BAB 2

LANDASAN TEORI

Untuk mendukung pembuatan skripsi ini, maka perlu dikemukakan hal-hal atau teori-teori yang berkaitan dengan permasalahan dan ruang lingkup pembahasan sebagai landasan dalam pembuatan skripsi ini.

2.1. E-Learning

E-learning adalah sebuah instruksi atau media yang dikirim ke perangkat digital seperti komputer atau perangkat *mobile* yang digunakan untuk menunjang pembelajaran. Ciri-ciri dari *e-learning* adalah seperti (Clark & Mayer, 2011):

- Pelajaran disimpan dalam CD-ROM, memori internal atau eksternal, atau di server pada internet atau intranet
- Berisi konten yang relevan dengan pelajaran yang digunakan
- Menggunakan media seperti tulisan dan gambar untuk mengirimkan konten
- Menggunakan metode instruksi seperti contoh, *practice*, atau *feedback*.
- Bisa membutuhkan pengajar (*synchronous e-learning*) atau *self-paced* (*asynchrounoue e-learning*).

2.2. Rapid Application Development (RAD)

Menurut Richard R. Pressman dan Bruce R. Maxim (2015) *Rapid Application Development* (RAD) adalah proses model pengembangan yang secara terus

menerus. RAD menekankan kepada "*high speed delivery*" yang dikembangakan dari *waterfall method* dimana pendekatannya menggunakan pendekatan secara komponen. RAD bergantung pada spesifikasi dan kebutuhan yang jelas, bila dua hal itu telah terpenuhi maka proses pengembangan akan cepat selesai. Dalam RAD, ada empat tahap yang dilakukan:

- **Requirement Planning Phase**

Pengguna, manajer, dan team *developer* berdiskusi untuk menemukan permasalahan, batasan, *scope* proyek, dan *system requirements*. Bila disepakati, proses pengembangan dapat maju ke tahap selanjutnya

- **User Design Phase**

Model sistem dikembangkan pada tahap ini dengan mendengarkan dari analis sistem dan pengguna.

- **Construction Phase**

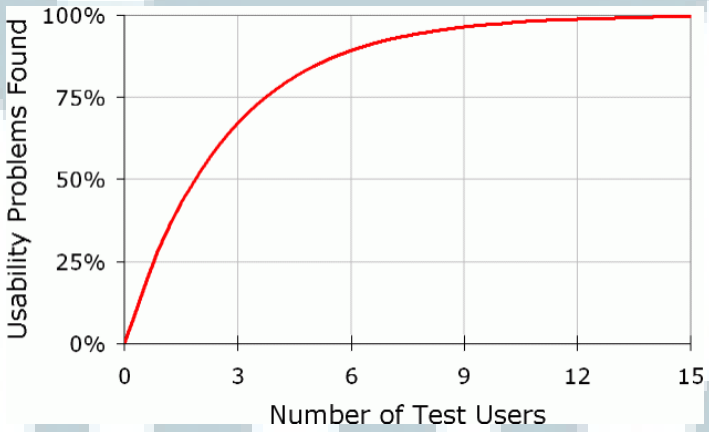
Sistem mulai dikembangkan pada tahap ini. *User* dan analis dapat tepat memberikan masukan dan permintaan perubahan seiring dengan perkembangan sistem.

- **Cutover Phase**

Pada tahap ini dilakukan konversi data, *testing*, pengalihan sistem lama ke sistem baru dan pelatihan pada pengguna sistem.

2.3. Usability Testing

Menurut Jeff Rubin dan Dana Chisnel (2008) *usability testing* adalah proses dimana sebuah kegiatan *testing* dilakukan dengan cara merekrut orang sebagai partisipan yang berfungsi menjadi wakil dari sebuah target *user* sistem yang akan dipakai. *Usability testing* bisa berbentuk berbagai macam mulai dari *test* yang memiliki jumlah sampel yang banyak dengan desain *test* yang kompleks sampai dengan *test* informal dengan hanya satu partisipan. Menurut Jakob Nielsen dan Thomas K. Launder (1993) *usability testing* cukup memerlukan 5 partisipan untuk mendapat hasil *testing* yang optimal.



