**LAPORAN DOCUMENTASI   
APLIKASI RESTOURAN**

**SMK SANGKURIANG 1 CIMAHI**

**

*DISUSUN OLEH :*

REGAN TSANY RIZQULLAH NIS : 102101701

M SYAHDAZAN A NIS : 102101688

HARJON S NIS : 102101682

SANDI SETIAWAN NIS : 102101704

1. **Kata Pengantar**

Assalamualaikum Wr.Wb,

Dalam rangka memenuhi syarat Ujian Komputer jurusan Rekayasa Perangkat Lunak, kami dengan hormat ingin menyampaikan terima kasih kepada Allah SWT atas berkah-Nya yang telah memandu kami dalam merancang aplikasi restouran ini.

Terima kasih yang sebesar-besarnya kami sampaikan kepada pembimbing kami yang telah memberikan arahan dan dukungan selama kurang lebih 3 tahun. Tanpa bimbingan beliau, proyek ini tidak akan terwujud.

Kami juga berterima kasih kepada sekolah kami atas dukungan dan fasilitas yang diberikan, yang telah membantu kami menyelesaikan laporan ini.

Semoga aplikasi restouran ini memberikan manfaat bagi industri kafe dan memenuhi persyaratan ujian kompetensi kami.

Terima kasih atas kesempatan ini.

1. **Latar Belakang**

Laporan Dokumentasi merupakan komponen penting yang diwajibkan bagi siswa-siswi yang mengikuti ujian komputer (UJIKOM), karena salah satu persyaratannya adalah penyusunan laporan dokumentasi agar dapat memenuhi syarat untuk mengikuti ujian tersebut

1. **Maksud dan Tujuan**

Maksud dan Tujuan diadakannya ujian ini adalah untuk menguji kemampuan siswa atau siswi dalam bidang perangkat lunak ini. Tujuan adanya Ujian Komputer ini antara lain:

1. Memenuhi syarat agar dapat lulus dari sekolah
2. Membekali pengalaman di dalam dunia IT
3. Meningkatkan serta memperluas pengalaman dalam mengatasi masalah komputer

Melatih kemampuan siswa agar terbiasa pada dunia IT yang sesungguhnya

1. **Gambaran Umum Aplikasi**

Aplikasi restouran ini adalah solusi teknologi yang dirancang untuk membantu pengelolaan efisien restouran. Aplikasi ini memungkinkan pemilik restouran untuk mengelola inventaris, mengoptimalkan proses pesanan, dan memberikan layanan pelanggan yang lebih baik. Dengan fitur-fitur manajemen inventaris yang otomatis, pemilik dapat dengan mudah mengelola stok bahan baku dan menghindari kekurangan persediaan. Selain itu, aplikasi ini menyediakan laporan yang berguna untuk pemantauan kinerja bisnis secara real-time. Dengan aplikasi ini, restouran dapat meningkatkan efisiensi operasional dan memaksimalkan pengalaman pelanggan.

**TOOLS YANG DIGUNAKAN**

|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **Nama**  **alat/komponen/bahan** | **Spesifikasi minimal** | **Jumlah** |
| **1** | **2** | **3** | **4** |
|  | **alat** |  |  |
| 1 | Komputer(PC/Laptop)yang  Difungsikan sebagai server | * Intel(R) Core(TM) i7-5600U CPU @ 2.60GHz (4 CPUs), ~2.6GHz * RAM : 8GB(4GB untuk yang memilih Platform mobile) * Keyboard | 1 |
| 2 | Mobile device | Mendukung android | 1 |
| 4 | Jaringan komputer | * Menghubungkan komputer server – client dan/atau mobile device |  |
|  | **Komponen** |  |  |
| 1 | Sistem Operasi | * OS(Windows 10 Pro 64-bit) | 1 |
| 2 | Aplikasi tools | * Visual Studio Code | 1 |
| 3 | Bahasa pemograman | * PHP/Javascript/CSS/HTML | 1 |
| 4 | Aplikasi server basis data | * phpMyAdmin | 1 |
| 5 | Aplikasi pendukung | * Xampp * 000webhosting * Draw.io * Balsamic | 1 |
|  | **Bahan** |  |  |
| 1 | Data – data referensi   * Data menu * Data pesanan * Data pengguna | Rekaan sendiri | Sesuai kebutuhan |
| 2 | ATK |  | Sesuai kebutuhan |

**Analisis Kebutuhan**

Mengumpulkan apa yang dibutuhkan secara lengkap untuk kemudian dianalisis guna mendefinisikan kebutuhan yang harus dipenuhi oleh program yang akan dibangun. Fase ini harus dikerjakan secara lengkap untuk bias menghasilkan desain yang lengkap. Analisi kebutuhan terbagi 2 yaitu :

**Kebutuhan Fungsional**

Kebutuhan Fungsional merupakan jenis kebutuhan yang berisi proses – proses apa saja yang nantinya dilakukan oleh aplikasi. Kebutuhan fungsional juga berisi informasi informasi apa saja yang harus ada dan dihasilkan oleh aplikasi.

