# BAB II
# KAJIAN LITERATUR

Pada bab ini peneliti akan membahas teori-teori yang peneliti butuhkan dalam pembuatan penelitian ini. Beberapa diantaranya mengenai zakat, sistem informasi, *Web Framework*, MVC, laravel, MySQL, UML, *Extreme Programming*, *black box testing*, *User Acceptance Testing* dan *Skala Likert*.

## 2.1 Tinjauan Pustaka

### 2.1.1 Zakat

Menurut Undang-undang Nomor 23 Tahun 2011 Tentang Pengelolaan Zakat, Zakat adalah harta yang wajib dikeluarkan oleh sorang muslim atau badan usaha untuk diberikan kepada yang berhak menerima sesuai dengan syariat islam [2].

Pada YBM PLN fokusnya dalam mengelola dana zakat maal yang dihimpun setiap bulannya sebesar 2,5% dari total penghasilan yang diterima oleh pegawai PLN.

### 2.1.2 Teori Sistem Informasi

Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengelolaan transaksi harian, mendukung operasi bersifat manajerial, dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan.

### 2.1.3 *Web Framework*

*Web Framework* adalah:“ kumpulan intruksi-intruksi yang dikumpulkan dalam class dan function-function dengan fungsi masing-masing untuk memudahkan developer dalam memanggilnya tanpa harus menuliskan syntax program yang sama berulang-ulang serta dapat menghemat waktu” [3].

### 2.1.4 MVC

*Framework* aplikasi web biasanya mengimplementasi pola MVC. Pola MVC memecah suatu aplikasi *Framework* menjadi 3 modul: *model*, *view*, dan *controller*. Model adalah bisnis proses dari aplikasi dan inti dari aplikasi. *View* adalah *User Interface* dari *controller*, yang mengatur tampilan untuk *User*. *Controller* mengimplementasi alur kontrol antara *view* dan *model* [4].

*Model-View-Controller* (MVC) adalah konsep yang diperkenalkan oleh penemu *Smalltalk* (Trygve Reenskaug) untuk mengenkapsulasi data bersama dengan pemrosesan (*model*), mengisolasi dari proses manipulasi (*controller*) dan tampilan (*view*) untuk dipresentasikan pada sebuah *User Interface*. *Framework* aplikasi web terdiri dari 3 komponen yang dipisah berdasarkan [5]:

1 *Model*

Digunakan untuk mengelola dan menyimpan informasi, serta memberitahu *User* ketika ada perubahan informasi. Model mengandung data dan fungsi yang berkaitan dengan pemrosesan data.

2. *View*

Bertanggung jawab untuk menampilkan informasikepada *User* melalui sebuah plat*Form* (*web*, *android*). *View* melekat pada model dan me-render isinya ke permukaan layar untuk ditampilkan ke *User*.

3. *Controller*

Bertugas menerima *input* dari *User* dan menginstruksikan *model* dan *view* untuk memproses aksi berdasarkan inputan tersebut. Sehingga *Controller* bertanggung jawab atas aksi yang dilakukan oleh *User* terhadap aplikasi.

### 2.1.5 Laravel

Laravel adalah *Framework* MVC yang memiliki fitur bundles, migrations, dan Artisan CLI. Laravel menyediakan sekumpulan tools yang lengkap dan menggabungkan arsitektur aplikasi dengan fitur-fitur terbaik dari berbagai *Framework* seperti CodeIgniter, Yii, ASP.NET MVC, Ruby on Rails, Sinatra, dan lainnya.

Laravel adalan *Framework* Open Source. Memiliki seperangkat fitur yang sangat banyak yang akan mempercepat pengembangan website. Jika sudah terbiasa dengan Core PHP dan Advance PHP, maka laravel akan lebih mempermudah pengembangan web, sehingga akan menghemat banyak waktu jika ingin mengembangkan situs dari awal. Tidak hanya itu, website yang dibangun dengan Laravel juga aman. Ini mencegah berbagai serangan yang dapat terjadi di website [6].

