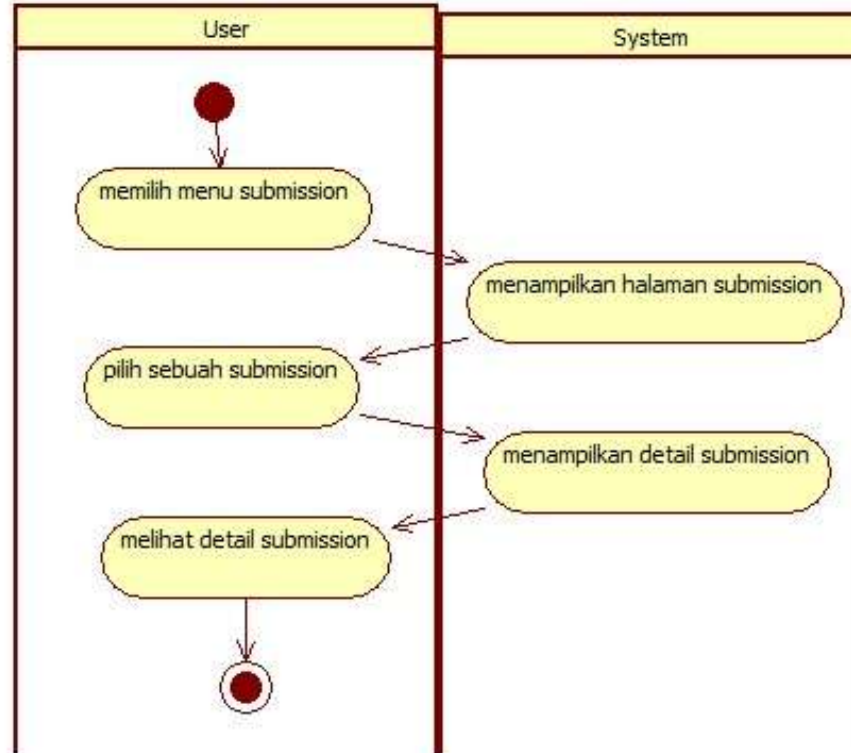
Activity diagram melihat *problem/soal* dimulai dengan memilih sebuah *problem/soal* pada halaman *detail contest/kelas*, lalu sistem akan menampilkan halaman *detail problem/soal* tersebut. Activity diagram melihat *problem/soal* disajikan pada Gambar 3.6.



Gambar 3.6 Activity diagram melihat problem/Soal

C. Activity melihat *submission*/jawaban

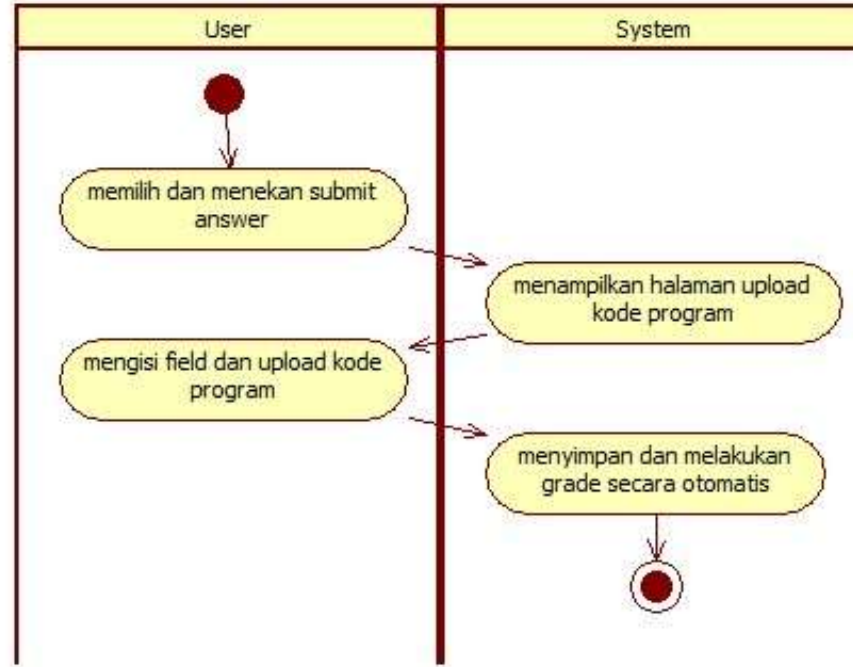
Activity diagram melihat *submission*/jawaban dimulai dengan memilih menu *submission*, lalu sistem akan menampilkan halaman utama *submission* dari sebuah *contest*/kelas yang sudah dipilih. Pilih sebuah jawaban/*submission*, selanjutnya sistem akan menampilkan *detail* status *submission*/jawaban dan *source code* dari jawaban/*submission* tersebut. *Activity diagram* melihat *submission*/jawaban disajikan pada Gambar 3.7.



Gambar 3.7 Activity diagram melihat *submission*/jawaban

D. Activity menambah *submission*

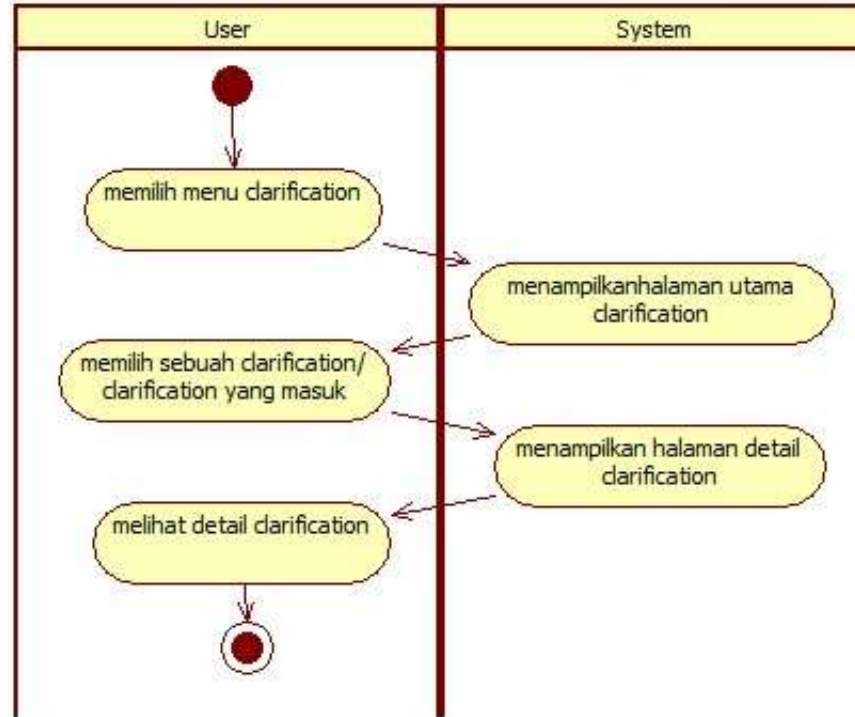
Activity diagram menambah *submission*/jawaban dimulai dengan memilih dan menekan *link submit answer* pada halaman *detail problem/soal*, lalu sistem akan menampilkan *form* unggah kode program. Setelah *user* mengisi *form* dan mengunggah kode program, sistem akan menyimpan *submission*/jawaban dari *user*/mahasiswa dan akan melakukan *compile* dan *running* program secara otomatis. Activity diagram menambah *submission* disajikan pada Gambar 3.8.



Gambar 3.8 *Activity diagram* tambah *submission*

E. *Activity* melihat *clarification*

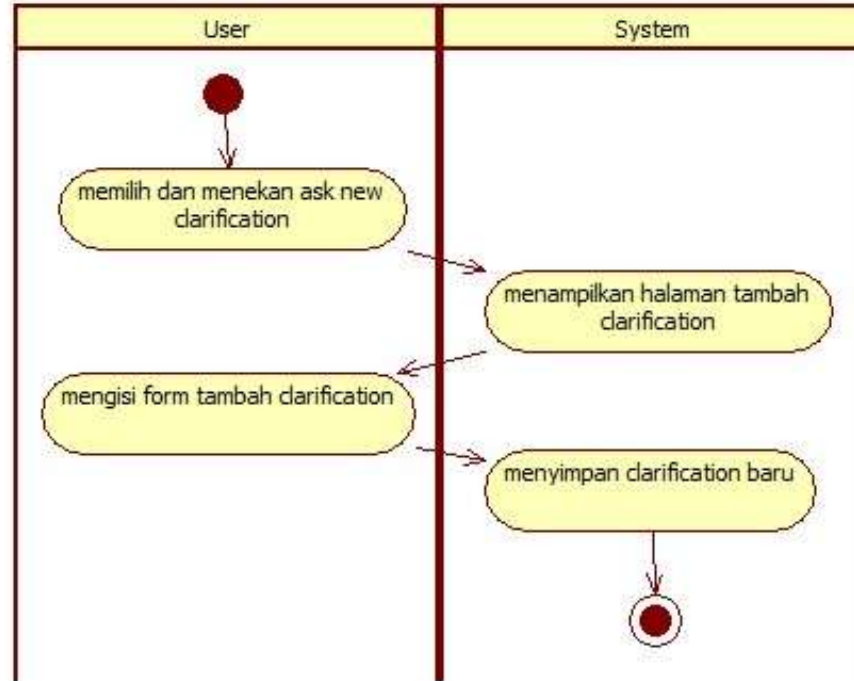
Activity diagram melihat *clarification* dimulai dengan memilih menu *clarification*, lalu sistem akan menampilkan halaman utama *clarification*. Pilih sebuah *clarification* atau *clarification* masuk, selanjutnya sistem akan menampilkan rincian dari *clarification* tersebut. *Activity diagram* melihat *clarification* disajikan pada Gambar 3.9.



Gambar 3.9 *Activity diagram* melihat *clarification*

F. *Activity* menambah *clarification*

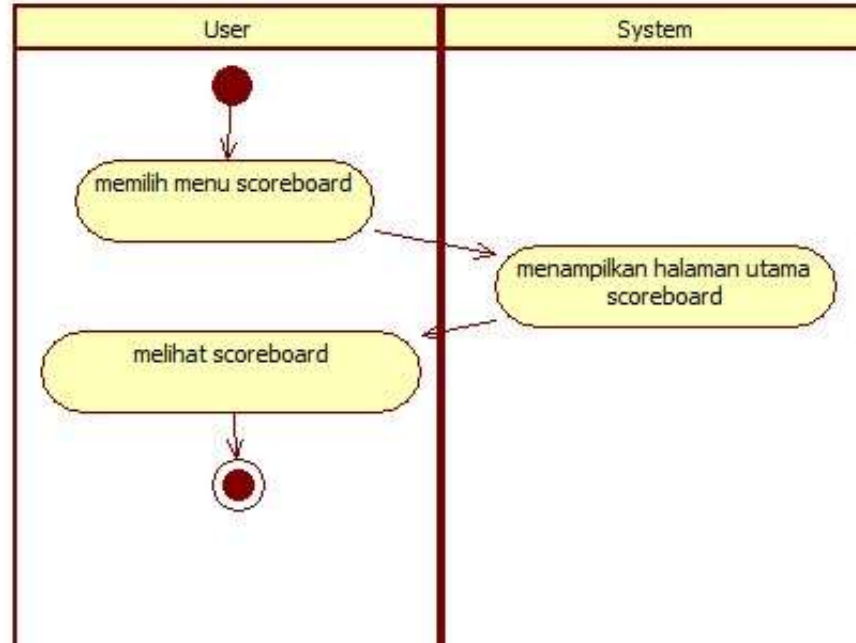
Activity menambah *clarification* dimulai dengan memilih *link ask new clarification* pada halaman utama *clarification*. Selanjutnya, sistem akan menampilkan halaman dan *form* tambah *clarification*. *User*/mahasiswa mengisi *form* pada halaman tambah *clarification* tersebut. Setelah selesai, sistem akan menyimpan *clarification* yang dibuat. *Activity diagram* menambah *clarification* disajikan pada Gambar 3.10.



Gambar 3.10 *Activity diagram* tambah clarification

G. *Activity* melihat scoreboard

Activity diagram melihat scoreboard dimulai dengan memilih menu scoreboard pada navigarion bar, lalu sistem akan menampilkan halaman utama scoreboard. *Activity diagram* melihat scoreboard disajikan pada Gambar 3.11.



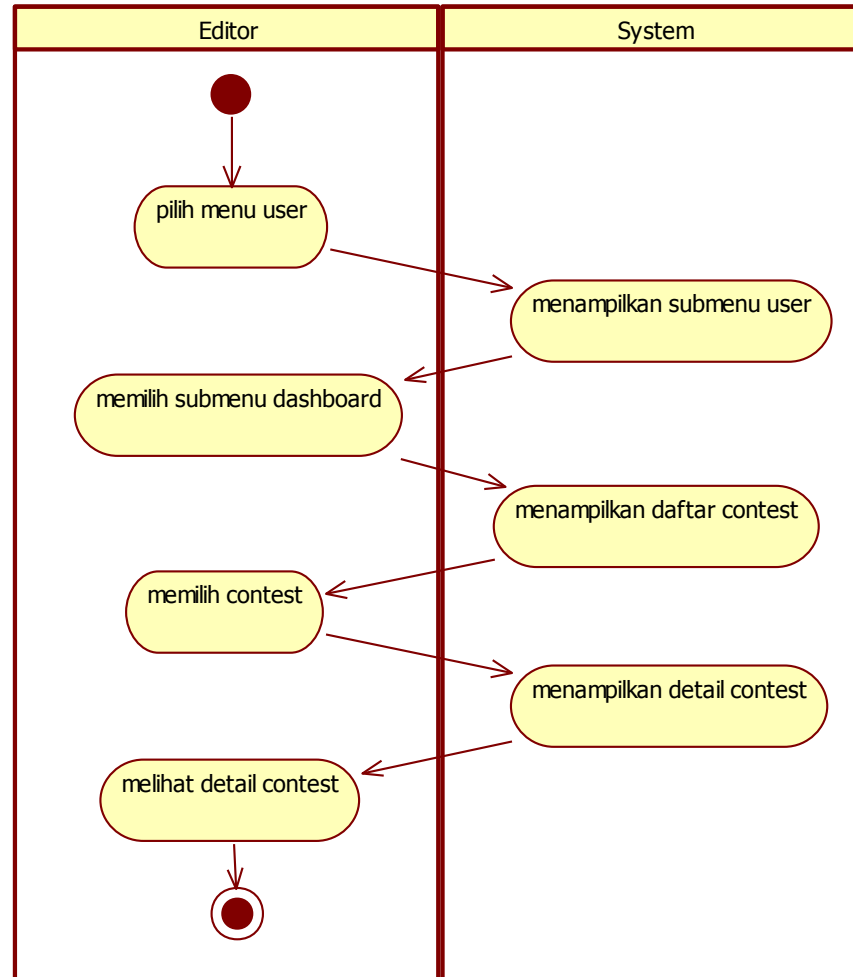
Gambar 3.11 *Activity diagram* melihat scoreboard

2. *Editor* atau dosen

Editor atau dosen dapat melakukan 20 transaksi pada sistem. Transaksi tersebut antara lain adalah sebagai berikut.

A. *Activity* melihat *contest*/kelas

Activity melihat *contest*/kelas dimulai dengan memilih menu *manage* yang ada di *navigation bar* paling kiri dan pilih submenu *contest*. Setelah selesai, sistem akan menampilkan halaman daftar *contest*/kelas. *Activity diagram* melihat *contest*/kelas disajikan pada Gambar 3.12.

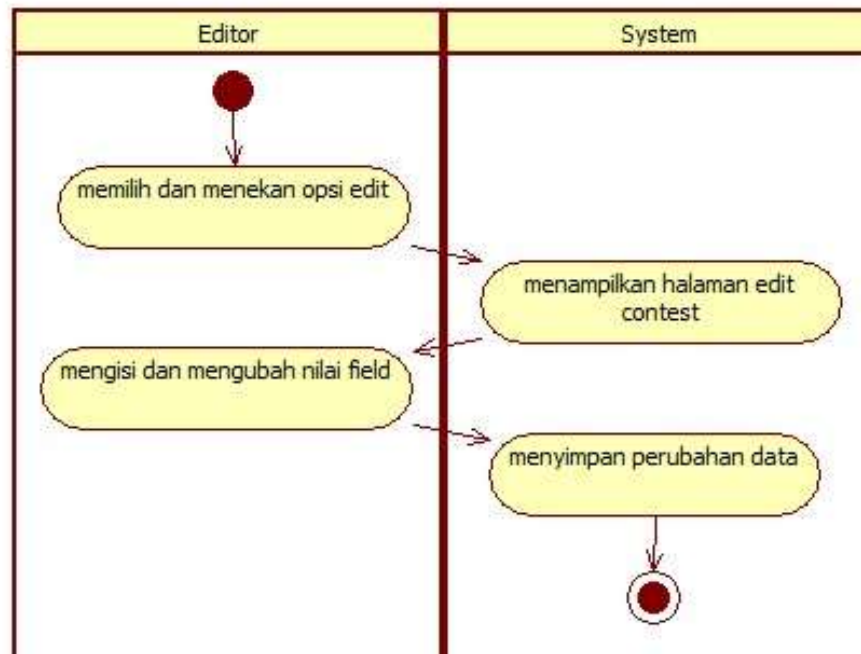


Gambar 3.12 *Activity diagram* melihat *contest*/kelas

B. *Activity diagram* mengubah *contest*/kelas

Activity mengubah *contest*/kelas dimulai dengan memilih dan menekan *opsi edit* pada sebuah *contest*/kelas yang akan diubah pengaturannya. Setelah memilih, sistem akan menampilkan halaman *edit contest*/kelas. *Editor* mengisi dan mengubah *informasi* yang ada pada *form* ubah data *informasi*. Setelah *editor* selesai mengubah *informasi*, sistem akan menyimpan hasil

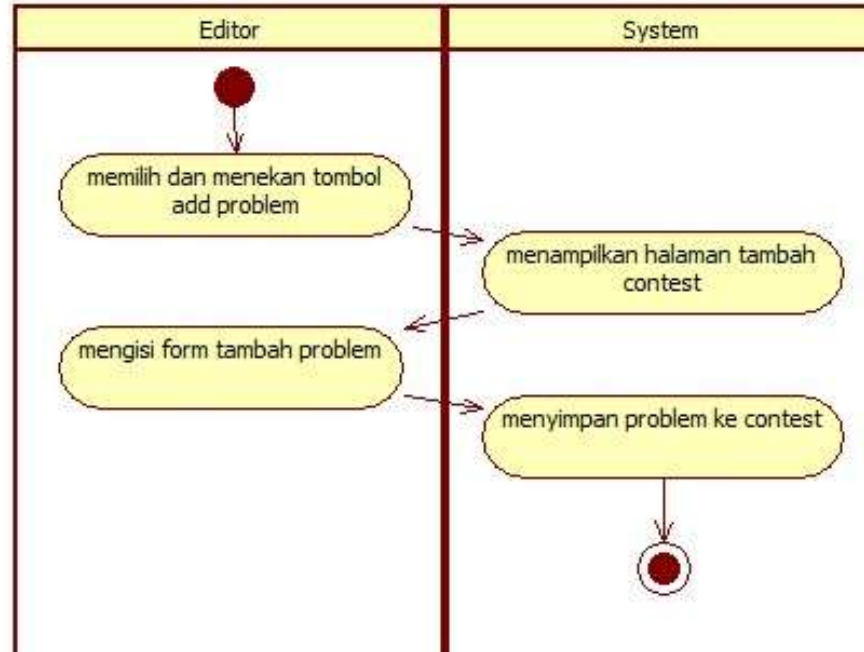
perubahan yang dilakukan oleh *editor*. *Activity* melakukan *edit contest* disajikan pada Gambar 3.13.



Gambar 3.13 *Activity diagram* mengubah *contest*/kelas

C. *Activity* menambah *problem*/soal ke dalam *contest*/kelas

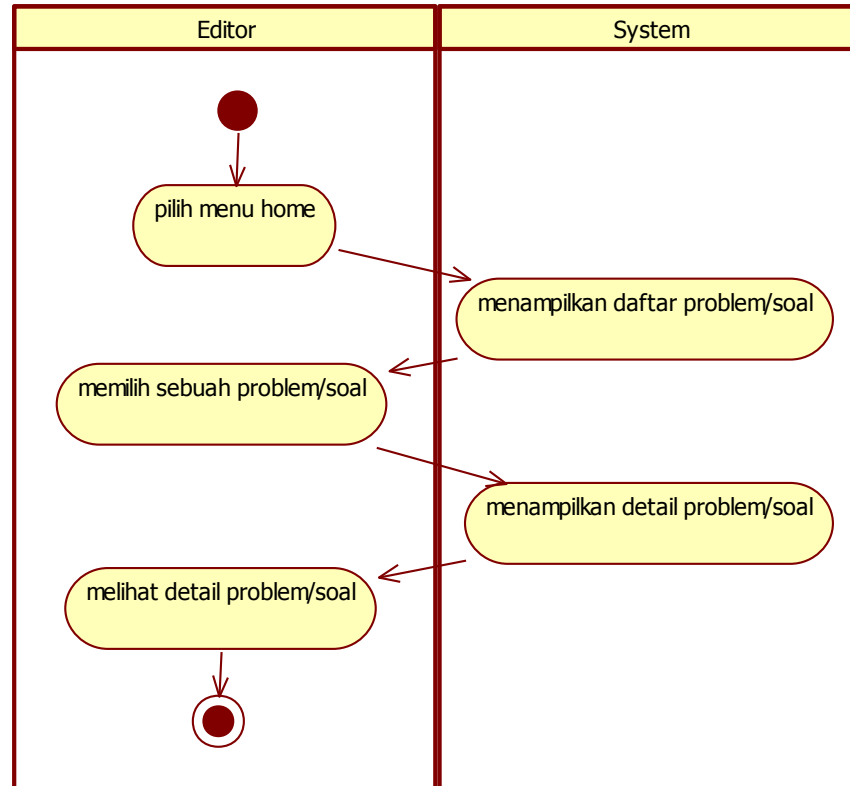
Activity menambah *problem*/soal ke dalam *contest*/kelas dimulai dengan memilih *opsi problem* pada halaman *detail contest*. Sistem menampilkan halaman tambah *problem*, *editor* memilih *problem*/soal yang sudah ditambahkan melalui *form* tambah *problem*/soal. Setelah selesai, sistem akan menyimpan *problem*/soal yang dipilih ke dalam *contest*. *Activity* menambah *problem*/soal ke dalam *contest*/kelas disajikan pada Gambar 3.14.



Gambar 3.14 *Activity diagram* menambah *problem* ke *contest*/kelas

D. *Activity* melihat *problem*/soal

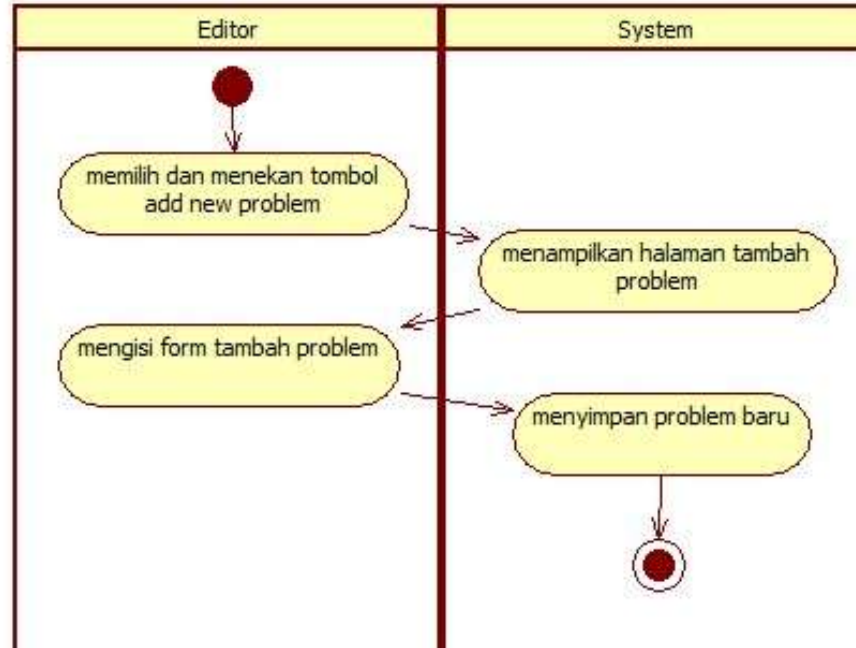
Activity melihat *problem*/soal dimulai dengan memilih menu *home* pada *navigation bar*, dan memilih submenu *problem*, sistem akan mengalihkan ke halaman daftar *problem*/soal. *Editor* memilih sebuah *problem*/soal, selanjutnya sistem akan menampilkan halaman *detail problem*/soal. *Activity* melihat *problem*/soal disajikan pada Gambar 3.15.



Gambar 3.15 Activity diagram melihat problem/soal

E. Activity menambah problem/soal

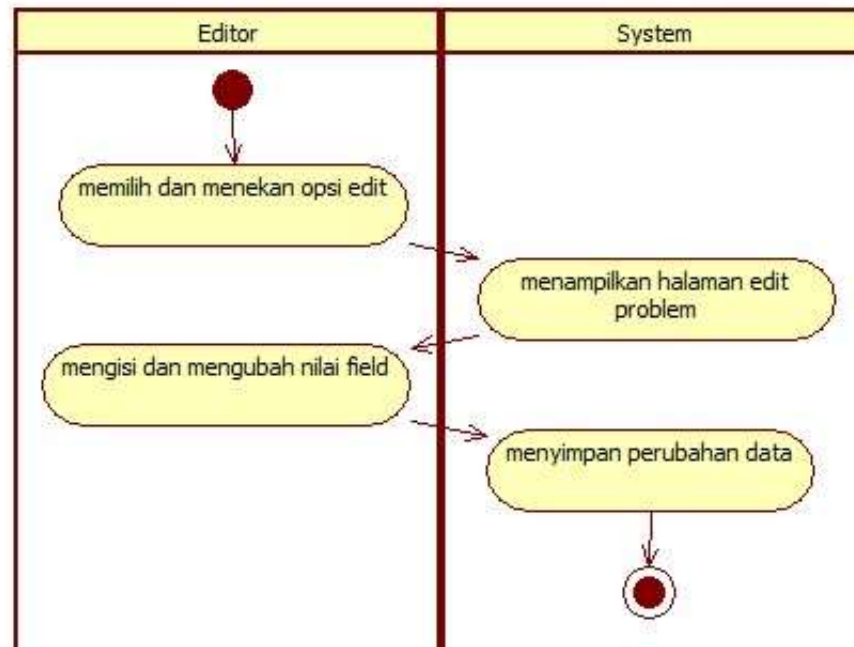
Activity menambah *problem/soal* dimulai dengan memilih *link add new problem* pada halaman *problem*. Sistem akan menampilkan halaman tambah *problem/soal*. Editor mengisi informasi yang akan diberikan untuk soal seperti deskripsi soal, *memory limit* dan *time limit* eksekusi program, batasan-batasan, serta contoh dan *format input output*. Activity menambah *problem/soal* disajikan pada Gambar 3.16.



Gambar 3.16 *Activity diagram* menambah *problem/soal*

F. *Activity* mengubah *problem/soal*

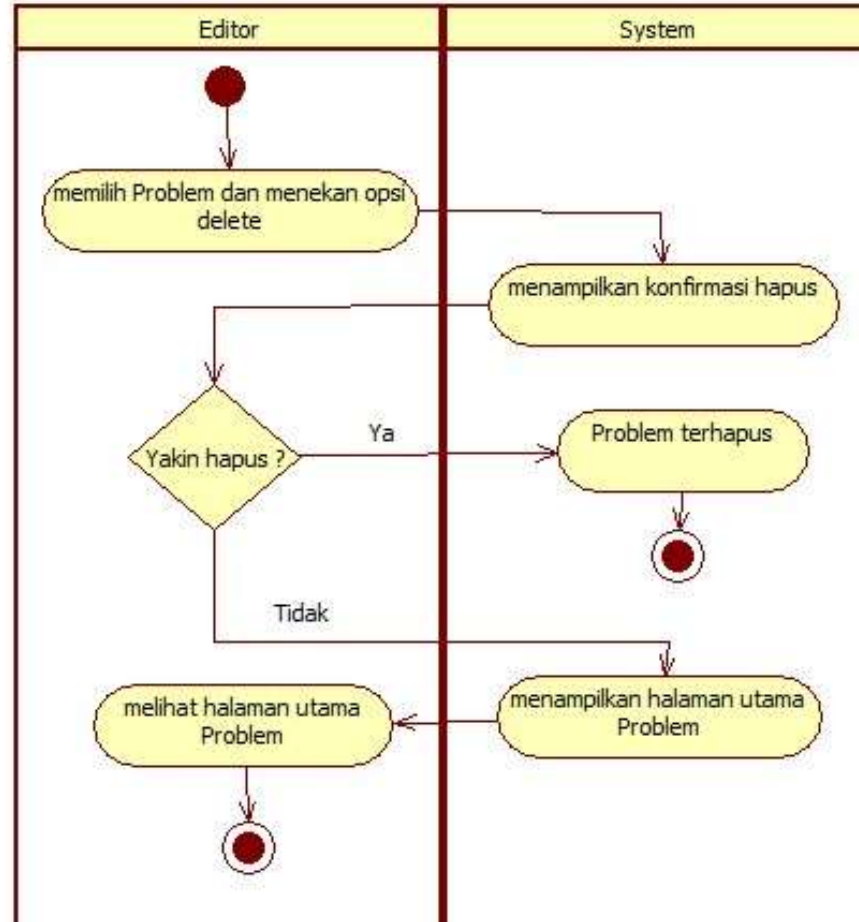
Activity mengubah *problem/soal* dimulai dengan memilih *opsi edit* pada sebuah *problem/soal*. Selanjutnya sistem akan menampilkan halaman *edit problem*. *Editor* mengisi dan melakukan perubahan terhadap *informasi* yang akan diubah seperti judul *problem/soal*, deskripsi *problem/soal*, *memory* dan *time limit*, dan sebagainya. *Activity* mengubah *problem/soal* disajikan pada Gambar 3.17.