Gambar 2.1. Komparasi Hasil Test

Berdasarkan pada Gambar 2.1, permasalahan pada *usability testing* dapat ditemukan sampai 33% persen dengan hanya satu partisipan, dua partisipan dapat mencapai 50%, dan lebih dari 5 partisipan tidak memberikan perubahan yang signifikan terhadap hasil *usability testing.*

2.4. PHP

PHP adalah akronim untuk *PHP: Hypertext Processor* dan merupakan salah satu bahasa pemrograman yang populer untuk membuat *website* (MacIntyre, 2010). PHP merupakan perangkat lunak *open-source* dan pada tahun 1996 ditambahkan integrasi *database*. Fungsi dari PHP dipisah menjadi tiga bagian (Tatroe, MacIntyre, & Lerdorf, 2013):

- ***Server-side Scripting***

PHP dirancang untuk membuat konten *web* secara dinamis. Untuk membuat HTML dengan PHP, dibutuhkan *parser* PHP dan sebuah *server web* yang akan mengirimkan informasi kode dokumen. PHP juga digunakan untuk membuat dokumen XML, grafik, animasi Flash, file PDF, dan lain-lain.

- **Command Line Scripting**

PHP bisa menjalankan *script* dari *command line* seperti PERL, awek, atau *shell* Unix. Penggunaan *command line* dengan PHP biasanya digunakan untuk sistem administrasi atau membuat *cron job*.

- **Aplikasi *Client-side GUI***

PHP juga dapat membuat aplikasi yang mempunyai *Graphical User Interface* dengan menggunakan PHP-GTK.

2.5. Laravel Framework

Laravel adalah *framework open-source* berbasis bahasa pemrograman PHP yang menggunakan model MVC (*Model-View-Controller*). Kelebihan Laravel adalah *syntactic sugar*, cara-cara berbeda untuk mengakses *database*, dan sistem *packaging* modular yang mempunyai *dependency manager*. Laravel memiliki pendekatan untuk membuat penamaan dan *syntax* yang mudah dimengerti dan elegan. Laravel menjadi salah satu *framework* yang paling terkenal dan paling banyak dipakai menggantikan *framework* lain seperti Yii, CakePHP, dan CodeIgniter. CodeIgniter yang dulu merupakan *framework* terbanyak yang dipakai berkurang peminatnya karena lambatnya proses implementasi fitur baru PHP seperti *namespace* dan *packaging* (Stauffer, 2017). Berikut adalah beberapa kelebihan Laravel sebagai *framework* PHP (Koster, 2016):

- **Sistem Konfigurasi**

Laravel didesain untuk membuat proses *development* mudah dan cepat. Laravel memberikan API yang simpel untuk digunakan. Sistem konfigurasinya juga dapat *support* kepada aplikasi *environment* yang berbeda-beda.

- **Sistem Error dan Logging**

Sistem Laravel memudahkan untuk melakukan *debugging* dan *logging* kepada satu atau lebih *file*.

- **Routing**

Sistem *Routing* (pemberian nama pada URL *website*) pada laravel memudahkan untuk melakukan fungsi basis data yang kompleks. URL juga dapat dibuat menjadi sebuah grup tertentu untuk memudahkan akses ke API.

- **Template Engine**

Laravel disertai dengan *template engine Blade* sehingga pembuatan tampilan HTML secara dinamis dapat dimudahkan dengan fungsi *layout, shortcode*, dan fungsi-fungsi lainnya.

- **Database Management**

Laravel memiliki sistem *Object-Relational Mapping* (ORM) yang mensimplifikasi pembuatan dan pengeditan *database*.

