BAB II

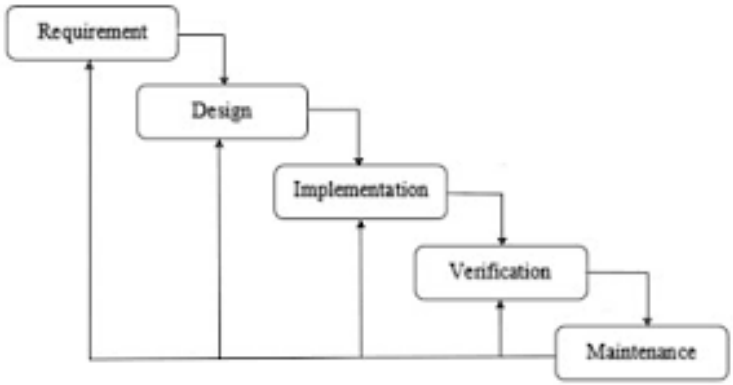
LANDASAN TEORI

2.1 Model Pengembangan Sistem

Dalam penelitian ini digunakan metode pengembangan perangkat lunak model air terjun (*Waterfall*).

Menurut Pressman dalam (Moenir & Yulianto, 2017) bahwa "model *waterfall* adalah mode klasik yang bersifat sistematis, berurutan dalam membangun *software*. Nama model ini sebenarnya adalah "*Linear Squental Model"*. Model ini sering disebut juga dengan "*Classic Life Cycle"* atau metode *waterfall*.

Menurut (Shalahuddin & Sukamto) dalam (Firmansyah & Udi , 2017) menjelaskan bahwa "model *waterfall* sering disebut model sekuensi linear atau alur hidup klasik. Pengembangan sistem dikerjakan secara terurut mulai dari anaslisi, desain, pengkodean, pengujian dan tahapan pendukung".



Sumber : Sukamto dan Shalahudin (2017)

Gambar II.1 Model *Waterfall*

Terdapat 5 tahapan metode *waterfall*, yaitu :

2.1.1 Analisa Kebutuhan Perangkat Lunak

Pada tahap analisis kebutuhan perangkat lunak ini penulis melakukan pengumpulan data dari dilakukannya observasi untuk mengamati sistem berjalan saat ini, dan melakukan wawancara untuk memperoleh data-data dengan wawancara langsung dengan *Manager Rental Head* di PT. Berdikari Pondasi Perkasa, selanjutnya melakukan pengumpulan data dengan cara studi pustaka mencari referensi dari sumber-sumber terpercaya yang berasal dari buku-buku, karya ilmiah, dan jurnal. Data-data yang didapatkan untuk pengembang perangkat lunaknya adalah dari *database* kendaraan bermotor.

2.1.2 Desain

Pada tahapan desain ini penulis merancang sistem secara luas dan menarik menggunakan program bantu dalam pembuatan perancangan program menggunakan *PHP, XAMPP, Visual Studio Code, Bootstrap* dan dalam perancangan program dalam pemodelan sistem menggunakan diagram *Entity Relationship Diagram* (ERD) dan *Logical Relationship Structure* (LRS).

2.1.3 Pembuatan Kode Program

Pada tahapan ini penulis menggunakan Bahasa pemrograman PHP dalam kode program mengguanakan program bantu aplikasi *Visual Studio Code* untuk merancang program yang akan dibuat. Desain harus dipindahkan kedalam program perangkat lunak, hasil dari tahapan ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahapan desain.

2.1.4 Pengujian

Pengujian fokus pada perangkat lunak dari segi logika dan fungsional serta memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan. Pengujian dalam web ini menggunakan teknik *Blackbox testing.*

2.1.5 Pendukung

Pada tahapan ini perangkat lunak tidak menutup kemungkinan mengalami perubahan ketika sudah diimplementasikan. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spefisikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat baru.

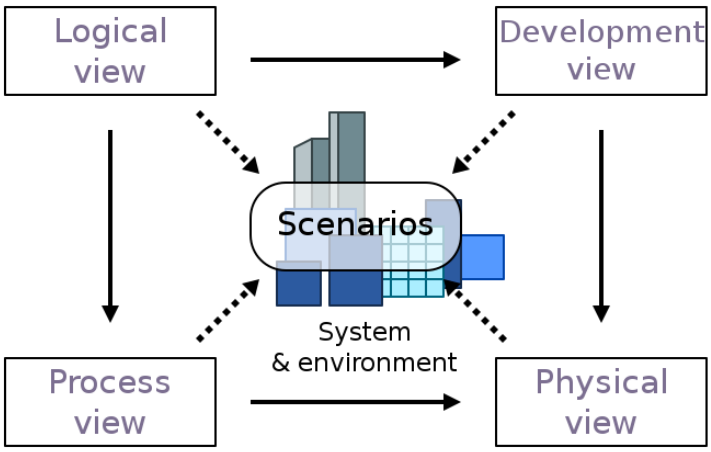
2.2 Metode Perancangan Sistem

Adapun rancangan sistem data yang dibuat pada Proyek Sistem Informasi SI-MOJAKEMO (Sistem *Monitoring* Pajak Kendaraan Bermotor) pada PT. Berdikari Pondasi Perkasa Jakarta adalah sebagai berikut :

2.2.1 UML (*Unified Modeling Language*)

Menurut (Munawar, 2018), *Unified Modeling Language* (UML) adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem berorientasi *object.* Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembang sistem untuk membuat

cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing)* dan mengkomunikasikan rancangan mereka dengan yang lainnya. UML dibangun atas model 4+1 *view*. Model ini didasarkan pada fakta bahwa struktur sebuah sistem dideskripsikan dalam 5 *view* dimana salah satu diantaranya *scenario*. *Scenario* ini memegang peran khusus untuk mengintegrasikan *content* ke *view* yang lain. Kelima *view* tersebut berhubungan dengan perspektif tertentu dimana sistem akan diuji. *View* yang berbeda akan menekankan pada aspek yang berbeda dari sistem yang mewakili ketertarikan sekelompok *stakeholder* tertentu.