Gambar 3.17 Activity diagram mengubah *problem/soal*

G. Activity menghapus *problem/soal*

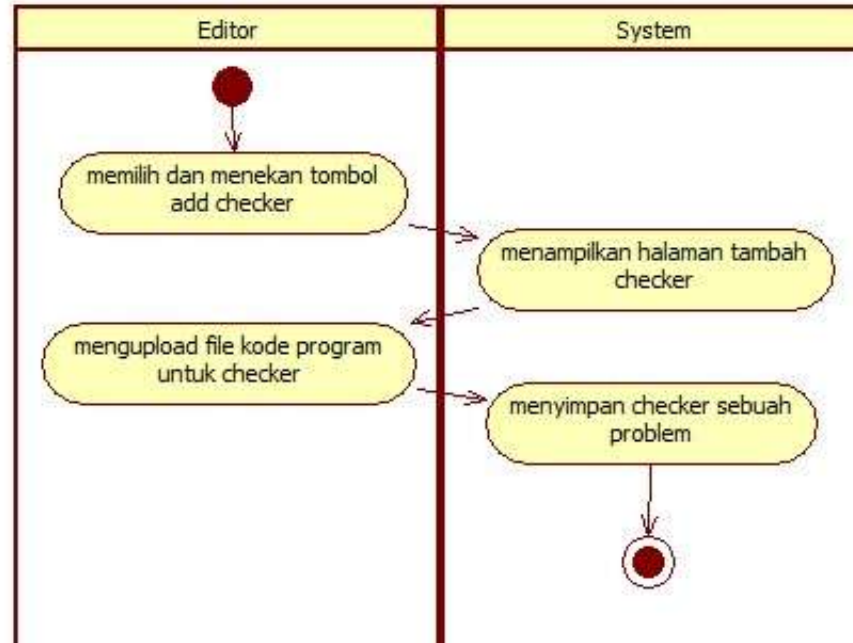
Activity menghapus *problem/soal* dimulai dengan memilih *problem* dan memilih *opsi delete*. Sistem akan menampilkan konfirmasi untuk menghapus *problem/soal*. Jika *editor* memilih ya, maka *problem* akan terhapus, jika *editor* memilih tidak, maka sistem akan langsung kembali ke halaman *problem/soal*. *Activity* menghapus *problem/soal* disajikan pada Gambar 3.18.



Gambar 3.18 Activity diagram menghapus problem/soal

H. Activity menambah checker

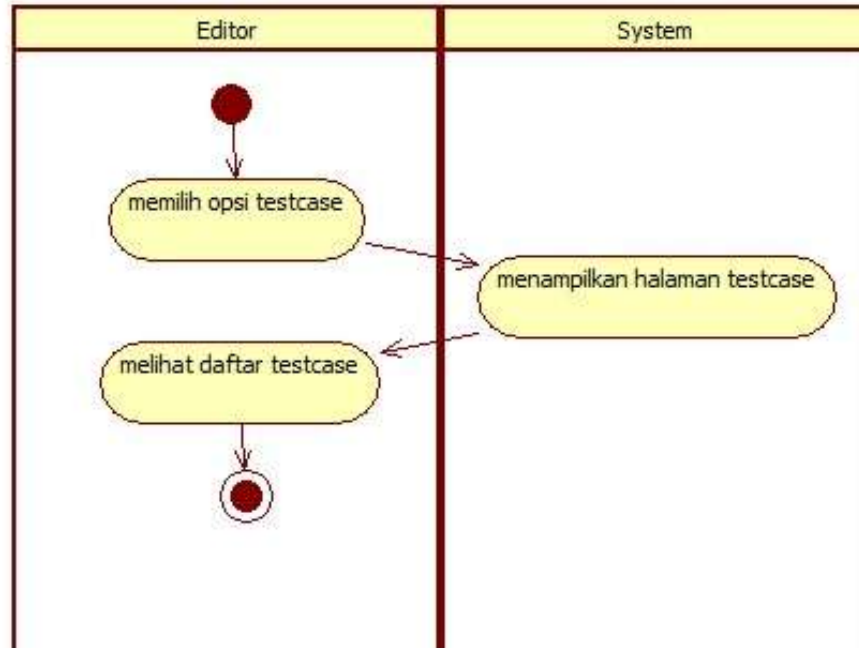
Activity menambah checker dimulai dengan memilih opsi checker pada halaman problem. Sistem akan menampilkan halaman tambah checker. Editor mengisi dan mengunggah berkas source code sebagai checker untuk problem/soal yang akan diberikan. Activity menambah checker disajikan pada Gambar 3.19.



Gambar 3.19 *Activity diagram* menambah checker

I. *Activity* melihat testcase

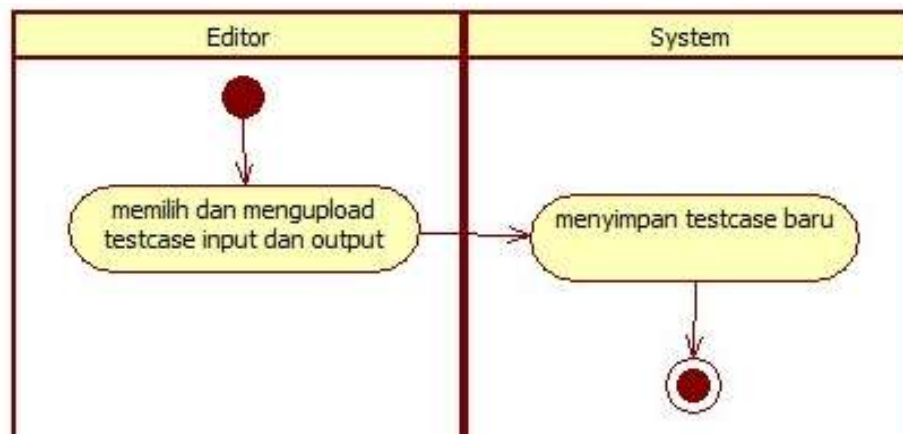
Activity melihat testcase dimulai dengan memilih opsi testcase pada halaman problem. Sistem akan menampilkan halaman daftar testcase. *Activity* melihat testcase disajikan pada Gambar 3.20.



Gambar 3.20 Activity diagram melihat testcase

J. Activity menambah testcase

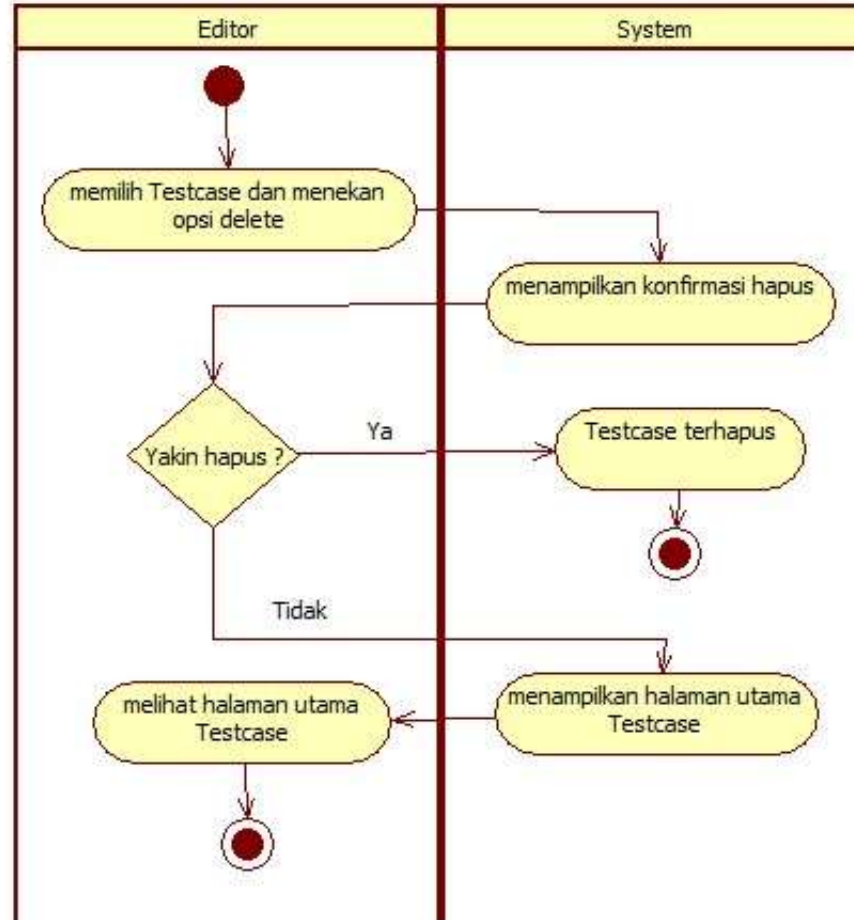
Activity menambah testcase dimulai dengan memilih dan mengunggah berkas testcase input dan testcase output pada halaman utama testcase. Setelah editor selesai mengunggah berkas testcase, sistem akan menyimpan testcase dari problem/soal. Activity menambah testcase disajikan pada Gambar 3.21.



Gambar 3.21 *Activity diagram* menambah *testcase*

K. *Activity* menghapus *testcase*

Activity menghapus *testcase* dimulai dengan memilih *testcase* dan memilih *opsi delete*. Sistem akan menampilkan konfirmasi untuk menghapus *testcase*. Jika *editor* memilih ya, maka *testcase* akan terhapus, jika *editor* memilih tidak, maka sistem akan langsung kembali ke halaman *testcase*. *Activity* menghapus *testcase* disajikan pada Gambar 3.22.

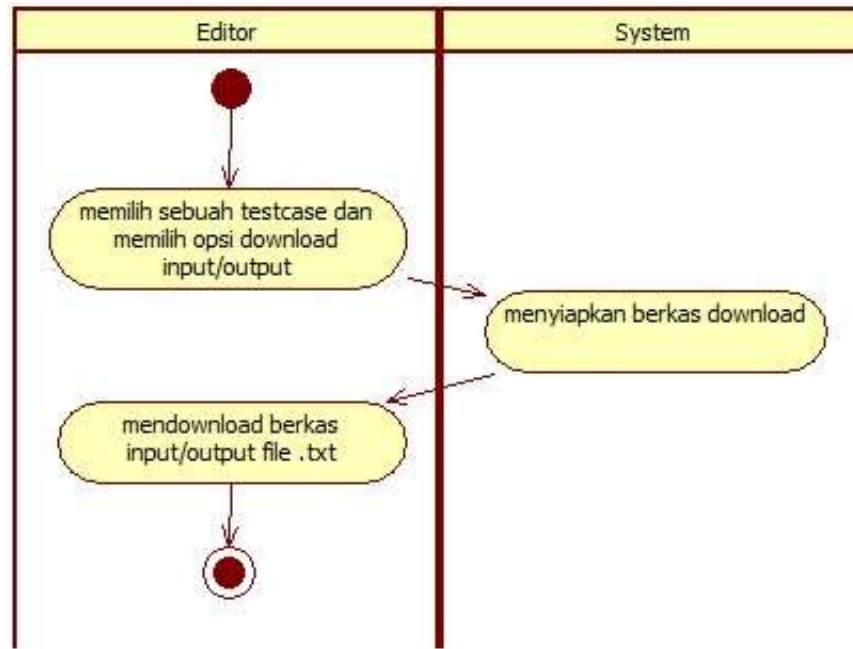


Gambar 3.22 Activity diagram menghapus testcase

L. Activity export testcase

Activity untuk *export testcase* dimulai dengan memilih *opsi download testcase input* dan *testcase output* pada halaman utama *testcase*. Setelah selesai, sistem akan menyiapkan berkas *download* berupa sebuah berkas dengan *format .txt* untuk diunduh. Setelah sistem selesai menyiapkan berkas,

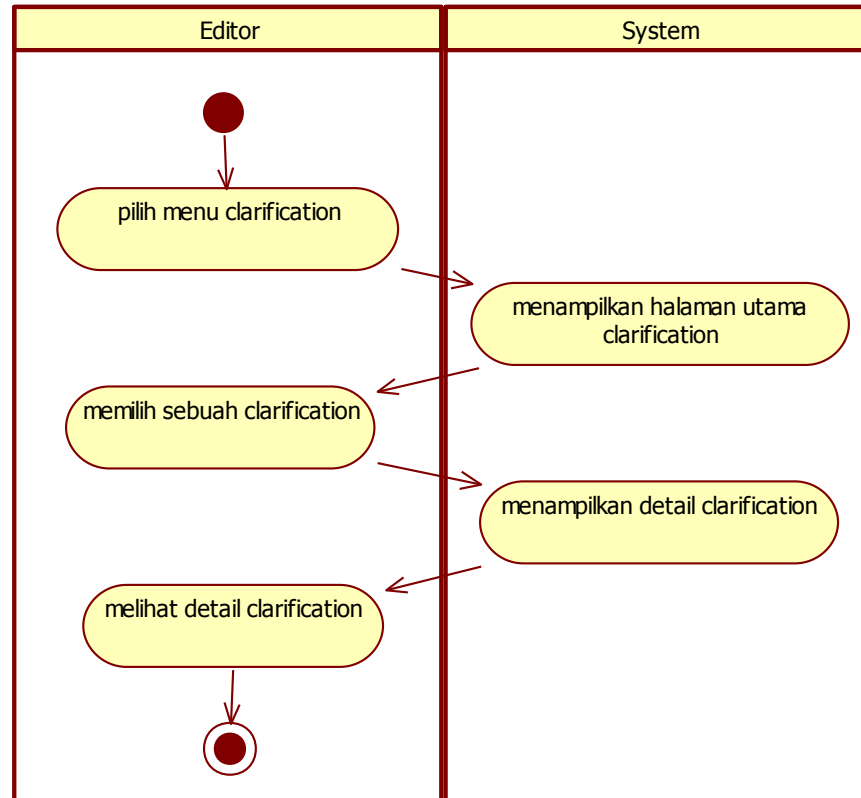
editor akan otomatis mengunduh berkas *text* dari sistem. *Activity diagram* untuk *export testcase* disajikan pada Gambar 3.23.



Gambar 3.23 *Activity diagram export testcase*

M. *Activity* melihat *clarification*

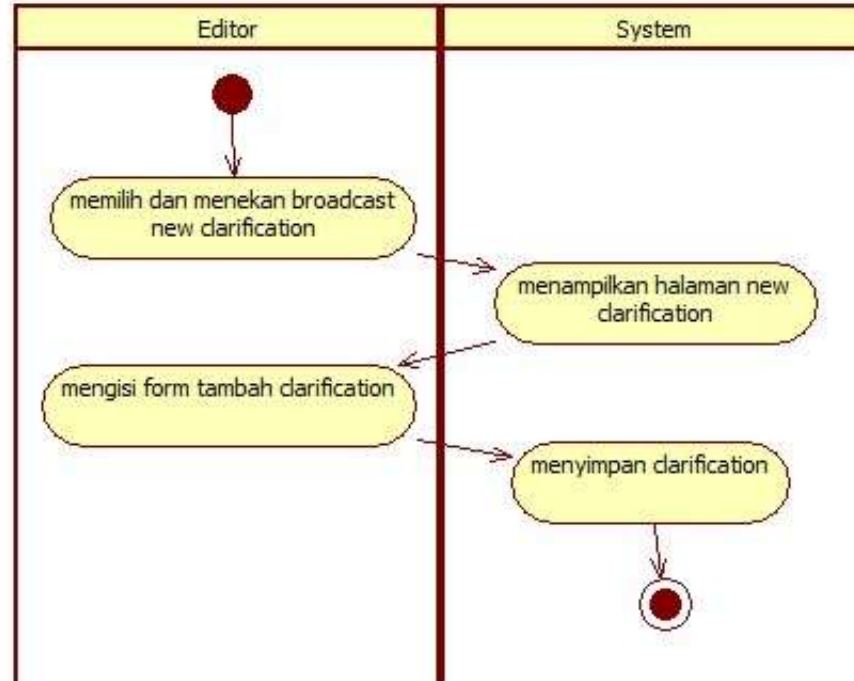
Activity melihat *clarification* dimulai dengan memilih menu *clarification* pada *navigation bar*, sistem akan menampilkan halaman daftar *clarification*. *Editor* memilih sebuah *clarification*, selanjutnya sistem akan menampilkan halaman *detail clarification*. *Activity* melihat *clarification* disajikan pada Gambar 3.24.



Gambar 3.24 Activity diagram melihat clarification

N. Activity menambah clarification

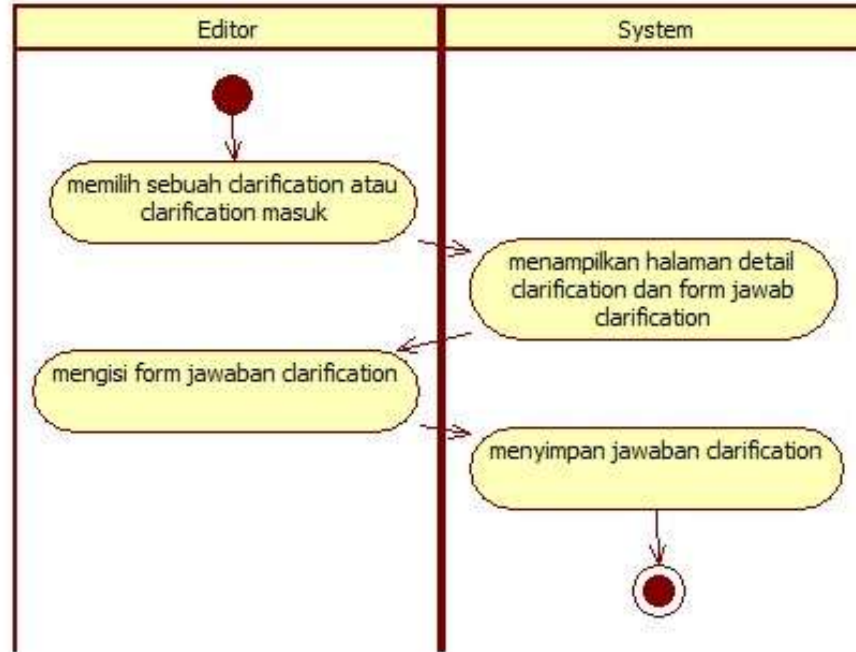
Activity menambah clarification dimulai dengan memilih *link broadcast new clarification* pada halaman utama *clarification*. Sistem akan menampilkan halaman tambah *clarification*. Editor mengisi informasi pada *form* tambah *clarification*. *Activity menambah clarification* disajikan pada Gambar 3.25.



Gambar 3.25 Activity diagram menambah clarification

O. Activity menjawab clarification

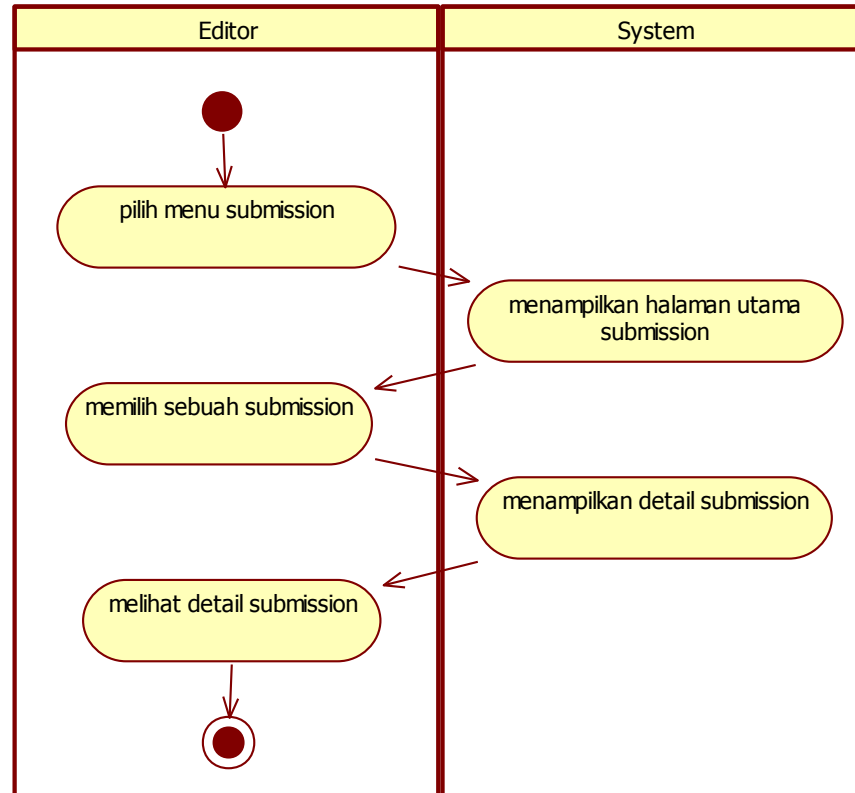
Activity menjawab clarification dimulai dengan memilih sebuah *clarification* yang masuk pada halaman utama *clarification*. Setelah memilih, sistem akan menampilkan halaman *detail clarification*. *Editor* mengisi *form answer clarification* di bawah *detail clarification* pada halaman *detail clarification*. Setelah selesai, tekan tombol *save* dan sistem akan menyimpan jawaban dari *clarification* yang masuk, dan akan dikirim ke *clarification* dari *contest* yang sesuai. *Activity menjawab clarification* disajikan pada Gambar 3.26.



Gambar 3.26 Activity diagram menjawab clarification

P. Activity melihat submission

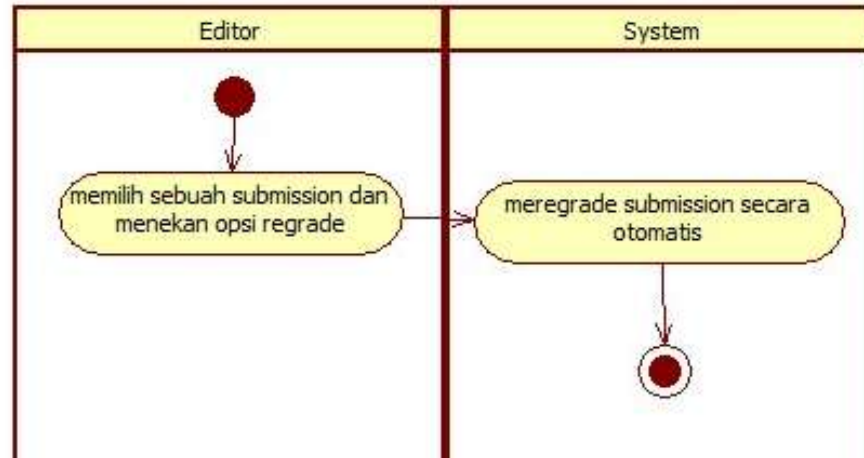
Activity melihat *submission* dimulai dengan memilih menu *submission* pada *navigation bar*, sistem akan mengalihkan ke halaman daftar *submission* yang masuk dari *user/mahasiswa*. *Editor* memilih sebuah *submission*, selanjutnya sistem akan menampilkan halaman *detail submission*. Activity melihat *submission* disajikan pada Gambar 3.27.



Gambar 3.27 Activity diagram melihat submission

Q. Activity melakukan *regrade submission*

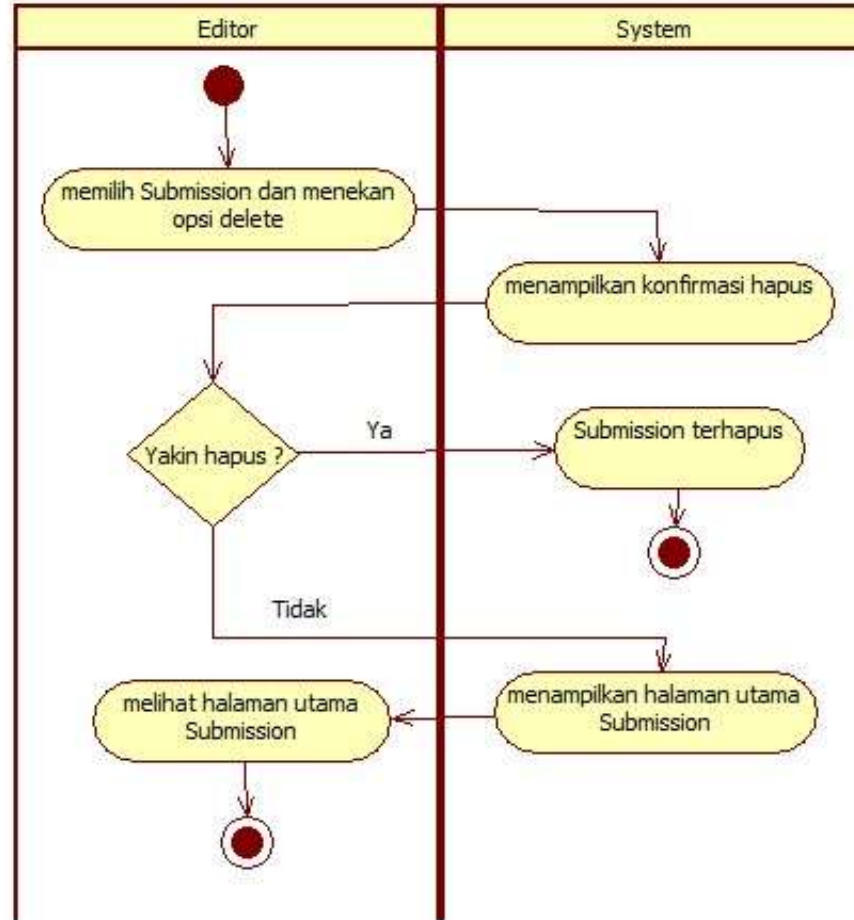
Activity melakukan *regrade submission* dimulai dengan memilih sebuah *submission* pada halaman utama *submission*. Setelah memilih *submission*, *editor* memilih *opsi regrade*. Sistem akan melakukan *regrade* atau *compile* dan *running* ulang kode program yang dikirimkan *user/mahasiswa* secara otomatis ke *submission* yang dipilih. *Activity* melakukan *regrade submission* disajikan pada Gambar 3.28.



Gambar 3.28 Activity diagram regrade submission

R. Activity menghapus submission

Activity menghapus *submission* dimulai dengan memilih *submission* dan memilih *opsi delete*. Sistem akan menampilkan konfirmasi untuk menghapus *submission*. Jika *editor* memilih ya, maka *submission* akan terhapus, jika *editor* memilih tidak, maka sistem akan langsung kembali ke halaman *submission*. Activity menghapus *submission* disajikan pada Gambar 3.29.

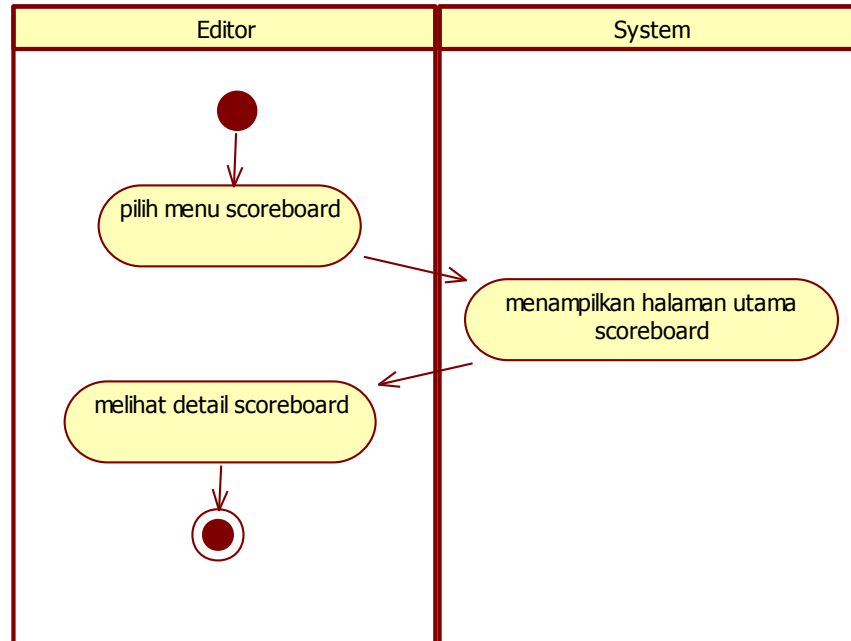


Gambar 3.29 Activity diagram menghapus *submission*

S. Activity melihat *scoreboard*

Activity melihat *scoreboard* dimulai dengan memilih menu *scoreboard* pada *navigation bar*, sistem akan menampilkan halaman utama *scoreboard*.

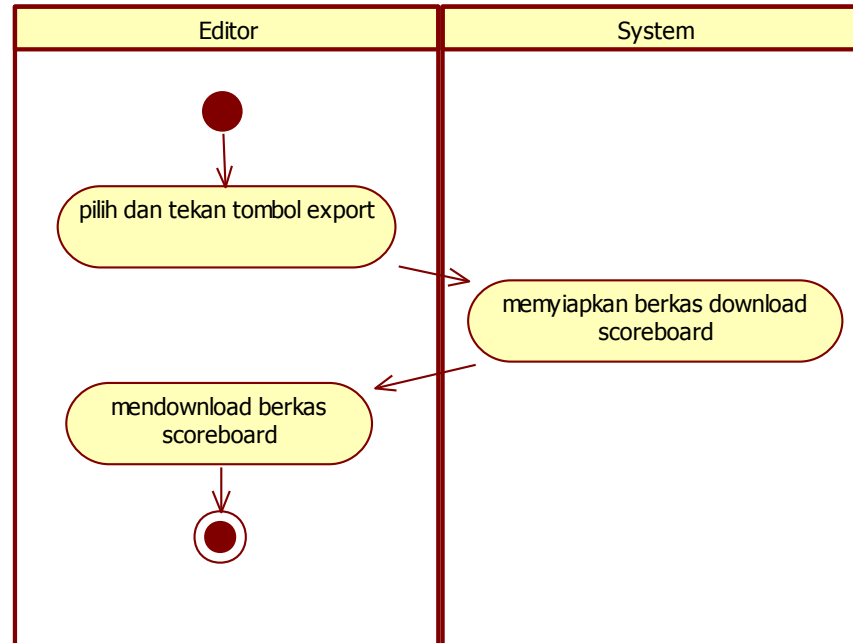
Activity melihat *scoreboard* disajikan pada Gambar 3.30.



