PP/018/Sms ke-VI/Revisi ke-3

LABORATORIUM
TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS AHMAD DAHLAN

PETUNJUK PRAKTIKUM

ANALISIS PERANCANGAN PERANGKAT LUNAK

Penyusun:
Murein Miksa Mardhia, S.T., M.T.
Arfiani Nur Khusna, S.T., M.Kom.

2020

KATA PENGANTAR

Terima kasih kami sampaikan kepada hadirat Allah SWT yang telah melimpahkan rahmat dan bimbingan-Nya sehingga kami dapat melengkapi MODUL PANDUAN PRAKTIKUM untuk mata kuliah ANALISIS PERANCANGAN PERANGKAT LUNAK (APPL) Tahun Akademik 2019/2020 di Program Studi Teknik Informatika, Universitas Ahmad Dahlan.

Materi yang disajikan dalam modul panduan praktikum ini telah disesuaikan dengan perencanaan silabus APPL, referensi utama (System Analysis and Design, Alan Dennis, 2012). Setiap pertemuan berisi penjelasan tentang teori yang berkaitan dengan materi yang diberikan dan penjelasan tentang tahapan kerja yang harus dilakukan saat berpraktik di laboratorium.

Kami menyadari bahwa masih banyak ketidaksempurnaan dalam penulisan ini. Kami selalu menerima kritik dan saran untuk meningkatkan kualitas pedoman dan implementasi praktikum.

Kami berterima kasih kepada seluruh tim dosen dan asisten laboratorium yang terlibat dalam pembuatan panduan praktikum ini. Semoga hasil yang diperoleh dari implementasi praktikum APPL melalui panduan praktikum ini dapat memberikan manfaat dan kontribusi dalam kemajuan ilmu pengetahuan.

Yogyakarta, Januari 2020

Tim Penyusun

DAFTAR PENYUSUN

Murein Miksa Mardhia, S.T., M.T.

Arfiani Nur Khusna, S.T., M. Kom.

HALAMAN REVISI

Yang bertanda tangan di bawah ini:

Nama : Murein Miksa Mardhia, S.T., M.T.

NIK/NIY : 60160960

Jabatan : Dosen Pengampu Matakuliah APPL

Dengan ini menyatakan pelaksanaan Revisi Petunjuk Praktikum APPL untuk Program Studi Teknik Informatika telah dilaksanakan dengan penjelasan sebagai berikut:

No	Keterangan Detail Revisi (Per Pertemuan)	Tanggal Revisi	Nomor Modul
1	a. Menambahkan materi class diagram	25 Juli 2018	PP/018/I/1
2	a. Memperbaiki template update dan menambah materi manajemen berkas	22 Januari 2020	PP/020/

Yogyakarta, Januari 2020
Penyusun

NIK/NIY. .....

HALAMAN PERNYATAAN

Yang bertanda tangan di bawah ini:

Nama : Murein Miksa Mardhia, S.T., M.T.

NIK/NIY : 60160960

Jabatan : Dosen Pengampu

Menerangkan dengan sesungguhnya bahwa Petunjuk Praktikum ini telah direview dan akan digunakan untuk pelaksanaan praktikum di Semester Genap Tahun Akademik 2019-2020 di Laboratorium Basis Data, Program Studi Teknik Informatika, Fakultas Teknologi Industri, Universitas Ahmad Dahlan.

Yogyakarta, 22 Januari 2020

Mengetahui,
Ketua Kelompok Keilmuan Relata

Drs. Tedy Setiadi, M.T.
NIK/NIY. 60030475

Kepala Laboratorium Basis Data

Lisna Zahrotun, S.T., M.Cs.
NIK/NIY. 60150773

VISI DAN MISI PRODI TEKNIK INFORMATIKA

VISI

Menjadi Program Studi Informatika yang diakui secara internasional dan unggul dalam bidang Informatika serta berbasis nilai-nilai Islam.

MISI

1. Menjalankan pendidikan sesuai dengan kompetensi bidang Informatika yang diakui nasional dan internasional
2. Meningkatkan penelitian dosen dan mahasiswa dalam bidang Informatika yang kreatif, inovatif dan tepat guna.
3. Meningkatkan kuantitas dan kualitas publikasi ilmiah tingkat nasional dan internasional
4. Melaksanakan dan meningkatkan kegiatan pengabdian masyarakat oleh dosen dan mahasiswa dalam bidang Informatika.
5. Menyelenggarakan aktivitas yang mendukung pengembangan program studi dengan melibatkan dosen dan mahasiswa.
6. Menyelenggarakan kerja sama dengan lembaga tingkat nasional dan internasional.
7. Menciptakan kehidupan Islami di lingkungan program studi.

TATA TERTIB LABORATORIUM TEKNIK INFORMATIKA

DOSEN/KOORDINATOR PRAKTIKUM

1. Dosen harus hadir saat praktikum minimal 15 menit di awal kegiatan praktikum dan menandatangani presensi kehadiran praktikum.
2. Dosen membuat modul praktikum, soal seleksi asisten, pre-test, post-test, dan responsi dengan berkoordinasi dengan asisten dan pengampu mata praktikum.
3. Dosen berkoordinasi dengan koordinator asisten praktikum untuk evaluasi praktikum setiap minggu.
4. Dosen menandatangani surat kontrak asisten praktikum dan koordinator asisten praktikum.
5. Dosen yang tidak hadir pada slot praktikum tertentu tanpa pemberitahuan selama 2 minggu berturut-turut mendapat teguran dari Kepala Laboratorium, apabila masih berlanjut 2 minggu berikutnya maka Kepala Laboratorium berhak mengganti koordinator praktikum pada slot tersebut.

PRAKTIKAN

1. Praktikan harus hadir 15 menit sebelum kegiatan praktikum dimulai, dan dispensasi terlambat 15 menit dengan alasan yang jelas (kecuali asisten menentukan lain dan patokan jam adalah jam yang ada di Laboratorium, terlambat lebih dari 15 menit tidak boleh masuk praktikum & dianggap Inhal).
2. Praktikan yang tidak mengikuti praktikum dengan alasan apapun, wajib mengikuti INHAL, maksimal 4 kali praktikum dan jika lebih dari 4 kali maka praktikum dianggap GAGAL.
3. Praktikan harus berpakaian rapi sesuai dengan ketentuan Universitas, sebagai berikut:
   a. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki-laki / Perempuan) dan Topi harus Dilepas.
   b. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
   c. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki-laki / Perempuan).
   d. Laki-laki tidak boleh memakai gelang, anting-anting ataupun aksesoris Perempuan.
4. Praktikan tidak boleh makan dan minum selama kegiatan praktikum berlangsung, harus menjaga kebersihan, keamanan dan ketertiban selama mengikuti kegiatan praktikum atau selama berada di dalam laboratorium (tidak boleh membuang sampah sembarangan baik kertas, potongan kertas, bungkus permen baik di lantai karpet maupun di dalam ruang CPU).
5. Praktikan dilarang meninggalkan kegiatan praktikum tanpa seizin Asisten atau Laboran.
6. Praktikan harus meletakkan sepatu dan tas pada rak/loker yang telah disediakan.
7. Selama praktikum dilarang NGENET/NGE-GAME, kecuali mata praktikum yang membutuhkan atau menggunakan fasilitas Internet.
8. Praktikan dilarang melepas kabel jaringan atau kabel power praktikum tanpa sepengetahuan laboran
9. Praktikan harus memiliki FILE Petunjuk praktikum dan digunakan pada saat praktikum dan harus siap sebelum praktikum berlangsung.
10. Praktikan dilarang melakukan kecurangan seperti mencontek atau menyalin pekerjaan praktikan yang lain saat praktikum berlangsung atau post-test yang menjadi tugas praktikum.

11. Praktikan dilarang mengubah setting software/hardware komputer baik menambah atau mengurangi tanpa permintaan asisten atau laboran dan melakukan sesuatu yang dapat merugikan laboratorium atau praktikum lain.
12. Asisten, Koordinator Praktikum, Kepala laboratorium dan Laboran mempunyai hak untuk menegur, memperingatkan bahkan meminta praktikan keluar ruang praktikum apabila dirasa anda mengganggu praktikan lain atau tidak melaksanakan kegiatan praktikum sebagaimana mestinya dan atau tidak mematuhi aturan lab yang berlaku.
13. Pelanggaran terhadap salah satu atau lebih dari aturan diatas maka Nilai praktikum pada pertemuan tersebut dianggap 0 (NOL) dengan status INHAL.

ASISTEN PRAKTIKUM

1. Asisten harus hadir 15 Menit sebelum praktikum dimulai (konfirmasi ke koordinator bila mengalami keterlambatan atau berhalangan hadir).
2. Asisten yang tidak bisa hadir WAJIB mencari pengganti, dan melaporkan kepada Koordinator Asisten.
3. Asisten harus berpakaian rapi sesuai dengan ketentuan Universitas, sebagai berikut:
   a. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki-laki / Perempuan) dan Topi harus Dilepas.
   b. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
   c. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki-laki / Perempuan).
   d. Laki-laki tidak boleh memakai gelang, anting-anting ataupun aksesoris Perempuan.
4. Asisten harus menjaga kebersihan, keamanan dan ketertiban selama mengikuti kegiatan praktikum atau selama berada di laboratorium, menegur atau mengingatkan jika ada praktikan yang tidak dapat menjaga kebersihan, ketertiban atau kesopanan.
5. Asisten harus dapat merapikan dan mengamankan presensi praktikum, Kartu Nilai serta tertib dalam memasukan/Input nilai secara Online/Offline.
6. Asisten harus dapat bertindak secara profesional sebagai seorang asisten praktikum dan dapat menjadi teladan bagi praktikan.
7. Asisten harus dapat memberikan penjelasan/pemahaman yang dibutuhkan oleh praktikan berkenaan dengan materi praktikum yang diasisteni sehingga praktikan dapat melaksanakan dan mengerjakan tugas praktikum dengan baik dan jelas.
8. Asisten tidak diperkenankan mengobrol sendiri apalagi sampai membuat gaduh.
9. Asisten dimohon mengkoordinasikan untuk meminta praktikan agar mematikan komputer untuk jadwal terakhir dan sudah dilakukan penilaian terhadap hasil kerja praktikan.
10. Asisten wajib untuk mematikan LCD Projector dan komputer asisten/praktikan apabila tidak digunakan.
11. Asisten tidak diperkenankan menggunakan akses internet selain untuk kegiatan praktikum, seperti Youtube/Game/Medsos/Streaming Film di komputer praktikan.

LAIN-LAIN

1. Pada Saat Responsi Harus menggunakan Baju Kemeja untuk Laki-laki dan Perempuan untuk Praktikan dan Asisten.
2. Ketidakhadiran praktikum dengan alasan apapun dianggap INHAL.
3. Izin praktikum mengikuti aturan izin SIMERU/KULIAH.

4. Yang tidak berkepentingan dengan praktikum dilarang mengganggu praktikan atau membuat keributan/kegaduhan.
5. Penggunaan lab diluar jam praktikum maksimal sampai pukul 21.00 dengan menunjukkan surat ijin dari Kepala Laboratorium Prodi Teknik Informatika.

Yogyakarta, 22 Januari 2020

Kepala Laboratorium Praktikum Teknik Informatika

Lisna Zahrotun, S.T., M.Cs.

NIK/NIY. 60150773

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

PRAKTIKUM 1: PENGENALAN SOFTWARE DAN MANAJEMEN BERKAS

Pertemuan ke : 1

Total Alokasi Waktu : 150 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 30 menit
- Praktikum : 90 menit
- Post-Test : 30 menit
- dst

Total Skor Penilaian : 100% (Bobot skor disesuaikan dengan RPS)
- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

1.1. TUJUAN DAN INDIKATOR CAPAIAN

Setelah mengikuti praktikum ini mahasiswa diharapkan:

1. Mampu memeriksa kelengkapan software pendukung praktikum di perangkat komputer masing-masing telah terpasang dan dapat berjalan dengan baik
2. Mampu membuat shared folder yang hanya dapat diakses oleh praktikan dan asisten praktikum di slot masing-masing.

Indikator ketercapaian diukur dengan:

1. Terpasangnya software pendukung tanpa kendala
2. Terbentuknya shared folder masing-masing praktikan untuk digunakan sebagai penilaian kepada asisten praktikum sepanjang pelaksanaan praktikum APPL.

1.2. TEORI PENDUKUNG

Dalam rangka pelaksanaan praktikum Analisis Perancangan Perangkat Lunak (APPL) semester ini, beberapa hal perlu dipersiapkan oleh asisten praktikum dan masing-masing praktikan. Persiapan ini dilakukan dengan tujuan menciptakan efektivitas pemantauan penugasan praktikan selama praktikum.

Sebelum pelaksanaan praktikum dimulai, pengelola praktikum telah mempersiapkan *software* pendukung yang akan digunakan seluruh praktikan sepanjang periode praktikum. Untuk pembuatan dokumen proses bisnis dan kebutuhan sistem, praktikan dapat menggunakan *software Word Editor* yang mendukung deskripsi dan justifikasi studi kasus. Untuk materi UML Diagrams, praktikan telah disediakan *software* pendukung Astah UML *Free Student Academic License* yang mendukung perancangan/desain masing-masing diagram berdasarkan fungsinya. Untuk materi Estimasi Biaya *Software* menggunakan Use Case Points, praktikan akan disediakan *software* Excel untuk melakukan simulasi perhitungan biaya dengan berbagai perhitungan matematis.

Selama 10 kali pertemuan praktikum matakuliah APPL, praktikan akan diminta mengerjakan tugas *pre-test, post-test* dan tugas mingguan praktikum sesuai dengan materi yang dijadwalkan. Pengumpulan tugas ini akan secara serentak dilakukan melalui berkas yang dapat dibagikan melalui aplikasi Google Drive. Asisten praktikum yang bertugas di masing-masing slot akan membuat tautan yang dapat diakses oleh praktikum secara *real-time* dimana praktikan dapat mengumpulkan pekerjaan mingguan mereka ke tautan tersebut yang otomatis terhubung dan dapat dicek langsung oleh akun Google asisten praktikum. Mekanisme pengumpulan tugas ini diharapkan dapat mempermudah praktikan dan asisten praktikum dalam mengelola berkas praktikum dan meningkatkan efektivitas proses penilaian berkas praktikan.