### 2.1.6 MySQL

MySQL adalah aplikasi sistem manajemen database SQL bersifat *Open Source* yang paling populer dan banyak digunakan. MySQL dikembangkan, didistribusikan, dan didukung oleh *Oracle Corporation*.

MySQL merupakan database *relasional* yang menyimpan data dalam tabel yang terpisah. Struktur database dikelola ke dalam *file* fisik yang dioptimalkan untuk kecepatan. MySQL menawarkan pemrograman yang fleksibel yang berkaitan dengan objek seperti database, *table*, *view*, *rows*, dan *columns*.

MySQL bersifat *Open Source* sehingga setiap orang memungkinkan untuk menggunakan dan memodifikasi MySQL. Siapapun boleh mengunduh MySQL di internet dan menggunakannya tanpa harus membayar apapun. MySQL menyediakan *source code* untuk dipelajari dan bisa diubah sesuai kebutuhan.

MySQL berjalan pada client dan server. Server MySQL sangat cepat, handal, *skalabilitas*, dan mudah digunakan sehingga bisa meng-handle database yang besar dengan lebih cepat. MySQL Server bisa berjalan di *desktop* atau *laptop*, *web server*, dan yang lainnya [7].

### 2.1.7 UML

*Unified Modelling Language* (UML) adalah sebuah bahasa yang telah menjadi standar dalam industri bahkan dalam pengembang *software* untuk *visualisasi*, merancang, dan mendokumentasikan *sistem software*. UML menawarkan sebuah standar untuk merancang model sebuah *sistem*.

UML dapat membuat model untuk semua jenis aplikasi *software*, di mana aplikasi tersebut dapat berjalan di *hardware*, *sistem operasi*, jaringan. UML

menggunakan konsep class dan *operation* dalam konsep dasarnya sehingga UML lebih cocok untuk bahasa *pemrograman* berorientasi objek (*Object Oriented Programming*). Walau begitu, UML masih bisa digunakan untuk modelling bahasa pemrograman berbasis prosedural.

Dalam modelling suatu *software*, maka dibutuhkan diagram yang menggambarkan rancangan dari suatu *software*. UML telah mendefinisikan diagram-diagram yang dibutuhkan sebagai berikut [8]:

1. *Use Case Diagram*

   *Use Case Diagram* menggambarkan *fungsionalitas* dari sebuah sistem. *Use Case Diagram* menjelaskan apa yang dilakukan oleh sistem. Sebuah *Use Case* merepresentasikan sebuah interaksi antara aktor dan sistem. *Use Case Diagram* membantu dalam menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan *client*, dan merancang *test case* untuk semua *fitur* yang ada pada sistem.

2. Class Diagram

   *Class Diagram* menggambarkan atribut atau properti dari suatu sistem, sekaligus menyediakan layanan untuk memanipulasi fungsi atau metode tersebut. *Class Diagram* menjelaskan struktur dan deskripsi *class*, *package*, dan *objek* beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain. *Class* jika *diinstansiasi* akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek.

3. State Diagram

   *State Diagram* menggambarkan *transisi* dan perubahan keadaan suatu objek pada sistem sebagai akibat dari interaksi yang diterima. Pada umumnya *State Diagram* menggambarkan *class* tertentu dan satu *class* dapat memiliki lebih dari satu *State Diagram*.

4. *Activity Diagram*

   *Activity Diagram* menggambarkan berbagai alur aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alur berawal, *decision* yang mungkin terjadi, dan bagaimana sistem berakhir. *Activity Diagram* juga menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity Diagram* menjelaskan proses-proses dan jalur-jalur aktivitas dari level

secara umum. Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas.

5. Sequence Diagram

   *Sequence Diagram* menggambarkan interaksi antar objek di dalam dan di sekitar sistem berupa *message* yang digambarkan terhadap waktu. *Sequence Diagram* terdiri antar dimensi *vertikal* (waktu) dan dimensi *horizontal* (objek-objek yang terkait). *Sequence Diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai *respons* dari sebuah *event* untuk menghasilkan *output* tertentu.