Gambar 3.30 *Activity diagram* melihat scoreboard

T. *Activity export scoreboard*

Activity untuk *export scoreboard* dimulai dengan memilih dan menekan tombol *export* pada halaman utama *scoreboard*. Setelah selesai, sistem akan menyiapkan berkas *download* berupa berkas berformat *.xls* atau *.xlsx* untuk diunduh. Setelah sistem selesai menyiapkan berkas, *editor* mengunduh berkas *excel* dari sistem. *Activity diagram export scoreboard* disajikan pada Gambar 3.31.



Gambar 3.31 *Activity diagram export scoreboard*

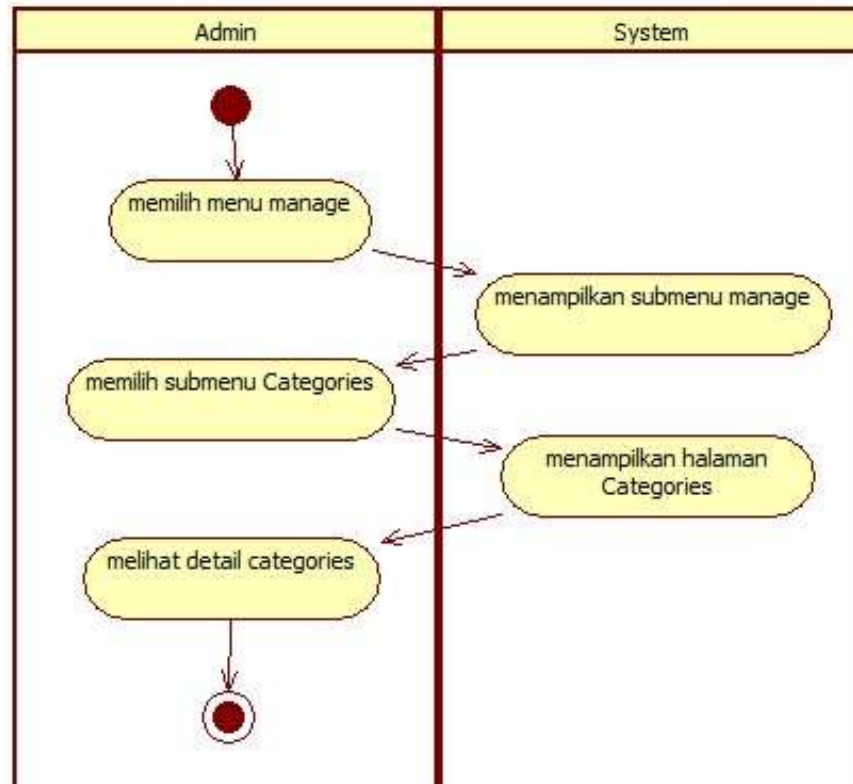
3. *Admin*

Admin merupakan *category* pengguna dengan tingkatan paling tinggi. *Admin* dapat melakukan 38 transaksi terhadap sistem. Transaksi yang dilakukan oleh *admin* adalah sebagai berikut.

A. *Activity* melihat *category*

Activity melihat *category* dimulai dengan memilih menu *manage* pada *navigation bar*. Sistem akan menampilkan submenu dari menu *manage*, *admin* memilih submenu *category*. Setelah selesai, sistem akan

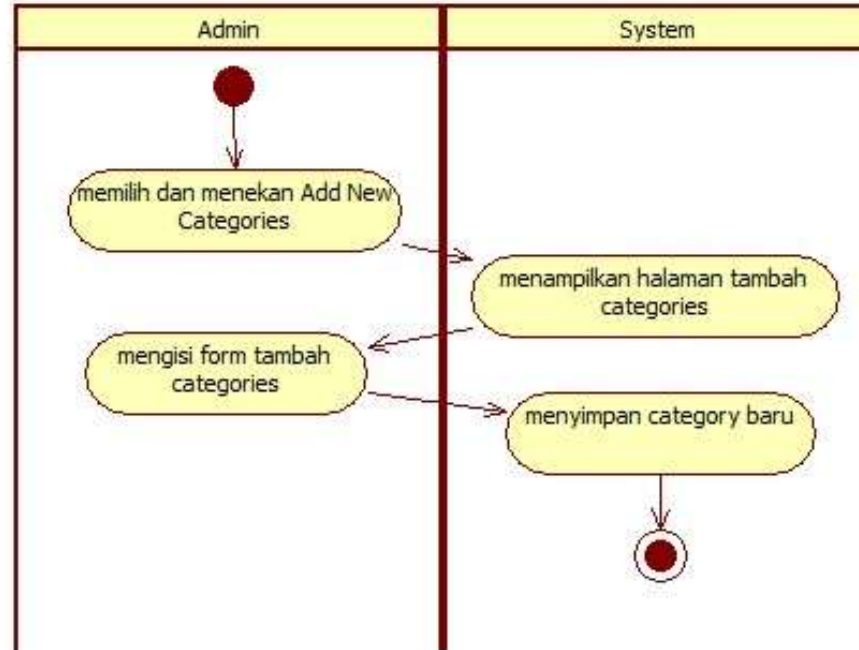
menampilkan halaman *category*. *Activity* melihat *category* disajikan pada Gambar 3.32.



Gambar 3.32 *Activity diagram* melihat *category*

B. *Activity* menambah *category*

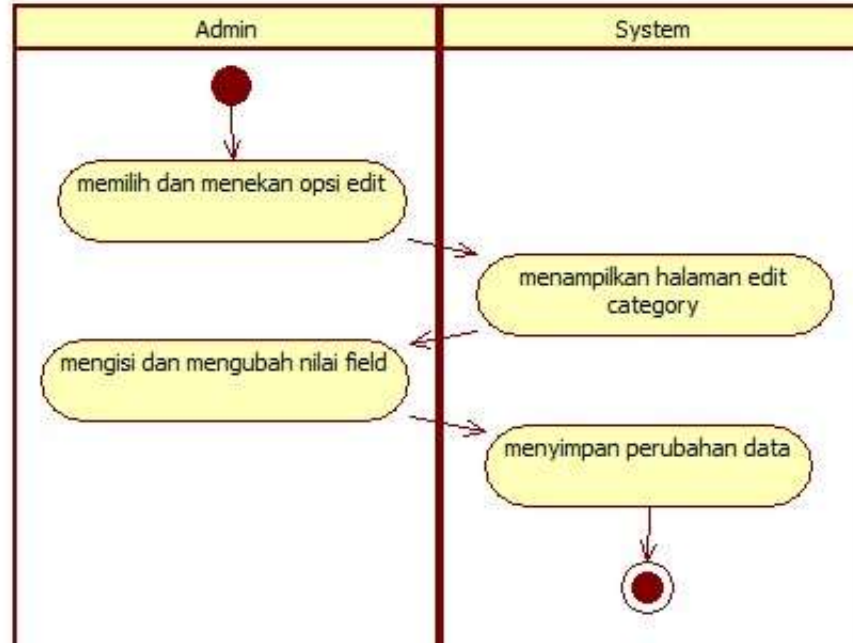
Activity menambah *category* dimulai dengan memilih dan menekan *link add new category* pada halaman utama *category*. Sistem akan menampilkan halaman dan *form* pengisian tambah *category*. *Admin* mengisi *form* tambah *category*. Setelah selesai, sistem akan menyimpan *category* yang baru ditambahkan. *Activity* menambah *category* disajikan pada Gambar 3.33.



Gambar 3.33 Activity diagram menambah category

C. Activity mengubah category

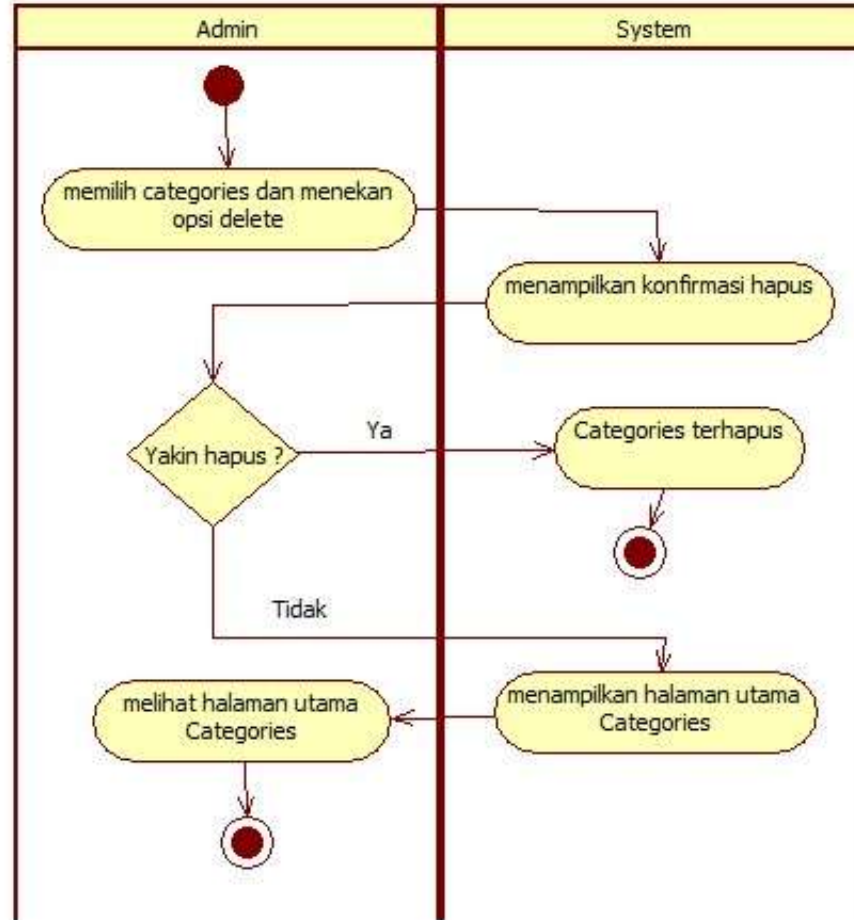
Activity mengubah category dimulai dengan memilih sebuah category, lalu memilih dan menekan opsi edit pada category yang sudah dipilih, sistem akan menampilkan halaman dan form edit category. Admin mengisi dan mengubah informasi yang ada pada form yang tersedia. Setelah selesai, sistem akan menyimpan hasil perubahan yang dilakukan oleh admin. Activity mengubah category disajikan pada Gambar 3.34.



Gambar 3.34 *Activity diagram* mengubah *category*

D. *Activity* menghapus *category*

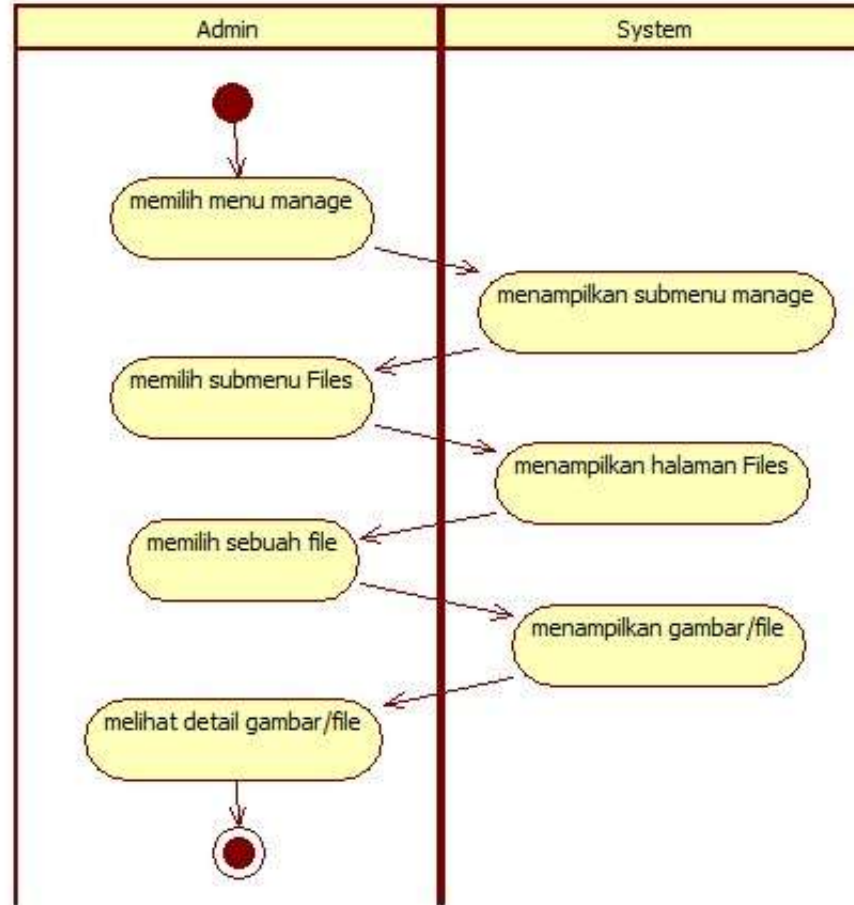
Activity menghapus *category* dimulai dengan memilih *category* dan memilih *opsi delete*. Sistem akan menampilkan konfirmasi untuk menghapus *category*. Jika *admin* memilih ya, maka *category* akan terhapus, jika *admin* memilih tidak, maka sistem akan kembali ke halaman *category*. *Activity* menghapus *category* disajikan pada Gambar 3.35.



Gambar 3.35 Activity diagram menghapus category

E. Activity melihat file

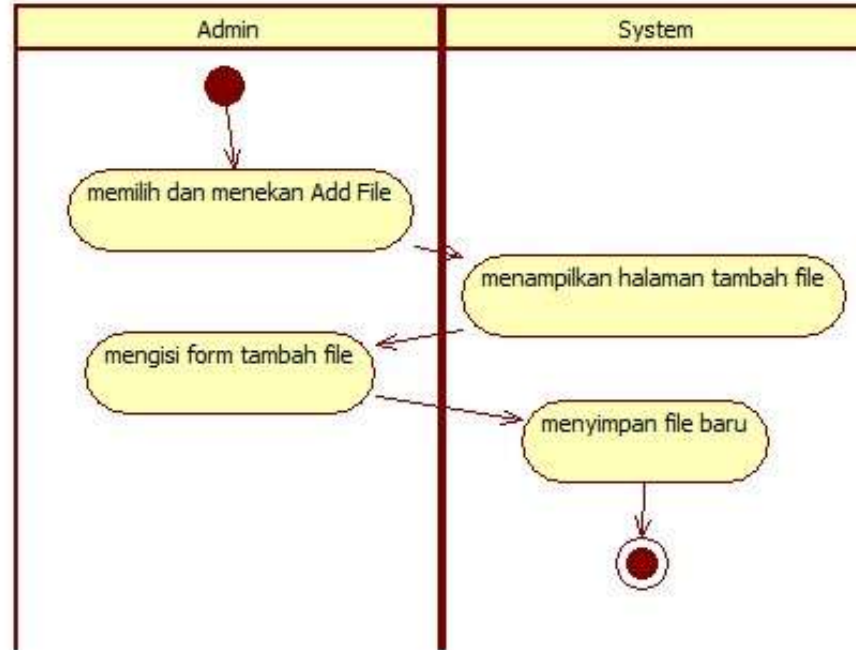
Activity melihat file dimulai dengan memilih menu *manage* pada *navigation bar*. Sistem akan menampilkan submenu dari menu *manage*, *admin* memilih submenu *file*. Setelah selesai, sistem akan menampilkan halaman utama *file*. *Admin* memilih sebuah *file*, dan sistem akan menampilkan gambar dari *file* tersebut. Activity melihat file disajikan pada Gambar 3.36.



Gambar 3.36 Activity diagram melihat file

F. Activity menambah file

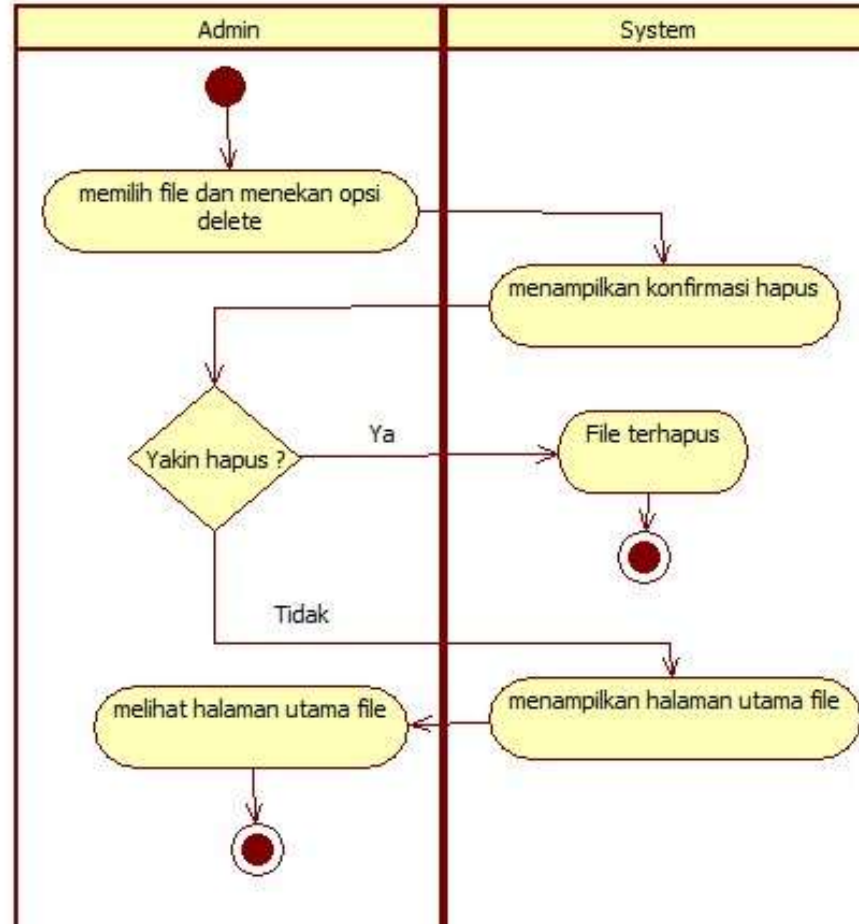
Activity menambah file dimulai dengan memilih dan menekan *link add new file* pada halaman utama file. Sistem akan menampilkan halaman dan *form* pengisian tambah file. Admin mengisi *form* tambah file. Setelah selesai, sistem akan menyimpan file yang baru ditambahkan. Activity menambah file disajikan pada Gambar 3.37.



Gambar 3.37 Activity diagram menambah file

G. Activity menghapus file

Activity menghapus file dimulai dengan memilih file dan memilih opsi delete. Sistem akan menampilkan konfirmasi untuk menghapus file. Jika admin memilih ya, maka file akan terhapus, jika admin memilih tidak, maka sistem akan langsung kembali ke halaman file. Activity menghapus file disajikan pada Gambar 3.38.

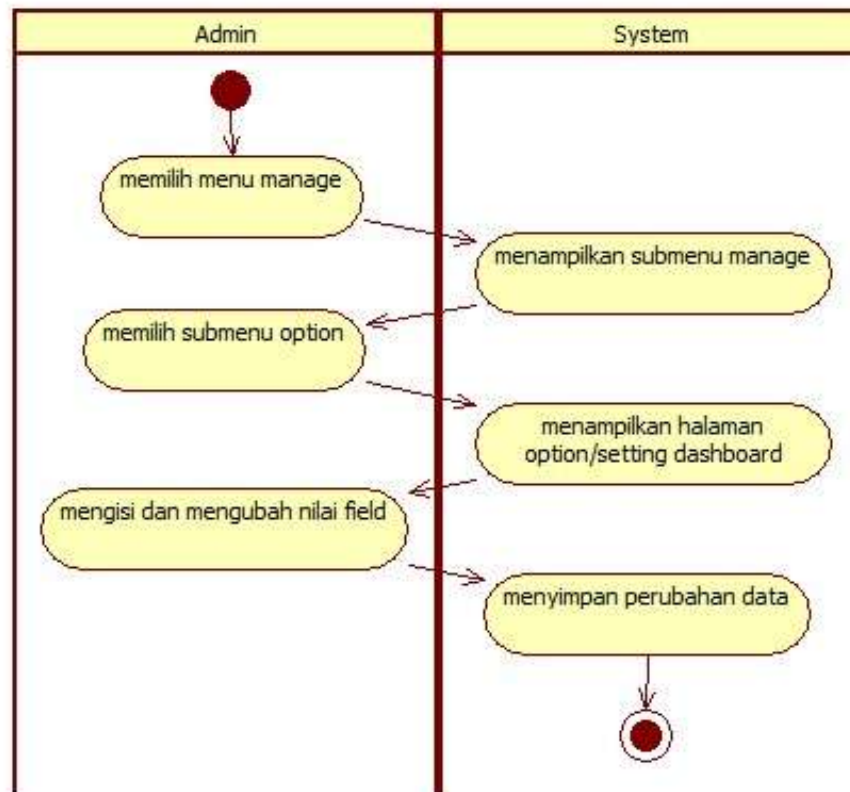


Gambar 3.38 Activity diagram menghapus file

H. Activity mengubah option

Activity mengubah option atau informasi dashboard dimulai dengan memilih menu *manage* pada *navigation bar*, lalu sistem akan menampilkan submenu dari menu *manage*. Admin memilih submenu *option*. Sistem akan menampilkan halaman dan *form edit option*. Admin mengisi dan mengubah informasi yang ada pada *form* yang tersedia. Setelah selesai, sistem akan

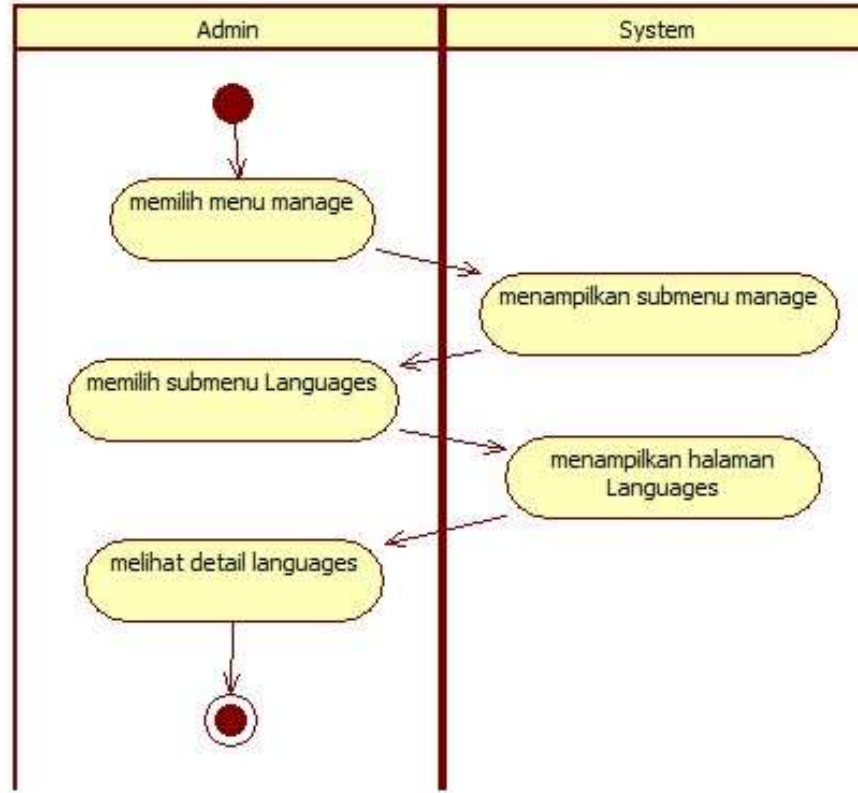
meysimpan hasil perubahan yang dilakukan oleh *admin*. *Activity* mengubah *option* disajikan pada Gambar 3.39.



Gambar 3.39 *Activity diagram* mengedit *option*

I. *Activity* melihat *languages*

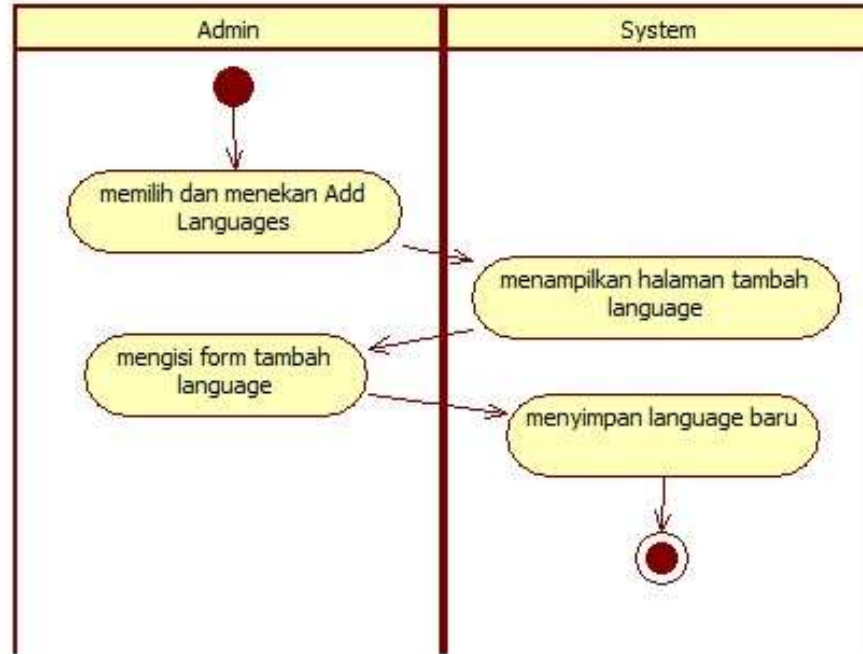
Activity melihat *languages* dimulai dengan memilih menu *manage* pada *navigation bar* dan memilih submenu *languages*. Setelah selesai, sistem akan menampilkan halaman utama *languages*. *Activity* melihat *languages* disajikan pada Gambar 3.40.



Gambar 3.40 Activity diagram melihat languages

J. Activity menambah languages

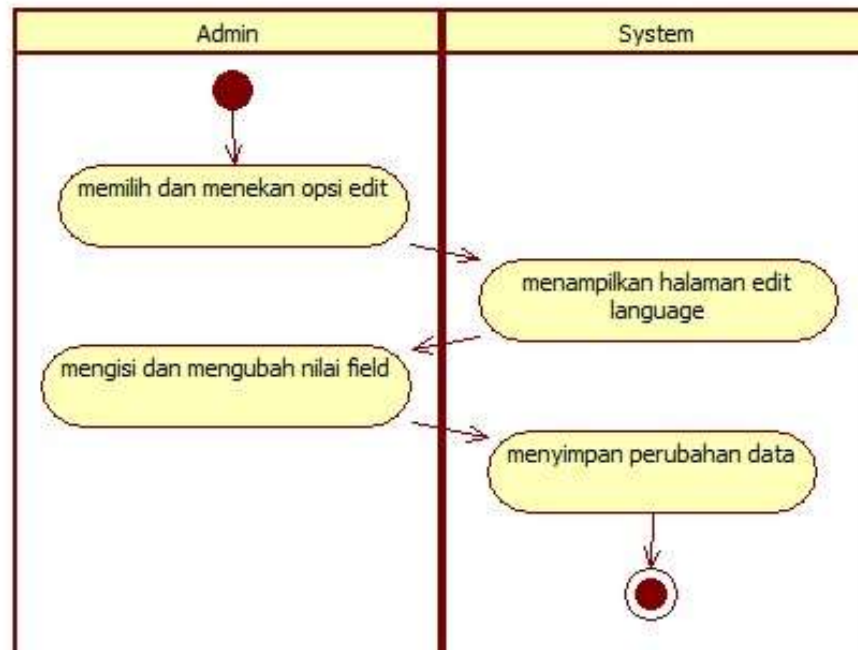
Activity menambah languages dimulai dengan memilih dan menekan link *add new languages* pada halaman utama *languages*. Sistem akan menampilkan halaman dan *form* pengisian tambah languages. Admin mengisi *form* tambah languages. Setelah selesai, sistem akan menyimpan languages yang baru ditambahkan. Activity menambah languages disajikan pada Gambar 3.41.



