BAB II
DASAR TEORI

BAB II
LANDASAN TEORI

2.1. Landasan Teori

2.2.1. Website

Secara terminologi, website adalah kumpulan dari halaman-halaman situs, yang biasanya terangkum dalam sebuah domain atau subdomain, yang tempatnya berada di dalam *World Wide Web* (www) di internet. Sebuah halaman web adalah dokumen yang ditulis dalam format HTML (*Hyper Text Markup Language*), yang hampir selalu bisa diakses melalui HTTP, yaitu protokol yang menyampaikan informasi dari server website untuk ditampilkan kepada para pemakai melalui web browser. Semua publikasi dari website-website tersebut dapat membentuk sebuah jaringan informasi yang sangat besar[1].

2.2.2. Fotografi

Fotografi (*photography*) berasak dari bahasa Yunani, yaitu dari kata *photo* (cahaya) dan *graphien* (menggambar). Fotografi secara umum dapat diartikan "Menggambar dengan cahaya". Prosesnya adalah foto tersebut merekam pantulan cahaya yang mengenai objek pada media yang peka dengan cahaya[2].

2.2.3. Rekrutmen

Rekrutmen adalah proses menarik dan mengidentifikasi calon karyawan yang berpotensi untuk mengisi posisi tertentu dalam organisasi. Menurut Dessler (2017), rekrutmen adalah proses menemukan atau menarik pelamar kerja yang berkualifikasi untuk mengisi posisi yang kosong dalam organisasi. Rekrutmen bertujuan untuk memastikan bahwa organisasi memiliki sumber daya manusia yang tepat, baik dalam hal jumlah maupun kualitas, untuk mencapai tujuan strategis organisasi[3].

2.2.4. Pekerja Lepas / *Freelancer*

Pekerja lepas atau biasa disebut istilahnya adalah *freelancer* adalah pekerja yang mengabdi kepada klien pada jangka waktu tertentu sesuai dengan ketentuan kontrak[3]. Dalam dunia kreatif banyak perusahaan ataupun perseorangan yang merekrut para *freelancer* dari *job* desain grafis , fotografi , hingga seni untuk melakukan pekerjaan. *Freelancer* kerapkali menjadi solusi untuk para perusahaan untuk mendapatkan *service* yang sesuai spesifikasi *job* dengan biaya yang murah.

2.2.5. Laravel 10

Laravel adalah *framework* aplikasi web berbasis PHP yang bersifat *open-source* dan dirancang untuk pengembangan aplikasi web dengan cara yang elegan dan mudah. Dikembangkan oleh Taylor Otwell, Laravel berusaha untuk membuat pengembangan web lebih mudah dengan menyediakan berbagai alat dan fitur yang terpadu[4].

1. *Routing*

   Laravel menyediakan sistem *routing* yang sederhana dan fleksibel untuk mendefinisikan rute aplikasi Anda. *Routing* ini memungkinkan Anda untuk membuat URL yang bersih dan mudah dibaca[4].

2. *Blade Templating Engine*

   Blade adalah mesin template yang ringan namun kuat yang disertakan dengan Laravel. *Blade* memungkinkan Anda untuk menggunakan sintaksis yang bersih dan sederhana untuk membuat tampilan yang dinamis[4].

3. *Eloquent* ORM

   Eloquent adalah ORM (*Object-Relational Mapping*) bawaan Laravel yang memudahkan interaksi dengan basis data. *Eloquent* memungkinkan Anda untuk bekerja dengan basis data menggunakan model PHP yang sederhana dan intuitif[4].

4. *Migration* dan *Schema Builder*

   Laravel menyediakan alat untuk migrasi dan pengelolaan skema basis data. Migration memungkinkan Anda untuk memversioning perubahan skema basis data, sementara *Schema Builder* memungkinkan Anda untuk membuat dan memodifikasi tabel menggunakan sintaksis PHP[4].

5. *Middleware*

   Middleware adalah lapisan antara HTTP *request* dan *response* yang memungkinkan Anda untuk menjalankan berbagai logika sebelum *request* diproses oleh aplikasi Anda. *Middleware* sering digunakan untuk otentikasi, *logging*, dan manipulasi *request*[4].