**Jenis Materi Rekayasa Perangkat Lunak**

1. **Perancangan Use Case**

*Use case* diagram adalah satu dari berbagai jenis diagram UML (*Unified Modelling Language*) yang menggambarkan hubungan interaksi antara sistem dan aktor. *Use Case* dapat mendeskripsikan tipe interaksi antara si pengguna sistem dengan sistemnya.  Tentunya, use case diagram merupakan sesuatu yang mudah dipelajari. Langkah awal untuk melakukan pemodelan, tentu perlunya suatu diagram yang mampu menjabarkan aksi aktor dengan aksi sistem itu sendiri, seperti yang terdapat pada use case diagram.

Adapun, fungsi dari use case diagram sebagai berikut:

* Berguna memperlihatkan proses aktivitas secara urut dalam sistem.
* Mampu menggambarkan proses bisnis, bahkan menampilkan urutan aktivitas pada sebuah proses.
* Sebagai *bridge*atau jembatan antara pembuat dengan konsumen untuk mendeskripsikan sebuah sistem.

Manfaat dari use case di antaranya:

* Menggunakannya sebagai kebutuhan verifikasi.
* Menjadi gambaran *interface* dari sebuah sistem karena setiap sistem yang dibangun haruslah memiliki interface.
* Mengidentifikasi siapa saja orang yang dapat berinteraksi dengan sistem, serta apa yang dapat dilakukan oleh sistem.
* Memberikan kepastian mengenai kebutuhan  sistem, sehingga tidak membingungkan.
* Memudahkan proses komunikasi antara domain *expert* dan end *user*.

## Komponen Use Case

### Sistem

Sebuah sistem  digambarkan ke dalam bentuk persegi. Fungsinya untuk membatasi use case dengan interaksi dari luar sistem.Sistem pada umumnya diberikan label yang sesuai. Namun, umumnya sistem ini tidaklah diberi gamba karena kita tidak terlalu memberikan makna pada sebuah diagram.

### Actor

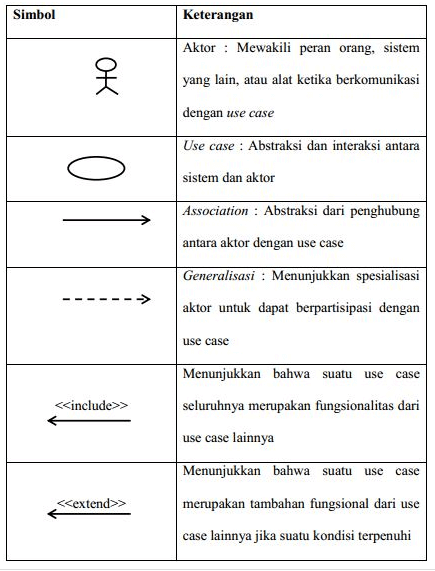
Banyak yang berspekulasi bahwa actor adalah bagian dari diagram. Padahal apabila kita mencari informasi lebih dalam mengenai soal ini, ternyata actor bukanlah bagian dari diagram.

Peran actor sangat penting, tentunya menciptakan use case jadi lebih mudah. Fungsi Actor menjelaskan siapa yang berinteraksi dengan sistem. Actor akan memberikan informasi kepada sistem, serta menerima informasi dari sistem. Keduanya bisa terjadi secara bersamaan. Aktor tidak memberikan kontrol terhadap sistem, namun hanya memberikan gambaran mengenai hubungannya dengan sistem.

Ternyata, inilah beberapa alasan mengapa actor dapat berhubungan dengan sistem lain:

* Jika terdapat relasi sistem lain dengan sistem yang sedang dibuat.
* Terdapat eksternal resource yang digunakan oleh sistem.
* Adanya kepentingan terhadap sistem, yaitu alur informasi baik penerima maupun arus sistem saling memiliki kepentingan.
* Terdapat seseorang atau pihak lain yang akan mengelola sistem.

**Simbol-Simbol Use Cas**



1. **Perancangan Activity Diagram**

Activity diagram, dalam bahasa Indonesia diagram aktivitas, yaitu diagram yang dapat memodelkan proses-proses yang terjadi pada sebuah sistem. Runtutan proses dari suatu sistem digambarkan secara vertikal. Activity diagram merupakan pengembangan dari *Use Case* yang memiliki alur aktivitas.

Alur atau aktivitas berupa bisa berupa runtutan menu-menu atau proses bisnis yang terdapat di dalam sistem tersebut. Dalam buku **Rekayasa Perangkat Lunak** karangan Rosa A.S mengatakan, “Diagram aktivitas tidak menjelaskan kelakuan aktor. Dapat diartikan bahwa dalam pembuatan activity diagram hanya dapat dipakai untuk menggambarkan alur kerja atau aktivitas sistem saja.”