Gambar 3.41 Activity diagram menambah languages

K. Activity mengubah languages

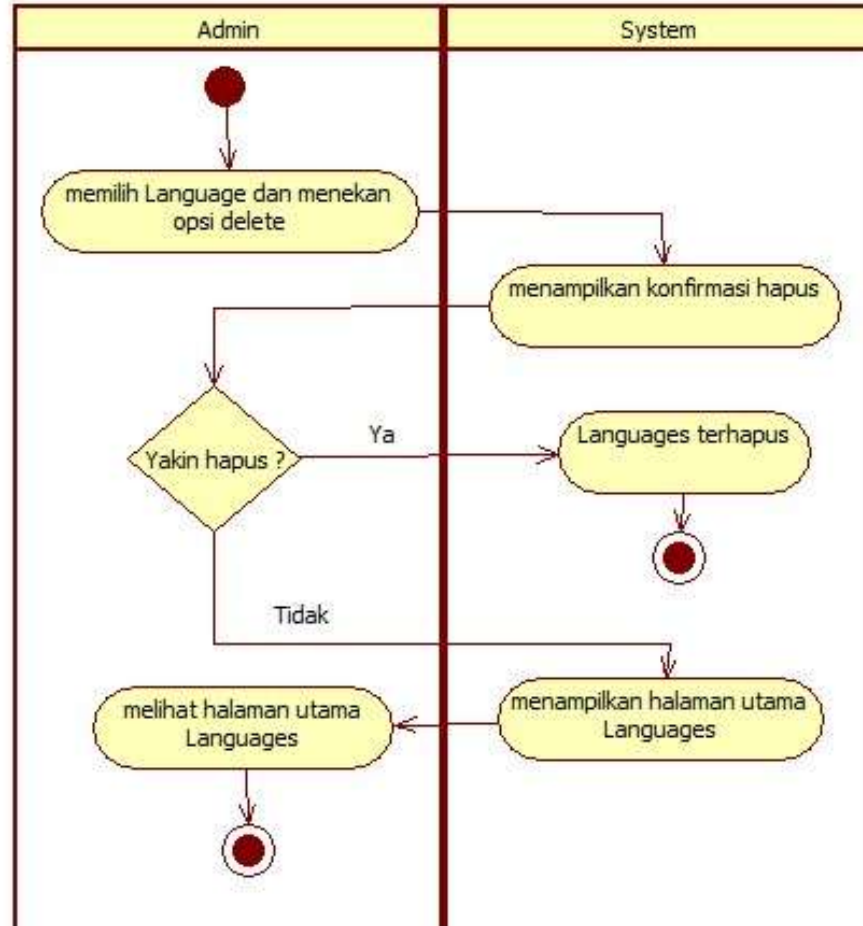
Activity mengubah languages dimulai dengan memilih sebuah languages, lalu memilih dan menekan opsi edit pada languages yang sudah dipilih, sistem akan menampilkan halaman dan form edit languages. Admin mengisi dan mengubah informasi yang ada pada form yang tersedia. Setelah selesai, sistem akan menyimpan hasil perubahan yang dilakukan oleh admin. Activity mengubah languages disajikan pada Gambar 3.42.



Gambar 3.42 Activity diagram mengedit languages

L. Activity menghapus languages

Activity menghapus *languages* dimulai dengan memilih *languages* dan memilih *opsi delete*. Sistem akan menampilkan konfirmasi untuk menghapus *languages*. Jika *admin* memilih ya, maka *languages* akan terhapus, jika *admin* memilih tidak, maka sistem akan kembali ke halaman *languages*. *Activity* menghapus *languages* disajikan pada Gambar 3.43.

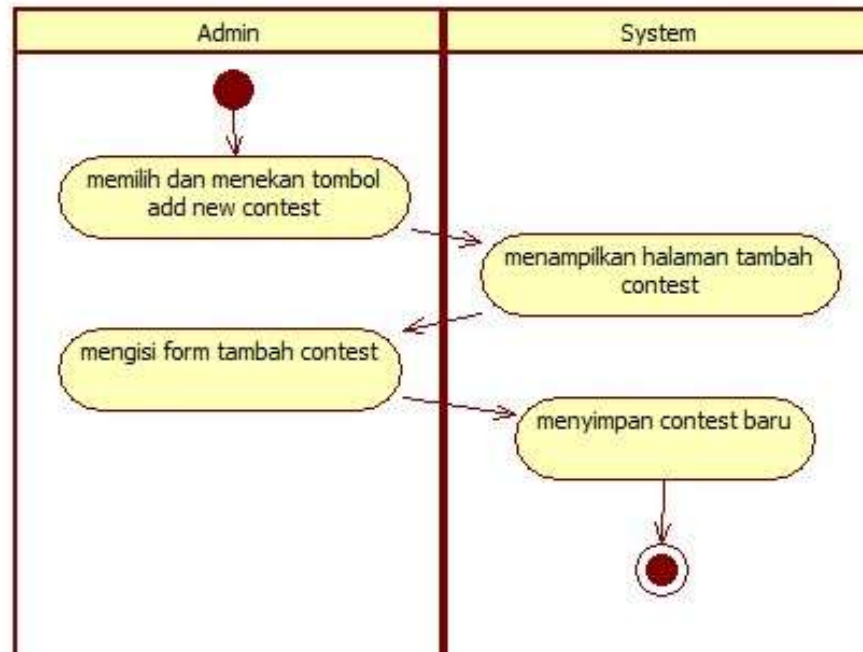


Gambar 3.43 Activity diagram menghapus languages

M. Activity menambah *contest*/kelas

Activity menambah *contest*/kelas dimulai dengan memilih dan menekan *link add new contest/kelas* pada halaman utama *contest/kelas*. Sistem akan menampilkan halaman dan *form* pengisian tambah *contest/kelas*. *Admin* mengisi *form* tambah *contest/kelas*. Setelah selesai, sistem akan menyimpan

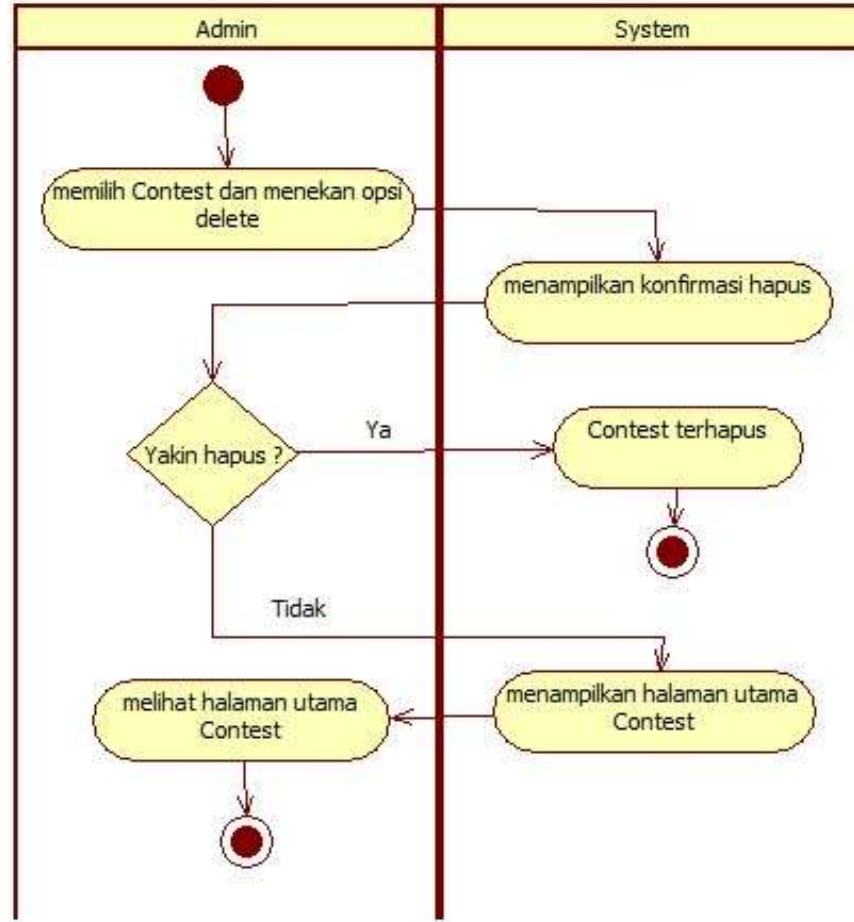
contest/kelas yang baru ditambahkan. *Activity* menambah *contest*/kelas disajikan pada Gambar 3.45.



Gambar 3.45 *Activity diagram* menambah *contest*/kelas

N. *Activity* menghapus *contest*/kelas

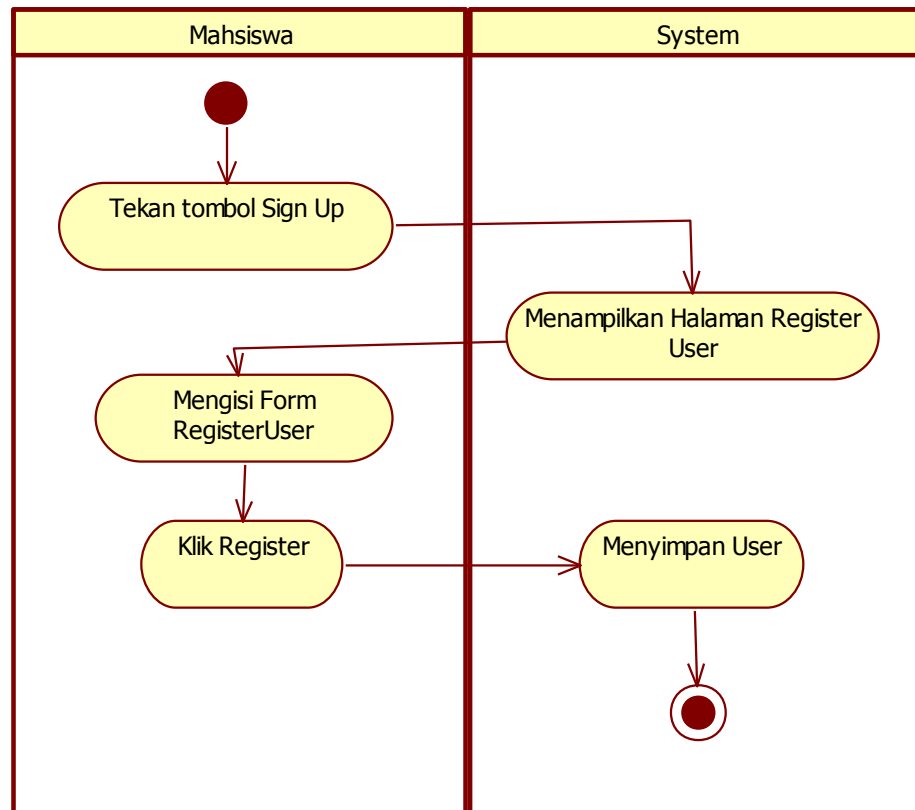
Activity menghapus *contest*/kelas dimulai dengan memilih *contest*/kelas dan memilih *opsi delete*. Sistem akan menampilkan konfirmasi untuk menghapus *contest*/kelas. Jika *admin* memilih ya, maka *contest*/kelas akan terhapus, jika *admin* memilih tidak, maka sistem akan langsung kembali ke halaman *contest*/kelas. *Activity* menghapus *contest*/kelas disajikan pada Gambar 3.46.



Gambar 3.46 Activity diagram menghapus contest/kelas

O. Activity register mahasiswa

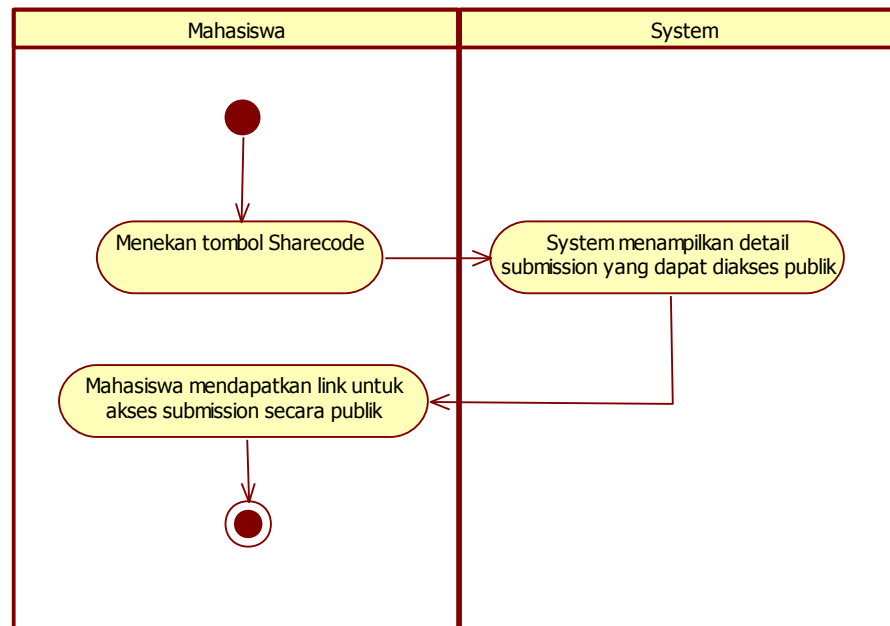
Activity register mahasiswa dimulai dengan menekan tombol *Sign Up* pada halaman *login*. Sistem akan mengarahkan ke halaman *register user*. Mahasiswa mengisi *form* dan menekan tombol *register*, sistem akan mengarahkan ke halaman *login*. Activity register mahasiswa disajikan pada Gambar 3.47.



Gambar 3.47 Activity diagram register mahasiswa

P. Activity sharecode

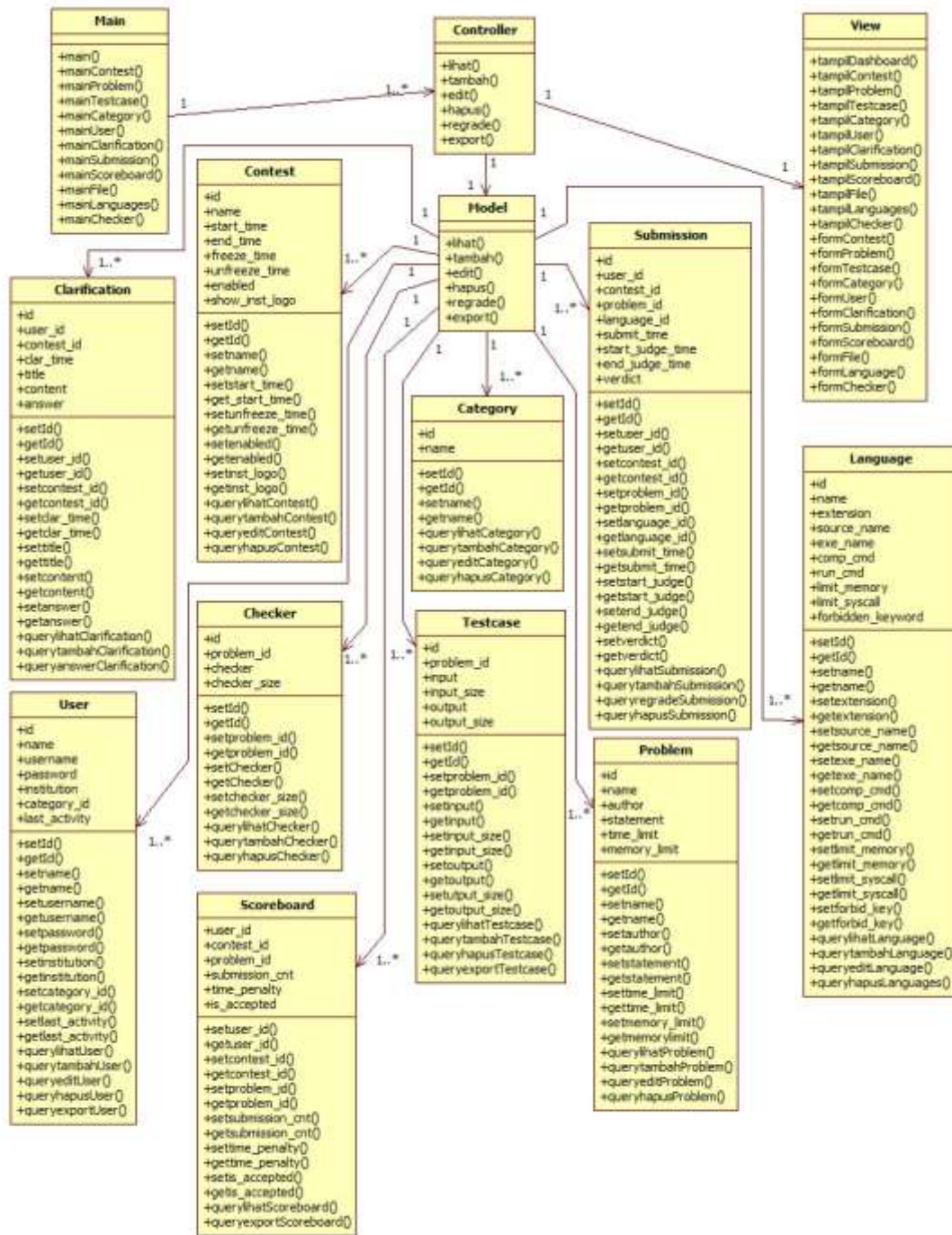
Activity sharecode dimulai dengan menekan tombol *link sharecode* pada halaman *detail submission*. Sistem akan mengarahkan ke halaman *detail submission* dengan *link url* yang berbeda dan dapat diakses oleh publik. *Activity sharecode* disajikan pada Gambar 3.48.



Gambar 3.48 Activity diagram sharecode

3.3.4.2. Class Diagram

Pada *class diagram* terdapat nama-nama *table* dan *attribute* yang digunakan pada sistem *grader*. *Class diagram* pada sistem ini menggunakan pendekatan *Model, View* dan *Controller* (MVC). *Class diagram* sistem *grader* disajikan pada Gambar 3.49.



Gambar 3.49 Class Diagram

3.3.4.3. *Sequence diagram*

Sequence diagram menggambarkan interaksi antara objek dan di sekitar sistem berupa pesan terhadap waktu dan merupakan hubungan antara sistem dengan pengguna. *Sequence diagram* digambarkan sesuai dengan cara sistem memproses permintaan dari *user*. *Sequence diagram* pada sistem *Grader* digambarkan dalam Gambar 3.50 sampai dengan 3.92.

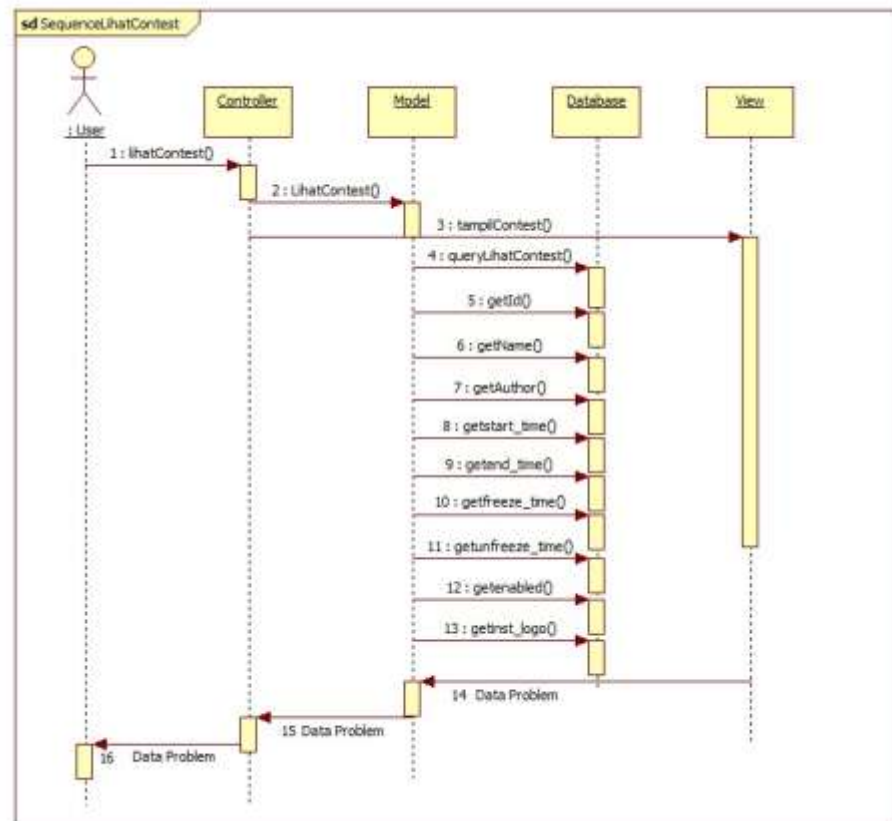
1. *User/Mahasiswa*

A. *Sequence diagram* melihat *Contest*

Sequence diagram melihat *contest* dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *User* memilih menu lihat *contest* pada *controller contest*.
- *Controller contest* mengakses *model contest* untuk mengambil data dari *database contest*.
- *Controller* menampilkan data *contest* ke *view contest*

Sequence diagram melihat *contest* disajikan pada Gambar 3.50.



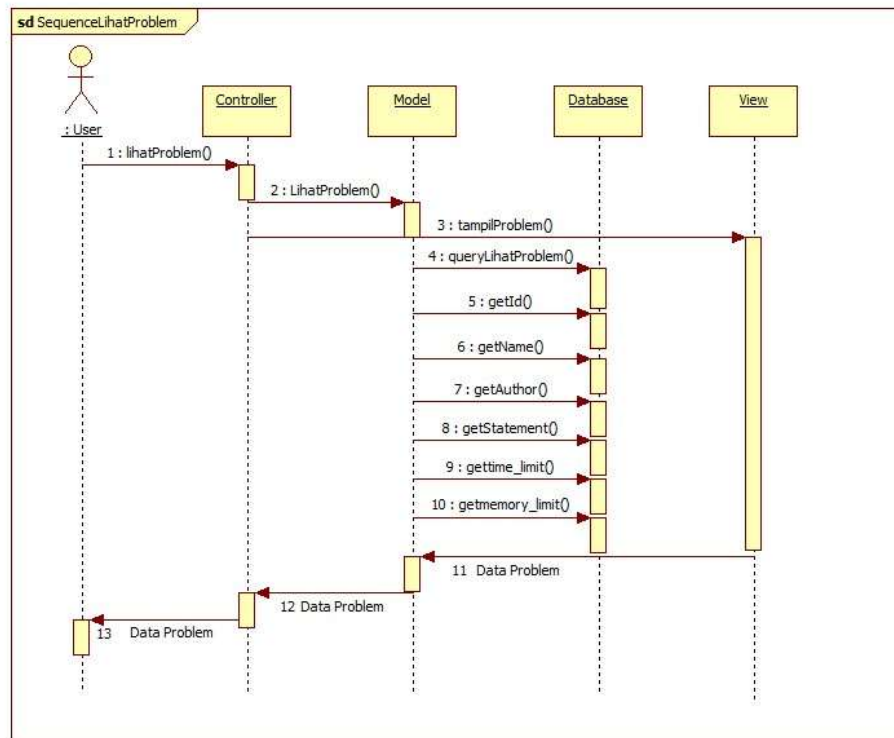
Gambar 3.50 *Sequence diagram* melihat contest

B. *Sequence diagram* melihat problem

Sequence diagram melihat *problem* dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *User* memilih menu lihat *problem* pada *controller problem*.
- *Controller problem* mengakses *model problem* untuk mengambil data dari *database contest*.
- *Controller* menampilkan data *problem* ke *view problem*.

Sequence diagram melihat *problem* disajikan pada Gambar 3.51.



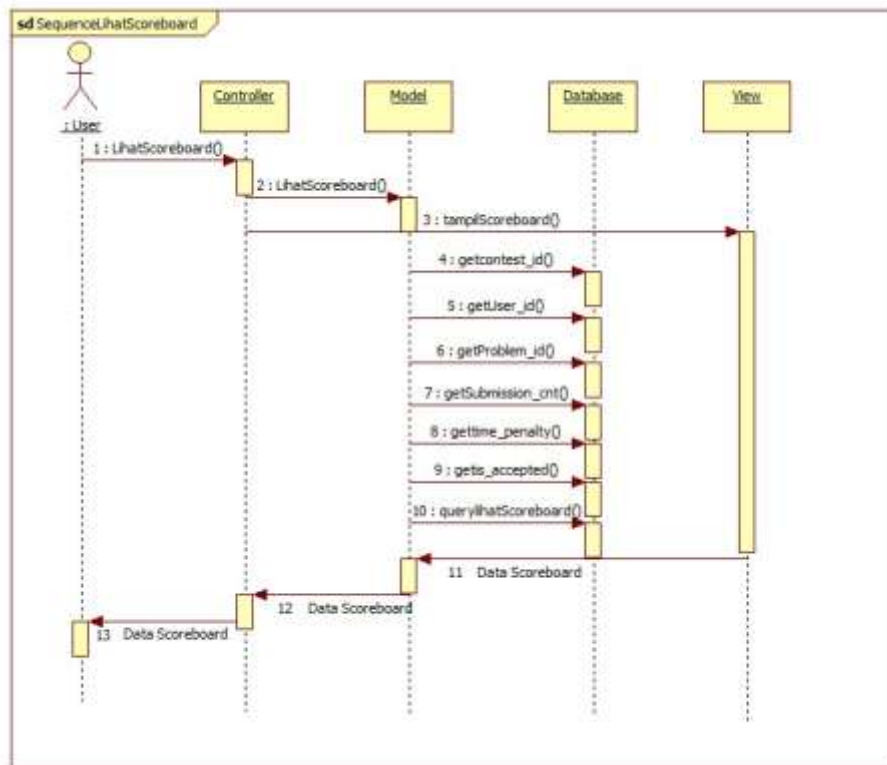
Gambar 3.51 *Sequence diagram* melihat problem

C. *Sequence diagram* melihat scoreboard

Sequence diagram melihat scoreboard dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *User* memilih menu lihat problem pada *controller scoreboard*.
- *Controller scoreboard* mengakses *model scoreboard* untuk mengambil data dari *database scoreboard*.
- *Controller* menampilkan data scoreboard ke *view scoreboard*

Sequence diagram melihat scoreboard disajikan pada Gambar 3.52.



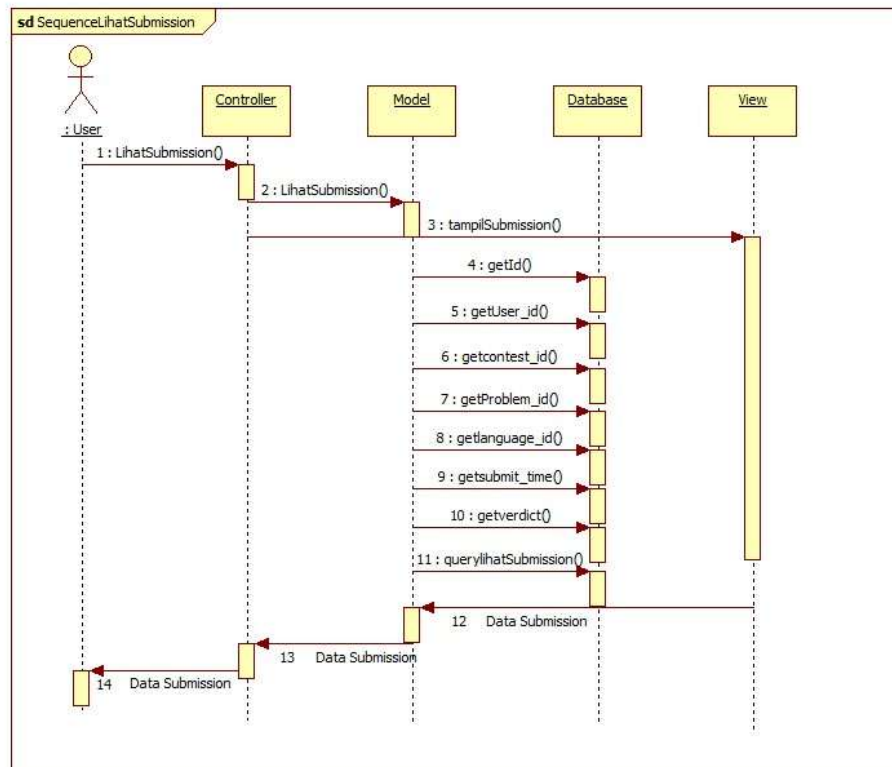
Gambar 3.52 *Sequence diagram* melihat scoreboard

D. *Sequence diagram* melihat submission

Sequence diagram melihat submission dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *User* memilih menu lihat *submission* pada *controller submission*.
- *Controller submission* mengakses *model submission* untuk mengambil data dari *database submission*.
- *Controller* menampilkan data *submission* ke *view submission*.

Sequence diagram melihat *submission* disajikan pada Gambar 3.53.



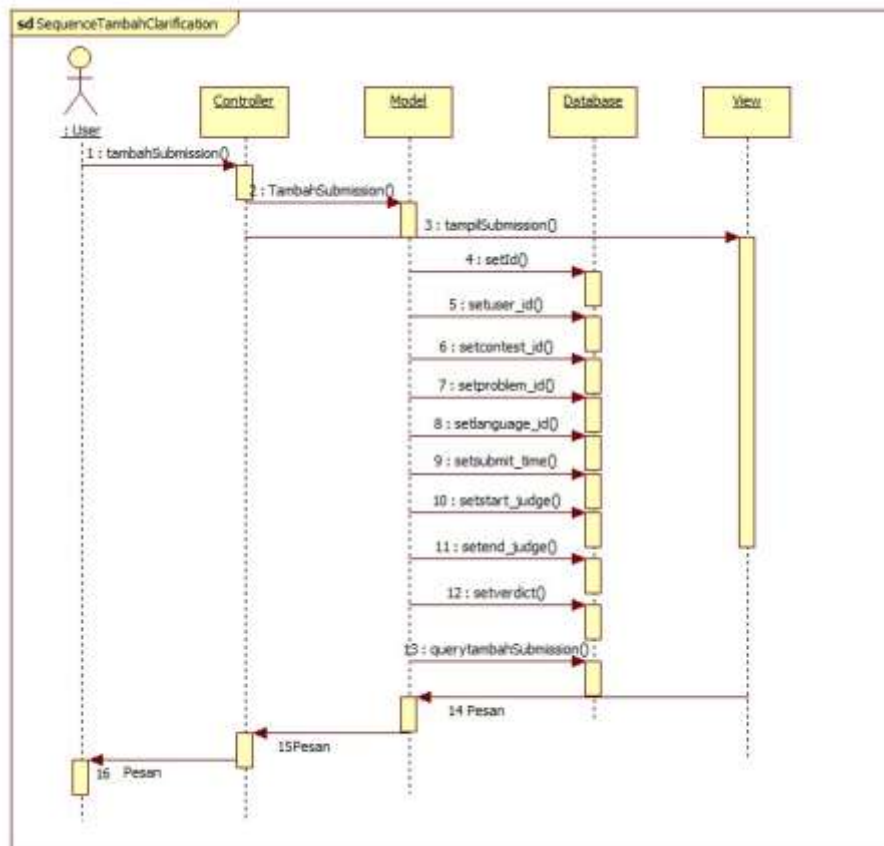
Gambar 3.53 *Sequence diagram* melihat *submission*

E. *Sequence diagram* menambah *submission*

Sequence diagram menambah *submission* dilakukan dengan 5 buah objek dengan proses sebagai berikut.