6. *Artisan* CLI

   Artisan adalah *command-line interface* yang disertakan dengan Laravel. *Artisan* menyediakan berbagai perintah yang membantu dalam proses pengembangan aplikasi, seperti membuat model, menjalankan *migrasi*, dan banyak lagi[4]

7. *Testing*

   Laravel memiliki dukungan bawaan untuk pengujian unit dan fungsional. Anda dapat menulis dan menjalankan tes untuk memastikan bahwa aplikasi Anda berfungsi dengan baik dan bebas dari *bug*[4].

2.2.6. Bootstrap 5

Bootstrap adalah *framework front-end open-source* yang digunakan untuk mendesain situs web dan aplikasi web. Dikembangkan oleh tim dari Twitter, Bootstrap menyediakan berbagai alat dan komponen yang memudahkan pengembangan antarmuka pengguna yang responsif dan modern. Versi terbaru, Bootstrap 5, membawa berbagai peningkatan dan fitur baru dibandingkan versi sebelumnya[5].

1. Sistem *Grid*:

   Bootstrap 5 menggunakan sistem *grid* yang fleksibel dan responsif untuk mengatur *layout* halaman. *Grid* sistem ini berbasis *Flexbox*, yang memudahkan pembuatan *layout* yang adaptif untuk berbagai ukuran layar[5].
2. Komponen:

   Bootstrap 5 menyediakan berbagai komponen UI yang siap digunakan, seperti tombol, kartu, *navbar*, *modals*, dan banyak lagi. Komponen-komponen ini dirancang untuk konsistensi dan kemudahan penggunaan[5].
3. Formulir:

   Bootstrap 5 menyederhanakan pembuatan formulir dengan berbagai elemen dan utilitas form yang dapat disesuaikan. Fitur ini mencakup validasi *form*, input grup, dan berbagai jenis input lainnya[5].
4. Utilitas CSS:

   Bootstrap 5 menyediakan berbagai utilitas CSS yang membantu dalam mengatur tampilan dan perilaku elemen, seperti *margin*, *padding*, teks, warna, dan lainnya. Utilitas ini memungkinkan pengembang untuk dengan cepat menyesuaikan desain tanpa menulis banyak CSS tambahan[5].
5. Ikon Bootstrap:

   Bootstrap 5 mendukung integrasi dengan Bootstrap *Icons*, sebuah set ikon *open-source* yang dirancang untuk bekerja dengan Bootstrap. Ikon ini dapat digunakan dalam berbagai komponen dan elemen[5].
6. JavaScript Plugins:

   Bootstrap 5 mencakup berbagai plugin JavaScript yang menambah interaktivitas pada elemen UI, seperti *modals*, *dropdowns*, *tooltips*, *popovers*, dan *carousels*. Plugin ini dirancang agar mudah digunakan dan dapat disesuaikan[5].

2.2.7. Rekayasa Perangkat Lunak

A. Metode Pengembangan Sistem

Metode yang digunakan untuk pengembangan sistem adalah model RAD. RAD merupakan model proses perangkat lunak yang menekankan pada daur pengembangan hidup yang singkat. RAD merupakan versi adaptasi cepat dari model *waterfall*, dengan menggunakan pendekatan konstruksi komponen. RAD merupakan gabungan dari bermacam-macam teknik terstruktur dengan teknik *prototyping* dan teknik pengembangan *joint application* untuk mempercepat pengembangan sistem atau aplikasi. Dari definisi konsep RAD ini, dapat dilihat bahwa pengembangan aplikasi dengan menggunakan metode RAD dapat dilakukan dalam waktu yang relatif lebih cepat[6].

Tahapan RAD terdiri dari 3 tahap yang saling tergantung satu sama lain, yaitu:

1) *Requirements planning* (Perencanaan Persyaratan)
   a) Pengguna dan analisis bertemu untuk mengindentifikasi tujuan dari suatu sistem.
   b) Berorientasi pada pemecahan masalah bisnis.
2) *Design Workshop*
   a) Fase desain menyempurnakan.
   b) Programmer dan analis membangun dan menunjukan tampilan visual desain dan alur kerja pengguna.
   c) Pengguna menanggapi *prototipe* kerja *actual*.
   d) Analsis menyempurnakan.
3) *Implementation* (Penerapan)

   Sebagai sistem yang baru dibangun, sistem baru akan diuji dan diperkenalkan kepada *stakeholder*. Proses pengujian perangkat lunak merupakan tujuan untuk program komputer berfungsi seperti yang diharapkan. Dalam pengetesannya ada beberapa teknik pengujian sistem yang dapat dilakukan, salah satunya adalah *Black-box testing*. *Blackbox testing* adalah metode pengujian perangkat lunak yang dilakukan tanpa melihat atau mengetahui bagaimana kode program dibuat atau bagaimana sistem diimplementasikan. Pengujian dilakukan dengan menguji fungsionalitas perangkat lunak dari sudut pandang pengguna, dengan menguji masukan dan keluaran yang dihasilkan oleh sistem.