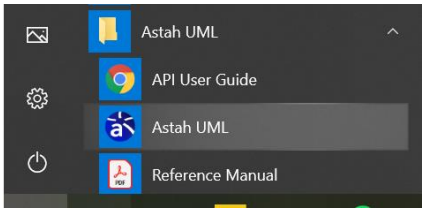
1.3. ALAT DAN BAHAN

1. Komputer
2. Word Editor (contoh: Ms. Word), Flowchart/Diagram Maker (contoh: Ms. Visio atau draw.io), Ms. Excel, dan UML Diagram Maker (Astah UML)

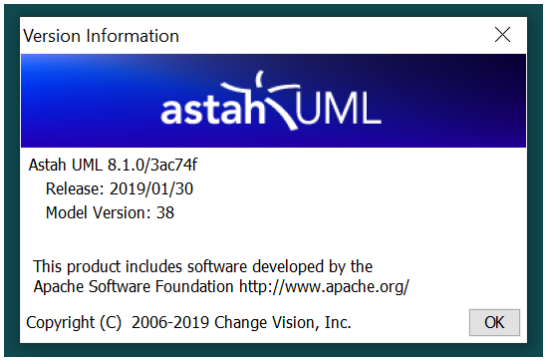
1.4 LANGKAH PRAKTIKUM

Pengecekan Software Astah UML

1. Klik Start Menu dan buka aplikasi Astah UML seperti pada Gambar 1.1 berikut. Version yang digunakan adalah Astah UML 8.1.0 dan seperti pada Gambar 1.2. Lisensi yang digunakan yaitu Free Student Academic License. Lisensi khusus pelajar ini dapat diperbarui melalui website resmi Astah http://astah.net/student-license-request dengan cara memasukkan identitas email berdomain universitas.

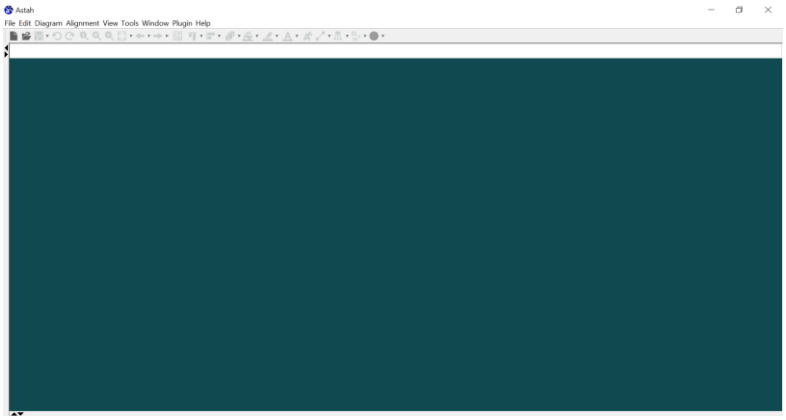


Gambar 1.1 Aplikasi Astah UML di Start Menu



Gambar 1.2 Software Version Astah UML

2. Apabila berhasil, maka jendela yang akan tampil adalah seperti pada Gambar 1.2.



Gambar 1.3 Jendela Aktif Astah UML

3. Apabila Gambar 1.2 tidak langsung muncul, segera hubungi asisten praktikum pada slot masing-masing. Kendala yang sering terjadi adalah *license key* yang belum aktif. Pastikan Anda langsung menghubungi asisten praktikum supaya praktikum selanjutnya tidak lagi menemui kendala di instalasi aplikasi Astah UML.

Manajemen berkas dengan Google Drive

1. Pastikan Anda telah mempunyai akun email dengan domain Gmail
2. Asisten praktikum akan memberikan instruksi untuk membuat shared folder di masing-masing slot praktikum.
3. Melalui tautan yang dibagikan, Anda diminta untuk membuat folder pribadi untuk meletakkan tugas-tugas praktikum Anda selama mengikuti praktikum APPL.
4. Instruksi detail mengenai tautan, penamaan file dan folder akan diinfokan oleh asisten praktikum di slot masing-masing. Pembuatan folder ini hanya dilakukan di pertemuan I, dan akan digunakan sepanjang pelaksanaan praktikum APPL.

1.5 TUGAS

Pada perangkat komputer Anda masing-masing:

1. Lakukan pengecekan sesuai yang diterangkan pada langkah praktikum
2. Periksa apabila terjadi kendala di semua software pendukung, baik Astah, Ms. Visio, Word Editor.
3. Pastikan keberadaan folder Anda di Google Drive telah terhubung dengan akun asisten praktikum sehingga saat pengumpulan tugas praktikum, berkas Anda ditemukan dan nilai Anda dapat terekam.

CONTOH
LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 1: PENGENALAN SOFTWARE DAN MANAJEMEN BERKAS

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 2: ANALISIS PROSES BISNIS

Pertemuan ke : 2

Total Alokasi Waktu : 150 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 30 menit
- Praktikum : 90 menit
- Post-Test : 30 menit
- dst

Total Skor Penilaian : 100% (Bobot skor disesuaikan dengan RPS)
- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

2.1 TUJUAN DAN INDIKATOR CAPAIAN

Setelah mengikuti praktikum ini mahasiswa diharapkan:

1. Mampu mengidentifikasi proses bisnis pada proyek yang sedang ditangani
2. Mampu menganalisis bagian per bagian proses bisnis terutama pada bagian yang bermasalah
3. Mampu mengidentifikasi alternatif solusi berupa proses bisnis yang diharapkan untuk menyelesaikan permasalahan di kondisi saat ini.

Indikator ketercapaian diukur dengan:

1. Tergambar proses bisnis saat ini
2. Hasil analisis dari permasalahan proses bisnis saat ini
3. Tergambar proses bisnis yang diharapkan untuk menyelesaikan permasalahan.

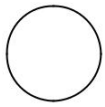
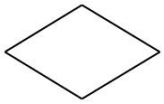
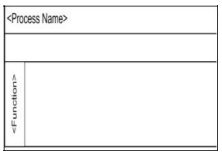
2.2 TEORI PENDUKUNG

BPMN merupakan suatu teknik untuk memodelkan dan manajemen proses bisnis. Sebelumnya, organisasi telah mempergunakan berbagai teknik dan tools untuk memodelkan dan mengelola proses bisnis. Tetapi teknik-teknik tersebut tidak memiliki standarisasi yang lengkap dan siklus hidup yang lengkap untuk mengontrol dan memandu perancangan dan eksekusi proses bisnis. Ada beberapa teknik yang dipergunakan dalam memodelkan proses, yang dibedakan menjadi beberapa tingkatan pemodelan, yaitu:

1. *Process maps*, yaitu pemodelan proses yang ditampilkan melalui flowchart sederhana atau grafik sederhana dari aktifitas-aktifitas.
2. *Process descriptions*, yaitu pemodelan proses dengan penggunaan *flowchart* yang diperluas. Pemodelan proses pada tingkatan ini sudah dilengkapi dengan penambahan informasi tetapi masih belum memadai untuk mendeskripsikan kinerja aktifitas yang sesungguhnya.

3. *Process models*, yaitu pemodelan proses bisnis melalui *flowchart* dilengkapi dengan informasi yang memadai sehingga proses yang dimodelkan dianalisis, disimulasi dan/atau dieksekusi.

Tabel 2.1 Elemen Utama dalam Diagram Proses Bisnis

Elemen	Deskripsi	Notasi
Event	Sesuatu yang terjadi selama proses bisnis berlangsung. *Event* mempengaruhi aliran proses bisnis dan biasanya memiliki suatu akibat atau dampak. *Event* dibedakan menjadi tiga, yaitu *Start, Intermediate* dan *End*	
Activity	Pekerjaan yang dilaksanakan oleh perusahaan. Jenis *Activity* yang merupakan bagian dari model proses, yaitu *process, subprocess* dan *task.*	
Gateway	Untuk mengontrol percabangan dan pertemuan dari *sequence flow*	
Sequence flow	Menunjukkan perintah yang akan dilaksanakan oleh suatu aktifitas dalam suatu proses	
Pool	suatu "*swimlane*" dan sebuah kontainer grafis untuk membagi serangkaian aktifitas dari *pool* lainnya	

2.3 ALAT DAN BAHAN

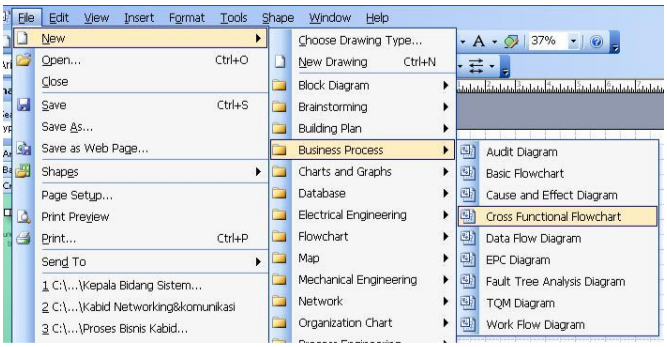
Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer
2. Software Pendukung: Word Editor, Astah
3. Diagram Maker: Ms Visio

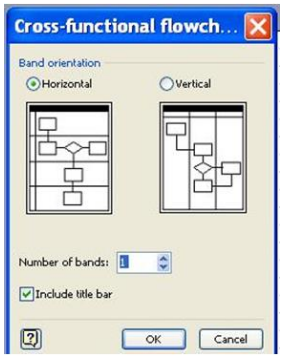
2.4 LANGKAH PRAKTIKUM

1. Berdasarkan studi kasus pengembangan aplikasi yang dipilih di kelas, buatkan proses bisnis kondisi saat ini.
2. Buatkan analisis permasalahan pada proses bisnis yang berjalan saat ini, berdasarkan hasil wawancara dengan customer.
3. Buat Proses bisnis yang diharapkan untuk memperbaiki proses bisnis saat ini, dengan memberikan solusi aplikasi yang Anda ambil.
4. Menggambarkan proses bisnis menggunakan alat bantu visio
   a. Buka visio

b. Buka file/new/Business process/cross Functional Flowchart



c. Pilih yang horizontal, number of bands diisi dengan berapa jumlah line yang akan dibuat



d. Setelah itu akan muncul nama proses dituliskan proses bisnis yang akan digambarkan. Function menunjukkan pelaku bisnis yang bertanggungjawab menjalankan proses bisnis.

e. Buat gambar proses bisnisnya dengan notasi yang sesuai.

f. Lakukan verifikasi dengan asisten praktikum yang bertugas.

2.5 TUGAS

Pada studi kasus yang telah ditentukan, uraikan proses bisnis saat ini yang dijalankan dengan menggambarkan alur kerjanya menggunakan Ms. Visio atau software diagram maker lainnya. Uraikan pelaku bisnis yang terlibat dan berikan keterangan detail apabila terdapat batasan-batasan yang diterapkan (misal: waktu, jumlah personel, biaya, dsb).

CONTOH
LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 2: PROSES BISNIS

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 3: FUNCTIONAL & NON-FUNCTIONAL REQUIREMENTS

Pertemuan ke : 3

Total Alokasi Waktu : 150 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 30 menit
- Praktikum : 90 menit
- Post-Test : 30 menit
- dst

Total Skor Penilaian : 100% (Bobot skor disesuaikan dengan RPS)
- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

3.1 TUJUAN DAN INDIKATOR CAPAIAN

Setelah mengikuti praktikum ini mahasiswa diharapkan:
1. Mampu menganalisis kebutuhan fungsional dan kebutuhan non fungsional
2. Mampu menganalisis bagian per bagian proses bisnis terutama pada bagian yang bermasalah

Indikator ketercapaian diukur dengan:
1. Hasil analisis kebutuhan fungsional dan non fungsional
2. Hasil pemetaan kebutuhan fungsional dan non fungsional yang dibutuhkan sistem

3.2 TEORI PENDUKUNG

Rekayasa kebutuhan *(Requirements engineering)* adalah fase terdepan dari proses rekayasa perangkat lunak, di mana kebutuhan perangkat lunak (*software requirements*) dari pengguna dan pelanggan dikumpulkan, dipahami dan ditetapkan. Para pakar *software engineering* sepakat bahwa rekayasa kebutuhan adalah suatu pekerjaan yang sangat penting. Fakta membuktikan bahwa kebanyakan kegagalan pengembangan perangkat lunak disebabkan karena adanya ketidakkonsistenan *(inconsistent)*, ketidaklengkapan *(incomplete)*, maupun ketidakbenaran *(incorrect)* dari spesifikasi kebutuhan (*requirements specification*).

Hasil dari fase *requirements engineering* terdokumentasi dalam SRS *(software requirements specification)* atau SKPL (spesifikasi kebutuhan perangkat lunak). SKPL berisi kesepakatan bersama tentang permasalahan yang ingin dipecahkan antara pengembang dan *customer*, dan merupakan titik awal menuju proses berikutnya yaitu *software design*.

Tipe Requirements

Kebutuhan *(requirements)* perangkat lunak seringkali diklasifikasikan ke dalam dua kategori:

1. *Functional Requirements* (Kebutuhan Fungsional)
   Merupakan pernyataan tentang sekumpulan layanan/fitur yang harus tersedia dalam perangkat lunak.
2. *Non-Functional Requirements* (Kebutuhan Non Fungsional)
   Terkait dengan kendala *(constraint)* dan kualitas dari perangkat lunak. Kualitas perangkat lunak adalah sifat atau karakteristik dari sistem yang stakeholders peduli dan karenanya akan mempengaruhi tingkat kepuasan terhadap sistem.

Tabel 3.1 Parameter Kebutuhan Non-Fungsional

Kode	Deskripsi
SKPL-NF1	*Availability* – ketersediaan aplikasi untuk dapat diakses oleh pengguna
SKPL-NF2	*Reliability* – kehandalan aplikasi, termasuk aspek teknis seperti koneksi, kebutuhan perangkat keras.
SKPL-NF3	*Ergonomy* – Desain aplikasi harus disesuaikan dengan kenyamanan pengguna.
SKPL-NF4	*Portability* – Keberpindahan aplikasi, sehingga dapat diakses oleh berbagai device.
SKPL-NF5	*Memory* – Kebutuhan aplikasi akan media penyimpanan.
SKPL-NF6	*Response time* – Waktu aplikasi untuk merespon request dari user.
SKPL-NF7	*Safety* – Keamanan data dari aplikasi, serta penggunaan aplikasi.
SKPL-NF8	*Security* – Keamanan aplikasi untuk melindungi data di dalamnya.
SKPL-NF9	*Communication* – Media bahasa yang digunakan oleh aplikasi.

3.3 ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer
2. Software Pendukung: Word Editor, Table Editor (Ms. Word, Ms. Excel)

3.4 LANGKAH PRAKTIKUM

1. Klasifikasikan daftar kebutuhan pelanggan ke dalam kategori kebutuhan fungsional dan kebutuhan non fungsional.
2. Berikan deskripsi mendetail dari masing-masing kebutuhan tersebut.

3.5 TUGAS

Buat daftar kebutuhan fungsional dan kebutuhan non fungsional menggunakan tabel berikut!