- *User* memilih tambah *submission* pada *controller submission*.
- *Controller submission* memanggil *model submission* untuk menyiapkan *query* tambah data
- *Model* menyimpan hasil *query* tambah data ke *database*
- Sistem menampilkan pesan berhasil.

Sequence diagram menambah *submission* disajikan pada Gambar 3.54.



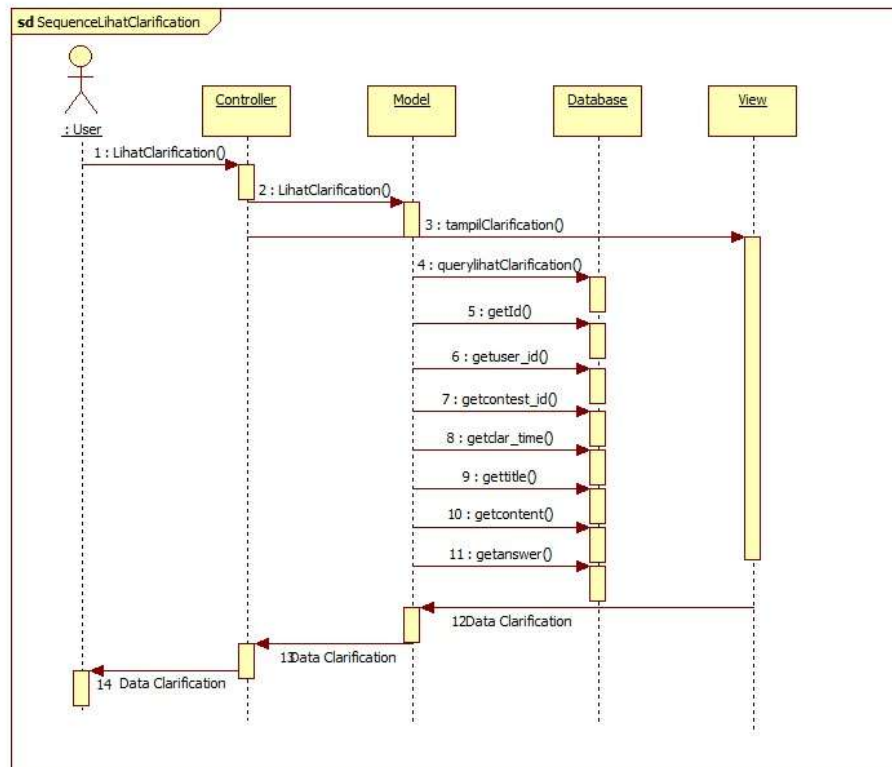
Gambar 3.54 *Sequence diagram* menambah *submission*

F. *Sequence diagram* melihat *clarification*

Sequence diagram melihat *clarification* dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *User* memilih menu lihat *clarification* pada *controller clarification*.
- *Controller clarification* mengakses *model clarification* untuk mengambil data dari *database clarification*.
- *Controller* menampilkan data *clarification* ke *view clarification*.

Sequence diagram melihat *clarification* disajikan pada Gambar 3.55.



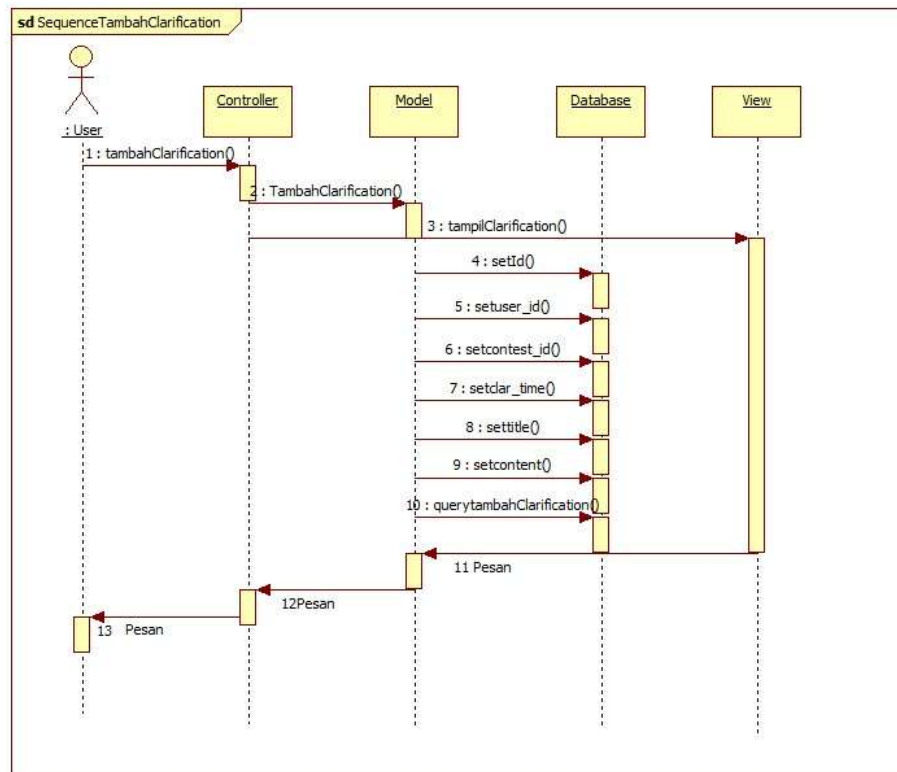
Gambar 3.55 *Sequence diagram* melihat *clarification*

G. *Sequence diagram* menambah *clarification*

Sequence diagram menambah *clarification* dilakukan dengan 5 buah objek dengan proses sebagai berikut.

- *User* memilih tambah *clarification* pada *controller clarification*.
- *Controller clarification* memanggil *model clarification* untuk menyiapkan *query* tambah data
- *Model* menyimpan hasil *query* tambah data ke *database*
- Sistem menampilkan pesan berhasil.

Sequence diagram menambah *clarification* disajikan pada Gambar 3.56.



Gambar 3.56 *Sequence diagram* menambah *clarification*

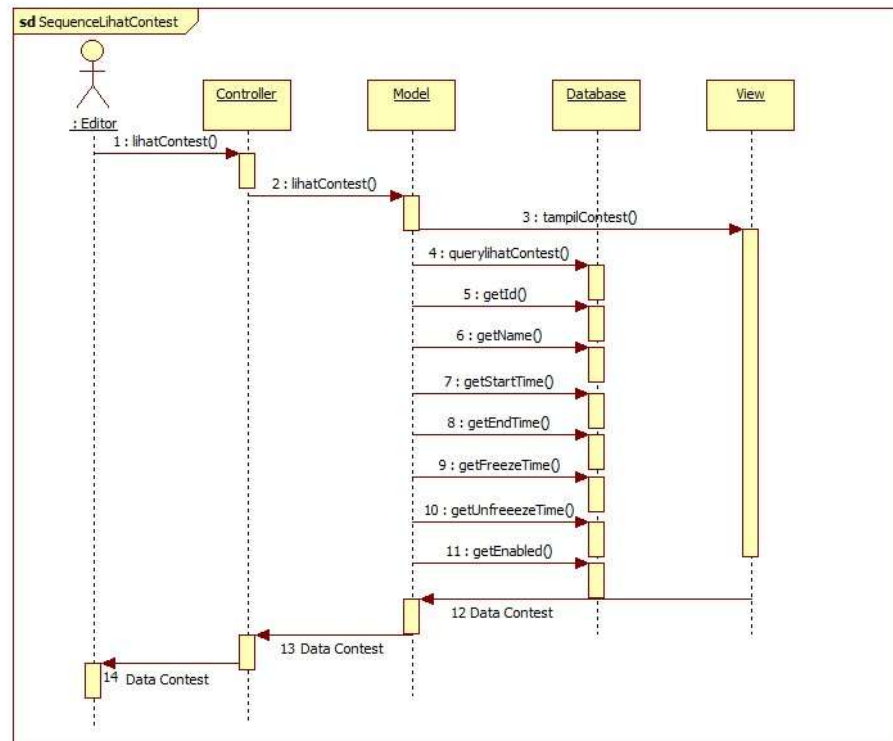
2. Editor/Dosen/Asisten Dosen

A. *Sequence diagram* melihat *contest*

Sequence diagram melihat *contest* dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *Editor* memilih menu lihat *contest* pada *controller contest*.
- *Controller contest* mengakses *model contest* untuk mengambil data dari *database contest*.
- *Controller* menampilkan data *contest* ke *view contest*.

Sequence diagram melihat *contest* disajikan pada Gambar 3.57.



Gambar 3.57 Sequence diagram melihat *contest*

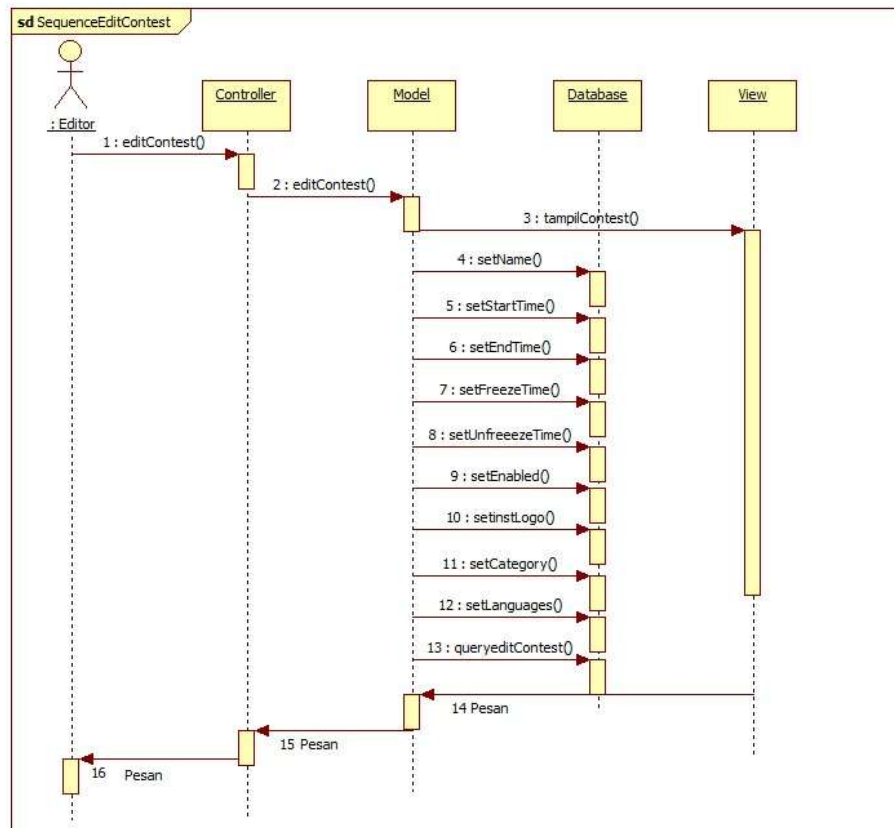
B. Sequence diagram mengubah *contest*

Sequence diagram mengubah *contest* dilakukan dengan 5 objek dengan proses sebagai berikut.

- *Editor* memilih menu *edit* pada *controller contest*
- *Controller* memanggil *model contest*
- *Controller* memanggil *view* untuk menampilkan data
- *Editor* mengisi *form*, *controller* memanggil *query edit* data dari *model*

- Sistem menampilkan pesan kepada *editor*.

Sequence diagram mengubah *contest* disajikan pada Gambar 3.58.



Gambar 3.58 *Sequence diagram* mengubah *contest*

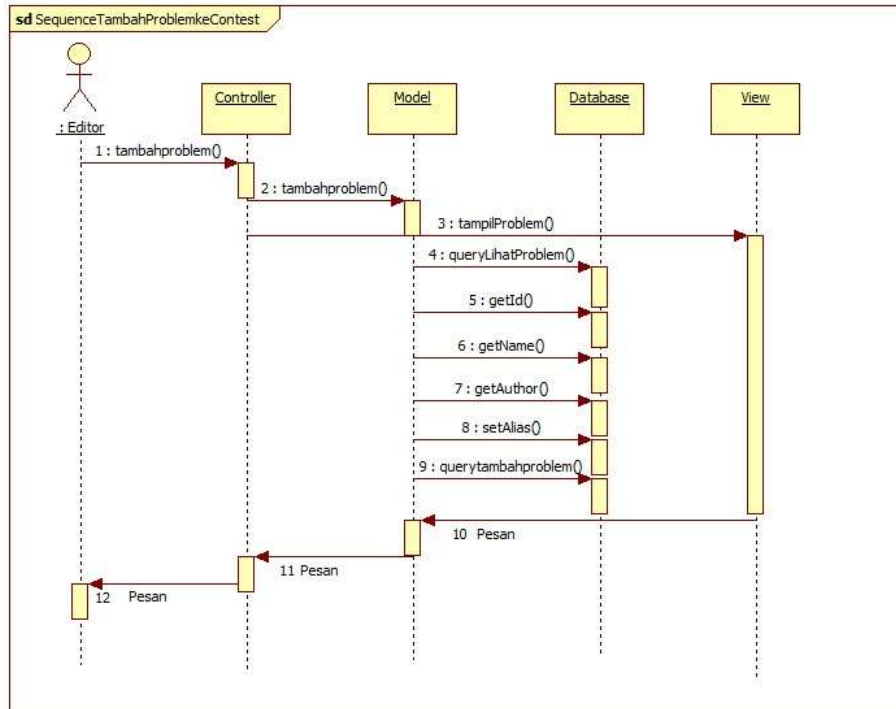
C. *Sequence diagram* menambah *problem* ke *contest*

Sequence diagram menambah *problem* ke dalam *contest* dilakukan dengan 5 buah objek dengan proses sebagai berikut.

- *Editor* memilih tambah *problem* pada *controller contest*.
- *Controller contest* memanggil *model contest* untuk menyiapkan *query* tambah data

- *Model* menyimpan hasil *query* tambah data ke *database*
- Sistem menampilkan pesan berhasil.

Sequence diagram menambah *problem* ke sebuah *contest* disajikan pada Gambar 3.59.



Gambar 3.59 *Sequence diagram* menambah *problem* ke *contest*

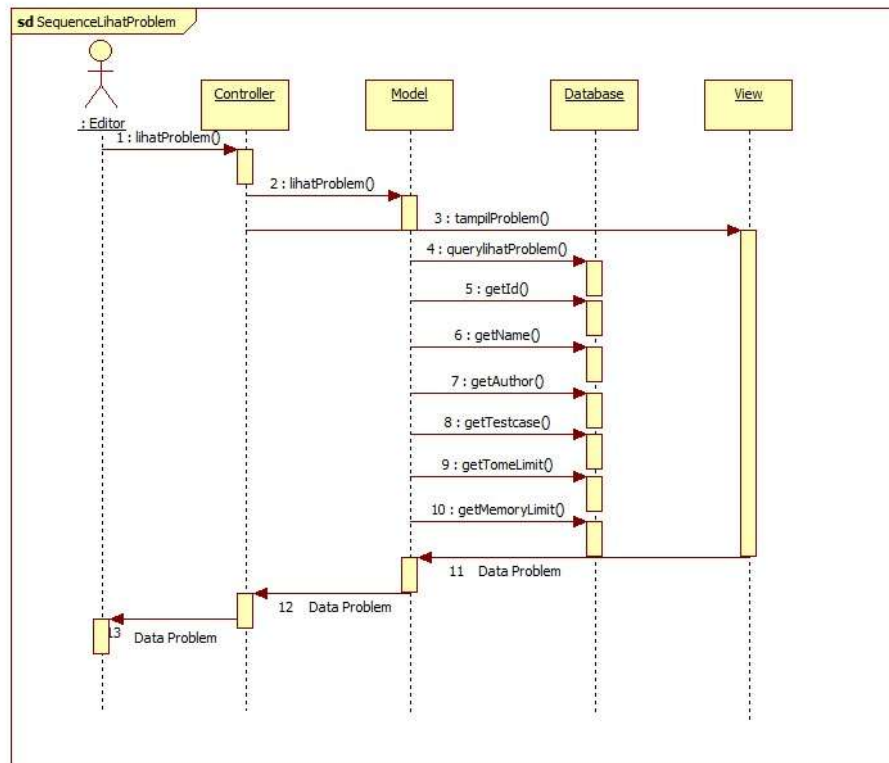
D. *Sequence diagram* melihat *problem*

Sequence diagram melihat *problem* dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *Editor* memilih menu lihat *problem* pada *controller problem*.
- *Controller problem* mengakses *model problem* untuk mengambil data dari *database problem*.

- *Controller* menampilkan data *problem* ke *view problem*.

Sequence diagram melihat *problem* disajikan pada Gambar 3.60.



Gambar 3.60 *Sequence diagram* melihat *problem*

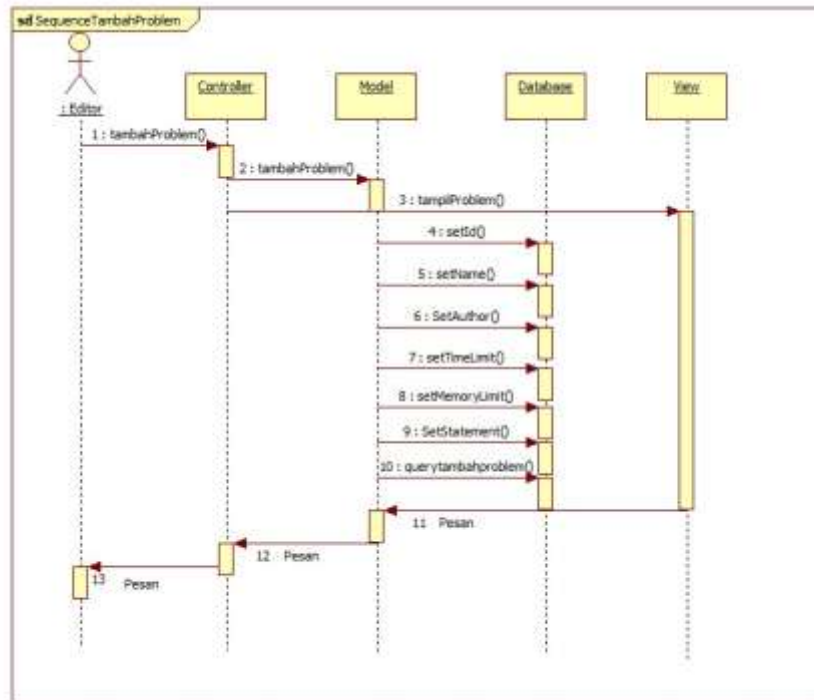
E. *Sequence diagram* menambah *problem*

Sequence diagram menambah *problem* dilakukan dengan 5 buah objek dengan proses sebagai berikut.

- *Editor* memilih tambah *problem* pada *controller problem*.
- *Controller problem* memanggil *model problem* untuk menyiapkan *query* tambah data
- *Model* menyimpan hasil *query* tambah data ke *database*

- Sistem menampilkan pesan berhasil.

Sequence diagram menambah *problem* disajikan pada Gambar 3.61.



Gambar 3.61 *Sequence diagram* menambah *problem*

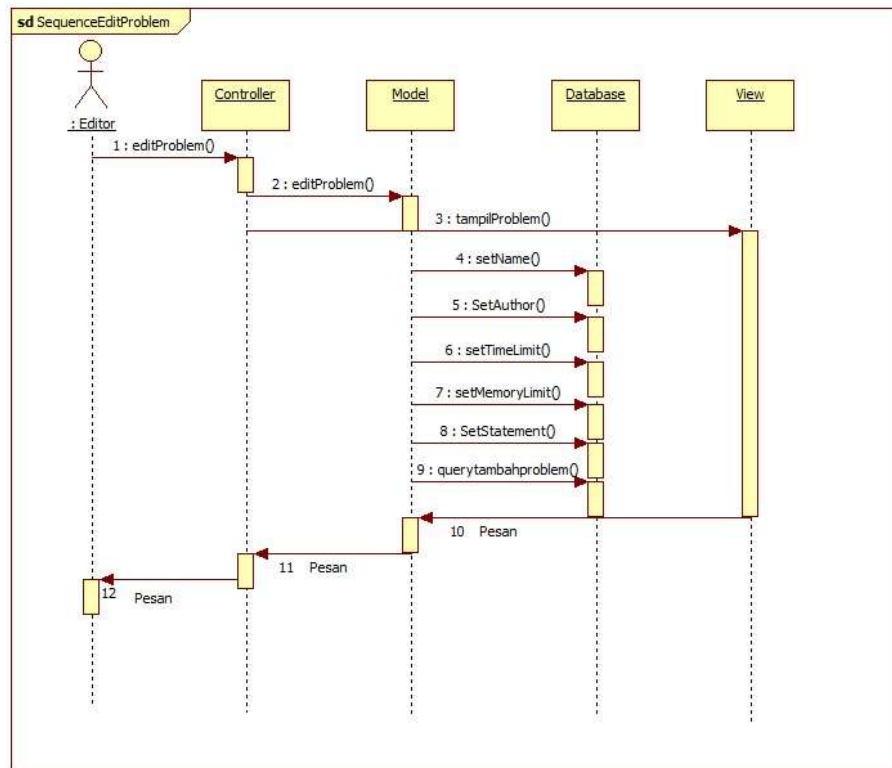
F. *Sequence diagram* mengubah *problem*

Sequence diagram mengubah *problem* dilakukan dengan 5 objek dengan proses sebagai berikut.

- *Editor* memilih menu *edit* pada *controller problem*
- *Controller* memanggil *model problem*
- *Controller* memanggil *view* untuk menampilkan data
- *Editor* mengisi *form*, *controller* memanggil *query edit* data dari *model*

- Sistem menampilkan pesan kepada *editor*.

Sequence diagram mengubah *problem* disajikan pada Gambar 3.62.



Gambar 3.62 *Sequence diagram* mengubah *problem*

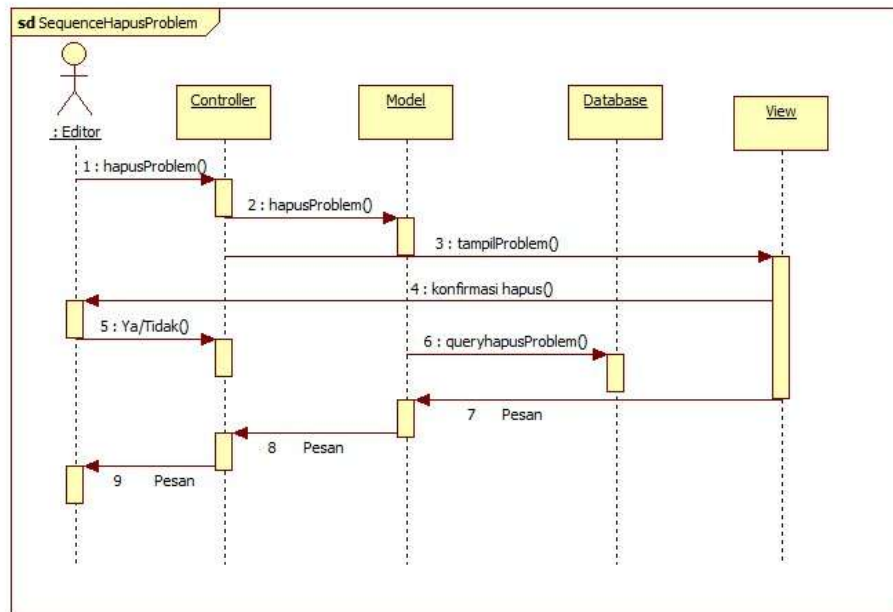
G. *Sequence diagram* menghapus *problem*

Sequence diagram menghapus *problem* dilakukan dengan 5 objek dengan proses sebagai berikut.

- *Editor* memilih menu hapus pada *controller problem*
- *Controller* akan memanggil *model problem*
- *Model* akan meneruskan ke *view* dan menampilkan konfirmasi hapus

- *Editor* memilih hapus atau tidak
- Sistem akan menampilkan pesan

Sequence diagram menghapus *problem* disajikan pada Gambar 3.63.



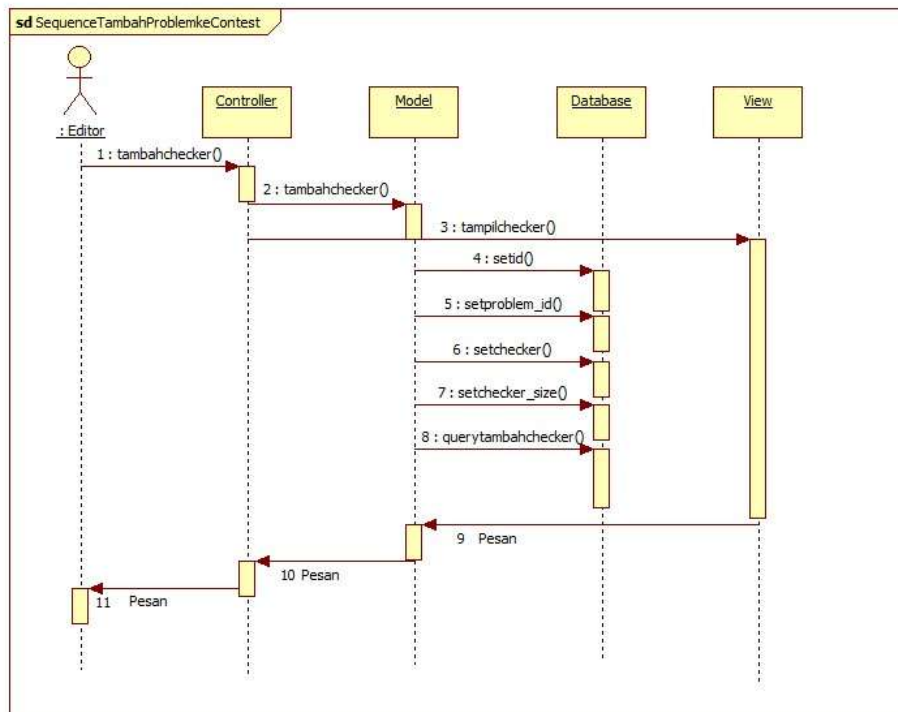
Gambar 3.63 *Sequence diagram* menghapus *problem*

H. *Sequence diagram* menambah *checker*

Sequence diagram menambah *checker* dilakukan dengan 5 buah objek dengan proses sebagai berikut.

- *Editor* memilih tambah *checker* pada *controller problem*.
- *Controller problem* memanggil *model problem* untuk menyiapkan *query* tambah data
- *Model* menyimpan hasil *query* tambah data ke *database*
- Sistem menampilkan pesan berhasil.

Sequence diagram menambah *checker* disajikan pada Gambar 3.64.



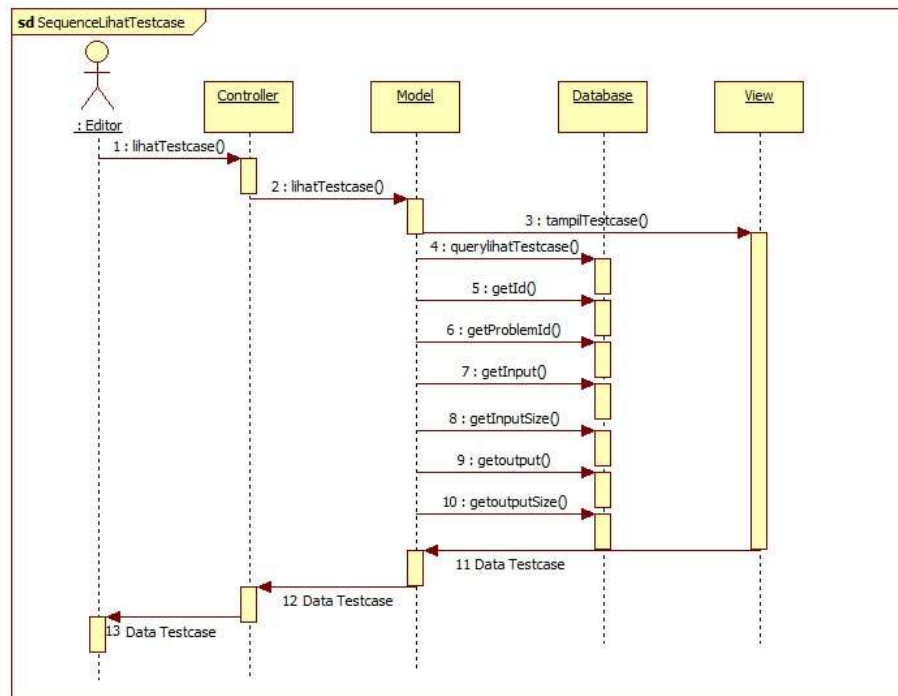
Gambar 3.64 *Sequence diagram* menambah *checker*

I. *Sequence diagram* melihat *testcase*

Sequence diagram melihat *testcase* dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *Editor* memilih menu lihat *testcase* pada *controller testcase*.
- *Controller testcase* mengakses *model testcase* untuk mengambil data dari *database testcase*.
- *Controller* menampilkan data *testcase* ke *view testcase*.

Sequence diagram melihat *testcase* disajikan pada Gambar 3.65.