B. Metode Pengujian Sistem

Pengujian sistem memiliki tujuan untuk mengetahui performa dari sistem itu sendiri. Sehingga didapatkan sebuah hasil yang jelas apakah aplikasi sudah sesuai dengan yang direncanakan atau masih ada fungsionalitas sistem yang belum berfungsi. *Black box testing* adalah pengujian perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan

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

C. *Tools* / Alat Bantu Penelitian

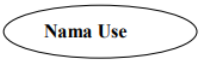
1. *Unified Modeling Language* (UML)

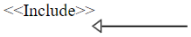
Unified Modeling Language atau UML adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemograman berorientasi objek[7]. Pemodelan sesungguhnya digunakan untuk penyederhanaan permasalahan yang kompleks menjadi lebih mudah dipahami. UML (*Unified Modeling Language*) memiliki diagram-diagram yang digunakan dalam pembuatan aplikasi berorientasi objek, diantaranya:

(a) *Use Case* Diagram

Use Case Diagram merupakan diagram yang harus dibuat pertama kali saat pemodelan perangkat lunak berorientasi objek dilakukan. Dibawah ini adalah simbol yang digunakan untuk membuat *use case* diagram[7], antara lain :

Tabel 2. 1 Simbol *Use Case* Diagram

No	Simbol	Nama	Keterangan
1		*Actor*	Merupakan pengguna dari sistem. Penamaan aktor menggunakan kata benda
2		*Use Case*	Merupakan pekerjaan yang dilakukan oleh aktor. Penamaan *use case* dengan kata kerja
3		*Association* / Asosiasi	Hubungan antara aktor dengan *use case*

No	Simbol	Nama	Keterangan
4		*Include*	Hubungan antara *use case* dengan *use case*. *Include* menyatakan bahwa sebelum pekerjaan dilakukan harus mengerjakan pekerjaan lain terlebih dahulu
5		*Extends*	Hubungan antara *use case* dengan *use case*. *Extends* menyatakan bahwa jika pekerjaan yang dilakukan tidak sesuai atau terdapat kondisi khusus, maka lakukan pekerjaan itu

2. Flowchart

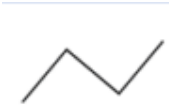
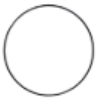
Flowchart merupakan sebuah diagram dengan simbol-simbol grafis yang menyatakan aliran algoritma atau proses yang menampilkan langkah-langkah yang disimbolkan dalam bentuk kotak, beserta urutan dengan menghubungkan masing-masing langkah tersebut menggunakan tanda panah[7]. Diagram ini bisa memberi solusi langkah demi langkah untuk penyelesaian masalah yang ada di dalam proses atau algoritma tersebut. Berikut akan dijelaskan mengenai simbol-simbol *flowchart* yang dibagi ke dalam 3 kategori, diantaranya :

(a) Simbol Arus (*Flow Direction Symbols*)

Biasanya simbol yang termasuk ke dalam kategori ini digunakan sebagai simbol penghubung. Beberapa simbol yang termasuk ke dalam kategori ini, yaitu :

Tabel 2. 2 Simbol Arus *Flowchart*

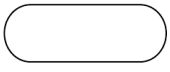
No	Simbol	Nama	Keterangan
1		*Flow Direction / Connecting Line*	Berfungsi untuk menghubungkan simbol yang satu dengan yang lainnya, menyatakan arus suatu proses

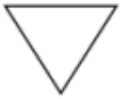
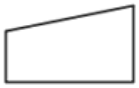
No	Simbol	Nama	Keterangan
2		*Communication Link*	Berfungsi untuk transmisi data dari satu lokasi ke lokasi lain
3		*Connector*	Digunakan untuk menyatakan sambungan dari proses yang satu ke proses berikutnya di halaman yang sama
4		*Offline Connector*	Digunakan untuk menyatakan sambungan dari proses yang satu ke proses berikutnya di halaman yang berbeda

(b) Simbol Proses (*Processing Symbols*)