Tabel Kebutuhan Fungsional

No	Kode	Deskripsi
1	SKPL-F1	
2		
3		
4		
dst		

Tabel Kebutuhan Non Fungsional

No	Kode	Parameter	Deskripsi
1	*SKPL-NF1*		
2			
3			
4			
5			
6			
7			
dst			

CONTOH
LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 3: FUNCTIONAL & NON-FUNCTIONAL REQUIREMENT

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 4: UML & USE CASE DIAGRAM

Pertemuan ke : 4

Total Alokasi Waktu : 150 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 30 menit
- Praktikum : 90 menit
- Post-Test : 30 menit
- dst

Total Skor Penilaian : 100% (Bobot skor disesuaikan dengan RPS)
- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

4.1. TUJUAN DAN INDIKATOR CAPAIAN

Setelah mengikuti praktikum ini mahasiswa diharapkan:

1. Mampu memahami Unified Modelling Language (UML) sebagai suatu aktifitas dan modeling untuk perangkat lunak
2. Mampu memahami analisis dan notasi Use Case
3. Mampu membuat analisa Use Case yang diharapkan untuk menyelesaikan permasalahan di kondisi saat ini.

Indikator ketercapaian diukur dengan:

1. Hasil analisis tertuang dalam gambar use case dari studi kasus yang ditentukan saat ini
2. Tergambar diagram use case yang diharapkan untuk menyelesaikan permasalahan.

4.2. TEORI PENDUKUNG

A. UML

Unified Modeling Language™ (UML®) adalah bahasa pemodelan visual standar yang digunakan untuk:

a. pemodelan bisnis dan sejenis proses
b. analisis, desain, dan implementasi sistem berbasis perangkat lunak.

UML merupakan bahasa umum untuk analis bisnis, arsitek dan pengembang perangkat lunak yang digunakan untuk menggambarkan, menentukan, desain, dan dokumen yang sudah ada atau proses bisnis baru, struktur dan perilaku artefak dari sistem perangkat lunak. UML adalah bahasa pemodelan standar, bukan proses pengembangan perangkat lunak. UML menjelaskan proses yang:

a. memberikan panduan untuk urutan kegiatan tim,
b. menentukan apa yang harus dikembangkan artefak,
c. mengarahkan tugas pengembang individu dan tim secara keseluruhan, dan

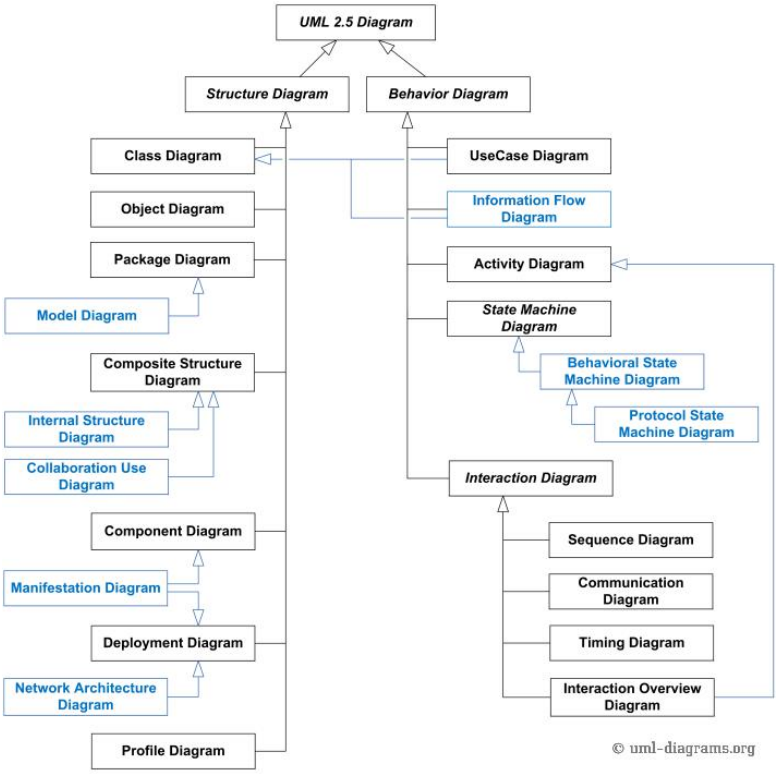
d. menawarkan kriteria untuk memantau dan mengukur produk dan kegiatan proyek.

Spesifikasi UML mendefinisikan dua jenis utama dari diagram UML: **diagram struktur** dan **diagram perilaku**.

Diagram struktur menunjukkan struktur statis dari sistem dan bagian-bagiannya pada abstraksi yang berbeda dan tingkat pelaksanaan dan bagaimana mereka berhubungan satu sama lain. Unsur-unsur dalam diagram struktur mewakili konsep yang bermakna dari suatu sistem, dan mungkin termasuk abstrak, dunia nyata dan konsep implementasi.

Diagram perilaku menunjukkan perilaku dinamis dari objek dalam suatu sistem, yang dapat digambarkan sebagai serangkaian perubahan ke sistem dari waktu ke waktu.

Diagram UML 2.5 dapat dikategorikan ke dalam hirarki seperti yang ditunjukkan Gambar 4.1. Catatan: item ditampilkan dalam warna biru bukan bagian resmi diagram taksonomi UML 2,5.



Gambar 4.1. Diagram UML 2.5

Keterangan beberapa diagram UML 2.5 sebagai berikut:

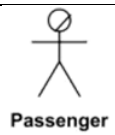
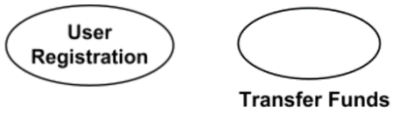
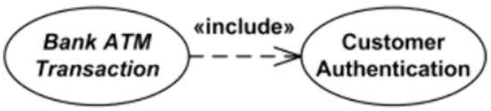
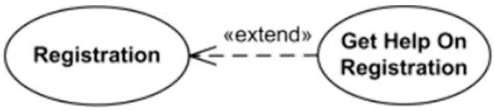
a. Use Case Diagram/Diagram Use Case: Diagram UML yang menunjukkan sekumpulan kasus penggunaan dan aktor dan hubungan mereka (Booch, Rumbaugh, dan Jacobson 2005).
b. Activity Diagram/Diagram Aktifitas: Diagram UML yang menggambarkan aliran kendali dan pengurutan di antara kegiatan.
c. Class Diagram/Diagram Kelas: Diagram UML yang menggambarkan tampilan statis suatu sistem dalam hal kelas dan hubungan antar kelas
d. State Transition Diagram: Representasi grafis dari *finite state machine* di mana *node* mewakili kondisi/keadaan dan garis mewakili transisi antar kondisi/keadaan.
e. Package Diagram/Diagram Paket: Menunjukkan paket dan hubungan antara paket.
f. Interaction Diagram/Diagram Interaksi: Diagram UML yang menggambarkan tampilan dinamis suatu sistem dalam hal objek dan urutan pesan yang saling dikirimkan. Communication diagram dan sequence diagram adalah dua jenis utama diagram interaksi.
g. Sequence Diagram: Diagram interaksi UML yang menggambarkan tampilan dinamis sebuah sistem di mana objek yang berpartisipasi dalam interaksi digambarkan secara horizontal, waktu direpresentasikan oleh dimensi vertikal, dan urutan pesan interaksi digambarkan dari atas ke bawah.

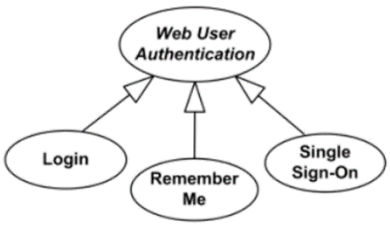
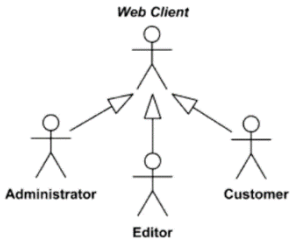
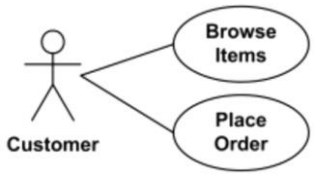
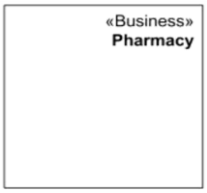
B. Use Case Diagram

Diagram use case biasanya disebut sebagai diagram perilaku yang digunakan untuk menggambarkan serangkaian tindakan (Use Case) bahwa beberapa sistem atau sistem (subjek) dapat melakukan bekerjasama dengan satu atau lebih pengguna eksternal dari sistem (aktor). Setiap penggunaan use case harus menyediakan beberapa hasil diamati dan berharga untuk para aktor atau pemangku kepentingan lain dari sistem.

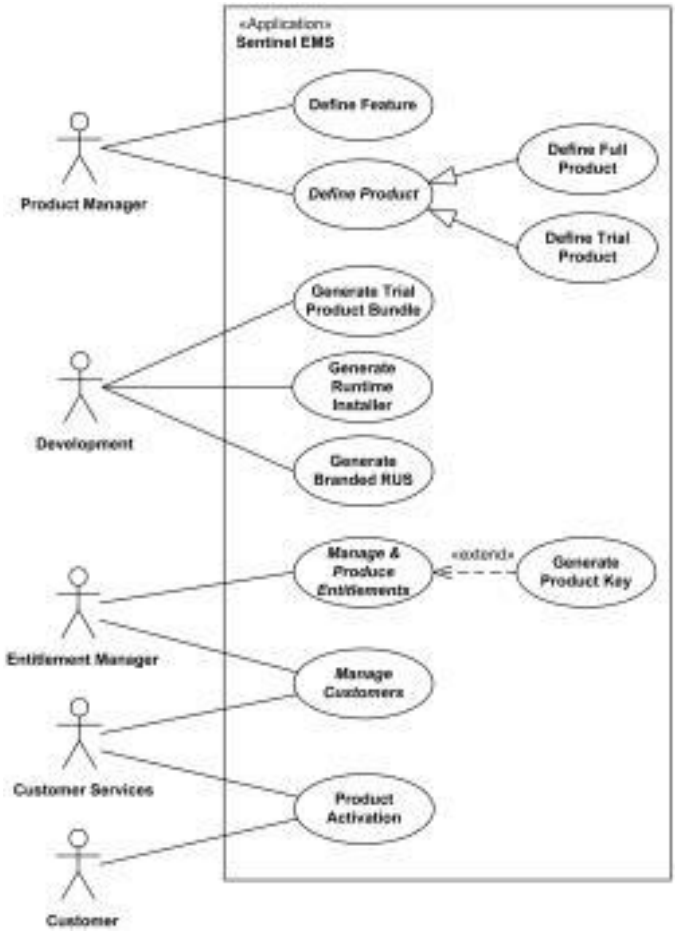
Tabel 4.1 Notasi yang Digunakan dalam Use Case Diagram

Notasi	Nama	Deskripsi
	Aktor	Seorang aktor adalah **perilaku** yang menentukan peran yang dimainkan oleh entitas eksternal yang berinteraksi dengan subjek (misalnya dengan bertukar sinyal dan data), pengguna manusia dari sistem yang dirancang, beberapa layanan sistem atau perangkat keras menggunakan lain dari subjek. DIgambarkan dengan notasi “sticky man”.

	Aktor eksternal	Sebuah pelaku usaha merupakan peran yang dimainkan oleh beberapa **orang atau sistem eksternal** untuk bisnis dimodelkan dan berinteraksi dengan bisnis. Digambarkan mirip seperti “sticky man” dengan garis.
	Use case	Menggambarkan **fungsionalitas** yang disediakan oleh sistem-sistem yang berasal dari daftar kebutuhan sistem. Digambarkan dengan objek elips. Penamaan use case bisa di dalam atau di bawah elips.
	Business use case	**Business use case** penggunaan bisnis use case untuk mendukung Pemodelan untuk mewakili fungsi bisnis, proses, atau kegiatan yang dilakukan dalam bisnis model. penggunaan bisnis kasus harus menghasilkan hasil nilai diamati untuk business actor.
	include	**Relasi Include** merupakan hubungan berarah antara dua use case yang mana digunakan untuk menunjukkan bahwa tingkah laku dari use case include adalah ditambahkan dalam tingkah laku use case dasar. Relasi antar use case ini merupakan relasi yang **diperlukan**, tidak opsional.
	extend	**Relasi Extend** adalah hubungan berarah yang menentukan bagaimana dan kapan perilaku didefinisikan dalam biasanya **tambahan (opsional)** penggunaan *extend use case* dapat dimasukkan ke dalam perilaku yang ditetapkan dalam kasus penggunaan yang berkepanjangan. Relasi antar use case ini merupakan relasi yang **optional, sebagai pelengkap.**

 	Generalisasi	**Relasi Generalisasi** merupakan Generalisasi antara use case anak use case **mewarisi** sifat dan perilaku induk use case dan mungkin menimpa induk use case.
	Asosiasi	**Relasi Asosiasi** menghubungkan antara aktor dan use case. Digambarkan dengan garis tanpa anak panah.
	Subject	Subjek adalah bisnis, perangkat lunak sistem, subsistem, komponen, perangkat, dll. Hal ini sangat penting untuk menentukan jenis sistem itu, dan apa yang ruang lingkup atau batas.

Contoh diagram use case menunjukkan beberapa pandangan sederhana dari perangkat lunak lisensi use case yang didukung oleh Sentinel EMS Aplikasi.