6. Collaboration Diagram

   *Collaboration Diagram* juga menggambarkan interaksi antar objek seperti *sequence diagram*, tetapi *Collaboration Diagram* menekankan pada peran masing-masing objek dan bukan pada waktu penyampaian *message*.

7. Component Diagram

   *Component Diagram* menggambarkan struktur dan hubungan antar komponen *software*, termasuk hal yang berkaitan dengan *software* seperti *dependency* di antaranya. Komponen *Software* adalah modul yang berisi *code*, baik berisi *source code* maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada *compile time*, *link time*, maupun *run time*.

8. Deployment Diagram

   *Deployment Diagram* menggambarkan detail bagaimana komponen di- *deploy* dalam *infrastruktur* sistem, di mana komponen akan terletak, bagaimana *infrastruktur* jaringan pada komponen tersebut, spesifikasi server, dan hal-hal lain yang bersifat *fisikal*.

### 2.1.8 *Extreme Programming*

*Extreme Programming (XP)* yang merupakan metode yang memiliki empat tahapan dalam pelaksanaannya, yaitu perencanaan, perancangan, pengkodean dan pengujian. Dengan melaui keempat tahap tersebut diharapkan hasil yang diperoleh menjadi maksimal dan dapat lebih membantu dalam proses yang dibutuhkan. Hasil

dari penelitian ini berupa aplikasi penyaluran dana zakat berbasis web menggunakan *Framework* laravel: studi kasus di yayasan baitul maal pln depok yang dapat memberikan kemudahan dalam mengambil keputusan program yang sudah diajukan oleh amil.[9]

Beberapa perbedaan *extreme programming* dengan metode pengembangan lainnya, diantaranya:

1. Memiliki perubahan yang cepat
2. Proyek yang memiliki resiko tinggi dengan tantangan yang berat
3. Tim developer memiliki sekitar 2 sampai 10orang
4. Developer mendapatkan permintaan dari pelanggan secara langsung

## 2.2 Metode Pengujian

### 2.2.1 Pengujian Fungsionalitas: *Black box* Testing

Pengujian *Software* sangat diperlukan untuk memastikan *software* yang sudah/sedang dibuat dapat berjalan sesuai dengan *fungsionalitas* yang diharapkan. Pengujian *software* merupakan elemen kritis dari jaminan kualitas *software* dan merupakan bagian yang tak terpisah dari siklus hidup *software* yang lain seperti analisis, desain, dan pengkodean[10].

Salah satu jenis pengujian *software* menggunakan metode *Black box Testing*. *Black box Testing* merupakan pengujian yang berfokus pada spesifikasi *fungsional software*, tester dapat mendefinisikan kumpulan kondisi *input* dan melakukan pengetesan pada spesifikasi *fungsional* program [11].

Cara kerja teknik *Black box Testing* terdiri dari 3 langkah:

1. *Input*

Persyaratan dan spesifikasi fungsional dari sistem akan diperiksa. Dokumen desain dan *source code* aplikasi juga diperiksa.

2. *Processing Unit*

Pada proses ini akan dilakukan uji kasus dengan inputan dan juga menjalankannya. Tahap ini juga akan dilakukan *load testing*, *stress testing*, *security review*, dan

*globalization testing*. Jika terdapat kekurangan maka akan diperbaiki dan diuji ulang.

3. *Output*

Setelah melalui semua tahap pengujian, maka akan muncul *output* yang diharapkan dan menyiapkan laporan laporan akhir dari pengujian.

### 2.2.2 Pengujian *User*: *User Acceptance Testing*

*User Acceptance Test* (UAT) merupakan suatu rangkaian proses pengujian di tingkat akhir *fase* pembangunan suatu aplikasi, di mana pengujian ini dilakukan oleh *end User* untuk mengetahui apakah aplikasi yang dibangun sudah sesuai dengan *User requirement*. Berikut merupakan contoh rencana pengujian yang akan dilakukan oleh *User* dengan menjawab pertanyaan yang tersedia setelah melakukan uji coba aplikasi.