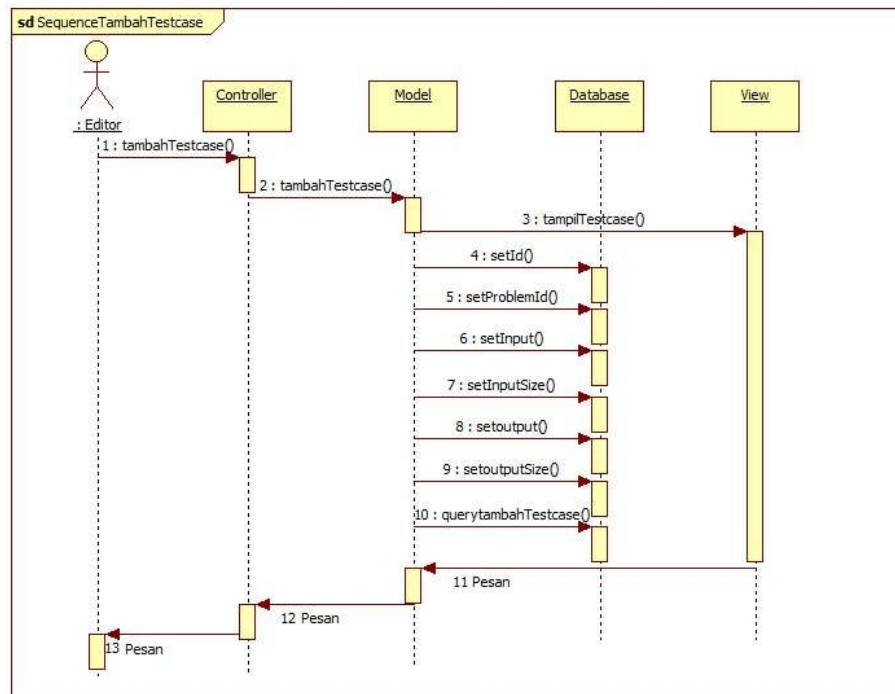
Gambar 3.65 *Sequence diagram* melihat *testcase*

J. *Sequence diagram* menambah *testcase*

Sequence diagram menambah *testcase* dilakukan dengan 5 buah objek dengan proses sebagai berikut.

- *Editor* memilih tambah *testcase* pada *controller testcase*.
- *Controller testcase* memanggil *model testcase* untuk menyiapkan *query* tambah data
- *Model* menyimpan hasil *query* tambah data ke *database*
- Sistem menampilkan pesan berhasil.

Sequence diagram menambah *testcase* disajikan pada Gambar 3.66.



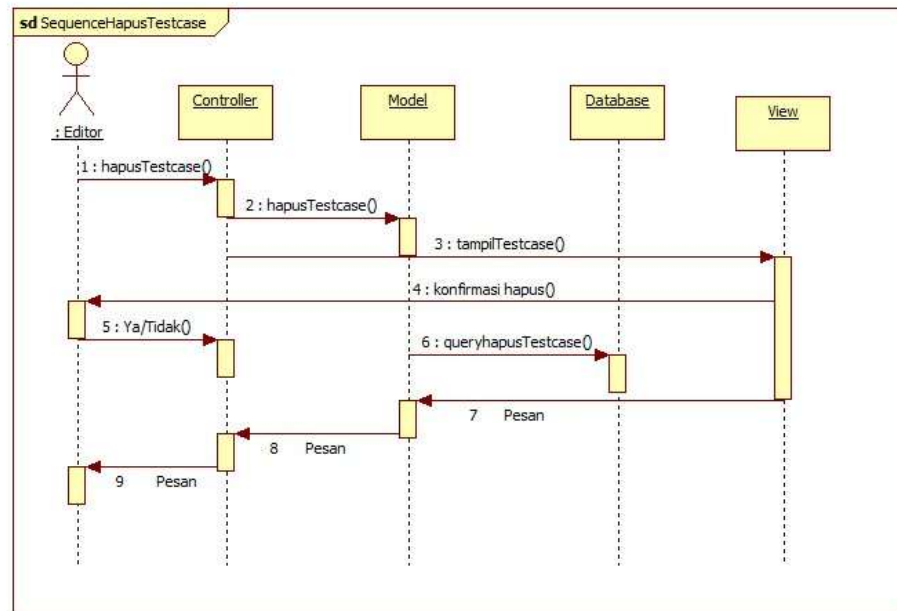
Gambar 3.66 *Sequence diagram* menambah *testcase*

K. *Sequence diagram* menghapus *testcase*

Sequence diagram menghapus *testcase* dilakukan dengan 5 objek dengan proses sebagai berikut.

- *Editor* memilih menu hapus pada *controller testcase*
- *Controller* akan memanggil *model testcase*
- *Model* akan meneruskan ke *view* dan menampilkan konfirmasi hapus
- *Editor* memilih hapus atau tidak
- Sistem akan menampilkan pesan

Sequence diagram menghapus *testcase* disajikan pada Gambar 3.67.



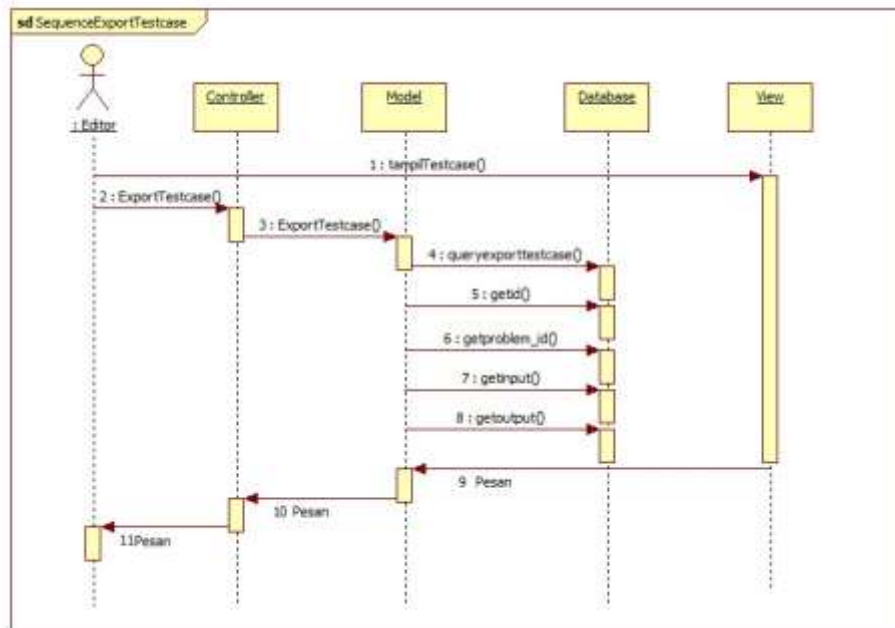
Gambar 3.67 *Sequence diagram* menghapus *testcase*

L. *Sequence diagram* export *testcase*

Sequence diagram export *testcase* dilakukan oleh 5 objek dengan proses sebagai berikut.

- *Editor* memilih *export testcase* pada *controller testcase*
- Sistem menyiapkan berkas *download*
- Sistem menampilkan pesan berhasil

Sequence diagram export *testcase* disajikan pada Gambar 3.68.



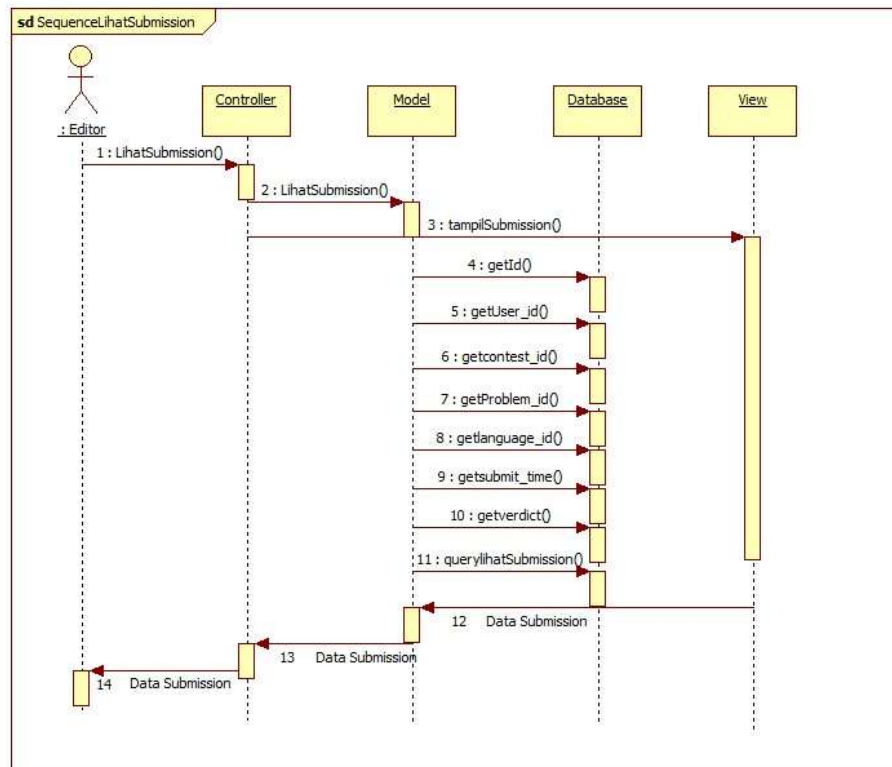
Gambar 3.68 *Sequence diagram export testcase*

M. *Sequence diagram melihat submission*

Sequence diagram melihat *submission* dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *Editor* memilih menu lihat *submission* pada *controller submission*.
- *Controller submission* mengakses *model submission* untuk mengambil data dari *database submission*.
- *Controller* menampilkan data *submission* ke *view submission*.

Sequence diagram melihat *submission* disajikan pada Gambar 3.69.



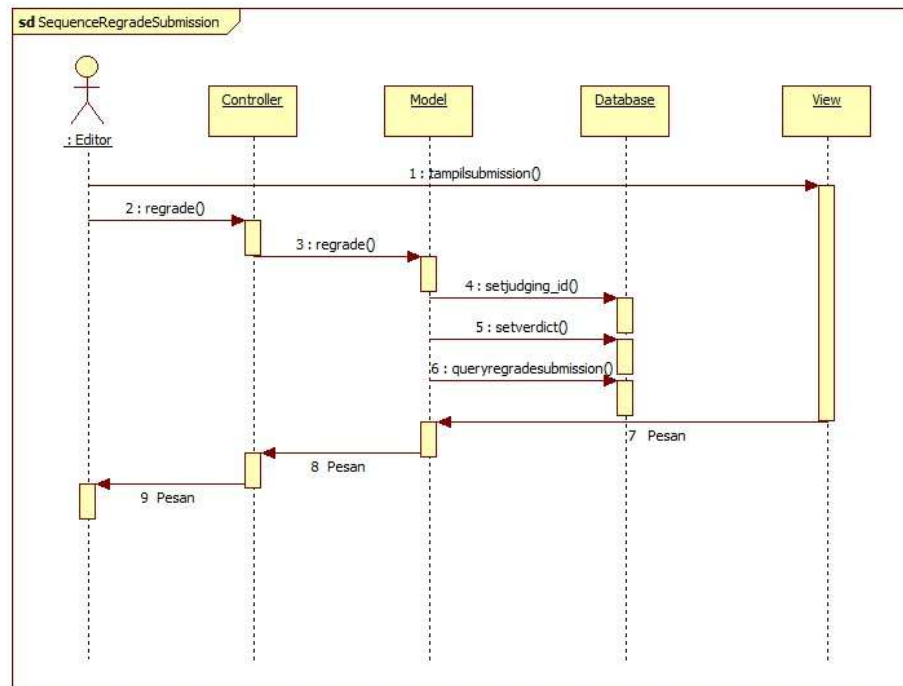
Gambar 3.69 *Sequence diagram* melihat submission

N. *Sequence diagram* regrade submission

Sequence diagram regrade submission dilakukan oleh 5 objek dengan proses sebagai berikut.

- *Editor* memilih menu *regrade* pada *controller submission*
- *Model* akan memberikan perintah ke sistem untuk melakukan proses *regrade*
- *Model* melakukan *query regrade submission* ke tabel *submission*
- Sistem menampilkan pesan berhasil

Sequence diagram regrade submission disajikan pada Gambar 3.70.



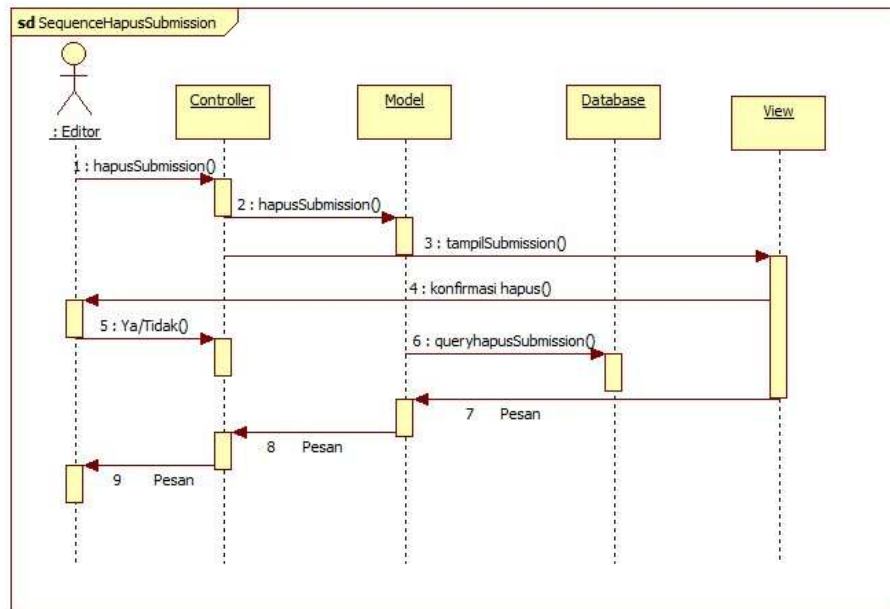
Gambar 3.70 *Sequence diagram regrade submission*

O. *Sequence diagram menghapus submission*

Sequence diagram menghapus *submission* dilakukan dengan 5 objek dengan proses sebagai berikut.

- *Editor* memilih menu hapus pada *controller submission*
- *Controller* akan memanggil *model submission*
- *Model* akan meneruskan ke *view* dan menampilkan konfirmasi hapus
- *Editor* memilih hapus atau tidak
- Sistem akan menampilkan pesan

Sequence diagram menghapus *submission* disajikan pada Gambar 3.71.



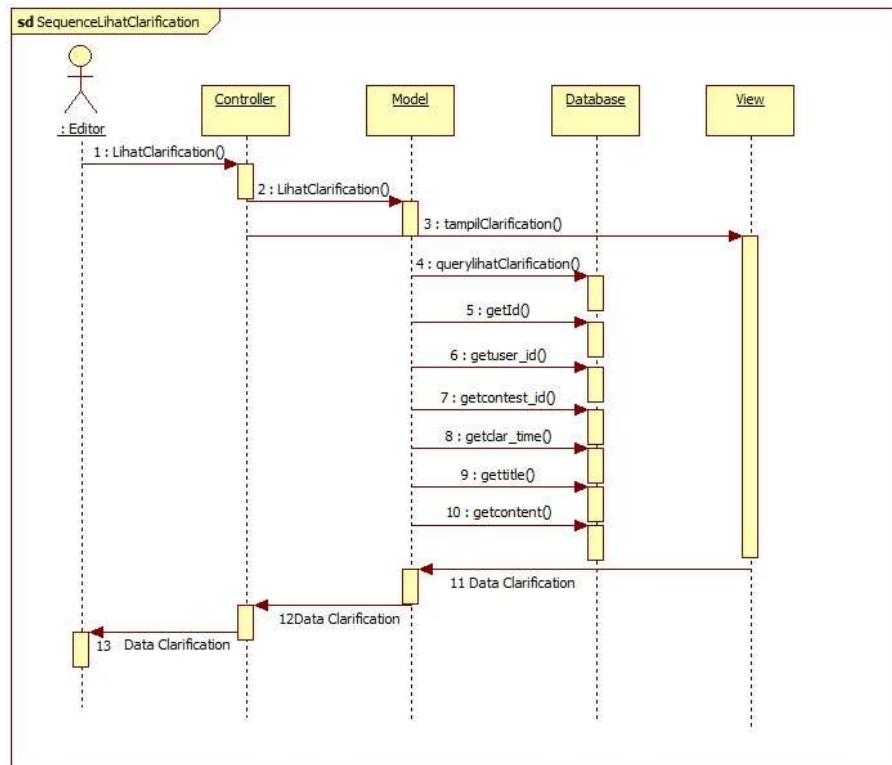
Gambar 3.71 *Sequence diagram* menghapus submission

P. *Sequence diagram* melihat clarification

Sequence diagram melihat clarification dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *Editor* memilih menu lihat clarification pada *controller clarification*.
- *Controller clarification* mengakses *model clarification* untuk mengambil data dari *database clarification*..
- *Controller* menampilkan data *clarification* ke *view clarification*.

Sequence diagram melihat clarification disajikan pada Gambar 3.72.



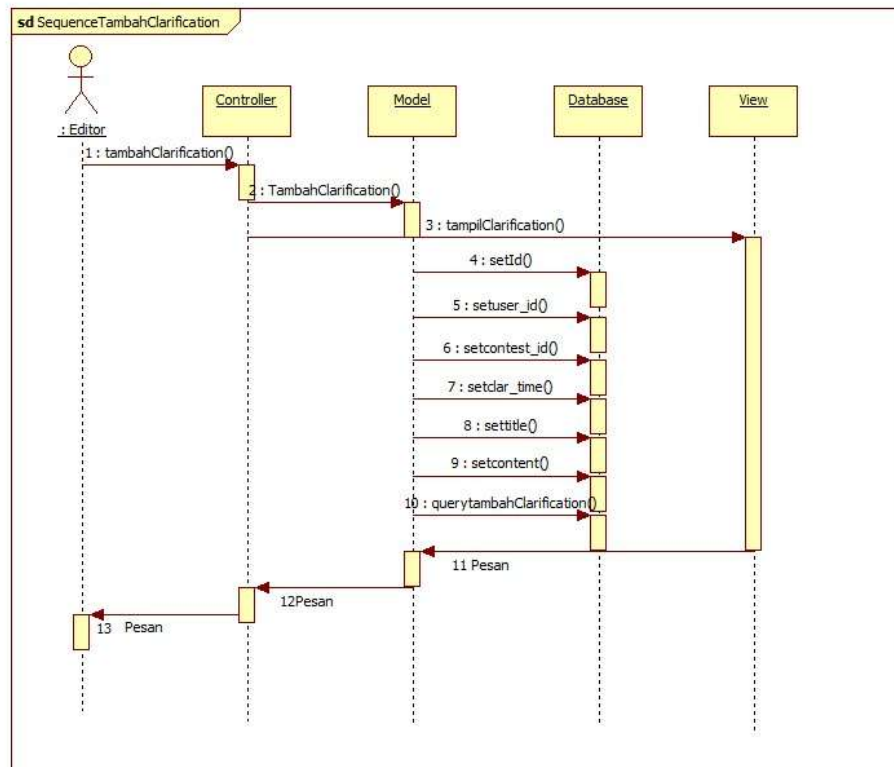
Gambar 3.72 *Sequence diagram* melihat *clarification*

Q. *Sequence diagram* menambah *clarification*

Sequence diagram menambah *clarification* dilakukan dengan 5 buah objek dengan proses sebagai berikut.

- *Editor* memilih tambah *clarification* pada *controller clarification*.
- *Controller clarification* memanggil *model clarification* untuk menyiapkan *query* tambah data
- *Model* menyimpan hasil *query* tambah data ke *database*
- Sistem menampilkan pesan berhasil.

Sequence diagram menambah *clarification* disajikan pada Gambar 3.73.



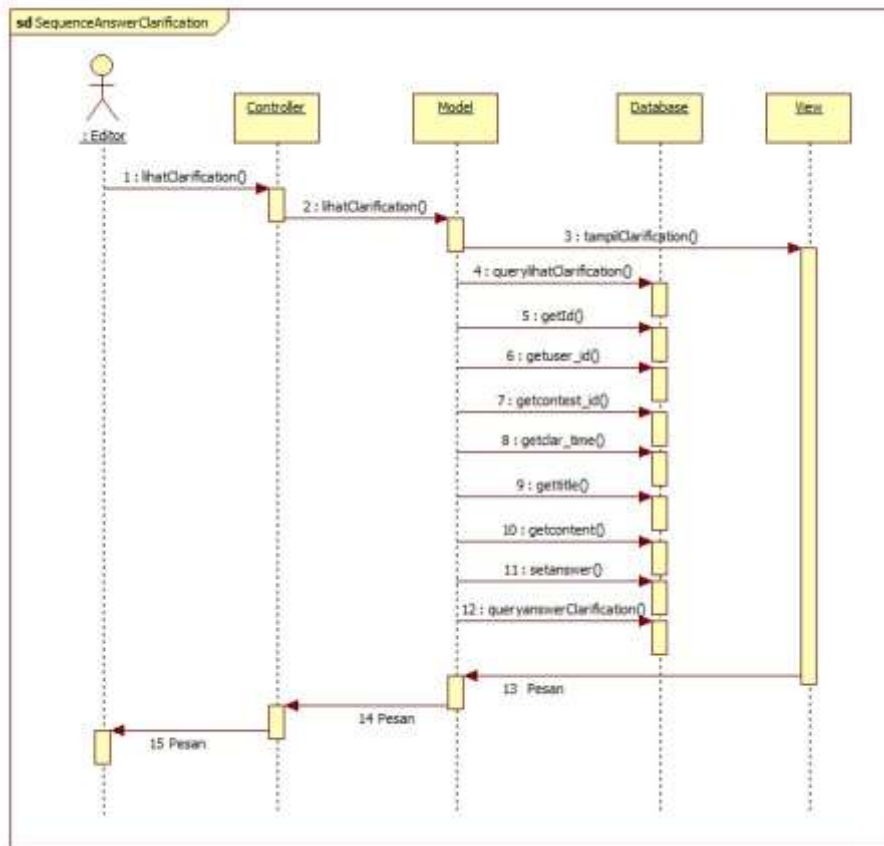
Gambar 3.73 Sequence diagram menambah clarification

R. Sequence diagram menjawab clarification

Sequence diagram menjawab clarification dilakukan dengan 5 buah objek dengan proses sebagai berikut.

- Editor memilih answer clarification pada controller clarification.
- Controller clarification memanggil model clarification untuk menyiapkan query tambah data
- Model menyimpan hasil query tambah data ke database
- Sistem menampilkan pesan berhasil.

Sequence diagram menjawab clarification disajikan pada Gambar 3.74.



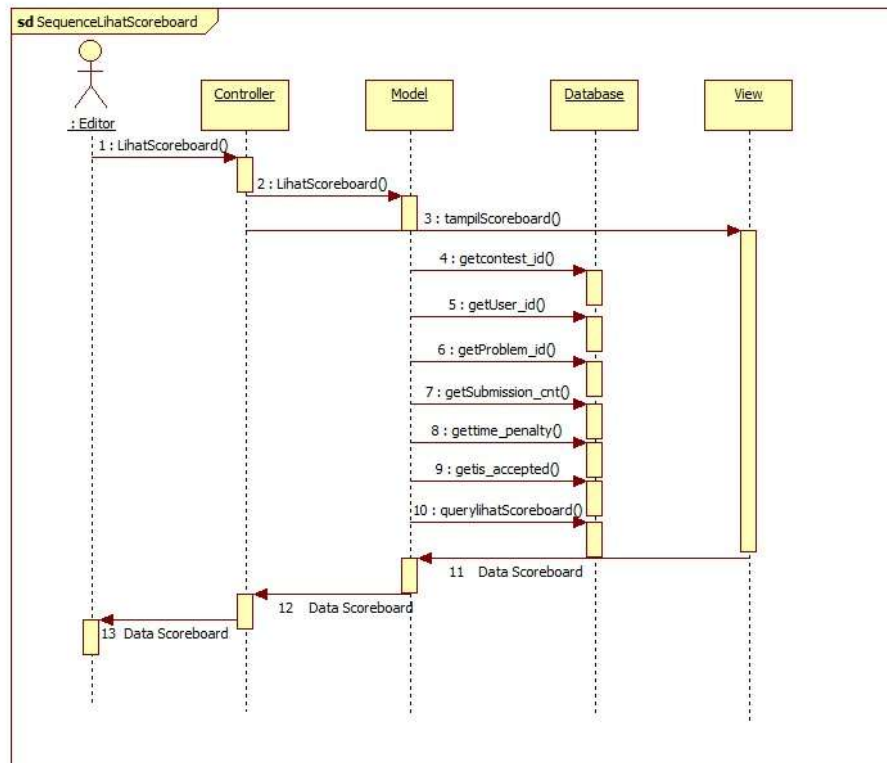
Gambar 3.74 *Sequence diagram* menjawab *clarification*

S. *Sequence diagram* melihat *scoreboard*

Sequence diagram melihat *scoreboard* dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *Editor* memilih menu lihat *scoreboard* pada *controller scoreboard*.
- *Controller scoreboard* mengakses *model scoreboard* untuk mengambil data dari *database scoreboard*.
- *Controller* menampilkan data *scoreboard* ke *view scoreboard*.

Sequence diagram melihat *scoreboard* disajikan pada Gambar 3.75.



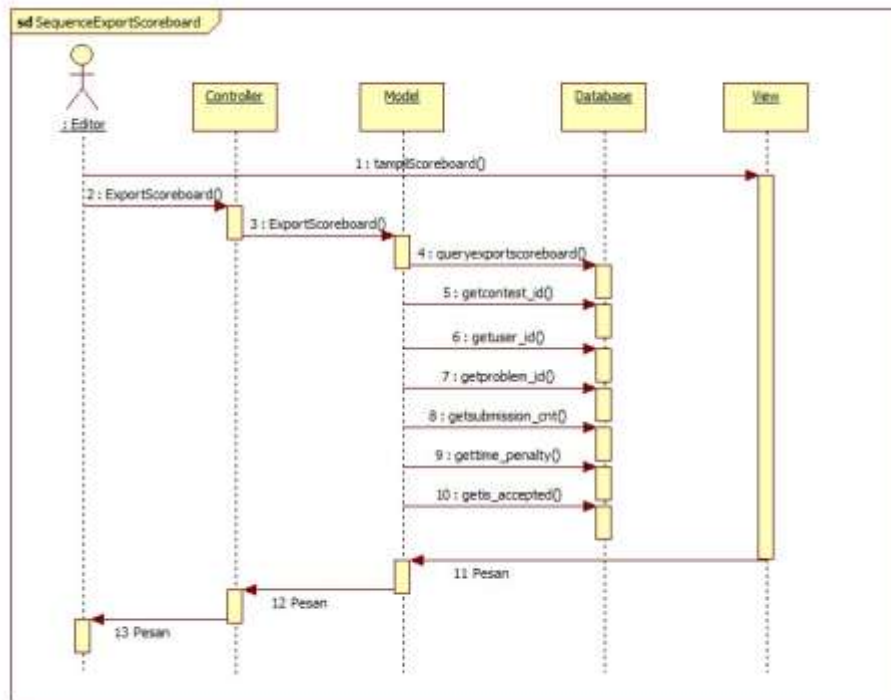
Gambar 3.75 *Sequence diagram* melihat scoreboard

T. *Sequence diagram* export scoreboard

Sequence diagram export scoreboard dilakukan oleh 5 objek dengan proses sebagai berikut.

- *Editor* memilih *export scoreboard* pada *controller scoreboard*
- Sistem menyiapkan berkas *download*
- Sistem menampilkan pesan berhasil

Sequence diagram export scoreboard disajikan pada Gambar 3.76.



Gambar 3.76 *Sequence diagram export scoreboard*

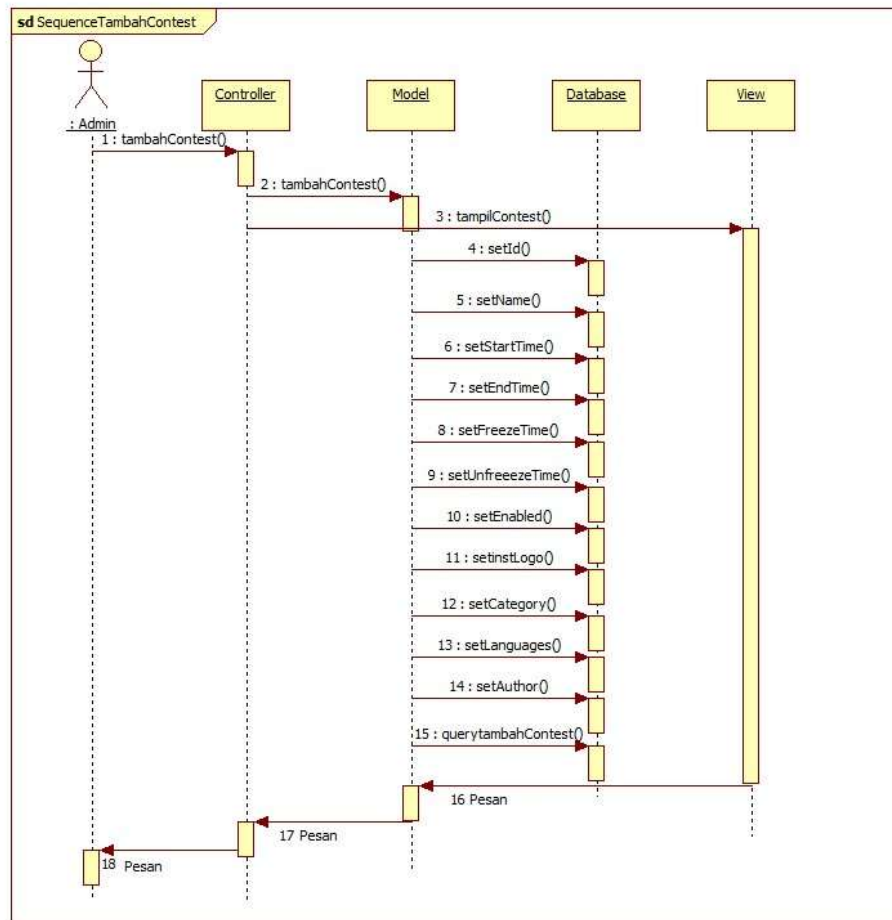
3. Admin

A. *Sequence diagram menambah contest*

Sequence diagram menambah *contest* dilakukan dengan 5 buah objek dengan proses sebagai berikut.