Gambar 4.2 Contoh Use Case Diagram

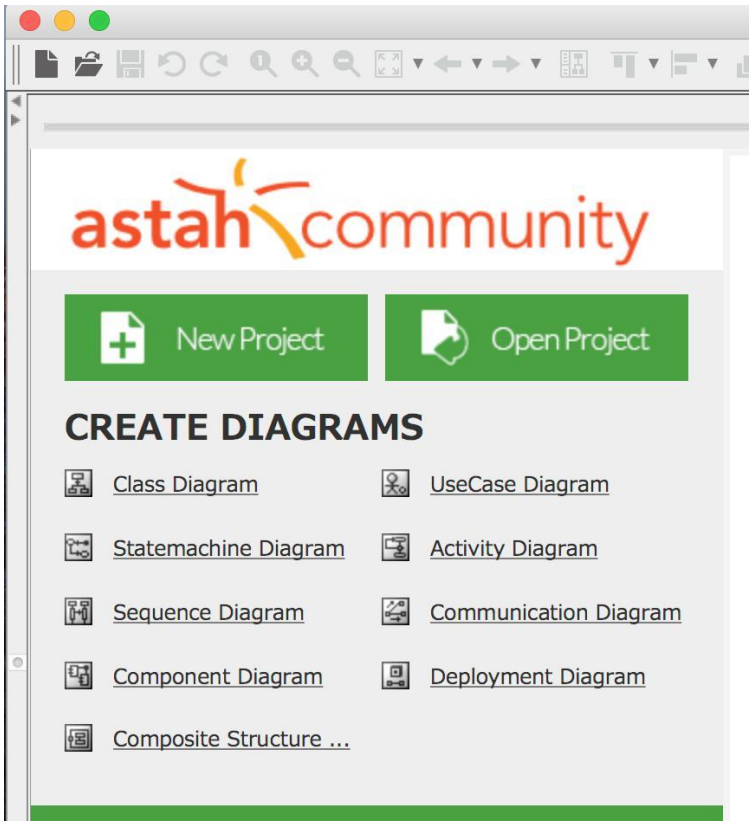
4.3. ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

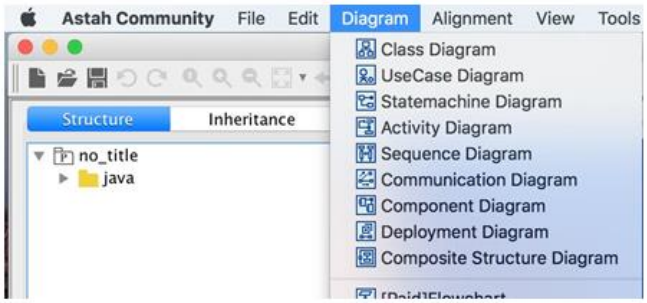
1. Komputer.
2. Software pendukung UML Diagram (Astah, Eclipse, dsb)

4.4. LANGKAH PRAKTIKUM

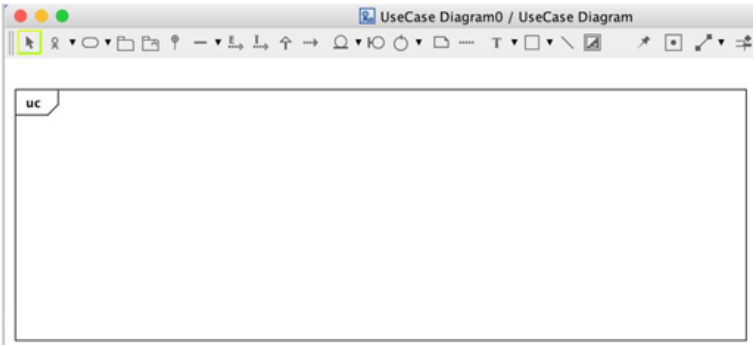
1. Perhatikan hasil observasi kebutuhan fungsional dari studi kasus yang telah ditentukan dalam masing-masing kelas praktikum.
2. Tentukan aktor yang telibat (contoh: manusia, sistem eksternal).
3. Dengan menggunakan Astah: Buat projek baru dengan klik New Project.



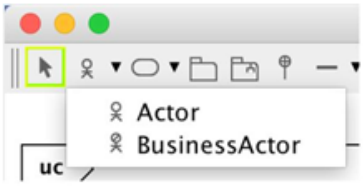
4. Pilih Menu Diagram dan pilih Use Case Diagram

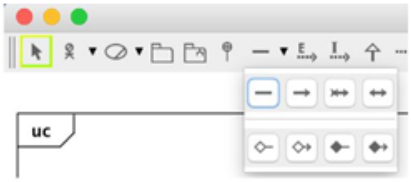


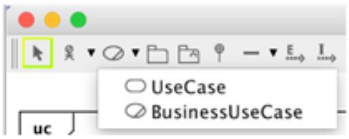
5. Notasi UseCase Diagram dapat ditemui pada toolbar jendela UseCase Diagram



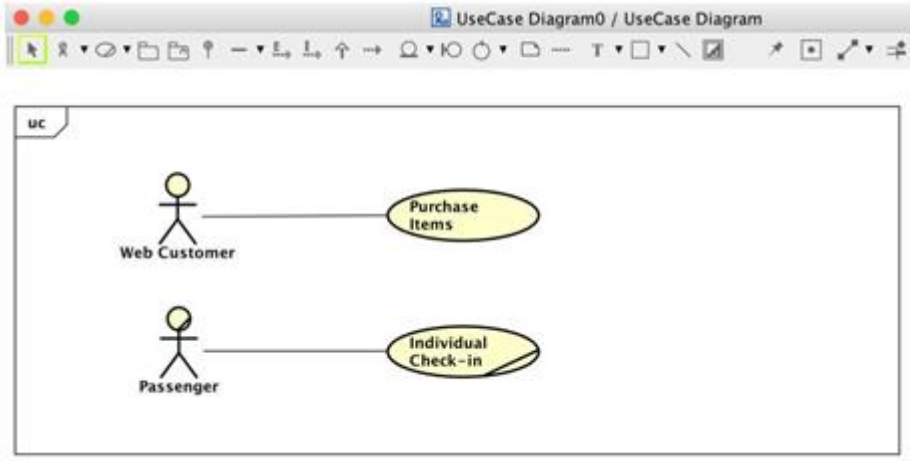
6. Untuk menggambar aktor, usecase atau relasi pilih ikon pada toolbar kemudian klik drag ke area subject







7. Buat gambar use casenya



4.5 TUGAS

1. Pahami setiap notasi yang digunakan dalam perancangan use case diagram. Asisten praktikum akan memberikan perintah untuk menjelaskan beberapa notasi dan fungsinya, serta contoh rancangannya untuk studi kasus lain.
2. Buat laporan use case diagram sesuai dengan tugas proyek tim Anda.

CONTOH
LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 4: UML dan USE CASE DIAGRAM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 5: ACTIVITY DIAGRAM

Pertemuan ke : 5

Total Alokasi Waktu : 150 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 30 menit
- Praktikum : 90 menit
- Post-Test : 30 menit
- dst

Total Skor Penilaian : 100% (Bobot skor disesuaikan dengan RPS)
- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

5.1 TUJUAN DAN INDIKATOR CAPAIAN

Setelah mengikuti praktikum ini mahasiswa diharapkan:

1. Mampu memahami definisi Activity Diagram dalam UML Diagrams
2. Mampu membuat rancangan Activity Diagram

Indikator ketercapaian diukur dengan:

1. Hasil analisis tertuang dalam gambar Activity Diagram dari studi kasus yang ditentukan saat ini
2. Tergambar Activity Diagram yang diharapkan untuk menyelesaikan permasalahan

5.2 TEORI PENDUKUNG

Activity Diagram/Diagram aktivitas adalah UML behavior diagram /diagram perilaku UML yang menunjukkan flow of control/aliran kontrol atau arus objek/object flow dengan penekanan pada urutan dan kondisi arus. Tindakan yang dikoordinasikan oleh model aktivitas dapat dimulai karena tindakan lain selesai dijalankan, karena objek dan data tersedia, atau karena beberapa kejadian di luar arus terjadi.

Activity

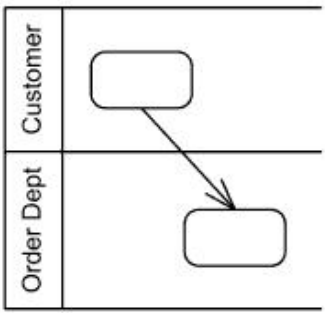
Activity/aktivitas adalah perilaku parameter yang ditunjukkan sebagai arus tindakan terkoordinasi. Aliran eksekusi dimodelkan sebagai node aktivitas yang dihubungkan oleh tepi aktivitas. Sebuah simpul bisa menjadi eksekusi dari perilaku bawahan, seperti perhitungan aritmatika, panggilan ke operasi, atau manipulasi isi objek. Activity nodes juga mencakup flow of control constructs, seperti synchronization, decision, dan concurrency control.

Dalam model berorientasi objek, aktivitas biasanya dipanggil secara tidak langsung sebagai metode yang terikat pada operasi yang dipanggil secara langsung.

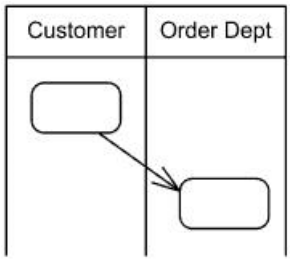
Activity Partition

Activity Partition/Partisi aktivitas adalah kelompok kegiatan untuk tindakan yang memiliki karakteristik umum. Partisi sering sesuai dengan **unit organisasi** atau **pelaku bisnis** dalam **model bisnis**.

Partisi aktivitas dapat ditunjukkan dengan menggunakan notasi **swimlane** - dengan dua, biasanya garis sejajar, baik horizontal atau vertikal, dan nama yang melabeli partisi dalam sebuah kotak di salah satu ujungnya. Setiap simpul aktivitas, mis. Tindakan dan tepi yang ditempatkan di antara garis-garis ini dianggap terkandung di dalam partisi (Gambar 5.1 dan Gambar 5.2).

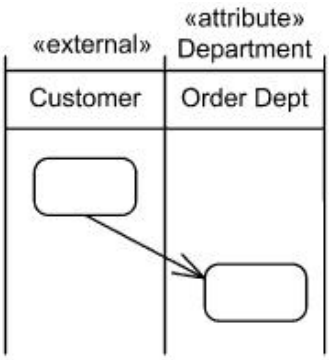


Gambar 5.1 Partisi Kegiatan Aktor sebagai Swimlanes Horizontal



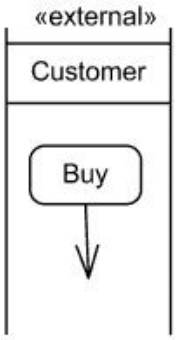
Gambar 5.2 Partisi Kegiatan Aktor sebagai Swimlanes Vertikal

Partisi dapat mewakili beberapa atribut dan subpartisinya - nilai spesifik atribut itu. Misalnya, partisi mungkin mewakili lokasi di mana perilaku dilakukan, dan sub-partisi akan mewakili nilai spesifik untuk atribut itu, seperti New York. Partisi hirarkis diwakili dengan menggunakan swimlanes untuk sub-partisi seperti diilustrasikan di Gambar 5.3.



Gambar 5.3 Partisi Hierarkis dengan Sub-Partisi

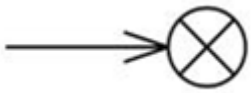
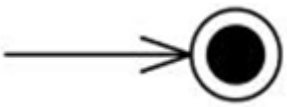
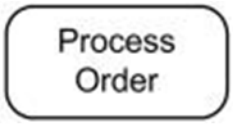
Partisi bisa mewakili entitas eksternal yang struktur partisinya tidak berlaku. Partisi eksternal adalah pengecualian yang disengaja terhadap peraturan untuk struktur partisi. Misalnya, dimensi mungkin memiliki partisi yang menunjukkan bagian pengklasifikasi terstruktur. Ini dapat memiliki partisi eksternal yang tidak mewakili salah satu bagian, namun merupakan klasifikasi yang benar-benar terpisah. Dalam pemodelan bisnis, partisi eksternal dapat digunakan untuk model entitas di luar bisnis. Bila aktivitas dianggap terjadi di luar domain model tertentu, partisi dapat diberi label dengan kata kunci «eksternal» seperti Gambar 5.4. Kapan pun sebuah aktivitas di swimlane ditandai «eksternal», ini menimpa perilakunya dan penunjukan dimensi.

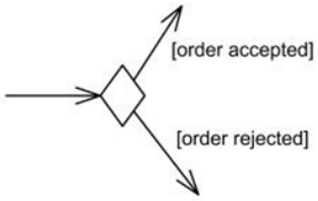
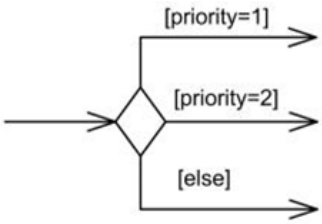
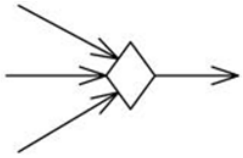
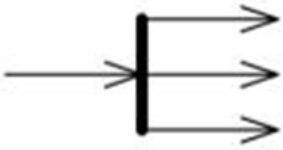
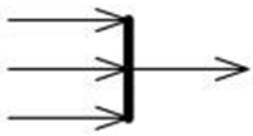


Gambar 5.4 Buy Action terjadi pada Partisi Eksternal Customer

Untuk lebih lanjut, Tabel 5.1 akan menjelaskan tentang notasi-notasi dalam activity diagram.

Tabel 5.1 Notasi yang Digunakan dalam Activity Diagram

Notasi	Penjelasan
 Activity initial node	**Initial node/**Simpul awal adalah node kontrol dimana arus dimulai saat aktivitas dipanggil. Aktivitas mungkin memiliki lebih dari satu simpul awal. Simpul awal ditampilkan sebagai lingkaran padat kecil.
 Flow final node	**Flow final node/**Arus simpul akhir adalah node akhir kontrol yang mengakhiri aliran. Notasi untuk node arus akhir adalah lingkaran kecil dengan X di dalamnya.
 Activity final node	**Activity final/**Aktivitas simpul terakhir adalah node akhir control yang menghentikan semua arus dalam suatu kegiatan. Aktivitas simpul terakhir ditampilkan sebagai lingkaran padat dengan lingkaran berongga di dalamnya.
 Action	Action/tindakan digambarkan sebagai persegi panjang dengan ujung melingkar. Nama tindakan atau deskripsi lainnya mungkin muncul dalam symbol.

 Decision	**Decision/**Keputusan digambarkan oleh node dengan dua tepi keluar. Decision node adalah node kontrol yang menerima token pada satu atau dua sisi yang masuk dan memilih satu tepi keluar dari satu atau lebih arus keluar.
 Decision	Keputusan node dengan tiga tepi keluar dan [else]. Untuk poin keputusan, panah "ELSE" yang telah ditentukan dapat didefinisikan paling banyak satu tepi keluar.
 Merge	Merge/Gabungan simpul dengan tiga sisi yang masuk dan tepi keluar tunggal. Merge node adalah node kontrol yang menyatukan beberapa arus masuk untuk menerima arus keluar tunggal. Tidak ada penggabungan token. Gabung seharusnya tidak digunakan untuk menyinkronkan arus bersamaan.
 Fork	Simpul fork/garpu dengan tepi aktivitas tunggal yang memasukinya, dan tiga sisi meninggalkannya. Fork node adalah node kontrol yang memiliki satu edge yang masuk dan beberapa tepi keluar dan digunakan untuk membagi arus masuk menjadi beberapa aliran bersamaan. Notasi untuk simpul garpu adalah segmen garis dengan tepi aktivitas tunggal yang memasukinya, dan dua atau lebih ujungnya meninggalkannya.
 Join	Join node/Node gabungan dengan tiga sisi aktivitas yang masuk dan satu sisi meninggalkannya. Penggabungan simpul adalah node kendali yang memiliki beberapa tepi masuk dan satu tepi keluar dan digunakan untuk menyinkronkan arus masuk bersamaan. Notasi untuk node join adalah segmen garis dengan beberapa tepi aktivitas yang memasukinya, dan hanya satu sisi yang meninggalkannya.