2.6. Vagrant

Vagrant adalah *tool* untuk membuat dan mengatur *virtual machine environment*. Vagrant mempunyai kelebihan untuk melakukan penginstallan automisasi sehingga mengurangi waktu untuk mempersiapkan *development environment* (VagrantUp, 2017).

Vagrant pertama kali dibuat sebagai proyek *open-source* yang digunakan untuk mensimplifikasi pembuatan *Virtual Machine* (VM). Vagrant dapat membuat *virtual machine* yang memiliki spesifikasi dengan kode tanpa membuat *binary* sehingga pengembang perangkat lunak dapat melakukan penarikan *source* dan langsung melakukan *development*. Aplikasi-aplikasi yang akan digunakan pada VM dapat didefinisikan di *file VagrantFile* (Thompson, 2015).

2.7. MySQL

Menurut Robert Nixon (2015) MySQL adalah sistem basis data yang digunakan untuk menyimpan data secara terorganisir dan data tersebut bisa dicari

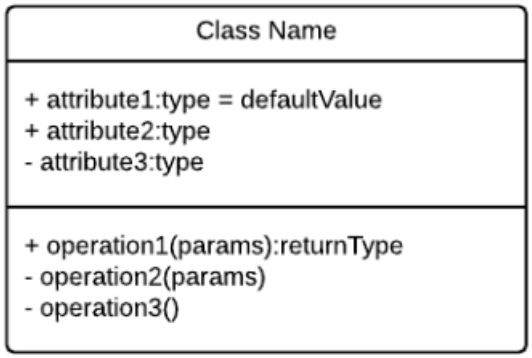
dengan cepat. SQL pada MySQL adalah singkatan untuk *Structured Query Language*. Bahasa ini didasarkan dari bahsa inggris dan juga digunakan oleh Oracle dan Microsoft SQL untuk melakukan *request* pada database dengan media *command* seperti “SELECT title FROM publications WHERE author = 'Charles Dickens';”.

2.8. Object-Oriented Analsis & Design (OOAD)

Object-Oriented Analsis & Design (OOAD) cara pembuatan desain sistem yang menggunakan *object* dan *class* sebagai konsep utamanya dengan empat tahap analisis yaitu: Membuat model untuk konteks aplikasinya, Menekankan kepada arsitektur, Menggunakan ulang *pattern*, dan menyesuaikan metode untuk projek tertentu. (Mathiassen, Madsen, Nielsen, & Strage, 2000). Dalam OOAD ada beberapa diagram yang digunakan, di antaranya adalah:

2.8.1. Class Diagram

Digunakan untuk mendeskripsikan struktur dari system dengan membuat model untuk *class*, atribut, operasi, dan relasi antar objek-objeknya.



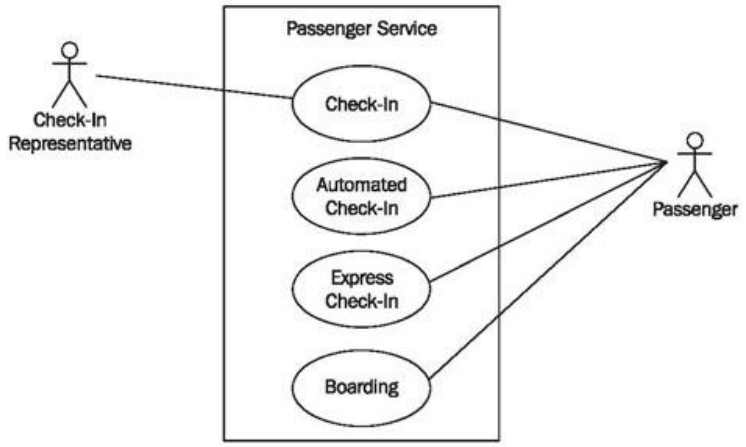
Gambar 2.2. Contoh Class Diagram

Class diagram dibagi menjadi tiga bagian (Mathiassen, Madsen, Nielsen, & Strage, 2000):

- *Upper Section:* Berisi nama dari *class tersebut.*
- *Middle Section:* Berisi atribut yang ada pada *class*.
- *Bottom Section:* Berisi operasi atau metode yang dapat dilakukan pada *class* yang ada pada diagram.