- Admin memilih tambah *contest* pada *controller contest*.
- *Controller contest* memanggil *model contest* untuk menyiapkan *query* tambah data
- *Model* menyimpan hasil *query* tambah data ke *database*
- Sistem menampilkan pesan berhasil.

Sequence diagram menambah *contest* disajikan pada Gambar 3.77.



Gambar 3.77 *Sequence diagram* menambah *contest*

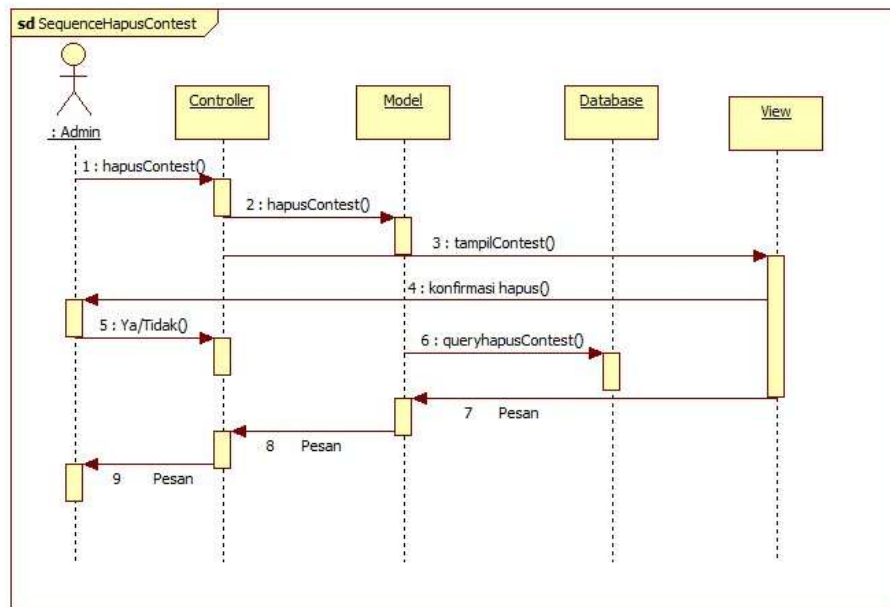
B. *Sequence diagram* menghapus *contest*

Sequence diagram menghapus *contest* dilakukan dengan 5 objek dengan proses sebagai berikut.

- *Admin* memilih menu hapus pada *controller contest*
- *Controller* akan memanggil *model contest*

- *Model* akan meneruskan ke *view* dan menampilkan konfirmasi hapus
- *Admin* memilih hapus atau tidak
- Sistem akan menampilkan pesan

Sequence diagram menghapus *contest* disajikan pada Gambar 3.78.



Gambar 3.78 *Sequence diagram* menghapus *contest*

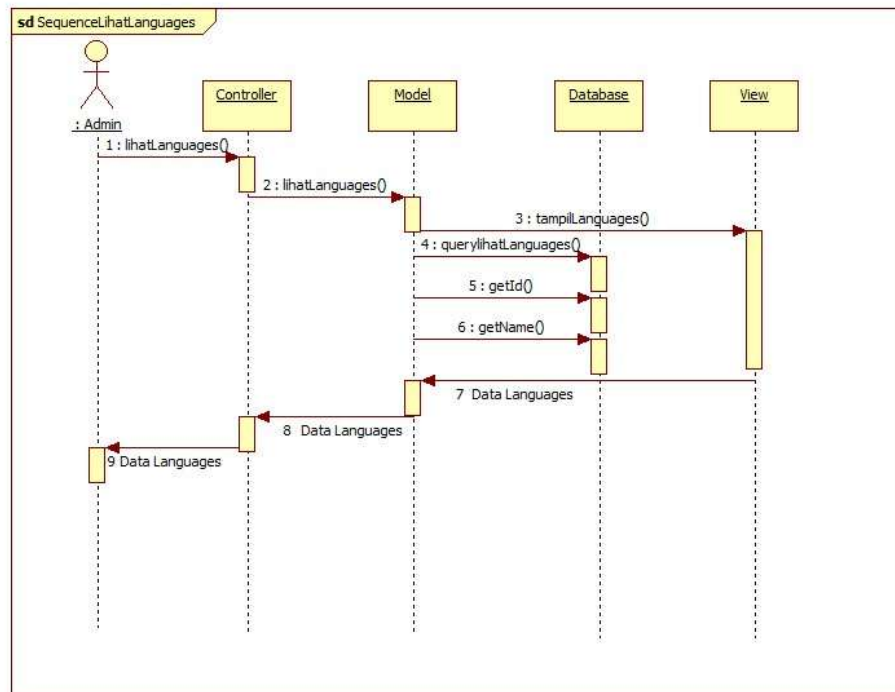
C. *Sequence diagram* melihat *languages*

Sequence diagram melihat *languages* dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *Admin* memilih menu lihat *languages* pada *controller languages*.
- *Controller languages* mengakses *model languages* untuk mengambil data dari *database languages*.

- *Controller* menampilkan data *languages* ke *view languages*.

Sequence diagram melihat *languages* disajikan pada Gambar 3.79.



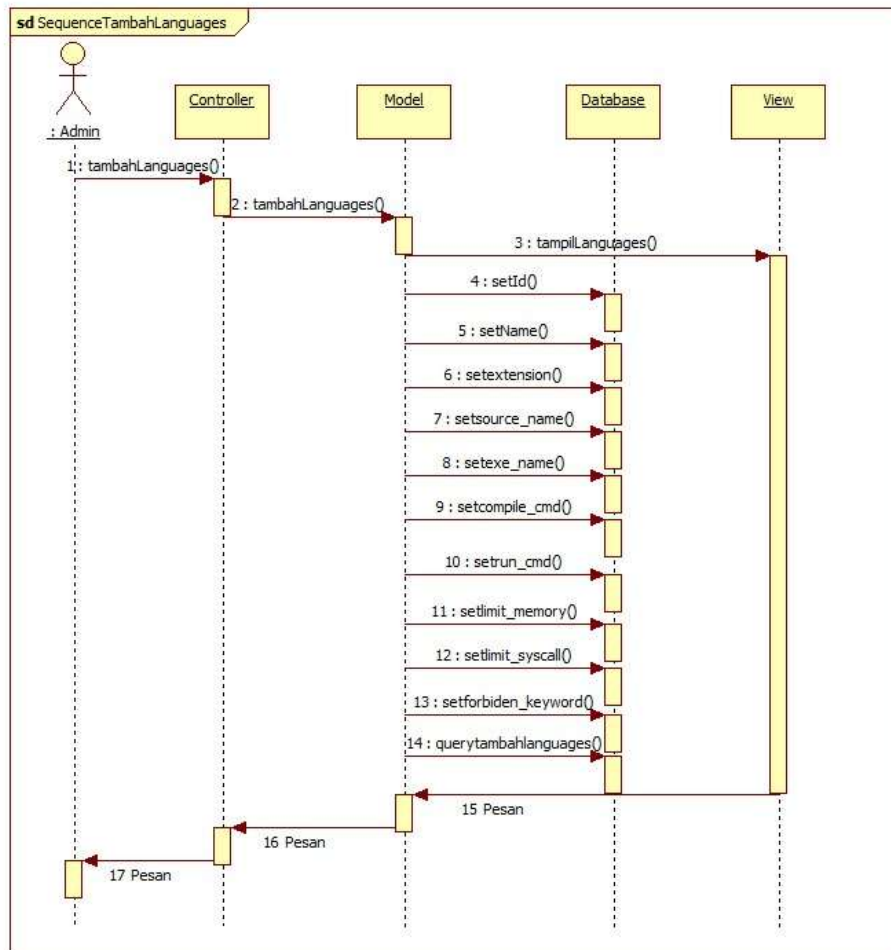
Gambar 3.79 *Sequence diagram* melihat *languages*

D. *Sequence diagram* menambah *languages*

Sequence diagram menambah *languages* dilakukan dengan 5 buah objek dengan proses sebagai berikut.

- *Admin* memilih tambah *languages* pada *controller languages*.
- *Controller languages* memanggil *model languages* untuk menyiapkan *query* tambah data
- *Model* menyimpan hasil *query* tambah data ke *database*
- Sistem menampilkan pesan berhasil.

Sequence diagram menambah *languages* disajikan pada Gambar 3.80.



Gambar 3.80 *Sequence diagram* menambah *languages*

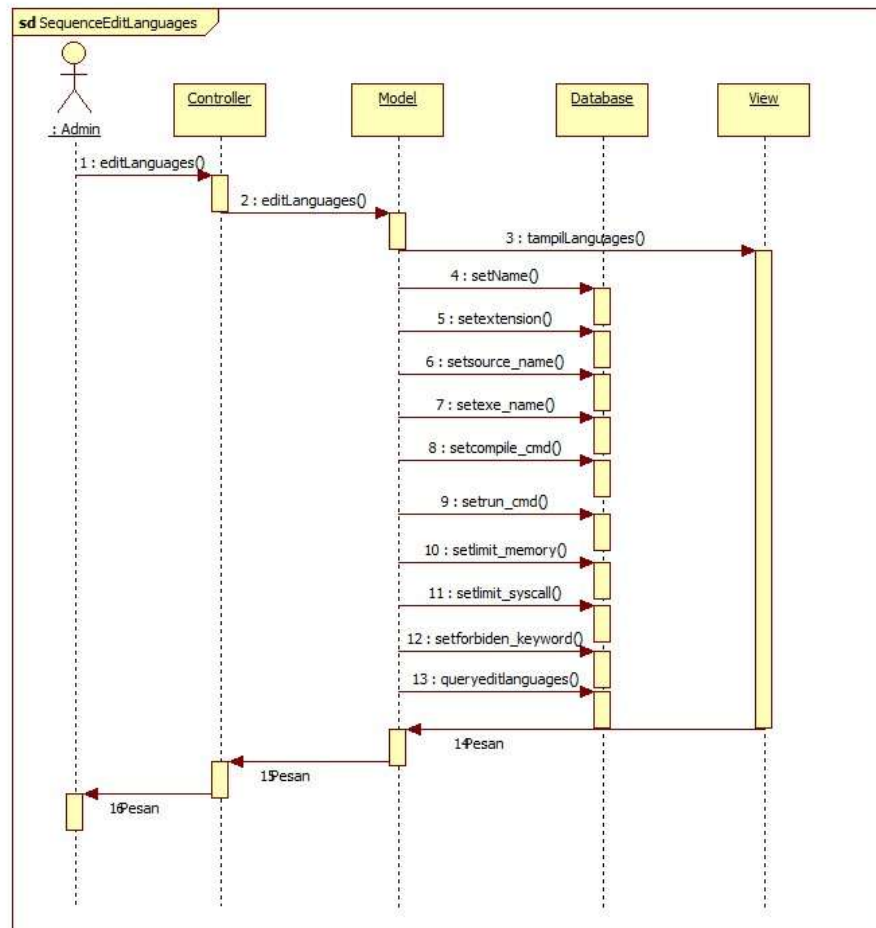
E. *Sequence diagram* mengubah *languages*

Sequence diagram mengubah *languages* dilakukan dengan 5 objek dengan proses sebagai berikut.

- *Admin* memilih menu *edit* pada *controller languages*
- *Controller* memanggil *model languages*

- *Controller* memanggil *view* untuk menampilkan data
- *Admin* mengisi *form*, *controller* memanggil *query edit* data dari *model*
- Sistem menampilkan pesan kepada *editor*.

Sequence diagram mengubah *languages* disajikan pada Gambar 3.81.



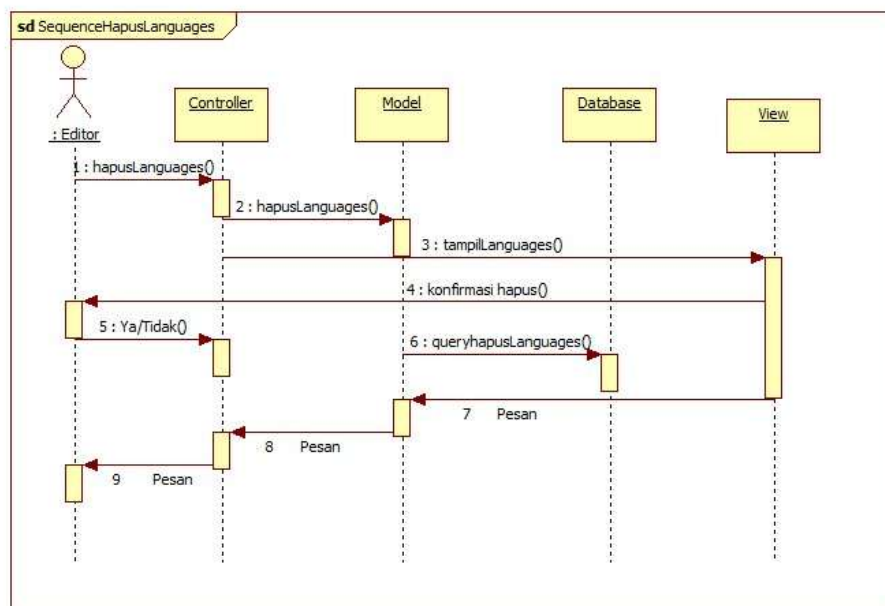
Gambar 3.81 *Sequence diagram* mengubah *languages*

F. *Sequence diagram* menghapus *languages*

Sequence diagram menghapus *languages* dilakukan dengan 5 objek dengan proses sebagai berikut.

- *Admin* memilih menu hapus pada *controller languages*
- *Controller* akan memanggil *model languages*
- *Model* akan meneruskan ke *view* dan menampilkan konfirmasi hapus
- *Admin* memilih hapus atau tidak
- Sistem akan menampilkan pesan

Sequence diagram menghapus *languages* disajikan pada Gambar 3.82.



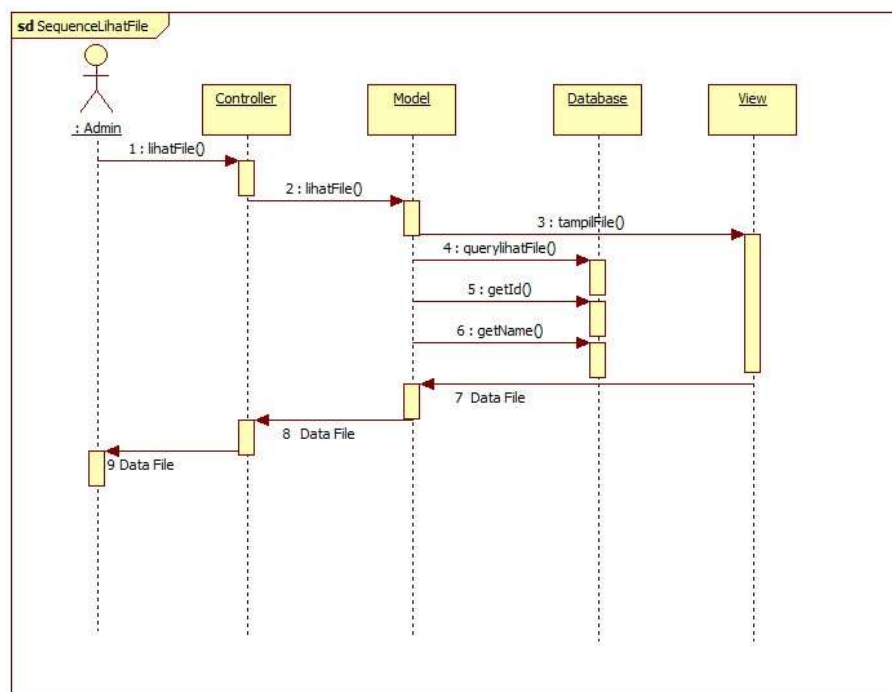
Gambar 3.82 *Sequence diagram* menghapus *languages*

G. *Sequence diagram melihat file*

Sequence diagram melihat *file* dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *Admin* memilih menu lihat *file* pada *controller file*.
- *Controller file* mengakses *model file* untuk mengambil data dari *database file*.
- *Controller* menampilkan data *file* ke *view file*.

Sequence diagram melihat *file* disajikan pada Gambar 3.83.



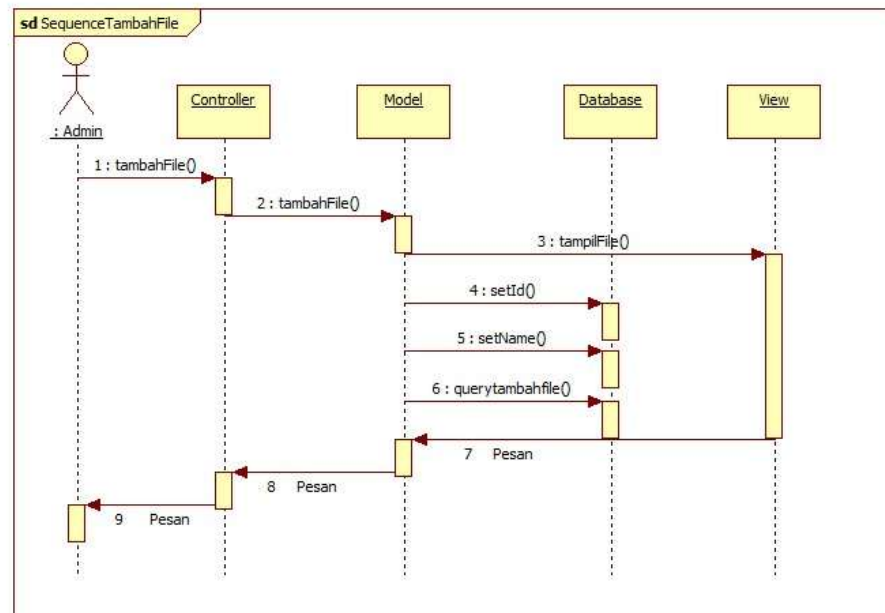
Gambar 3.83 *Sequence diagram* melihat *file*

H. *Sequence diagram* menambah *file*

Sequence diagram menambah *file* dilakukan dengan 5 buah objek dengan proses sebagai berikut.

- *Admin* memilih tambah *file* pada *controller file*.
- *Controller file* memanggil *model file* untuk menyiapkan *query* tambah data
- *Model* menyimpan hasil *query* tambah data ke *database*
- Sistem menampilkan pesan berhasil.

Sequence diagram menambah *file* disajikan pada Gambar 3.84.



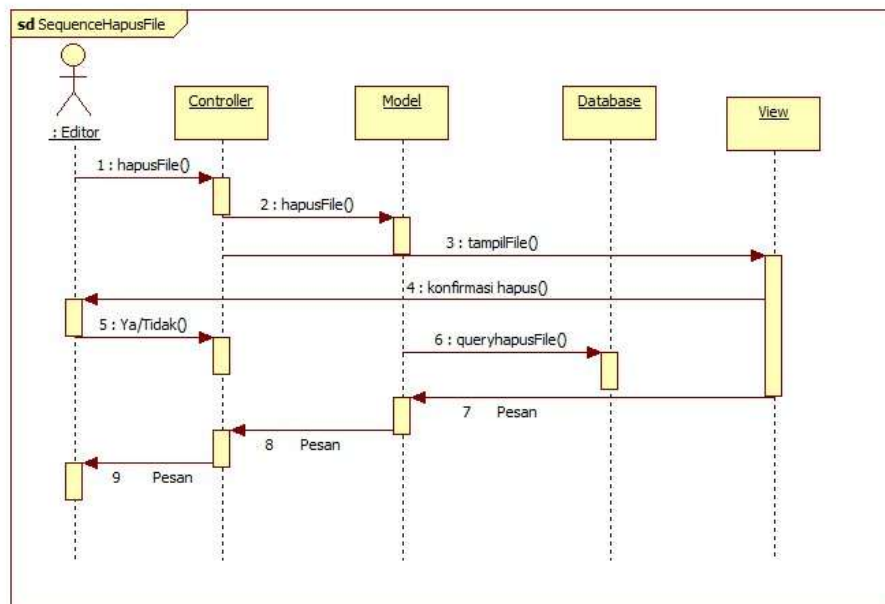
Gambar 3.84 *Sequence diagram* menambah *file*

I. *Sequence diagram* menghapus *file*

Sequence diagram menghapus *file* dilakukan dengan 5 objek dengan proses sebagai berikut.

- *Admin* memilih menu hapus pada *controller file*
- *Controller* akan memanggil *model file*
- *Model* akan meneruskan ke *view* dan menampilkan konfirmasi hapus
- *Admin* memilih hapus atau tidak
- Sistem akan menampilkan pesan

Sequence diagram menghapus *file* disajikan pada Gambar 3.85.



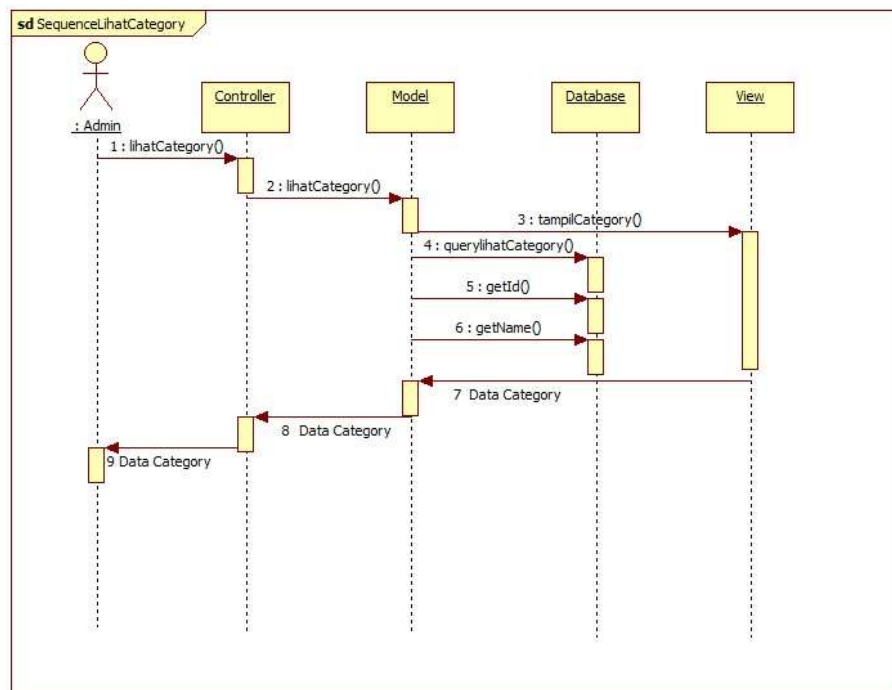
Gambar 3.85 *Sequence diagram* menghapus *file*

J. *Sequence diagram melihat category*

Sequence diagram melihat *category* dilakukan dengan adanya 5 objek dengan proses sebagai berikut.

- *Admin* memilih menu lihat *category* pada *controller category*.
- *Controller category* mengakses *model category* untuk mengambil data dari *database category*.
- *Controller* menampilkan data *category* ke *view category*.

Sequence diagram melihat *category* disajikan pada Gambar 3.86.



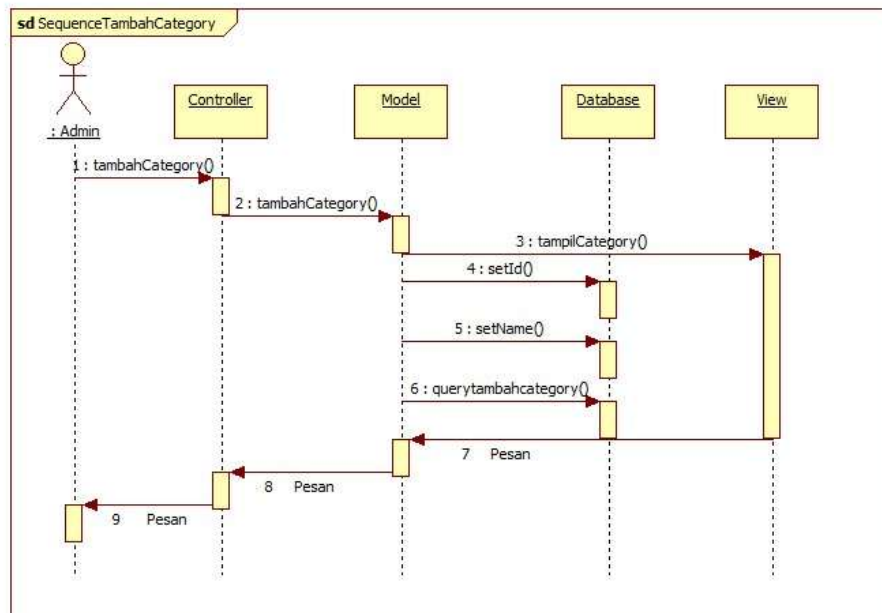
Gambar 3.86 *Sequence diagram* melihat *category*

K. *Sequence diagram* menambah *category*

Sequence diagram menambah *category* dilakukan dengan 5 buah objek dengan proses sebagai berikut.

- *Admin* memilih tambah *category* pada *controller category*.
- *Controller category* memanggil *model category* untuk menyiapkan *query* tambah data
- *Model* menyimpan hasil *query* tambah data ke *database*
- Sistem menampilkan pesan berhasil.

Sequence diagram menambah *category* disajikan pada Gambar 3.87.



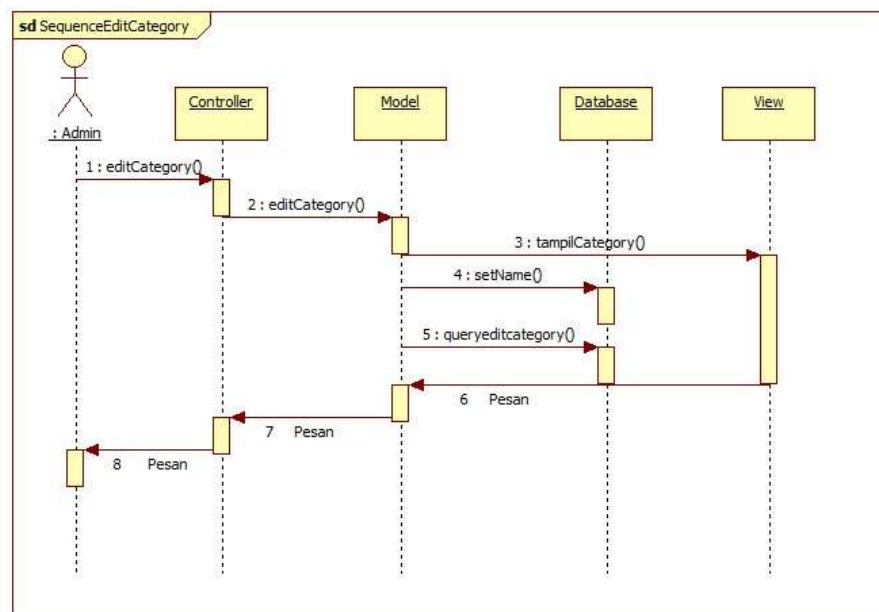
Gambar 3.87 *Sequence diagram* menambah *category*

L. *Sequence diagram* mengubah *category*

Sequence diagram mengubah *category* dilakukan dengan 5 objek dengan proses sebagai berikut.

- *Admin* memilih menu *edit* pada *controller category*
- *Controller* memanggil *model category*
- *Controller* memanggil *view* untuk menampilkan data
- *Admin* mengisi *form*, *controller* memanggil *query edit* data dari *model*
- Sistem menampilkan pesan kepada *editor*.

Sequence diagram mengubah *category* disajikan pada Gambar 3.88.



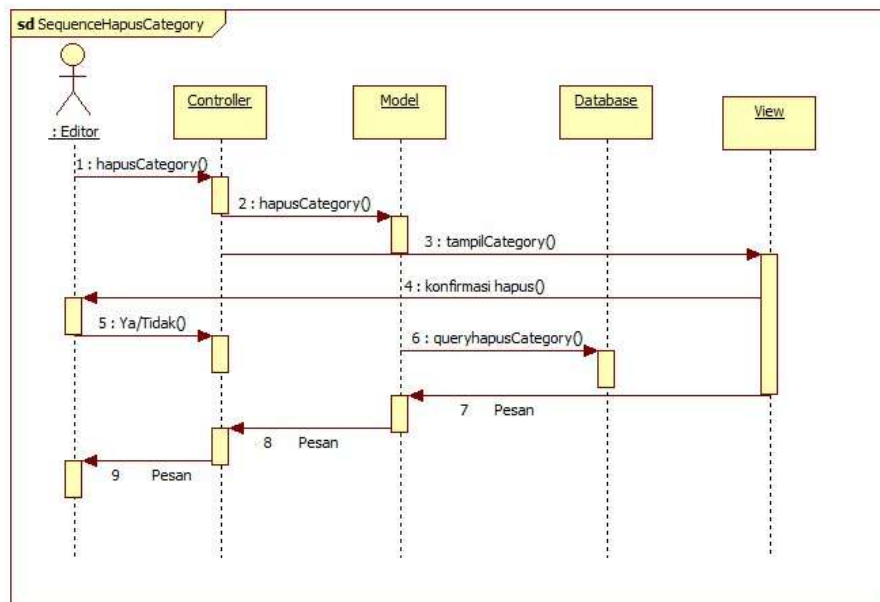
Gambar 3.88 *Sequence diagram* mengubah *category*

M. *Sequence diagram* menghapus *category*

Sequence diagram menghapus *category* dilakukan dengan 5 objek dengan proses sebagai berikut.

- *Admin* memilih menu hapus pada *controller category*
- *Controller* akan memanggil *model category*
- *Model* akan meneruskan ke *view* dan menampilkan konfirmasi hapus
- *Admin* memilih hapus atau tidak
- Sistem akan menampilkan pesan

Sequence diagram menghapus *category* disajikan pada Gambar 3.89.