Tabel 2. 1 UAT

| No | Fitur | Deskripsi |
|---|---|---|
| 1 | *Dashboard* | Menampilkan informasipenyaluran dana zakat |
| 2 | *Search PM* - Lembaga | *User* mencari PM berdasarkan Nama Lembaga |
| 3 | *Search PM* – Individu | *User* mencari PM berdasarkan NIK |
| 4 | Kelola *User* Admin | *User* mengelola data *User* admin untuk setiap kantor |
| 5 | Kelola Mitra | *User* mengelola data mitra |
| 6 | Kelola Pegawai | *User* mengelola data pegawai di setiap kantor |
| 7 | Kelola Kantor | *User* mengelola data kantor |
| 8 | Kelola Pengajuan Individu | *User* bisa mengelola berkas individu |
| 9 | Kelola Pengajuan Lembaga | *User* bisa mengelola berkas lembaga |
| 10 | Kelola Pengajuan Program | *User* bisa mengelola program |
| 11 | Disposisi | *User* bisa mendisposisikan berkas |
| 12 | *Login* | *User* bisa *login* menggunakan akun |
| 13 | *Account* | *User* bisa mengubah informasiakun |
| 14 | *Password* | *User* bisa mengubah *password* |

### 2.2.3 *Skala Likert*

*Skala Likert* adalah skala yang digunakan untuk mengukur persepsi, sikap atau pendapat seseorang kelompok mengenai sebuah peristiwa atau fenomena sosial, berdasarkan definisi operasional yang telah ditetapkan oleh peneliti. Dengan *Skala Likert*, *variabel* yang akan diukur dijabarkan menjadi *indikator variabel*.

Dalam pengukuran *Skala Likert*, terdapat dua bentuk pertanyaan yaitu[12]:

1. Pertanyaan Positif
   Digunakan untuk mengukur *skala* positif. Pertanyaan positif diberi skor 4, 3, 2, dan 1.
2. Pertanyaan Negatif
   Digunakan untuk mengukur *skala* negatif. Pertanyaan negatif diberi skor 1, 2, 3, dan 4.

Bentuk jawaban *Skala Likert* antara lain: Sangat Baik, Baik, Cukup, Tidak Cukup.

## 2.3 Penelitian Terkait

### 2.3.1 Tabel Penelitian Terkait

Penulisan penelitian ini tak lepas dari berbagai inspirasi lain dari penelitian-penelitian sebelumnya yang mempunya latar belakang yang sama.

1. Pada penelitian pembuatan "Perancangan Sistem InformasiZakat Berbasis Web" disimpulkan bahwa Pembuatan website pengelolaan zakat berbasis web ini memudahkan petugas pengelola zakat dalam hal penginputan data sehingga meminimalisir terjadinya redudansi data dalam hal penginputan data muzaki, mstahik dan transaksi penerimaan dan penyaluran zakat yang sebelumnya dilakukan secara manual [13].
2. Pada penelitian "Pembangunan Aplikasi Monitoring *Budget Event Organizer* Pada PT Indi Notokreasi berbasis *web* menggunakan php *Framework laravel*" disimpulkan bahwa Aplikasi *monitoring budget event organizer* berbasis *web* dapat berfungsi dengan baik. Hal ini dinyatakan dari hasil pengujian fungsional *black-box* telah berjalan 100%, dan hasil

pengujian UAT dinyatakan 83% fitur aplikasi berjalan, dan 17% fitur diterima dengan catatan[14].

3. Pada penelitian "Sistem Informasi Pengelolaan Dana Desa Pada Desa *Hilizoliga* Berbasis Web" disimpulkan bahwa Dengan pengolahan dana desa yang dilakukan secara komputerisasi dapat pengolahan dana desa menjadi lebih cepat, tepat guna, *efektif*, efesien, serta *transparan*[15].
4. Pada penelitian "Rancang Bangun Sistem Informasi *Event* Keagamaan Berbasis *Web* Menggunakan *Framework* Laravel" disimpulkan bahwa Aplikasi sistem informasi*event* keagamaan yang dikembangkan berbasis *web* menggunakan *Framework Laravel* mampu menyajikan *event* keagamaan[16].