2.8.2. *Use Case Diagram*

Use Case merupakan sebuah abstraksi dari sebuah interaksi pada sebuah sistem. *Use Case Diagram* digunakan untuk menjelaskan apa saja aktor-aktor dalam sistem dan operasi apa saja yang bisa dilakukan.



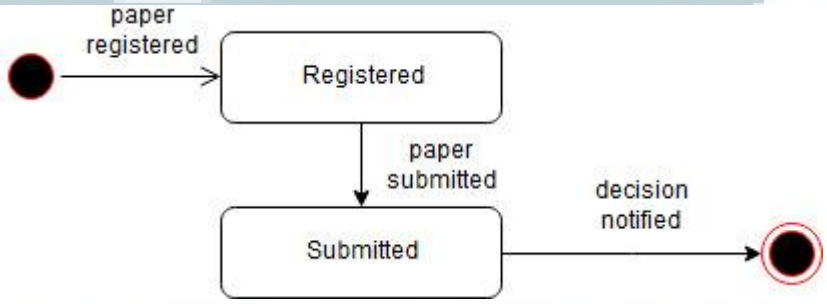
Gambar 2.3. Contoh Use Case Diagram

Komponen yang ada pada *use case diagram* adalah (Mathiassen, Madsen, Nielsen, & Strage, 2000):

- Aktor, direpresentasikan dengan *stick figure* atau persegi: Pengguna yang akan menggunakan sistem. Aktor bisa merupakan orang, organisasi, atau sistem eksternal yang berinteraksi dengan sistem yang dirancang.
- *Use Case*, digambarkan dengan bentuk oval: Proses yang terjadi di dalam sistem. Proses digambarkan alurnya mulai dari atas sampai ke bawah.
- *Line* atau garis: Menggambarkan partisipasi yang terjadi antara pengguna dengan *use case* yang berhubungan.

2.8.3. *Statechart Diagram*

Statechart Diagram adalah diagram yang menggambarkan keadaan(*state*) dari *classes* yang ada.



Gambar 2.4. Contoh Statechart Diagram

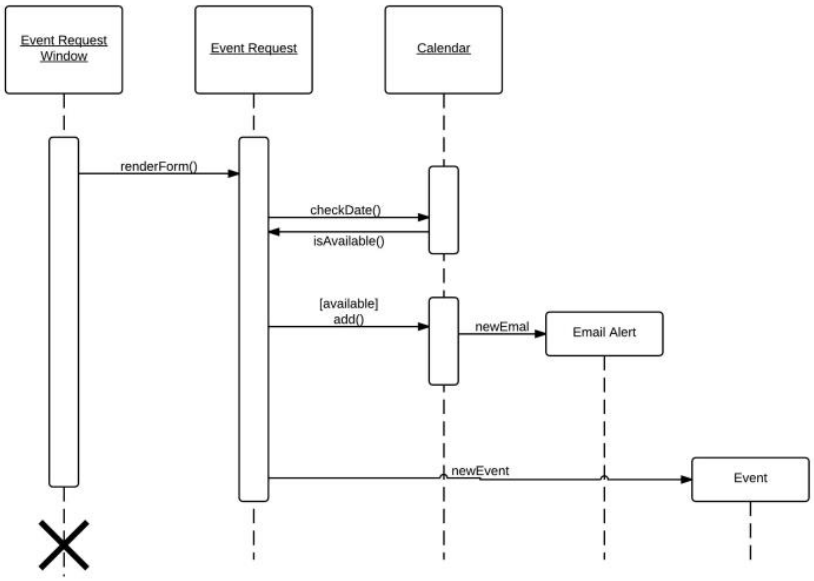
Simbol yang digambarkan pada *statechart diagram* adalah (Mathiassen, Madsen, Nielsen, & Strage, 2000).