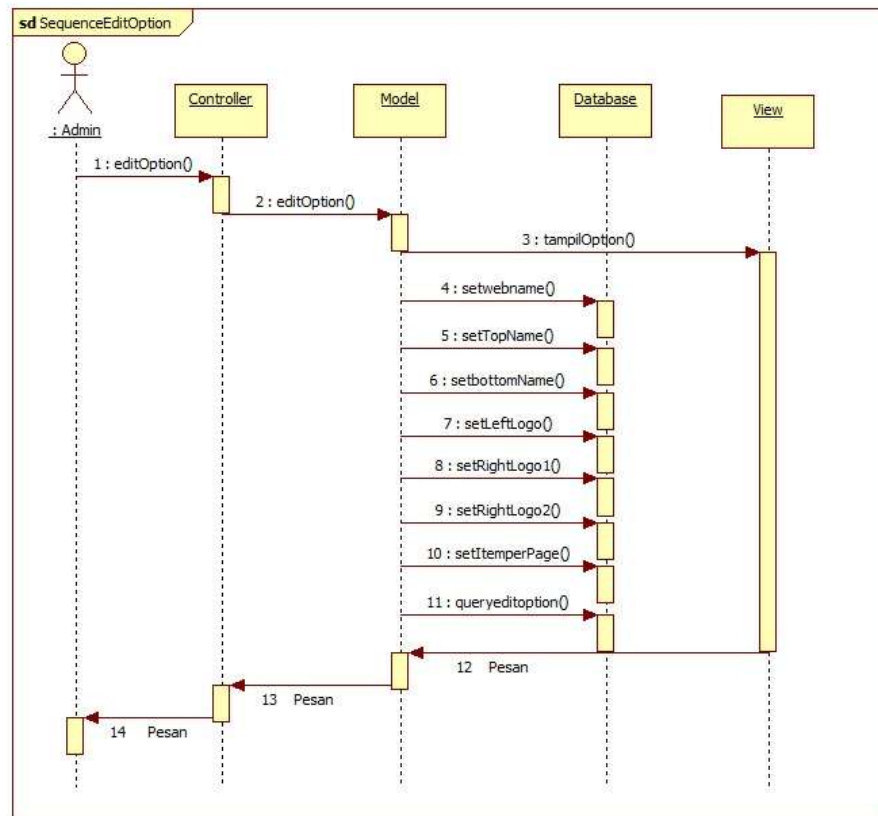
Gambar 3.89 *Sequence diagram* menghapus *category*

N. Sequence diagram mengubah option

Sequence diagram mengubah contest dilakukan dengan 5 objek dengan proses sebagai berikut.

- Admin memilih menu *edit* pada controller option
- Controller memanggil model option
- Controller memanggil view untuk menampilkan data
- Admin mengisi form, controller memanggil query edit dari model
- Sistem menampilkan pesan kepada editor.

Sequence diagram mengubah option disajikan pada Gambar 3.90.



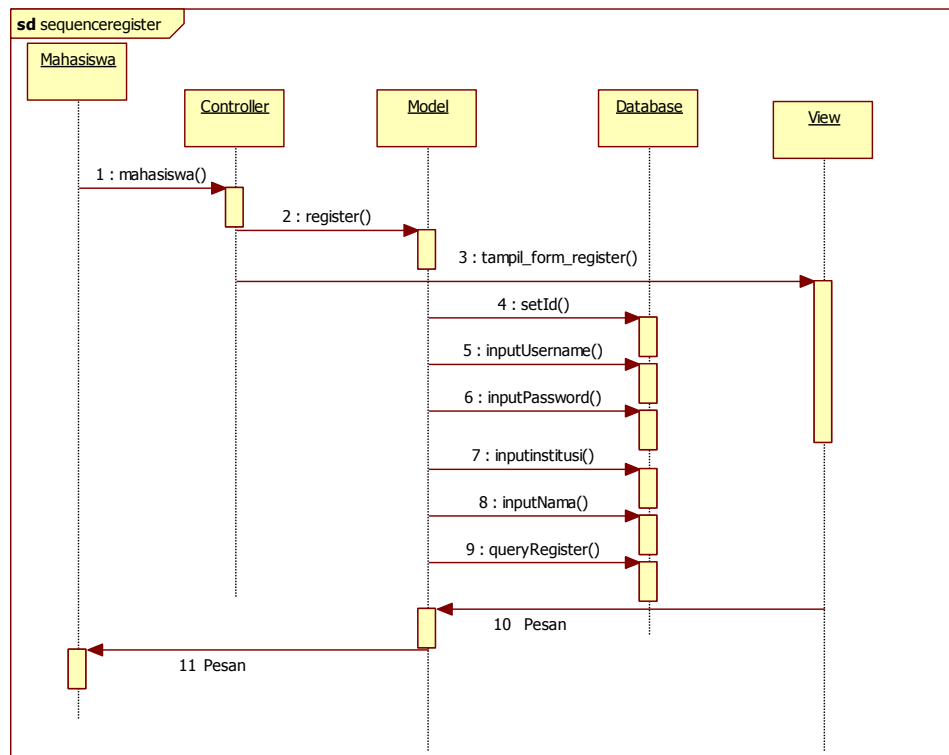
Gambar 3.90 Sequence diagram mengubah option

O. *Sequence diagram register mahasiswa*

Sequence diagram mengubah *contest* dilakukan dengan 4 objek dengan proses sebagai berikut.

- Mahasiswa memilih *register* pada *controller* mahasiswa.
- *Controller mahasiswa* memanggil *model user* untuk menyiapkan *query register*.
- *Model* menyimpan hasil *query register* mahasiswa ke *database*
- Sistem mengarahkan ke halaman *login*.

Sequence diagram register mahasiswa disajikan pada Gambar 3.91.



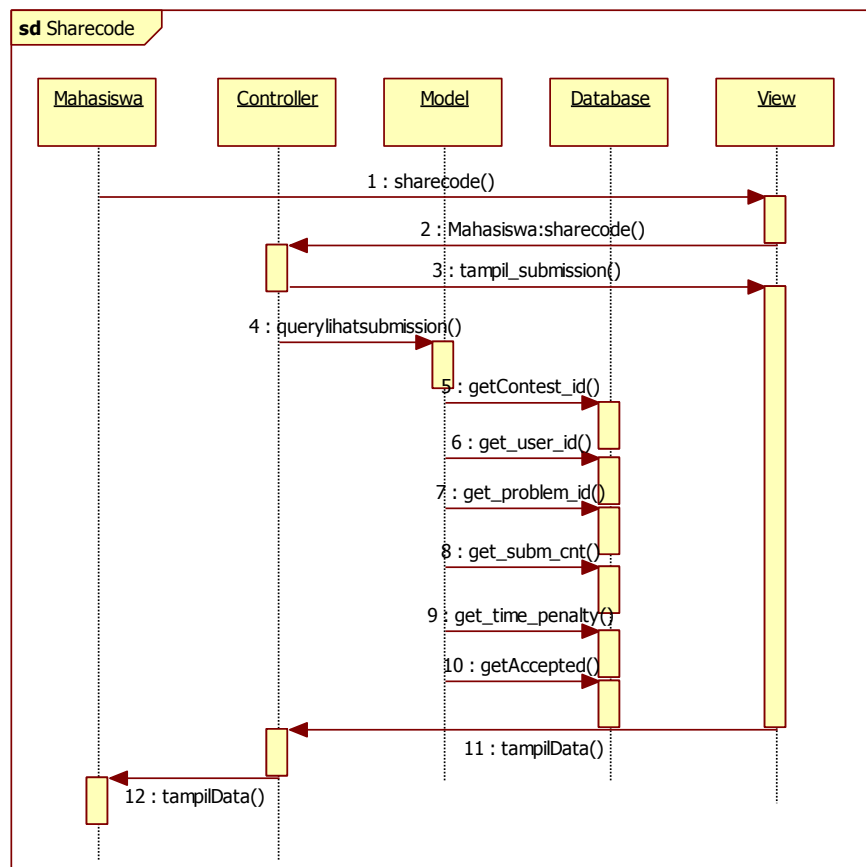
Gambar 4.91 *Sequence diagram register* mahasiswa

P. *Sequence diagram sharecode*

Sequence diagram sharecode dilakukan dengan 4 objek dengan proses sebagai berikut.

- Mahasiswa memilih sharecode *di controller mahasiswa*
- *Controller* memanggil *model submission*
- *Controller* memanggil *view* untuk menampilkan data
- *Database* menyiapkan data untuk ditampilkan di *view*.

Sequence diagram mengubah *option* disajikan pada Gambar 3.92.



Gambar 4.92 *Sequence diagram sharecode*

3.3.5. *Coding System*

Pada tahap ini akan dilakukan pembuatan kode program (*coding*), tahap *coding* sistem dengan mengimplementasikan permasalahan yang sudah didapatkan ke dalam sistem berdasarkan desain yang sudah dibuat. Pada proses *coding* sistem, digunakan bahasa pemrograman PHP dengan menggunakan *framework CodeIgniter*. *Database* yang digunakan untuk membuat sistem ini adalah *MySQL database*.

3.3.6. *Testing*

Setelah proses *coding* selesai, maka dilakukan proses pengujian dengan menggunakan metode *black box testing*. *Black box testing* merupakan tahap pengujian yang memperhatikan fungsionalitas dari sistem. Proses ini dilakukan untuk memastikan sistem yang dibuat telah bekerja sesuai dengan ketentuan yang telah ditetapkan. Pada pengujian ini, dilakukan dengan membagi domain masukan ke beberapa *testcase* yang dibuat. Pengujian ini dibagi menjadi 3 aktor, yaitu *admin*, *editor* dan *mahasiswa/contestant*.

Tabel 3.1 Tabel rencana pengujian pada aktor *admin*

No.	Kelas Uji	Daftar Pengujian	Kasus Uji	Hasil Yang diharapkan
1.	<i>Login</i>	<i>Login User</i>	Tidak mengisi salah satu <i>field</i>	Menolak dan memberi informasi untuk mengisi <i>field</i>
			Mengisi <i>username</i> dan/atau	Sistem menolak, dan memberi informasi bahwa
			<i>password</i> yang salah	<i>username/password</i> salah
			Mengisi <i>field username</i> dan <i>password</i> yang sesuai	Berhasil, diarahkan ke halaman <i>dashboard</i>
2.	Mengelola Kelas atau <i>Contest</i>	Melihat Kelas / <i>Contest</i>	Memilih menu <i>manage</i> dan memilih submenu <i>contest</i>	Menampilkan halaman <i>contest</i> / kelas yang di kelola
		Menambah <i>contest</i> / kelas	Tidak mengisi beberapa <i>field mandatory</i>	Menampilkan pesan <i>error</i>
			Mengisi semua <i>field mandatory</i>	Berhasil menyimpan dan diarahkan ke halaman <i>contest</i>
		Mengedit Kelas / <i>Contest</i>	Memilih <i>icon</i> pensil pada list <i>contest</i> /kelas yang tersedia di halaman <i>contest</i>	Mengarahkan ke halaman <i>edit</i> kontes / kelas
			Menekan tombol simpan, tanpa mengubah data	Data tersimpan dan diarahkan ke halaman kontes
			Mengubah data yang ada dan menekan tombol simpan	Data tersimpan dan diarahkan ke halaman kontes

Tabel 3.1. Tabel rencana pengujian pada aktor *admin* (Lanjutan)

3.	Mengelola <i>Problem / Soal</i>	Melihat soal / <i>problem</i>	Memilih menu <i>manage</i> dan memilih submenu <i>problem</i>	Menampilkan halaman dan <i>list problem</i>
		Menambah <i>problem / soal</i>	Tidak mengisi salah satu <i>field</i> yang tersedia	Sistem tidak menyimpan data, dan menampilkan pesan kesalahan
			Mengisi seluruh <i>field</i> yang tersedia	Sistem akan menyimpan <i>problem/soal</i> dan
				diarahkan ke halaman <i>problem</i>
		Mengedit soal / <i>problem</i>	Mengubah beberapa data di <i>field</i> dan mengosongkan <i>field</i>	Sistem akan menolak dan meminta <i>editor</i> untuk mengisi <i>field</i>
			Mengubah data dan menyimpan	Sistem akan menyimpan perubahan
		Menambah <i>checker</i>	Menekan tombol <i>add</i> tanpa mengunggah <i>file</i>	Menampilkan pesan <i>error</i>
			Menekan tombol <i>add</i> dengan mengunggah <i>file</i>	Data disimpan
		Menambah <i>test case</i>	Mengunggah salah satu <i>file</i> , <i>input</i> atau <i>output</i> , atau tidak mengunggah keduanya	Menampilkan pesan <i>error</i>

Tabel 3.1. Tabel rencana pengujian pada aktor *admin* (Lanjutan)

			Mengunggah <i>file</i> dengan ekstensi selain <i>.txt</i>	
			Mengunggah <i>file</i> dengan ekstensi <i>.txt</i>	Data <i>test case</i> berhasil di simpan dan di tampilkan di atas <i>form</i>
		Mengunduh <i>test case</i>	Menekan <i>icon download</i> pada halaman <i>testcase</i>	Sistem menyiapkan <i>file download</i> dengan ekstensi <i>.txt</i>
4.	Mengelola Bahasa	Melihat bahasa	Memilih menu <i>manage</i> dan submenu <i>languages</i>	Menampilkan halaman <i>languages</i>
		Menambah bahasa	Mengosongkan beberapa <i>field mandatory</i>	Menampilkan pesan <i>error</i>
			Mengisi semua <i>field mandatory</i>	Menampilkan pesan berhasil dan diarahkan ke halaman <i>languages</i>
5.	Mengelola <i>Category</i>	Melihat <i>category</i>	Memilih menu <i>manage</i> dan submenu <i>categories</i>	Menampilkan halaman <i>category</i>
		Menambah <i>category</i>	Tidak mengisi <i>field</i> dan menekan tombol simpan	Menampilkan pesan <i>error</i>
			Mengisi <i>field</i> dan menyimpan	Menampilkan pesan berhasil dan mengarahkan ke halaman <i>category</i>

Tabel 3.1. Tabel rencana pengujian pada aktor *admin* (Lanjutan)

6.	Mengelola <i>User</i>	Melihat <i>User</i>	Memilih menu <i>manage</i> dan memilih submenu <i>user</i>	Menampilkan semua <i>user</i> dari kategori kontes yang dikelola
		Menambah <i>user</i>	Tidak mengisi beberapa <i>field</i> yang tersedia	Menampilkan pesan <i>error</i>
			Mengisi semua <i>field</i> dengan <i>username</i> yang sudah tersedia	Menampilkan pesan <i>error</i> bahwa <i>username</i> sudah digunakan
			Mengisi semua <i>field</i> dengan data baru	Data disimpan
		Mencetak data <i>user</i>	Memilih tombol <i>print user</i>	Sistem menyiapkan <i>file</i> dengan ekstensi <i>.xls</i>
7.	Mengelola <i>Options</i>	Mengubah informasi <i>website</i>	Mengosongkan salah satu atau semua <i>field mandatory</i>	Menampilkan pesan <i>error</i>
			Mengisi semua <i>field mandatory</i>	Menampilkan informasi bahwa data berhasil di simpan
8.	Mengelola <i>Files</i>	Menambah <i>file</i> (gambar)	Menekan tombol <i>add</i> tanpa mengunggah <i>file</i>	Menampilkan pesan <i>error</i>
			Mengunggah <i>file</i> dan menekan tombol <i>add</i>	Menampilkan pesan berhasil dan diarahkan ke halaman <i>files</i>
9.	Mengelola <i>Submission</i>	Melihat <i>submission</i>	Memilih <i>icon</i> lihat <i>submission</i>	Menampilkan <i>detail submission</i> yang di pilih

Tabel 3.1. Tabel rencana pengujian pada aktor *admin* (Lanjutan)

		<i>Regrade submission</i>	Memilih <i>icon regrade submission</i>	Mengubah status <i>verdict</i> dan melakukan <i>grader</i> ulang
10.	Mengelola <i>Clarification</i>	Melihat <i>clarification</i>	Memilih menu <i>clarification</i>	Menampilkan halaman <i>clarification</i>
		Menambah <i>clarification</i>	Tidak mengisi judul atau konten <i>clarification</i>	Menampilkan pesan kesalahan
			Mengisi judul dan konten <i>clarification</i>	Sistem menyimpan <i>clarification</i> , dan diarahkan ke halaman <i>clarification</i>
		Menjawab <i>clarification</i>	Memilih <i>clarification</i> , dan mengisi jawaban di kolom jawaban	Menyimpan jawaban dan mengarahkan ke halaman <i>clarification</i>
11.	Melihat <i>Scoreboard</i>	Melihat <i>Scoreboard</i>	Memilih menu <i>scoreboard</i>	Menampilkan halaman <i>scoreboard admin</i>
		<i>export scoreboard</i>	Memilih kelas dan memilih tombol <i>export scoreboard</i>	Sistem menyiapkan <i>file</i> berekstensi <i>.xls</i>

Tabel 3.2 Tabel rencana pengujian pada aktor *editor*

No	Kelas Uji	Daftar Pengujian	Kasus Uji	Hasil Yang diharapkan
1.	<i>Login</i>	<i>Login User</i>	Tidak mengisi salah satu <i>field</i>	Menolak dan memberi informasi untuk mengisi <i>field</i>
			Mengisi <i>username</i> dan/atau <i>password</i> yang salah	Sistem menolak, dan memberi informasi bahwa <i>username/password</i> salah
			Mengisi <i>field username</i> dan <i>password</i> yang sesuai	Berhasil, diarahkan ke halaman <i>dashboard</i>
2.	Mengelola Kelas atau <i>Contest</i>	Melihat Kelas / <i>Contest</i>	Memilih menu <i>manage</i> dan memilih submenu <i>contest</i>	Menampilkan halaman <i>contest</i> / kelas yang di kelola
		Mengedit Kelas / <i>Contest</i>	Memilih <i>icon</i> pensil pada list <i>contest</i> /kelas yang tersedia di halaman <i>contest</i>	Mengarahkan ke halaman <i>edit</i> kontes / kelas
			Menekan tombol simpan, tanpa mengubah data	Data tersimpan dan diarahkan ke halaman kontes
			Mengubah data yang ada dan	Data tersimpan dan diarahkan ke halaman kontes
			menekan tombol simpan	
3.	Mengelola <i>Problem</i> / Soal	Melihat soal / <i>problem</i>	Memilih menu <i>manage</i> dan memilih submenu <i>problem</i>	Menampilkan halaman dan list <i>problem</i>

Tabel 3.2 Tabel rencana pengujian pada aktor *editor* (Lanjutan)

		Menambah <i>problem / soal</i>	Tidak mengisi salah satu <i>field</i> yang tersedia	Sistem tidak menyimpan data, dan menampilkan pesan kesalahan
			Mengisi seluruh <i>field</i> yang tersedia	Sistem akan menyimpan <i>problem/soal</i> dan diarahkan ke halaman <i>problem</i>
		Mengedit soal / <i>problem</i>	Mengubah beberapa data di <i>field</i> dan	Sistem akan menolak dan meminta <i>editor</i> untuk mengisi <i>field</i>
			mengosongkan <i>field</i>	
			Mengubah data dan menyimpan	Sistem akan menyimpan perubahan
		Menambah <i>checker</i>	Menekan tombol <i>add</i> tanpa mengunggah <i>file</i>	Menampilkan pesan <i>error</i>
			Menekan tombol <i>add</i> dengan mengunggah <i>file</i>	Data disimpan
		Menambah <i>test case</i>	Mengunggah salah satu <i>file</i> , <i>input</i> atau <i>output</i> , atau tidak mengunggah keduanya	Menampilkan pesan <i>error</i>
			Mengunggah <i>file</i> dengan ekstensi selain <i>.txt</i>	
			Mengunggah <i>file</i> dengan ekstensi <i>.txt</i>	Data <i>test case</i> berhasil di simpan dan di tampilkan di atas <i>form</i>

Tabel 3.2 Tabel rencana pengujian pada aktor *editor* (Lanjutan)

		Mengunduh <i>test case</i>	Menekan <i>icon download</i> pada halaman <i>testcase</i>	Sistem menyiapkan <i>file download</i> dengan ekstensi <i>.txt</i>
4.	Mengelola <i>User</i>	Melihat <i>User</i>	Memilih menu <i>manage</i> dan memilih submenu <i>user</i>	Menampilkan semua <i>user</i> dari kategori kontes yang dikelola
		Menambah <i>user</i>	Tidak mengisi beberapa <i>field</i> yang tersedia	Menampilkan pesan <i>error</i>
			Mengisi semua <i>field</i> dengan <i>username</i> yang sudah tersedia	Menampilkan pesan <i>error</i> bahwa <i>username</i> sudah digunakan
			Mengisi semua <i>field</i> dengan data baru	Data disimpan
		Mencetak data <i>user</i>	Memilih tombol <i>print user</i>	Sistem menyiapkan <i>file</i> dengan ekstensi <i>.xls</i>
5.	Mengelola <i>Submission</i>	Melihat <i>submission</i>	Memilih <i>icon</i> lihat <i>submission</i>	Menampilkan <i>detail submission</i> yang di pilih
		<i>Regrade submission</i>	Memilih <i>icon regrade submission</i>	Mengubah status <i>verdict</i> dan melakukan <i>grader</i> ulang
6.	Mengelola <i>Clarification</i>	Melihat <i>clarification</i>	Memilih menu <i>clarification</i>	Menampilkan halaman <i>clarification</i>
		Menambah <i>clarification</i>	Tidak mengisi judul atau konten <i>clarification</i>	Menampilkan pesan kesalahan

Tabel 3.2 Tabel rencana pengujian pada aktor *editor* (Lanjutan)

			Mengisi judul dan konten <i>clarification</i>	Sistem menyimpan <i>clarification</i> , dan diarahkan ke halaman <i>clarification</i>
		Menjawab <i>clarification</i>	Memilih <i>clarification</i> , dan mengisi jawaban di kolom jawaban	Menyimpan jawaban dan mengarahkan ke halaman <i>clarification</i>
7.	Melihat <i>Scoreboard</i>	Melihat <i>Scoreboard</i>	Memilih menu <i>scoreboard</i>	Menampilkan halaman <i>scoreboard editor</i>
		<i>export scoreboard</i>	Memilih kelas dan memilih tombol <i>export scoreboard</i>	Sistem menyiapkan <i>file</i> berekstensi <i>.xls</i>

Tabel 3.3 Tabel rencana pengujian pada aktor *contestant*/mahasiswa

No	Kelas Uji	Daftar Pengujian	Kasus Uji	Hasil Yang diharapkan
1.	<i>Register</i>	<i>Register User</i> (mahasiswa)	Tidak mengisi salah satu <i>field</i> yang disediakan	Sistem menolak menyimpan
			Mengisi <i>username</i> (NPM) dengan yang sudah ada	Sistem menolak dan memberi informasi bahwa <i>username</i> sudah digunakan
			Mengisi semua <i>field</i> yang tersedia dengan benar	<i>Register</i> berhasil dan di arahkan ke halaman <i>login</i>

Tabel 3.3 Tabel rencana pengujian pada aktor *contestant*/mahasiswa (Lanjutan)

2.	<i>Login</i>	<i>Login User</i>	Tidak mengisi salah satu <i>field</i>	Menolak dan memberi informasi untuk mengisi <i>field</i>
			Mengisi <i>username</i> dan/atau <i>password</i> yang salah	Sistem menolak, dan memberi informasi bahwa <i>username / password</i> salah
			Mengisi <i>field username</i> dan <i>password</i> yang sesuai	Berhasil, diarahkan ke halaman <i>dashboard</i>
3.	Melihat Kelas / <i>Contest / Soal</i>	Melihat soal/kelas	Memilih daftar kelas/ <i>contest</i> yang tersedia dan memilih soal	Menampilkan informasi berupa <i>list</i> soal dan <i>detail</i> soal
4.	Mengelola <i>Submission</i>	Melihat <i>Submission</i>	Memilih <i>submission</i> di menu <i>submission</i>	Memberikan informasi mengenai <i>submission</i> yang dipilih
		Menambah <i>submission</i>	Tidak memilih bahasa pemrograman	Menampilkan pesan kesalahan
			Tidak meng- <i>upload file code</i> program	Sistem tidak menyimpan dan memberi pesan kesalahan
			Memilih bahasa pemrograman dan meng- <i>upload code</i> program	Sistem menyimpan <i>submission</i> dan memengarahkan ke halaman <i>submission</i>
5.	Mengelola <i>Clarification</i>	Melihat <i>clarification</i>	Memilih menu <i>clarification</i>	Menampilkan halaman <i>clarification</i> , dan daftar <i>clarification</i>

Tabel 3.3 Tabel rencana pengujian pada aktor *contestant*/mahasiswa (Lanjutan)

		Menambah <i>clarification</i>	Tidak mengisi judul atau konten <i>clarification</i>	Menampilkan pesan kesalahan
			Mengisi judul dan konten <i>clarification</i>	Sistem menyimpan <i>clarification</i> , dan diarahkan ke halaman <i>clarification</i>
6.	Melihat <i>Scoreboard</i>	Melihat <i>Scoreboard</i>	Memilih menu <i>scoreboard</i>	Menampilkan halaman <i>scoreboard</i> peserta

3.3.7. Release

Release merupakan proses akhir dari pengembangan sistem menggunakan metode *Extreme Programming*. tahap *release* dilakukan jika tahap *planning*, *design*, *coding* dan tahap *testing* sudah selesai dilakukan. Dalam proses *release*, akan ada lebih dari satu kali proses *planning*, *design*, *coding* dan *testing*. Pada *Extreme Programming*, *release* akan dilaksanakan jika semua kebutuhan sudah terpenuhi melalui tahap *testing*. Jika pada tahap *testing*, masih terdapat kebutuhan sistem yang belum terpenuhi, maka proses diulang mulai dari proses *planning* hingga pengujian kembali.

V. SIMPULAN DAN SARAN

5.1. Simpulan

Berdasarkan penelitian yang sudah dilakukan, dapat disimpulkan beberapa hal seperti berikut.

1. Sistem *grader* telah berhasil dikembangkan dan diimplementasikan ke *server* laboratorium jurusan Ilmu Komputer FMIPA Universitas Lampung dan dapat diakses pada laman www.lab.ilkom.unila.ac.id/grader.
2. Halaman *register* mahasiswa berjalan dengan baik.
3. Fitur *share code* berhasil dan menghasilkan sebuah *link* baru yang dapat diakses tanpa melakukan *login* terlebih dahulu.
4. Sistem dikembangkan dengan 3 kategori pengguna, yaitu *admin*, *editor* dan *contestant*.
5. Sistem dapat digunakan untuk semua bahasa pemrograman.
6. *Admin* dan *editor* dapat mengunduh *scoreboard* ke dalam *file excel* berekstensi *.xls*.
7. Hasil pengujian yang diujikan kepada tiga kategori pengguna berhasil dilakukan.

5.2. Saran

Adapun saran yang dapat diberikan untuk penelitian ini adalah sebagai berikut.

1. Melakukan *update* untuk *user interface* sistem agar lebih *user friendly*.
2. Menambahkan beberapa *table* pada *database* agar sesuai dengan sistem yang sudah ada, seperti informasi mahasiswa, dosen, dan informasi lainnya.
3. Mengintegrasikan sistem *grader* agar dapat terhubung dengan sistem yang sudah ada seperti BeOn, Kulon, serta VClass.
4. Membuat *REST API* agar dapat dikembangkan ke versi *mobile*.
5. Maksimalkan fitur *clarification* agar dapat bertanya antar *contestant*.
6. Menambahkan halaman untuk dapat melakukan pembuatan program di sistem sebagai alternatif selain pilihan mengunggah *source code*.

DAFTAR PUSTAKA

- Alatas, Husein. 2013. *Responsive Web Design Dengan PHP & Bootstrap*. Yogyakarta: Lokomedia.
- Daqiqil, Ibnu Id. 2011. *Framework Codeigniter sebuah panduan dan best practice*. Pekanbaru: koder.web.id.
- Dyazin, Cahya. 2015. *Membangun Aplikasi Online Shop dengan CodeIgniter untuk Pemula*. Yogyakarta : Andi.
- Enterprise, Jubilee. 2016. *Pemrograman Bootstrap untuk Pemula*. Yogyakarta: Gramedia.
- Indrajani. 2014. *Pengantar Sistem Basis Data Case Study All in One*. Jakarta: Gramedia.
- MOE Contest Environment. 2015. *The MOE Contest Environment*. <https://www.ucw.cz/moe/> diakses 21 Desember 2017 pkl. 21.27
- Pressman, Roger S. 2014. *Software Engineering a practitioner's approach*. McGraw-Hill: New York.
- Pribadi, Awan Basuki. 2016. *Menguasai CodeIgniter Kasus Membangun Aplikasi Perpustakaan*. Yogyakarta: Lokomedia.
- Raharjo, Budi. 2015. *Belajar Otodidak Framework CodeIgniter Teknik Pemrograman Web dengan PHP dan Framework CodeIgniter 3*. Bandung: Informatika.
- Regrader. 2015. *Regrader : An ICPC-style programming contest system that works*. <https://github.com/fushar/regrader>. Diakses pada 20 Maret 2018 pukul 20.00.
- Rosa A.S dan Salahuddin M, 2013. *Modul Pembelajaran Rekayasa Perangkat Lunak (Terstruktur dan Berorientasi Objek)*. Bandung:Modula.

- Septian, Gungun. 2011. *Trik Pintar Menguasai Codeigniter*. Jakarta: PT Elex Media Komputindo.
- Sholih. 2006. *Pemodelan Sistem Informasi Berorientasi Objek dengan UML*. Yogyakarta: Graha Ilmu.
- Subagia, Anton. 2016. *Membuat Web dengan PHP 7 dan Database PDO MySQLi*. Jakarta: Gramedia.
- Whitten, Jeffrey L. dan Lonnie Bentley D. 2007. *Systems Analysis and Design for the Global Enterprise Seventh Edition*. New York: McGraw-Hill.
- Widodo, Massus Subekti. 2006. *Requirements Management Pada Extreme Programming*. Yogyakarta: Seminar Nasional Aplikasi Teknologi Informasi.