Activity diagram mesti digunakan sejajar (*horizontal*) dengan teknik pemodelan lainnya, seperti diagram *Use Case* dan diagram *State*. Kamu bisa menggunakan *activity* diagram agar dapat memodelkan alur kerja sistem dengan baik. Activity diagram berfungsi juga untuk menganalisis diagram *use case*dengan cara mendeskripsikan aktor, tindakan yang perlu dilakukan, dan kapan harus terjadi.

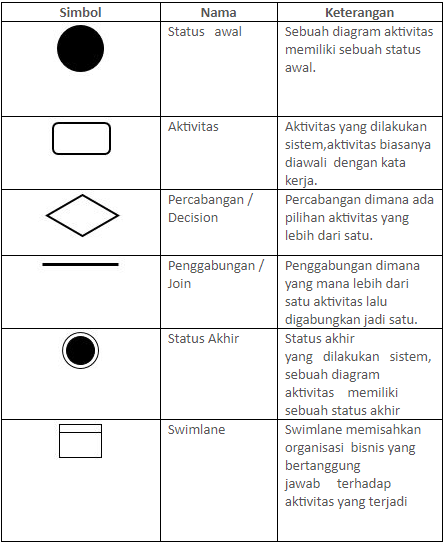
Diagram ini menggambarkan sebuah algoritma dan pemodelan sekuensial yang kompleks dengan proses paralel. Selanjutnya mari kita bahas mengenai tujuan dari pada *activity* diagram itu sendiri.

### Tujuan Activity Diagram

Berikut beberapa tujuan dari activity diagram:

1. Menjelaskan urutan aktivitas dalam suatu proses.
2. Di dalam dunia bisnis biasanya digunakan untuk modeling (memperlihatkan urutan proses bisnis).
3. Mudah dalam memahami proses yang ada dalam sistem secara keseluruhan.
4. Merupakan metode perancangan yang terstruktur, mirip dengan *Flowchart*maupun *Data Flow Diagram* (DFD).
5. Mengetahui aktivitas aktor/pengguna berdasarkan *use case/*diagram yang dibuat sebelumnya.

**Komponen – komponen activity diagram**



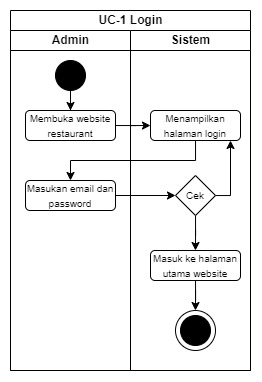
### Berikut penjelasan lengkapnya mengenai komponen-komponen pada *activity*diagram di atas :

* ***Start Point* atau*Initial State*(Titik Mulai/Status Awal)***Start Point* adalah lingkaran hitam kecil. Biasanya digunakan untuk menandakan status awal, tindakan awal, atau titik awal aktivitas untuk setiap activity diagram.
* ***Activity* (Aktivitas)**Activity merupakan aktivitas yang dilakukan atau sedang terjadi dalam sistem. Biasanyadiawali dengan “kata kerja” dari aktivitas yang dilakukan.
* ***Decision*atau Percabangan**Percabangan atau *decision* merupakan suatu titik atau *point*yang mengindikasikan suatu kondisi di mana adanya kemungkinan dalam perbedaan transisi. Hal tersebut diperlukan ketika sistem yang dimiliki memiliki beberapa kemungkinan atau jalan alternatif.
* ***Synchronization****Synchronization*dibagi menjadi 2 bagian, yaitu *fork*dan *join*.
  + *Fork*(percabangan) digunakan untuk memecah *behaviour* (tingkah laku) menjadi *activity*atau *action* (aksi) secara paralel.
  + *Join*(penggabungan) digunakan untuk menghubungkan kembali *activity* dengan *action*secara paralel.
* ***Merge***Menggabungkan *flow*yang sudah dipecah menjadi beberapa bagian oleh suatu *flow*.
* ***Swimlanes***Memecah activity diagram menjadi kolom dan baris untuk membagi tanggung jawab objek-objek yang melakukan suatu aktivitas.
* ***Transition***Digunakan untuk menunjukan aktivitas selanjutnya dan sebelumnya.
* **Notasi akhir (*end state*)**Notasi akhir digunakan untuk menandakan proses tersebut berakhir. Pada UML, notasi akhir dapat  digambarkan dengan simbol sebuah *bull’s eye* (mata sapi).

Terkadang menggunakan percabangan (*decision*) dengan *fork*adalah hal yang keliru. Sebab *Decision*digunakan untuk memecah aktivitas yang bersifat kondisional. Contohnya pilihan **Ya** atau **Tidak**, jika opsi **Ya,**maka terjadi aksi baru dan jika **Tidak,** maka menolak aksi baru. Sedangkan *fork*digunakan untuk memecah behaviour menjadi aktivitas yang paralel, contohnya seperti pengguna dapat memilih, menambah, mengubah, serta bisa juga menghapus.

Contoh dari activity diagram coffe shop

**UC – 01 Melakukan Proses Login**

****