- Kotak: Menggambarkan keadaan/*state* dari sebuah *class*.
- Lingkaran Hitam: Melambangkan awal *state* dari sebuah *class*.
- Arah Panah: Melambangkan perubahan sebuah *class* dari *state* ke *state*.
- Arah Panah Melingkar: Melambangkan *fungsi* yang dapat dilakukan saat *class* pada *state* tersebut.

- Lingkaran Hitam dengan garis mengelilingi: Melambangkan akhir dari sebuah *state*.

2.8.4. *Sequence Diagram*

Sequence Diagram adalah diagram yang digunakan untuk mendeskripsikan tahapan yang terjadi pada *event-event.*



Gambar 2.5. Contoh Sequence Diagram

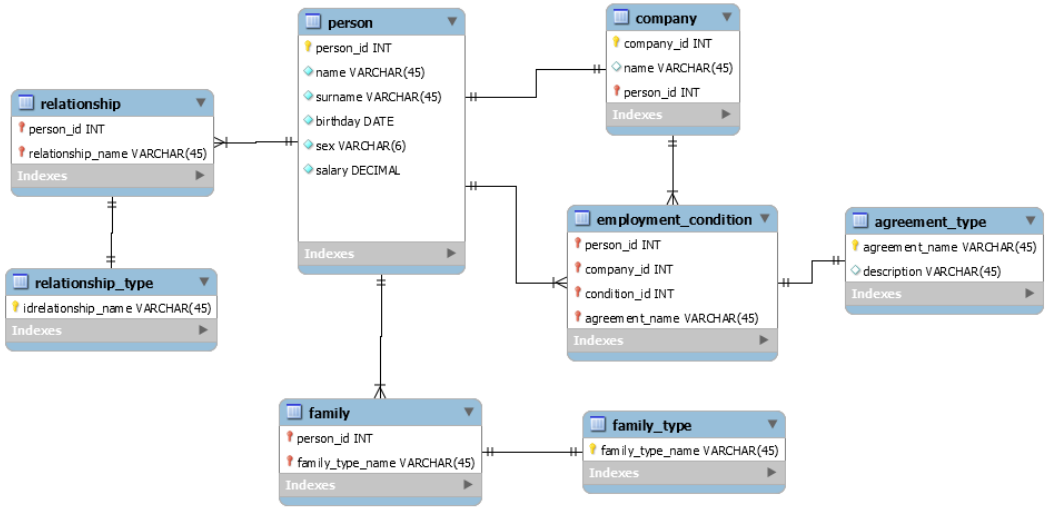
Simbol yang digambarkan pada *sequence diagram* adalah (Mathiassen, Madsen, Nielsen, & Strage, 2000):

- Persegi: Mendeskripsikan *event* atau *object* yang terdapat pada sistem.
- Persegi Panjang: Mendeskripsikan waktu yang dibutuhkan untuk menyelesaikan *event.*
- Garis putus-putus: Mendeskripsikan waktu yang berlangsung selama proses *event* terjadi.

- Arah Panah: Instruksi kepada *event* lain.
- Arah Panah putus-putus: Respon dari sebuah objek ke objek yang lain.
- Lambang "X": Mendeskripsikan berakhirnya sebuah *event*.

2.9. Entity Relationship Diagram (ERD)

Entity Relationship Diagram adalah alat untuk memodelkan data semantik yang digunakan untuk memberikan gambaran abstrak untuk mendeskripsikan data. ERD juga dapat digunakan untuk mendokumentasikan sebuah basis data dengan cara men-*reverse-engineering* basis data tersebut. (Sikha & Earp, 2003)