Sesuai dengan namanya, simbol proses digunakan untuk menyatakan simbol yang berkaitan dengan serangkaian proses yang dilakukan. Berikut beberapa simbol yang termasuk ke dalam bagian proses, yaitu :

Tabel 2. 3 Simbol Proses

No	Simbol	Nama	Keterangan
1		*Processing*	Digunakan untuk menunjukkan pengolahan yang akan dilakukan dalam komputer
2		*Manual Operation*	Digunakan untuk menunjukkan pengolahan yang tidak dilakukan oleh komputer
3		*Decision*	Digunakan untuk memilih proses yang akan dilakukan berdasarkan kondisi tertentu
4		*Predefined Process*	Digunakan untuk mempersiapkan penyimpanan yang sedang/akan digunakan dengan memberikan harga awal
5		*Terminal*	Digunakan untuk memulai atau mengakhiri program

No	Simbol	Nama	Keterangan
6		*Offline Storage*	Berfungsi untuk menunjukkan bahwa data akan disimpan ke media tertentu
7		*Manual Input Symbol*	Digunakan untuk menginputkan data secara manual dengan keyboard

(c) Simbol I/O (*Input/Output*)

Simbol yang termasuk ke dalam bagian *input-output* berkaitan dengan masukan dan keluaran. Berikut beberapa simbol yang termasuk, yaitu :

Tabel 2. 4 Simbol *Input/Output*

No	Simbol	Nama	Keterangan
1		*Input / Output*	Digunakan untuk menyatakan *input* dan *output* tanpa melihat jenisnya
2		*Punched Card*	Digunakan untuk menyatakan masukan dan keluaran yang berasal dari *card*
3		*Disk Storage*	Digunakan untuk menyatakan masukan dan keluaran yang berasal dari *disk*
4		*Magnetic Tape*	Digunakan untuk menyatakan masukan dan keluaran yang berasal dari pita magnetis
5		*Document*	Digunakan untuk menyatakan masukan dan keluaran yang berasal dari dokumen
6		*Display*	Digunakan untuk menyatakan masukan dan keluaran melalui layar monitor

2.2.8. Pemograman Berorientasi Objek

Pemograman Berorientasi Objek atau *Object Oriented Programming* (OOP) merupakan pemograman yang berorientasikan kepada objek, di mana semua data dan fungsi dibungkus dalam *class-class* atau *object-object*[7]. Setiap *object* dapat menerima pesan,

memroses data, mengirim, menyimpan, dan memanipulasi data. Beberapa object berinteraksi dengan saling memberikan informasi satu terhadap yang lainnya.

Masing-masing *object* harus berisikan informasi mengenai dirinya sendiri dan dapat dihubungkan dengan *object* lain. Penggunaan pemograman berorientasi objek sangat banyak sekali, misalnya Java, PHP, Perl, C#, dan lainnya. Dalam konsep pemograman berorientasi objek dikenal beberapa istilah umum, yaitu [7]:

a. Kelas (*class*)

   Kelas adalah kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statik dan himpunan objek yang sama mungkin diciptakan oleh kelas tersebut. Sebuah kelas mempunyai sifaf (atribut), kelakuan (operasi/*method*), hubungan (*relationship*) dan arti. Suatu kelas dapat diturunkan ke kelas yang lain, dimana atribut dan kelas semula dapat diwariskan ke kelas yang baru.

b. Objek (*object*)

   Objek adalah abstraksi dan sesuatu yang mewakili dunia nyata seperti benda, manusia, satuan organisasi, tempat, kejadian, struktur, status, atau hal-hal lain yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi dan mempunyai operasi yang dapat diterapkan.

c. Metode (*method*)

   Method adalah fungsi atau prosedur yang dibuat oleh seorang programmer didalam suatu *class*. Pada sebuah *method* didalam sebuah kelas juga memiliki izin akses seperti atribut, yaitu *private*, *public*, dan *protected*. Sebuah kelas boleh memiliki lebih dari satu *method* dengan nama yang sama asalkan memiliki parameter masukan yang berbeda sehingga *compiler* atau *interpreter* dapat mengenali method mana yang dipanggil.

d. Atribut (*attribute*)

   Atribut dari sebuah kelas adalah variabel *global* yang dimiliki sebuah kelas. Atribut dapat berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek. Atribut dipunyai secara individual oleh sebuah objek, misalnya berat, jenis, nama, dan sebagainya. Atribut sebaiknya bersifat privat untuk menjaga enkasulapsi.

e. Enkapsulasi (*encapsulation*)

   Pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.

f. Pewarisan (*inheritance*)

Mekanisme yang memungkin suatu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dari dirinya

g. Polimorpisme (*polymorphism*)

Kemampuan suatu objek untuk digunakan dibanyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.