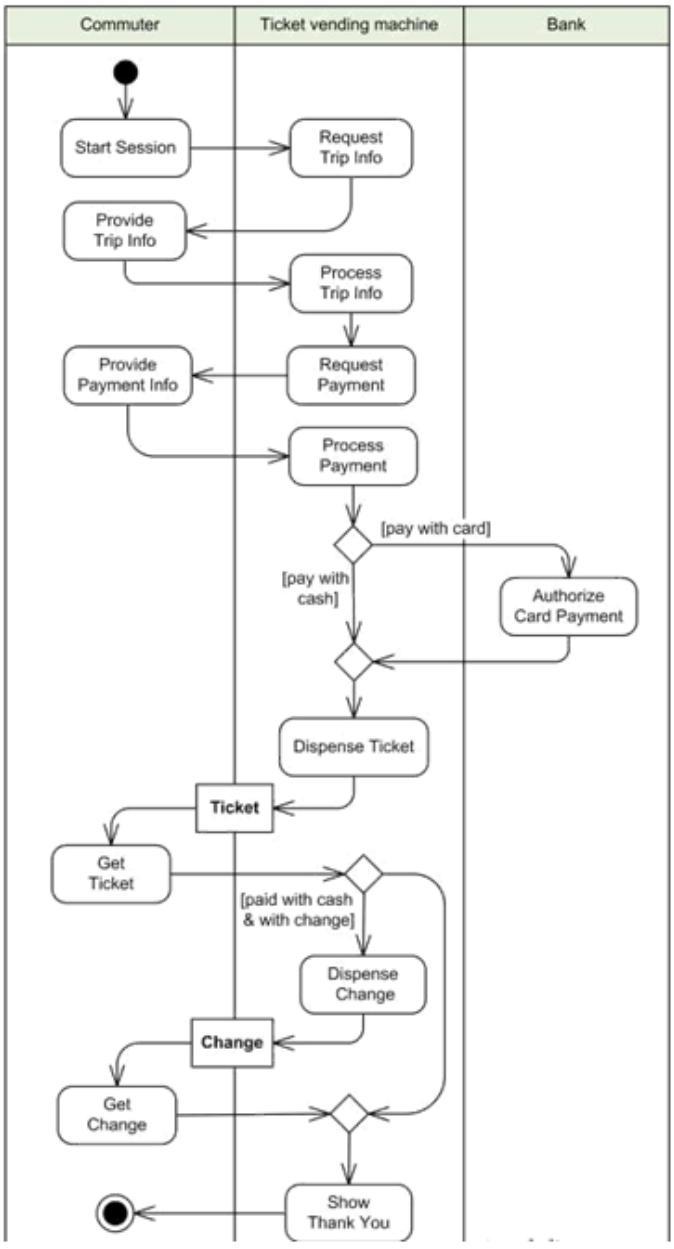
Contoh Kasus: Ticket Vending Machine

Gambar 5.6 adalah contoh diagram aktivitas UML yang menggambarkan perilaku kasus penggunaan Tiket Pembelian.

Kegiatan dimulai oleh pelaku komuter yang perlu membeli tiket. Mesin penjual tiket akan meminta informasi perjalanan dari Commuter. Informasi ini akan mencakup nomor dan jenis tiket, mis. Apakah itu tiket bulanan, tiket satu arah atau bulat, nomor rute, nomor tujuan atau zona, dll.

Berdasarkan info yang disediakan info tiket mesin penjual akan menghitung pembayaran jatuh tempo dan meminta opsi pembayaran. Pilihan tersebut meliputi pembayaran secara tunai, atau dengan kartu kredit atau debit. Jika pembayaran dengan kartu dipilih oleh Commuter, aktor lain, Bank akan berpartisipasi dalam kegiatan tersebut dengan memberi otorisasi pembayaran.

Setelah pembayaran selesai, tiket dibagikan kepada Commuter. Pembayaran tunai bisa mengakibatkan beberapa perubahan karena, sehingga perubahan tersebut dibagikan kepada Commuter dalam kasus ini. Mesin penjual tiket akan menampilkan beberapa layar "Terima Kasih" di akhir aktivitas.



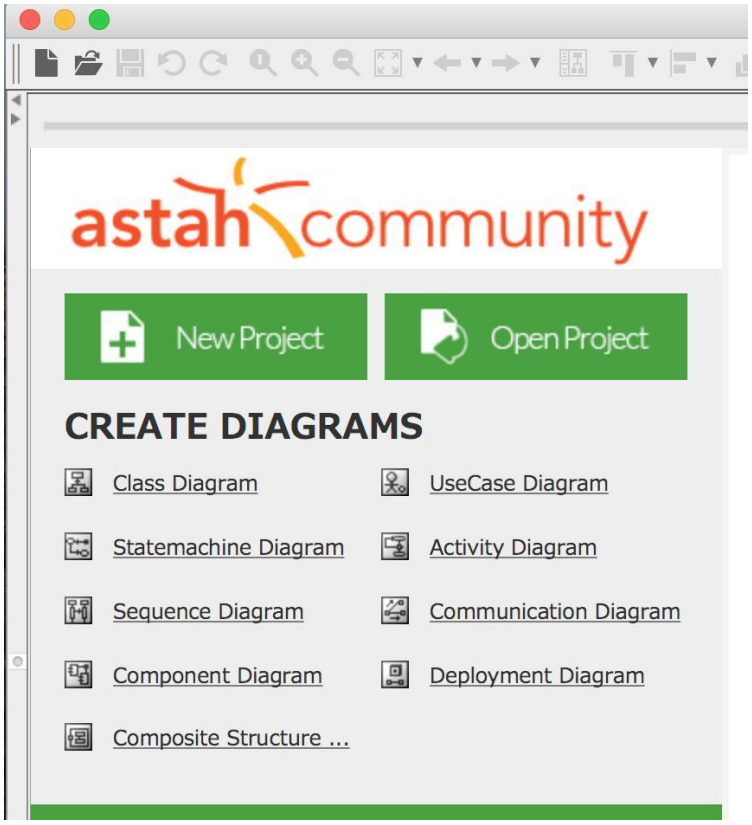
Gambar 5.5 Contoh Activity Diagram untuk Kasus Pembelian Tiket Otomatis

5.3 ALAT DAN BAHAN

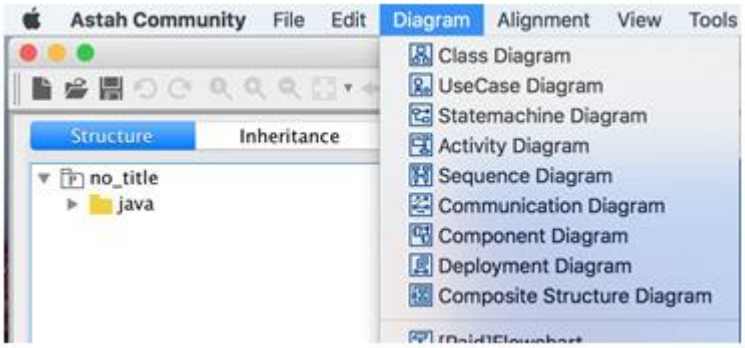
1. Komputer
2. Software Pendukung: Astah

5.4 LANGKAH PRAKTIKUM

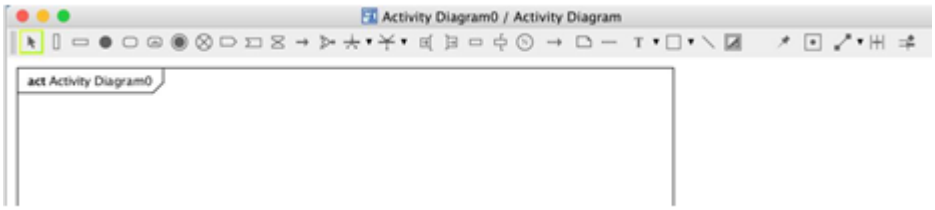
1. Berdasarkan studi kasus pengembangan aplikasi yang telah ditentukan di kelas praktikum, buatkan activity diagram untuk studi kasus tersebut di kondisi saat ini.
2. Buatkan analisis activity diagram, berdasarkan hasil pencatatan fungsionalitas dengan calon pengguna.
3. Rancang activity diagram menggunakan alat bantu Astah Community
4. Buat Project Baru



5. Pilih Menu Diagram dan pilih Activity Diagram



6. Notasi activity diagram dapat ditemui pada toolbar jendela activity diagram



7. Buat gambar activity diagramnya dengan drag and drop komponen notasi activity diagram di bagian toolbar. Gunakan Partial activity untuk mendeklarasikan aktivitas yang dilakukan tiap aktor dalam sistem yang terlibat di aktivitas tersebut.

5.5 TUGAS

1. Pahami setiap notasi yang digunakan dalam perancangan use case diagram. Asisten praktikum akan memberikan perintah untuk menjelaskan beberapa notasi dan fungsinya, serta contoh rancangannya untuk studi kasus lain.
2. Buat laporan activity diagram sesuai dengan tugas proyek tim Anda.

CONTOH
LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 5: ACTIVITY DIAGRAM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 6: USER INTERFACE DESIGN

Pertemuan ke : 6

Total Alokasi Waktu : 150 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 30 menit
- Praktikum : 90 menit
- Post-Test : 30 menit
- dst

Total Skor Penilaian : 100% (Bobot skor disesuaikan dengan RPS)
- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

6.1 TUJUAN DAN INDIKATOR CAPAIAN

Setelah mengikuti praktikum ini mahasiswa diharapkan:

1. Memahami konsep dasar *User Interface* (UI)
2. Merancang *user interface* dari rancangan diagram aktifitas
3. Mengimplementasikan diagram aktifitas ke dalam alur *interface*.

Indikator ketercapaian diukur dengan:

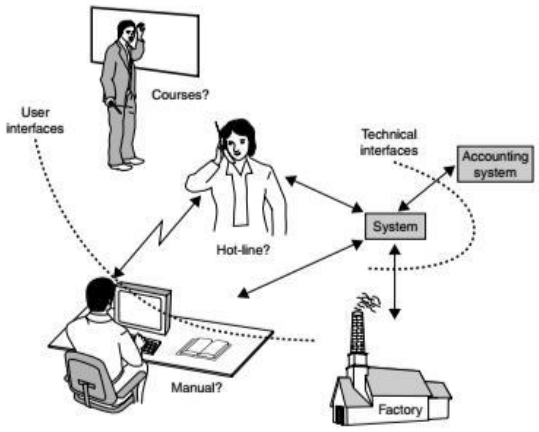
1. Fungsionalitas yang akan membutuhkan interaksi lewat UI telah diidentifikasi
2. Desain UI yang diturunkan dari diagram aktifitas telah dirancang dan diimplementasikan

6.2 TEORI PENDUKUNG

User *Interface* (UI) adalah bagian dari sistem yang dapat dilihat, didengar ataupun dirasakan oleh pengguna. Sedangkan bagian lainnya tersembunyi, misalnya adalah dimana basis data disimpan. Meskipun pengguna tidak melihat bagian yang tersembunyi, mereka membayangkan apa yang terjadi di 'belakang layar'. Apa yang mereka bayangkan mengenai apa yang terjadi ini seringkali mempunyai arti penting ketika pengguna menemui situasi dimana sistem tidak bekerja dengan seharusnya.

Saat kita menggunakan komputer, kita menjalankannya dengan memberikan perintah, umumnya melalui *mouse* dan *keyboard*. Komputer akan merespon perintah kita, biasanya dengan memperlihatkan sesuatu di layer atau membuat suatu suara. Terkadang situasi tersebut terbalik, komputer yang memberikan perintah dan kita yang memberikan respon.

Sebuah bioskop adalah sebuah contoh – penonton tidak berinteraksi langsung dengan filmnya. Contoh lain adalah *smoke detector* – alat ini bekerja dengan terus menerus mendeteksi. Interaksi dengan komputer ini terjadi melalui adanya User *Interface* (UI). Dalam sebuah komputer PC pada umumnya, aplikasi UI yang ada terdiri dari layar, *keyboard*, *mouse* dan *speaker* (Gambar 6.1).



Gambar 6.1 Sistem Interface

Pada sistem yang lebih canggih, *interface* yang ditampilkan mungkin meliputi suara melalui mikrofon; tombol khusus, lampu dan tampilan layar; sarung tangan elektrik yang bisa mendeteksi gerakan jari tangan; dan sensor mata yang dapat mendeteksi arah gerak bola mata terhadap layar.

Tujuan dari aktifitas perancangan *interface* ini antara lain:

1. Untuk memperoleh desain dan tampilan *interface* dengan dasar fundamental yang kuat, yang langsung diturunkan sejak dari fase pertama analisis perancangan sistem perangkat lunak, khususnya dari fase diagram aktifitas.
2. Untuk memberikan ruang bagi pengembang untuk mengetahui bagaimana pandangan pengguna terhadap desain *interface* yang diusulkan, dan untuk memperbaiki error (jika ada).

Untuk menciptakan dukungan kerjasama / kolaborasi yang baik antara perancang *User Interface* dan programmer (*software developer*).

Best Practices Membangun *Interface*

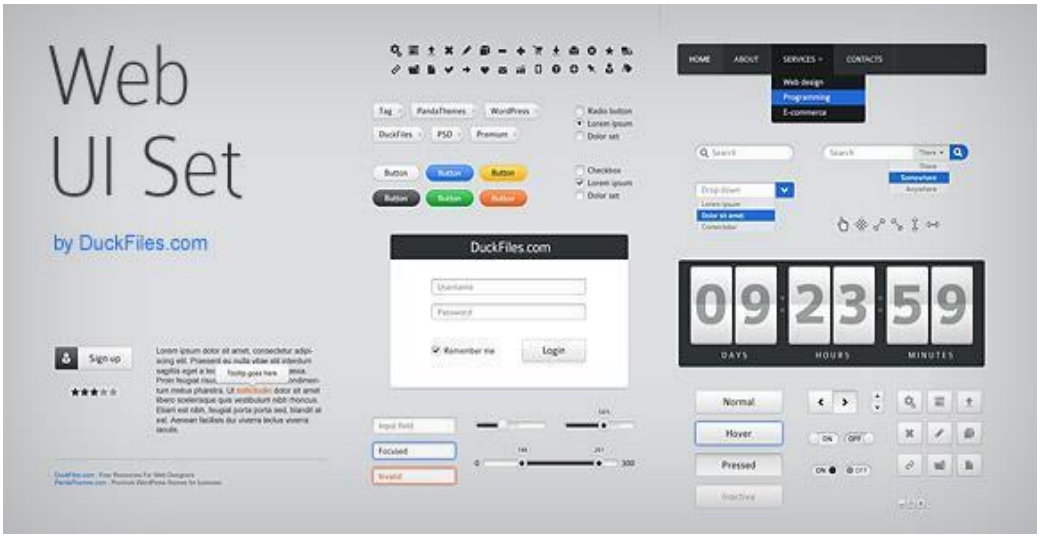
Ketika membangun sebuah sistem telah sampai pada tahapan perancangan *interface*, beberapa hal yang harus diketahui antara lain:

- **Buat *interface* yang sederhana.**

Tidak ada *interface* yang paling baik untuk semua pengguna, yang penting hindari elemen yang tidak perlu dan menggunakan Bahasa yang jelas ketika menggunakan label dan menuliskan pesan (*message*).