Gambar 2.6 Contoh Entity Relationship Diagram

2.10. System Usability Scale (SUS)

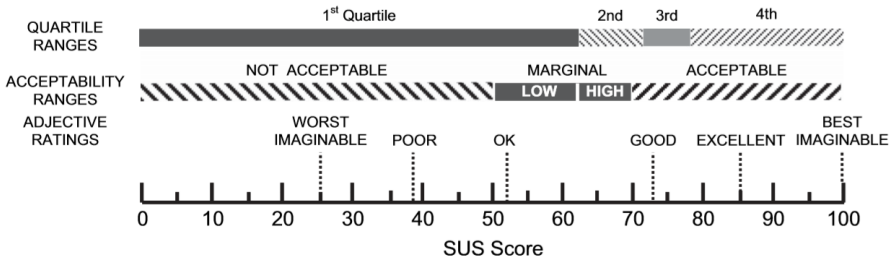
System Usability Scale (SUS) merupakan metode *usability testing* untuk mengukur *usability* dalam sebuah sistem. SUS merupakan sebuah kuesioner berisi

sepuluh pernyataan dengan 5 respon dari “Sangat Tidak Setuju” sampai “Sangat Setuju”. Pernyataan adalah sebagai berikut:

1. Saya akan sering menggunakan sistem ini.
2. Saya merasa sistem ini kompleks.
3. Saya merasa sistem ini mudah untuk digunakan.
4. Saya merasa butuh orang teknis untuk menggunakan sistem ini.
5. Saya merasa fitur dalam sistem ini terintergrasi dengan bagus.
6. Saya merasa terlalu banyak ketidakkonsistenan dalam sistem ini.
7. Saya merasa orang akan cepat mudah mengerti menggunakan sistem ini.
8. Saya merasa sistem ini merepotkan untuk digunakan.
9. Saya merasa nyaman menggunakan sistem ini.
10. Saya memerlukan banyak hal sebelum menggunakan sistem ini.

Untuk menghitung skor dari hasil *testing*. Jumlah nilai dari hasil *testing* dijumlahkan. Untuk pernyataan nomor 1, 3, 5, 7 kontribusi nilainya adalah posisi nilai di pernyataan dikurang satu. Sedangkan untuk pernyataan nomor 2,4,6,8 dan 10 kontribusinya adalah 5 dikurang dari posisi pernyataan. Nilai itu lalu dikalikan untuk mendapatkan hasil dari *testing* SUS. (Usability.gov, 2017)

Nilasi SUS yang cukup baik adalah diantara 52 dan 69, nilai yang baik adalah nilai SUS yang memiliki nilai diatas 70 atau 80. Nilai yang hampir sempurna adalah diatas 90, dan nilai dibawah 50 memerlukan perhatian yang lebih terhadap *usability* sistemnya dan dianggap sebagai nilai yang tidak bisa diterima. (Aaron Bangor, 2008)



Gambar 2.7 Komparasi nilai Mean SUS

Sumber: (Aaron Bangor, 2008)

2.11. Moodle

Moodle adalah *Course Management System (CMS)* berbasis *open-source* yang digunakan universitas swasta dan negeri, komunitas, bisnis, dan pengajar individu untuk menambah teknologi *web* pada proses ajar-mengajarnya. *Moodle* tidak memiliki biaya dan dapat diunduh di websitenya pada https://moodle.org. *Moodle* mempunyai dua arti, yang pertama adalah *Modular Object-Oriented Dynamic Learning Enviroment* dan juga bisa berarti proses belajar dengan meneliti sesuatu sehingga menciptakan kreativitas dan pendalaman (Cole & Foster, 2008).

Moodle diciptakan oleh Martin Dougiamas yang seorang insinyur computer dan pengajar yang membuat CMS pada universitas di Perth, Australia. Fitur-fitur utama *Moodle* antara lain (Moodle, 2018):

- *Personalized Dashboard* yang menampilkan informasi *user* secara *personalized*.

- *Tools* untuk membantu kolaborasi dan aktivitias ajar-mengajar.

- Kalender.
- *File Manager* untuk penyimpanan data.
- *Text Editor* yang simpel dan intuitif.
- Sistem notifikasi bila ada *event* yang terjadi pada sistem.
- *Progress tracking* seperti sistem cek nilai dan lain-lain.