Sumber : Munawar (2018)

Gambar II.2 Model 4+1 *View*

1. *Scenario*

Scenario menggambarkan interaksi diantara obyek dan diantara proses. *Scenario* ini digunakan untuk identifikasi elemen arsitektur, ilustrasi, dan validasi desain arsitektur serta sebagai titik awal untuk pengujian *prototype* arsitektur.

2. Development View

Development View menjelaskan sebuah sistem dari perspektif *programmer* dan terkonsentrasikan ke manajemen perangkat lunak. *View* ini dikenal juga sebagai *implementation view.* Diagram UML yang termasuk dalam *development view* diantaranya adalah *component diagram* dan *package diagram*.

3. Logical View

Logical View dengan fungsionalitas sistem yang dipersiapkan untuk pengguna akhir. *Logical View* mendeskripsikan struktur logika yang mendukung fungsi-fungsi yang dibutuhkan di *use case. Logical View* ini berisi *object diagram, class diagram, state machine diagram* dan *composite structure diagram.*

4. Physical View

Physical View mengambarkan sistem dari perspektif sistem *engineer*. Focus dari *physical view* adalah topologi sistem perangkat lunak. Diagram UML yang termasuk dalam *physical view* ini adalah *development diagram* dan *tinning diagram.*

5. Process View

Process View berhubungan erat dengan aspek dinamis dari sistem, proses yang terjadi di sistem dan bagaimana komunikasi yang terjadi di sistem serta tingkah laku sistem saat dijalankan. *Process view* menjelaskan apa itu *concurrency,* distribusi integrasi, kinerja dan lain-

lain. *Process View* ini berisi *activity diagram, communication diagram, sequence diagram,* dan *interaction overview diagram.*

Menurut (Rosa A.S & M Shalahuddin , 2018) dalam buku yang berjudul Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek, *Unified Modeling Language* (UML) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek.

UML muncul karena adanya kebutuhan pemodelan visual untuk mengspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak. *Unified Modeling Language* merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung.

2.2.2 *Use Case Diagram*

Menurut (Munawar, 2018), *Use case diagram* adalah salah satu jenis diagram pada UML yang mengambarkan interaksi antara sistem dan *actor*. *Use case diagram* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah *system* dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah *system* dipakai.

2.2.3 *Class Diagram*

Class Diagram adalah salah satu diagram yang paling popular bagi *software engineering*. *Class Diagram* mewakili entitas kunci dalam bisnis

serta domain teknis. *Class Diagram* sifatnya sangat structural dan statis. *Class Diagram* dapat menunjukan kelas yang ada pada tingkat bisnis, juga dengan kelas pada tingkat teknis yang berasal dari bahasa impelementasi (misalnya *Java* atau C++) (Unhelkar, 2018).

2.2.4 PHP (*Hypertext Preprocessor*)

Menurut (Supono & Putratama, 2018) mengemukakan bahwa "PHP (*Hypertext Preprocessor)* adalah suatu bahasa pemrograman yang digunakan untuk menterjemahkan basis kode program menjadi kode mesin yang dapat dimengerti oleh komputer yang bersifat *server-side* yang ditambahkan ke HTML".

Berdasarkan pengertian diatas, maka dapat disimpulkan bahwa PHP (*Hypertext Preprocessor)* merupakan bahasa pemograman yang mengolah *database*, *content website* sehingga *website* yang dibuat merupakan *web* dinamis, dan PHP merupakan bahasa pemograman yang dikombinasikan dengan HTML.

2.2.5 XAMPP

Menurut (Supono & Putratama, 2018) mendefinisikan bahwa "XAMPP terdiri dari apache *web server*, MySQL, PHP, Perl, FTP *server* dan phpMyAdmin".

Berdasarkan pengertian dari para ahli di atas, dapat disimpulkan bahwa XAMPP merupakan *software* yang digunakan sebagai *server* untuk mengeksekusi fungsi *website* tanpa jaringan *internet.*

2.2.6 Bootstrap

Bootstrap merupakan sebuah f*ront-end framework* yang mengedapankan tampilan untuk *mobile device phone* (Wandela & Elisa, 2019). Kelebihan saat memakai *Bootstrap* yaitu diciptakan memakai *Less*, teknologi CSS yang sederhana dan mudah untuk digunakan. Dengan adanya *Less*, maka pengembang bisa melakukan akses yang lebih mudah informasi dan fungsi warna, variabel, dan operasi penggunaan (Christian et al., 2018).

2.3 Metode Perancangan Basis Data

2.3.1 *Entity Relationship Diagram (ERD)*

Menurut (A S & Shalahddin, 2018) menyatakan bahwa "ERD atau *Entity Relationship Diagram* dikembangkan berdasarkan teori himpunan dalam bidang matematik, ERD digunakan untuk pemodelan basis data relasional, ERD digunakan untuk pemodelan basis data relasional".

2.3.2 *Logical Record Structure* (LRS)

Menurut jurnal Sistem Informasi Kependudukan di Rukun Tetangga 04/08 Kelurahan Utan Panjang Berbasis *Web* 2017 Ibnu Rusdi & Mohamad Abi Mashabi LRS (*Logical Record Structure)* adalah representasi dari struktur *record-record* pada tabel yang terbentuk dari hasilkan antar himpunan entitas.