- **Ciptakan elemen UI yang konsisten dan umum diketahui.**

Pengguna akan merasa lebih nyaman dan dapat berinteraksi dengan sistem secara lebih cepat. Selain itu, penting juga untuk menciptakan pola dalam kalimat Bahasa yang digunakan, dalam layout dan keseluruhan perancangan untuk alasan efisiensi. Apabila pengguna sudah memahami bagaimana caranya melakukan sesuatu melalui *interface* tersebut, mereka akan jadi terbiasa dalam mengoperasikan sistem / PL tersebut. Gambar 6.2 menyajikan contoh elemen-elemen yang umum diketahui:



Gambar 6.2 Contoh Elemen-Elemen User Interface [sumber: "User Interface Elements Free"]

- **Buatlah layout halaman yang mempunyai tujuan**

Pertimbangkan hubungan antara hal-hal yang ingin ditampilkan di sebuah halaman dan bagaimana struktur halaman berdasarkan bagian mana yang penting. Penempatan informasi tersebut harus dilakukan dengan hati-hati dengan tujuan untuk menarik perhatian pengguna terhadap bagian informasi yang paling penting.

- **Gunakan warna dan tekstur yang taktis.**

Pengembang *interface* dapat mengarahkan atau mengembalikan perhatian terhadap suatu hal melalui warna, pencahayaan, dan tekstur, bergantung kepada tujuan yang ingin dicapai.

- **Gunakan tipografi untuk menciptakan kemampuan pemahaman pengguna.**

Tipografi adalah seni mengatur atau menggunakan huruf, kata atau paragraf pada ruang yang tersedia. Berhati-hatilah dalam menggunakan jenis font, ukuran font dan pengaturan tata letak teks untuk meningkatkan kemampuan memahami pengguna.

- **Pastikan bahwa sistem/PL selalu mengkomunikasikan apa yang terjadi.**

Selalu berikan informasi kepada pengguna mengenai lokasi, aksi, perubahan (bila ada) atau mengenai adanya *error*. Penggunaan berbagai macam elemen UI untuk mengkomunikasikan sebuah status aktifitas, dan mungkin saja langkah selanjutnya sebagai respon atas status yang terjadi.

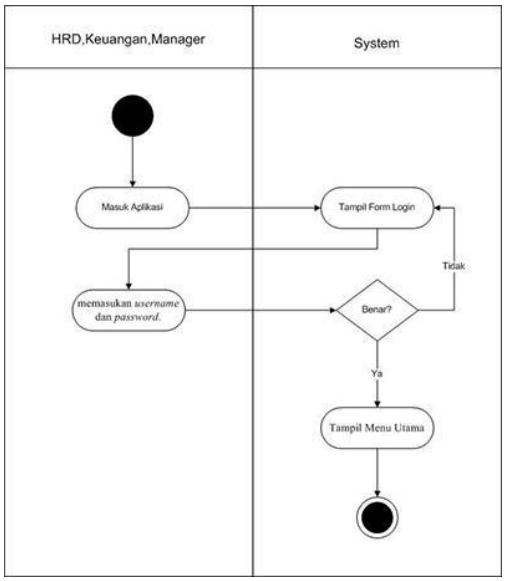
- **Pikirkan mengenai pengaturan awal (*default*).**

Dengan mengantisipasi tujuan yang ingin dicapai pengguna dari sistem / PL, pengembang *interface* dapat membuat sebuah pengaturan awal (*default*) yang bisa mengurangi beban pengguna. Hal ini menjadi penting ketika ada rancangan form, dan ada isian *default* yang akan otomatis tertulis saat pengguna tidak mengisi *form* tersebut.

Memetakan Aktifitas ke Dalam Rancangan UI

Di praktikum ini, kita akan belajar merancang *interface* yang diturunkan dari diagram aktifitas. Langkah-langkah yang dapat dilakukan antara lain:

1. Siapkan diagram aktifitas dari sistem perangkat lunak yang sudah dirancang sebelumnya. Contoh di bawah ini adalah diagram aktifitas untuk aktifitas login, seperti yang ditunjukkan pada Gambar 6.3.



Gambar 6.3 Contoh Diagram Aktifitas untuk Log In [https://www.dumetschool.com/blog/Apa-Itu-Activity-Diagram]

2. Dari setiap diagram aktifitas yang dibuat, buat dokumen aktifitas yang mendata tentang fungsionalitas apa saja yang dimiliki. Dari masing-masing fungsionalitas tersebut, selanjutnya diturunkan ke dalam elemen-elemen UI. Dokumen aktifitas dibuat berdasarkan siapa saja yang menjadi subjek pelaku (aktor). Contoh dokumen aktifitas bisa dilihat seperti pada Gambar 6.4.

Judul Dokumen : .....
Jumlah aktifitas : M
Jumlah aktifitas yang perlu UI : N
1. Nama aktifitas : .....
Fungsionalitas : .....
Elemen UI : .....
Catatan : .....
2. Nama aktifitas : .....
Fungsionalitas : .....
Elemen UI : .....
Catatan : .....
.....
.....
.....
N. Nama aktifitas : .....
Fungsionalitas : .....
Elemen UI : .....
Catatan : .....

Gambar 6.4 Dokumen Aktifitas [diadaptasi dari "Producing Graphical User Interface from Activity Diagrams"]

3. Jumlah aktifitas diisi dengan berapa jumlah aktifitas yang ada di setiap actor, misal berjumlah M. sedangkan jumlah aktifitas yang perlu UI diisi dengan berapa jumlah M yang akan berinteraksi dengan pengguna lewat *interface*, misal berjumlah N. M dan N tidak harus selalu sama. Contoh:

Judul Dokumen : User (HRD, Keu, Man)
Jumlah aktifitas : 2
Jumlah aktifitas yang perlu Interaksi UI : 1
1. **Nama aktifitas** : Memasukkan UsName & Pwd
Fungsionalitas : (1) user masuk sistem dengan mengetikkan username & password
(2) user submit data dengan tombol Login
(3) user dapat meminta bantuan bila lupa UsName atau Password
Elemen UI : (1) form textField UsName & Password, (2) Button Submit, (3) Link "Forgot
Catatan : -

Gambar 6.5 Contoh Dokumen Aktifitas untuk Log In

4. Fungsionalitas diisi dengan kemampuan apa saja yang harus dilakukan oleh aktifitas terkait. Misal, fungsionalitas registrasi, menambah atau menghapus data barang. Elemen UI disesuaikan dengan fungsionalitas yang tadi sudah didata.
5. Catatan digunakan untuk memperjelas interaksi yang terjadi antar elemen yang harus dapat dilakukan lewat *interface*.
6. Setiap elemen UI yang merepresentasikan setiap fungsionalitas kemudian digambarkan dalam rancangan *mockup* sederhana (Gambar 6.6 untuk contoh *mockup* fungsionalitas Login).

Gambar 6.6 Mockup UI untuk Fungsi Log In

6.3 ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Software Pendukung: Astah, Word Editor, UI Designer: Balsamiq/Justinmind

6.4 LANGKAH PRAKTIKUM

1. Berdasarkan studi kasus pengembangan aplikasi yang dipilih di kelas, rancanglah antarmuka yang diperlukan sesuai dengan diagram aktifitas yang dibuat sebelumnya.
2. Siapkan diagram aktifitas yang telah dibuat dari masing-masing studi kasus yang dipilih.
3. Buat dokumen aktifitas untuk setiap aktifitas yang memiliki fungsionalitas yang harus ditampilkan melalui *interface*.
4. Dari tabel dokumen yang dibuat di poin (3), buat rancangan/desain *interface (mockup)* nya.
5. Lakukan verifikasi fungsionalitas dan rancangan *interface* yang diajukan dengan asisten praktikum

6.5 TUGAS

1. Pahami setiap notasi yang digunakan dalam perancangan User Interface dari Activity Diagram. Asisten praktikum akan memberikan perintah untuk menjelaskan beberapa simbol navigasi dan fungsinya, serta bagaimana contoh rancangannya untuk studi kasus lain.
2. Buat laporan perancangan UI untuk studi kasus masing-masing sesuai pada langkah-langkah praktikum.

CONTOH
LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 6: USER INTERFACE DESIGN

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 7: CLASS DIAGRAM (KELAS & ATRIBUT)

Pertemuan ke : 7

Total Alokasi Waktu : 150 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 30 menit
- Praktikum : 90 menit
- Post-Test : 30 menit
- dst

Total Skor Penilaian : 100% (Bobot skor disesuaikan dengan RPS)
- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

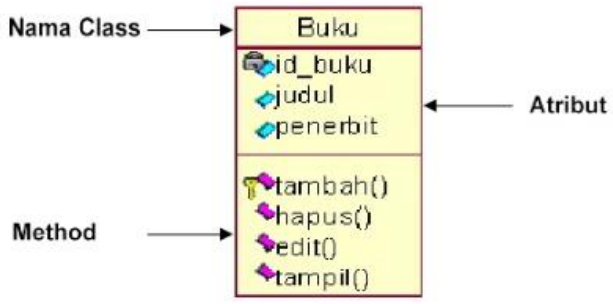
7.1 TUJUAN DAN INDIKATOR CAPAIAN

Setelah mengikuti praktikum ini mahasiswa diharapkan:
1. Mampu memahami analisis dan notasi Class diagram untuk kelas dan atribut
2. Mampu membuat rancangan kelas dan atribut dalam class diagram

Indikator ketercapaian diukur dengan:
1. Hasil rancangan tertuang dalam class diagram dari studi kasus yang ditentukan saat ini
2. Tergambar class diagram yang diharapkan untuk menyelesaikan permasalahan.

7.2 TEORI PENDUKUNG

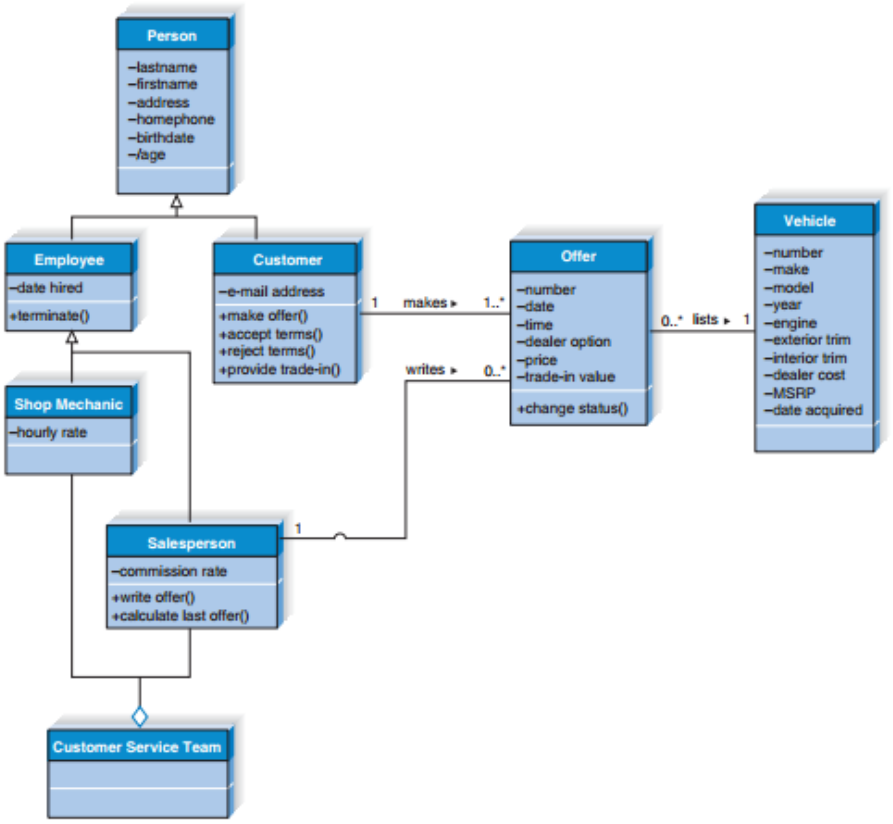
Class diagram (Gambar 7.1) digunakan untuk menggambarkan tentang kelas-kelas serta paket yang ada di dalam sistem yang sedang dikembangkan.



Gambar 7.1 Struktur Class Diagram

Class

Class diagram adalah sebuah diagram untuk memodelkan kelas-kelas dan relasinya seperti halnya Entity Relationship Diagram (ERD). Berbeda halnya dengan ERD yang hanya menggambarkan atribut, Class Diagram menggambarkan kelas-kelas yang meliputi atribut, behavior dan state.



Gambar 7.2 Contoh Class Diagram untuk Sistem Persewaan Kendaraan

Pembangun utama dalam class Diagram adalah kelas, yang menyimpan dan mengelola informasi. Pada Gambar 7.2, contoh dari kelas-kelas antara lain Person, Employee dan Vehicle. Setiap kelas digambarkan dengan menggunakan persegi panajng yang di dalamnya dibagi menjadi 3 bagian. Paling atas dituliskan nama kelas, kemudian bagian kedua adalah atribut dan bagian terbawah menjadi tempat operasi (methods).

Atribut

Atribut adalah properti dari kelas untuk menyimpan informasi. Bila dilihat pada Gambar 7.2, sebuah kelas *Person* mempunyai atribut *lastname, firstname, address, phone, birthdate* dan *age*. Mungkin suatu saat nanti akan ada atribut turunan; misalnya atribut yang diperoleh dari atribut lainnya atau dari hasil kalkulasi. Atribut seperti ini tidak perlu disimpan, tetapi hanya perlu diberi tanda slash (/) seperti yang dilihat di atribut "/age", sebagai hasil pengurangan dari atribut *birthdate* dengan *currentdate.*

Term and Definition	Symbol
A class ▪ Represents a kind of person, place, or thing about which the system must capture and store information. ▪ Has a name typed in bold and centered in its top compartment. ▪ Has a list of attributes in its middle compartment. ▪ Has a list of operations in its bottom compartment. ▪ Does not explicitly show operations that are available to all classes.	Class name –Attribute name –/derived attribute name +Operation name ()
An attribute ▪ Represents properties that describe the state of an object. ▪ Can be derived from other attributes, shown by placing a slash before the attribute's name.	Attribute name /derived attribute name
A method ▪ Represents the actions or functions that a class can perform. ▪ Can be classified as a constructor, query, or update operation. ▪ Includes parentheses that may contain special parameters or information needed to perform the operation.	Operation name ()
An association ▪ Represents a relationship between multiple classes, or a class and itself. ▪ Is labeled by a verb phrase or a role name, whichever better represents the relationship. ▪ Can exist between one or more classes. ▪ Contains multiplicity symbols, which represent the minimum and maximum times a class instance can be associated with the related class instance.	1..* verb phrase 0..1

Gambar 7.3 Notasi yang Digunakan untuk Class Diagram

7.3 ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

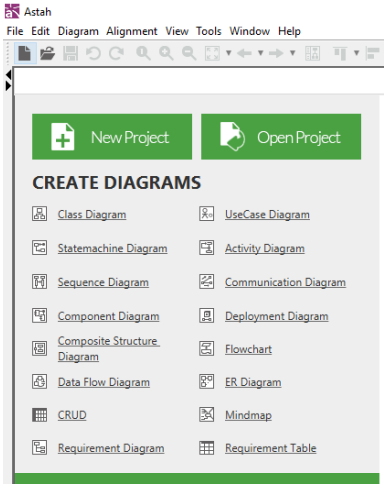
1. Komputer
2. Software pendukung: Astah

7.4 LANGKAH PRAKTIKUM

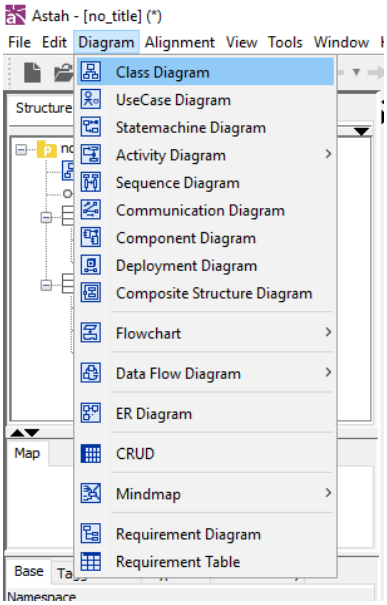
Berdasarkan studi kasus yang ditentukan di kelas praktikum, kita akan merancang class diagram sesuai dengan objek-objek yang terlibat.