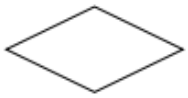
2.2.9. Database

Database adalah sekumpulan informasi yang diatur supaya mudah dicari. Dalam arti umum *database* adalah sekumpulan data yang diproses dengan bantuan komputer yang memungkinkan data dapat diakses dengan mudah dan tepat[7]. *Database* adalah kumpulan dari tabel-tabel yang berisi data-data yang saling berkaitan.

A. *Entity Relationship Diagram* (ERD)

Entity Relationship Diagram (ERD) merupakan diagram yang digambarkan dalam bentuk grafis dalam pembuatan *database* yang menghubungkan hubungan antar data yang satu dengan data yang lain[7]. Fungsi ERD adalah untuk membantu pembuatan *database* dan memberikan gambaran terhadap kerja *database* yang akan dibuat. Simbol ERD dapat dilihat pada Tabel 2.5.

Tabel 2. 5 Simbol *Entity Relationship Diagram*

No	Simbol	Nama	Keterangan
1		*Entity*	Simbol yang menyatakan himpunan entitas ini bisa berupa suatu elemen lingkungan, sumber daya, atau transaksi yang begitu pentingnya bagi perusahaan sehingga didokumentasikan dengan data
2		*Attribute*	Simbol terminal ini untuk menunjukkan nama-nama atribut yang ada pada suatu *entity*
3		*Primary Key*, *Attribute*	Simbol attribut yang digaris bawahi, berfungsi sebagai *key* (kunci) diantara nama-nama atribut yang ada pada suatu *entity*
4		*Relationship*	Simbol ini menyatakan relasi ini digunakan untuk menunjukkan hubungan yang ada antara entiti yang satu dengan entitas yang lainnya
5		*Link*	Simbol berupa garis ini digunakan sebagai

No	Simbol	Nama	Keterangan
			penghubung antara himpunan relasi dengan himpunan entitas dan himpunan entitas dengan atributnya

B. Query

1. DDL (*Data Definition Language*)

DDL (*Data Definition Language*) merupakan suatu perintah yang digunakan untuk menciptakan struktur data atau untuk membangun *database*. DDL mempunyai tugas untuk membuat untuk objek SQL dan menyimpan definisinya dalam tabel. Contoh objek tersebut yaitu *table*, *view*, dan *index*. DDL mempunyai fungsi untuk melakukan perubahan struktur tabel, mengubah dan nama tabel. Berikut perintah-perintah yang ada dalam golongan DDL:

(a) *Create*

Digunakan untuk membuat *database*, tabel, dan objek lain dalam *database*. Contoh query membuat *database* dan tabel: *CREATE DATABASE* keuangan;

(b) *Alter*

Digunakan untuk memodifikasi tabel, seperti mengubah nama tabel, *field*, dan menambah nama *field*. Contoh *query* untuk menambah *field* pada tabel di database: *ALTER TABLE users ADD* alamat *VARCHAR* (40);

(c) *Drop*

Digunakan untuk menghapus *database*, tabel, dan objek lain dalam *database*. Contoh *query* untuk menghapus *database*: *DROP DATABASE* keuangan;

2. DML (*Data Manipulation Language*)

DML (*Data Manipulation Language*) merupakan *database* yang digunakan untuk melakukan modifikasi dan pengambilan data pada suatu *database*. Pengolahan ini meliputi:

(a) *Insert*

Digunakan untuk melakukan penambahan data baru ke dalam sebuah tabel, contohnya: *INSERT INTO users* (*'id','username','password'*) *VALUES* ('1','amanda'MD5('12345'));

(b) *Select*

Digunakan untuk pengambilan data, contohnya: *SELECT * FROM users WHERE username* = 'amanda';

(c) *Update*

Digunakan untuk melakukan perubahan data, contohnya: *UPDATE users SET username* =’amanda’ *WHERE* id = ‘1’;

(d) *Delete*

Digunakan untuk melakukan penghapusan data, contohnya: *DELETE FROM users WHERE username* = ‘amanda’;