Tabel 2. 2 Penelitian Terdahulu

| No | Judul | Peneliti | Tahun | Kesimpulan |
|---|---|---|---|---|
| 1 | Perancangan Sistem InformasiZakat Berbasis Web | Deddy Supriadi dan Leli Fitriani | 2018 | Pembuatan website pengelolaan zakat berbasis web ini memudahkan petugas pengelola zakat dalam hal penginputan data sehingga meminimalisir terjadinya redudansi data dalam hal penginputan data muzaki, mstahik dan transaksi penerimaan dan penyaluran zakat yang sebelumnya dilakukan secara manual |

| No | Judul | Peneliti | Tahun | Kesimpulan |
|---|---|---|---|---|
| 2 | Pembangunan Aplikasi *Monitoring Budget Event* Organizer Pada PT Indi Notokreasi berbasis *web* menggunakan php *Framework laravel* | Muhammad Tarmizi | 2018 | Aplikasi monitoring *budget event organizer* berbasis *web* dapat berfungsi dengan baik. Hal ini dinyatakan dari hasil pengujian fungsional *black-box* telah berjalan 100%, dan hasil pengujian UAT dinyatakan 83% fitur aplikasi berjalan, dan 17% fitur diterima dengan catatan. |
| 3 | Sistem InformasiPengelolaan Dana Desa Pada Desa *Hilizoliga* Berbasis *Web* | Sorang Pakpahan | 2020 | Dengan pengolahan dana desa yang dilakukan secara komputerisasi dapat pengolahan dana desa menjadi lebih cepat, tepat guna, efektif, efesien, serta transparan. |
| 4 | Rancang Bangun Sistem Informasi*Event* Keagamaan Berbasis *Web* Menggunakan *Framework* Laravel | Aufa Billah Putra Jazama | 2019 | Aplikasi sistem informasi*event* keagamaan yang dikembangkan berbasis *web* menggunakan *Framework Laravel* |

| No | Judul | Peneliti | Tahun | Kesimpulan |
|---|---|---|---|---|
| | | | | mampu menyajikan *event* keagamaan |

### 2.3.2 Posisi Penelitian

Pada penelitian ini posisi penelitian yang dilakukan di antara penelitian yang telah dilakukan sebelumnya, tertera pada table 2.

Tabel 2. 3 Posisi Penelitian

| Web | MySQL | Laravel | *Extreme Programming* |
|---|---|---|---|
| Deddy Supriadi dan Leli Fitriani, 2018 (Universitas Bina Sarana *Informatika*) Rancang Bangun Sistem InformasiAdministrasi Pengelolaan Dana Masjid Pada Yayasan Al-Muhajirin, Tanggerang | | | |
| Sorang Pakpahan, 2020 (Universitas Katolik Santo Thomas) Sistem InformasiPengelolaan Dana Desa Pada Desa *Hilizoliga* Berbasis *Web* | | | |
| Muhammad Tarmizi, 2018 (Sekolah Tinggi Teknologi Terpadu Nurul Fikri) Pembangunan Aplikasi Monitoring *Budget Event Organizer* Pada PT Indi Notokreasi berbasis *web* menggunakan php *Framework laravel* | | | |
| Aufa Billah Putra Jazama, 2019(Sekolah Tinggi Teknologi Terpadu Nurul Fikri) Rancang Bangun Sistem | | | |

| | | | |
|---|---|---|---|
| **Web** | **MySQL** | **Laravel** | ***Extreme Programming*** |
| Informasi*Event* Keagamaan Berbasis *Web* Menggunakan *Framework Laravel* | | | |
| TB.M Billal Zakky Surya, 2022 (Sekolah Tinggi Teknologi Terpadu Nurul Fikri) Rancang Bangun Aplikasi Penyaluran Dana Zakat Berbasis Web Menggunakan *Framework Laravel* Studi Kasus Di Yayasan Baitul Maal PLN Depok. | | | |