Gunakan alat bantu Astah Community untuk merancang Class Diagramnya.

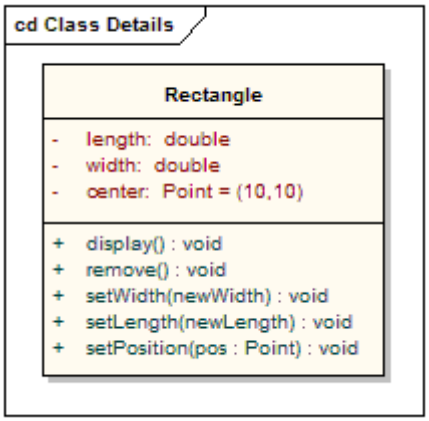
1. Buat Project Baru



2. Pilih Menu Diagram dan pilih class diagram



3. Notasi class diagram dapat ditemui pada toolbar jendela class diagram



4. Buat komponen class dengan memilih menu icon di toolbar

7.5 TUGAS

1. Pahami setiap notasi yang digunakan dalam perancangan Class Diagram. Asisten praktikum akan memberikan perintah untuk menjelaskan beberapa notasi dan fungsinya (kelas, atribut dan tipe data), serta bagaimana contoh rancangannya untuk studi kasus lain.
2. Buat laporan perancangan Class Diagram untuk pendefinisian berapa kelas yang dibutuhkan dan atribut yang melekat ke masing-masing kelas.

CONTOH
LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 7: CLASS DIAGRAM (KELAS & ATRIBUT)

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 8: CLASS DIAGRAM (RELASI)

Pertemuan ke : 8

Total Alokasi Waktu : 150 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 30 menit
- Praktikum : 90 menit
- Post-Test : 30 menit
- dst

Total Skor Penilaian : 100% (Bobot skor disesuaikan dengan RPS)
- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

8.1 TUJUAN DAN INDIKATOR CAPAIAN

Setelah mengikuti praktikum ini mahasiswa diharapkan:

1. Mampu memahami analisis dan notasi Class diagram untuk relasi antar kelas
2. Mampu membuat rancangan class diagram secara utuh

Indikator ketercapaian diukur dengan:

1. Hasil rancangan tertuang dalam class diagram dari studi kasus yang ditentukan saat ini
2. Tergambar class diagram yang diharapkan untuk menyelesaikan permasalahan.

8.2 TEORI PENDUKUNG

Generalisasi dan Agregasi

Generalisasi menunjukkan hubungan beberapa kelas sebagai *subclass* (child) dan *superclass* (*parent*), dimana atribut yang berada di *superclass* akan diturunkan dan digunakan juga oleh *subclass*. Generalisasi digambarkan dengan garis panah dari *subclass* ke kelas *superclass* dengan anak panah mengarah ke *superclass*. Pada gambar 6.2 di Praktikum 6, terdapat hubungan generalisasi pada hubungan antara kelas Person sebagai *superclass* dan kelas Employee dan kelas Customer sebagai kelas-kelas *subclass*.

Agregasi digunakan saat beberapa kelas direlasikan menjadi sebuah kelas baru. Misalnya pada kelas Customer Care Team. Kelas tersebut dibangun dengan tujuan aktivitas pelayanan pelanggan yang ingin disediakan dealer, yang terdiri dari teknisi dan pemasaran. Penggambaran agregasi dilakukan dengan simbol diamond yang diletakkan di kelas yang merepresentasikan agregasi (Cust Care Team). Kelas-kelas yang beragregasi kemudian dihubungkan dengan garis ke arah objek belahketupat.

Association

Fungsi utama dari Class Diagram adalah untuk menunjukkan asosiasi atau relasi yang dimiliki kelas satu dengan lainnya. Relasi ini digambarkan dengan melalui garis yang menghubungkan kotak kelas. Asosiasi ini sama dengan yang ditemukan dalam ERD.

Multiplicity

Multiplicity menunjukkan bagaimana setiap instance dalam objek berhubungan dengan instance lain. Penomoran diletakkan di ujung panah untuk menandakan jumlah minimum dan maksimum sebuah instance dapat berelasi. Pada umumnya, kelas-kelas saling berelasi dengan kondisi yang normal, seperti 1..1, 1..N atau N..N, tetapi tidak menutup kemungkinan adanya angka spesifik yang dicantumkan sebagai jumlah minimum/maksimum instance yang boleh berelasi. Gambar 8.1 menunjukkan ilustrasi notasi multiplicity dalam Class Diagram.

Instance(s)	Representation of Instance(s)	Diagram Involving Instance(s)	Explanation of Diagram
Exactly one	1	Department — 1 — Boss	A department has one and only one boss.
Zero or more	0..*	Employee — 0..* — Child	An employee has zero to many children.
One or more	1..*	Boss — 1..* — Employee	A boss is responsible for one or more employees.
Zero or one	0..1	Employee — 0..1 — Spouse	An employee can be married to zero or one spouse.
Specified range	2..4	Employee — 2..4 — Vacation	An employee can take between two to four vacations each year.
Multiple, disjoint ranges	1..3, 5	Employee — 1..3, 5 — Committee	An employee is a member of one to three or five committees.

Gambar 8.1 Jenis Multiplicity dan Notasinya

8.3 ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer
2. Software pendukung: Astah

8.4 LANGKAH PRAKTIKUM

Berdasarkan studi kasus yang ditentukan di kelas praktikum, kita akan merancang class diagram sesuai dengan objek-objek yang terlibat. Gunakan alat bantu Astah Community untuk merancang Class Diagramnya.

8.5 TUGAS

1. Pahami setiap notasi yang digunakan dalam perancangan Class Diagram. Asisten praktikum akan memberikan perintah untuk menjelaskan beberapa notasi (jenis relasi) dan fungsinya, serta bagaimana contoh rancangannya untuk studi kasus lain.
2. Lengkapi laporan perancangan Class Diagram di pertemuan sebelumnya untuk pendefinisian relasi yang menghubungkan antar kelas.

CONTOH LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 8: CLASS DIAGRAM (RELASI)

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 9: SEQUENCE DIAGRAM

Pertemuan ke : 9

Total Alokasi Waktu : 150 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 30 menit
- Praktikum : 90 menit
- Post-Test : 30 menit
- dst

Total Skor Penilaian : 100% (Bobot skor disesuaikan dengan RPS)
- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

9.1 TUJUAN DAN INDIKATOR CAPAIAN

Setelah mengikuti praktikum ini mahasiswa diharapkan:
1. Mampu memahami analisis dan notasi sequence diagram
2. Mampu membuat rancangan sequence diagram

Indikator ketercapaian diukur dengan:
1. Hasil rancangan tertuang dalam gambar sequence diagram dari studi kasus yang ditentukan saat ini
2. Tergambar sequence diagram yang diharapkan untuk menyelesaikan permasalahan

9.2 TEORI PENDUKUNG

Sequence diagram adalah bentuk diagram interaksi yang menunjukkan objek sebagai jalur kehidupan yang mengalir menuruni halaman, dengan interaksi mereka dari waktu ke waktu ditunjukkan sebagai pesan yang ditarik sebagai panah dari sumber lifeline ke garis hidup target. Sequence diagram bagus untuk menunjukkan objek mana yang berkomunikasi dengan objek lain; Dan pesan apa yang memicu komunikasi tersebut. Sequence diagram tidak dimaksudkan untuk menunjukkan logika prosedural yang kompleks.

Sequence diagram adalah diagram interaksi yang menunjukkan bagaimana objek beroperasi satu sama lain dan dalam urutan apa. Ini adalah konstruksi dari sebuah bagan urutan pesan. Sequence diagram menunjukkan interaksi objek yang diatur dalam urutan waktu. Ini menggambarkan objek dan kelas yang terlibat dalam skenario dan urutan pesan dipertukarkan antara objek yang dibutuhkan untuk menjalankan fungsi skenario Sequence diagram biasanya dikaitkan dengan realisasi kasus penggunaan dalam Tampilan Logis dari sistem yang sedang dikembangkan. Sequence diagram kadang disebut diagram acara atau skenario acara. Diagram urutan menunjukkan, sebagai garis vertikal paralel (lifelines), proses atau objek yang berbeda yang hidup bersamaan, dan, sebagai panah horisontal, pesan

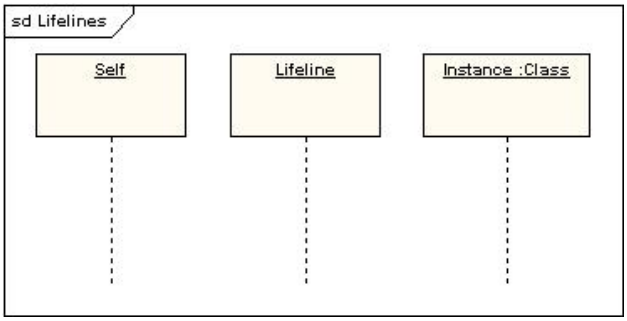
dipertukarkan di antara keduanya, sesuai urutan kemunculannya. Hal ini memungkinkan spesifikasi skenario runtime sederhana secara grafis.

Sequence diagram adalah jenis diagram interaksi yang paling umum, yang berfokus pada pertukaran pesan antara sejumlah lifelines. Sequence diagram menggambarkan interaksi dengan memusatkan perhatian pada urutan pesan yang dipertukarkan, bersamaan dengan spesifikasi kejadian yang sesuai pada jalur kehidupan.

Lifeline

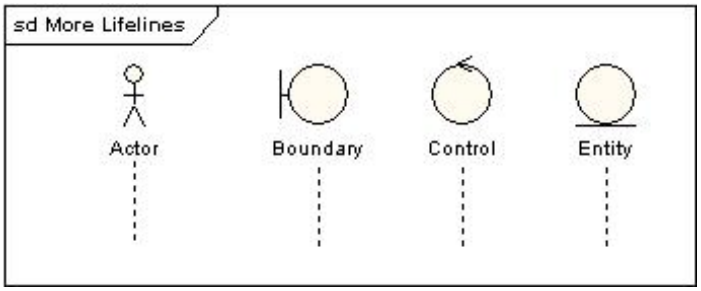
Lifeline (Gambar 9.1) adalah elemen bernama yang mewakili peserta individual dalam interaksi. Sementara bagian dan fitur struktural mungkin memiliki multiplisitas lebih besar dari 1, garis hidup hanya mewakili satu entitas yang berinteraksi.

Lifeline mewakili peserta individual dalam sequence diagram. Lifeline biasanya memiliki persegi panjang yang berisi nama objeknya. Jika namanya adalah "Self", itu menunjukkan lifeline mewakili penggolong yang memiliki sequence diagram.



Gambar 9.1 Penggambaran Lifeline Versi Nama Objek dalam Sequence Diagram

Terkadang sequence diagram akan memiliki *lifeline* dengan simbol elemen aktor di atasnya. Ini biasanya akan terjadi jika sequence diagram dimiliki oleh use case. Batasan, kontrol dan elemen entitas dari diagram dapat juga memiliki *lifeline* seperti pada Gambar 9.2 yang menggambarkan lifeline dalam *Boundary* (Batasan), *control* (kendali), *entity* (entitas).

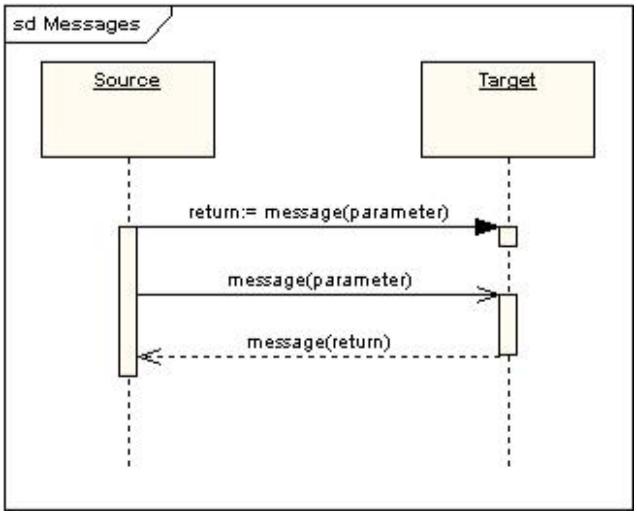


Gambar 9.2 Penggambaran Lifeline Versi Use Case dalam Sequence Diagram

Messages/pesan

Pesan ditampilkan sebagai tanda panah. Pesan bisa lengkap, hilang atau ditemukan; Sinkron atau asinkron; Panggilan atau sinyal Pada diagram berikut, pesan pertama adalah pesan sinkron (dilambangkan dengan panah padat) lengkap dengan pesan balik implisit; Pesan

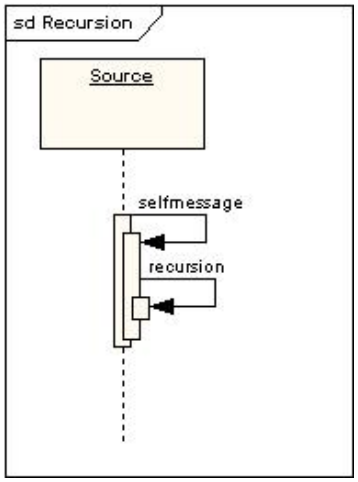
kedua adalah asinkron (dilambangkan dengan panah baris), dan yang ketiga adalah pesan kembali asinkron (dilambangkan dengan garis putus-putus) seperti gambar 9.3.



Gambar 9.3 Penggambaran Aliran Pesan dalam Sequence Diagram

Self Message

Self Message dapat mewakili panggilan rekursif suatu operasi, atau satu metode memanggil metode lain yang termasuk dalam objek yang sama. Hal ini ditunjukkan pada Gambar 9.4 sebagai menciptakan fokus pengendalian yang terpusat dalam kejadian eksekusi the lifeline.



Gambar 9.4 Penggambaran Self Message (Pesan ke Diri Sendiri) dalam Sequence Diagram

9.3 ALAT DAN BAHAN

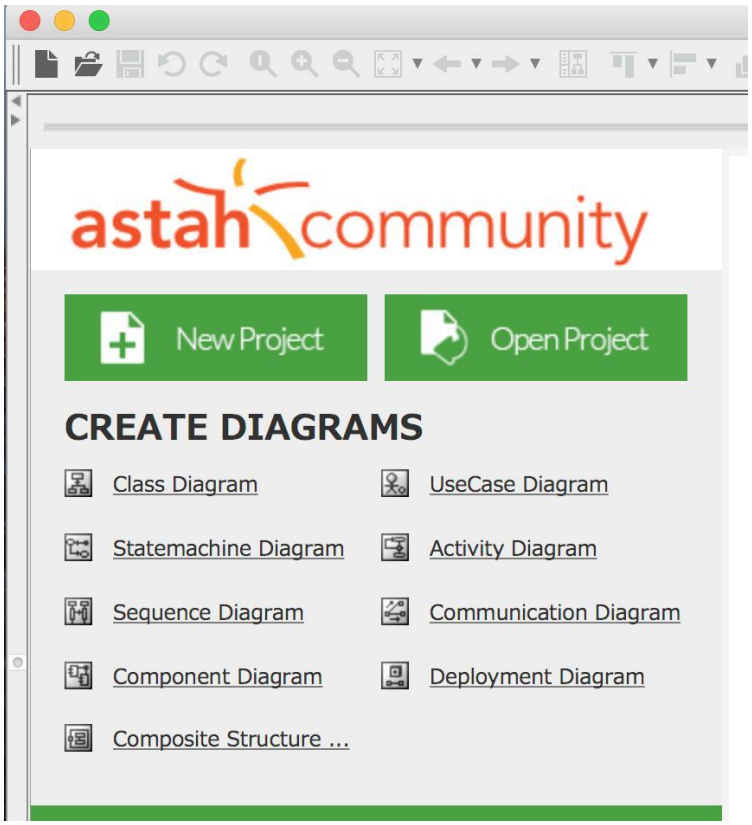
Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Software Pendukung: Astah

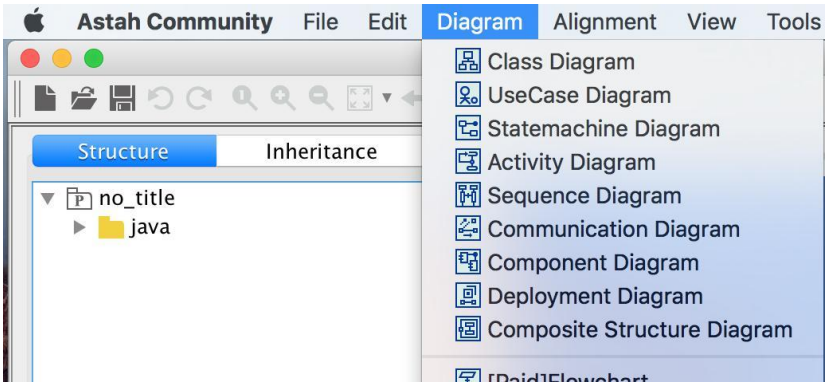
9.4 LANGKAH PRAKTIKUM

1. Berdasarkan studi kasus pengembangan aplikasi yang dipilih di kelas, buatkan sequence diagram

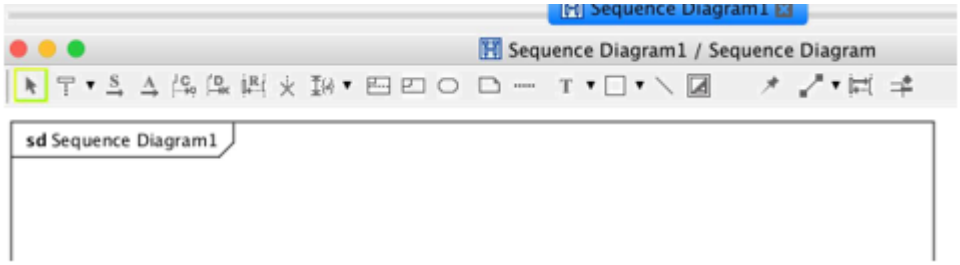
2. Menggambarkan sequence diagram menggunakan alat bantu astah community
3. Buat Project Baru



4. Pilih Menu Diagram dan pilih sequence diagram



5. Notasi sequence diagram dapat ditemui pada toolbar jendela sequence diagram



6. Buat gambar sequence diagram nya

9.5 TUGAS

1. Pahami setiap notasi yang digunakan dalam perancangan Sequence Diagram. Asisten praktikum akan memberikan perintah untuk menjelaskan beberapa notasi dan fungsinya, serta bagaimana contoh rancangannya untuk studi kasus lain.
2. Lengkapi laporan perancangan Sequence Diagram untuk studi kasus tim Anda.

CONTOH
LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 9: SEQUENCE DIAGRAM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 10: SOFTWARE COSTING ESTIMATION (USE CASE POINTS)

Pertemuan ke : 10

Total Alokasi Waktu : 150 menit (Alokasi waktu disesuaikan dengan RPS)
- Pre-Test : 30 menit
- Praktikum : 90 menit
- Post-Test : 30 menit
- dst

Total Skor Penilaian : 100% (Bobot skor disesuaikan dengan RPS)
- Pre-Test : 20 %
- Praktikum : 50 %
- Post-Test : 30 %

10.1 TUJUAN DAN INDIKATOR CAPAIAN

Setelah mengikuti praktikum ini mahasiswa diharapkan:

1. Mampu memahami ragam Software Costing dan Estimasi
2. Mampu menghitung Software Costing dan Estimasi berbasis usce case
3. Mampu mengidentifikasi alternatif solusi berupa proses bisnis yang diharapkan untuk menyelesaikan permasalahan di kondisi saat ini.

Indikator ketercapaian diukur dengan:

1. Hasil analisis tertuang dalam perhitungan dari studi kasus yang ditentukan saat ini;
2. Terhitungnya Software Costing dan Estimasi yang diharapkan untuk menyelesaikan permasalahan.

10.2 TEORI PENDUKUNG

Model Use Case Point (UCP) pertama kali dipopulerkan oleh Kerner pada tahun 1993. Model UCP terinspirasi oleh model Function Points (FP) tetapi dengan manfaat dari analisis persyaratan dalam proses Objectory. UCP dimulai dengan mengukur fungsionalitas sistem berdasarkan pada model use case dalam sebuah hitungan yang disebut dengan Unadjusted Use Case Point (UUCP). Faktor teknis yang terlibat dalam mengembangkan fungsi ini dinilai, mirip dengan FP. Langkah terakhir dalam estimasi, namun tidak dari FP dan itu adalah faktor yang disebut Environmental Factorbaru yang diusulkan oleh penulis. Faktor ini tampaknya sangat penting menurut pengalaman pengguna Objectory (Karner 1993).

UUCP - Unadjusted Use Case Point

Untuk menghitung UUCP dilakukan dengan menilai setiap aktor pada Use Case.Hasil penilaian berupa nilai sederhana, rata-rata atau kompleks dengan bantuan dari Tabel 10.1 dan setiap case digunakan dengan bantuan dari Tabel 10.1.

Tabel 10.1 Penilaian terhadap Aktor berdasarkan Definisi dan Bobot

Complexity	Definition	Weight
SIMPLE	An actor is simple if it represents another system with a defined application programming interface	1
AVERAGE	An actor is average if it is: 1. An interaction with another system through a protocol 2. A human interaction with a line terminal.	2
COMPLEX	An actor is complex if it interacts through a graphical user interface.	3

Tabel 10.2 Penilaian terhadap Use Case berdasarkan Definisi dan Bobot

Complexity	Definition	Weight
SIMPLE	A use case is simple if it has 3 or less transactions including alternative courses. You should be able to realise the use case with less than 5 analysis objects.	5
AVERAGE	A use case is average if it has 3 to 7 transactions including alternative courses. You should be able to realise the use case with 5 to 10 analysis objects.	10
COMPLEX	A use case is complex if it has more than 7 transactions including alternative courses. The use case should at least need 10 analysis objects to be realised.	15

Nilai UUCP dihitung dengan menjumlahkan hasil dari Tabel 10.1 dan Tabel 10.2 dengan rumus berikut:

$$UUCP = \sum_{i=1}^{6} n_i * W_i \quad (1)$$

dimana n_i adalah jumlah *item* dari berbagai *i*.

TCF - Technical Complexity Factor

TCF diperoleh dengan memberikan penilaian terhadap Tabel 10.3 dengan skala 0, 1, 2, 3, 4, dan 5 pada masing-masing itemnya. Nilai TCF dihitung dengan rumus berikut:

$$TCF = C_1 + C_2 \sum_{i=1}^{13} F_i * W_i \quad (2)$$

Dimana nilai C_1 = 0.6 dan nilai C_2 = 0.01. Nilai C_1 merupakan sebuah konstanta dan bobot yang diusulkan oleh Albrecht pada tahun 1979 tetapi C_1 diturunkan dari 0.65 menjadi 0.6 agar sesuai dengan jumlah faktor (Karner 1993). F_i adalah sebuah faktor yang dinilai pada skala 0, 1, 2, 3, 4 dan 5. 0 berarti tidak relevan dan 5 berarti ini sangat penting. Jika faktor tidak penting atau tidak relevan maka akan memiliki nilai 3. Jika semua faktor mempunyai nilai 3 maka nilai TCF akan setara dengan 1.

Tabel 10.3 Faktor yang Berkontribusi terhadap Kompleksitas

Fi	Factors Contributing to Complexity	Wi
F1	Distributed systems	2
F2	Application performance objectives, in either response or throughput	1
F3	End user efficiency (on-line)	1
F4	Complex internal processing	1
F5	Reusability, the code must be able to reuse in other applications	1
F6	Installation ease	0.5
F7	Operational ease, usability	0.5
F8	Portability	2
F9	Changeability	1
F10	Concurrency	1
F11	Special security features.	1
F12	Provide direct access for third parties	1
F13	Special user training facilities	1

EF - Environmental Factor

EF membantu untuk mengestimasi seberapa efisien proyek tersebut. Faktor ini adalah bentuk yang sama sebagai faktor teknis. EF dihitung berdasarkan Tabel 10.4 yang menjelaskan bobot nilai dari masing-masing faktor yang berkontribusi terhadap efisiensi. Nilai EF dihitung dengan rumus berikut:

$$EF = C_1 + C_2 \sum_{i=1}^{8} F_i * W_i \quad (3)$$

Dimana nilai C_1 = 1.4 dan nilai C_2 = -0.03. Fi adalah sebuah faktor yang dinilai pada skala 0, 1, 2, 3, 4 dan 5. 0 berarti tidak relevan dan 5 berarti ini sangat penting. Jika faktor tidak penting atau tidak relevan maka akan memiliki nilai 3. Jika semua faktor mempunyai nilai 3 maka nilai EF akan setara dengan 1.

Tabel 10.4 Faktor yang Berkontribusi terhadap Efisiensi

Fi	Factors Contributing to Efficiency	Wi
F1	Familiar with Objectory	1.5
F2	Part time workers	-1
F3	Analyst capability	0.5
F4	Application experience	0.5
F5	Object oriented experience	1
F6	Motivation	1
F7	Difficult programming language	-1
F8	Stable requirements	2

Hasil dan Analisis

Use Case Point (UCP) dihitung dengan rumus berikut:

$$UCP = UUCP * TCF * EF \quad (4)$$

Berdasarkan UCP dihitung dengan melihat statistik dari proyek-proyek sebelumnya untuk melihat berapa banyak sumber daya yang dibutuhkan per UCP. Setelah itu dikalikan jumlah UCP dengan *Mean Resources needed per UCP* (MR). Nilai ini juga dilihat dengan menggunakan *Standard Deviation of the MR* (SDMR) untuk melihat seberapa baik estimasinya.

10.3 ALAT DAN BAHAN

Alat dan bahan yang digunakan dalam praktikum ini yaitu:

1. Komputer
2. Software pendukung: Ms Excel, Astah

10.4 LANGKAH PRAKTIKUM

1. Persiapkan Use Case Diagram Anda
2. Lakukan perhitungan Use Case Point dengan alat bantu yang disediakan

10.5 TUGAS

1. Pahami setiap komponen yang digunakan dalam estimasi biaya perangkat lunak dengan Use Case Points. Asisten praktikum akan memberikan perintah untuk menjelaskan beberapa istilah dan definisinya, serta bagaimana contoh estimasinya untuk studi kasus lain.
2. Lengkapi laporan estimasi *software costing* Use Case Points untuk studi kasus tim Anda.

CONTOH
LEMBAR JAWABAN PRE-TEST / POST-TEST / EVALUASI PRAKTIKUM 10: SOFTWARE COSTING ESTIMATION

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

DAFTAR PUSTAKA

http://www.uml-diagrams.org/

Chemuturi, M. (2009). *Software estimation best practices, tools & techniques: A complete guide for software project estimators*. J. Ross Publishing.

https://www.slideshare.net/labsirkel/bab-iii-class-diagram, diakses 4 Mei 2018.

Dennis, Alan, Barbara Haley Wixom, and David Tegarden. Systems analysis and design: An object-oriented approach with UML. John Wiley & Sons, 2015.

Dennis, Alan, Barbara Haley Wixom, and Roberta M. Roth. Systems analysis and design. John Wiley & Sons, 2014.

http://www.sparxsystems.com.au/resources/uml2_tutorial/uml2_classdiagram.html

https://sourcemaking.com/uml/modeling-it-systems/structural-view/class-diagram

Jesse James Garrett's *The Elements of User Experience: User-Centered Design for the Web and Beyond (2nd Edition)*

"Apa Itu Activity Diagram" https://www.dumetschool.com/blog/Apa-Itu-Activity-Diagram diakses 13 April 2017

"User *Interface* Elements" https://www.usability.gov/how-to-and-tools/methods/user-*interface*-elements.html diakses 13 April 2017

"User *Interface* Elements Free" http://clipart.me/24018/70-user-*interface*-elements-free-psd diakses pada 13 April 2017

E. K. Elberkawi and M. M. Elammari, "Producing Graphical User *Interface* from Activity Diagrams," *Int. Sci. Index, Comput. Inf. Eng.*, vol. 9, no. No:3, pp. 667–672, 2015.

Soren Lauesen. 2005. *User Interface Design: A Software Engineering Perspective*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.


LABORATORIUM
TEKNIK INFORMATIKA
UNIVERSITAS AHMAD DAHLAN


UNIVERSITAS AHMAD DAHLAN