# BAB III

**PELAKSANAAN KERJA PRAKTEK**

## 3.1 Analisis Permasalahan

Saat ini PT PLN ULP Cilegon saat ini belum mempunyai website yang dapat menampilkan data dan chart yang berkaitan operasional mereka. Hal ini menyebabkan kesulitan dalam memantau, menganalisis, dan mengambil Keputusan berdasarkan data yang ada. Selain itu, Perusahaan juga belum memiliki system yang terintegrasi dan terstandarisasi untuk mengelola data dan chart yang berasala dari berbagai sumber dan format

## 3.2 Usulan Pemecahan Masalah

membuat website dashboard yang dapat menampilkan data dan chart yang relevan, akurat, dan terkini dengan menggunakan teknologi PHP, Laravel, ChartJS, dan mysql. PHP Laravel adalah framework web yang popular dan mudah digunakan, yang menyediakan fitur-fitur seperti routing, templating, authentication, dan testing. ChartJS adalah library Javascript yang dapat menampilkan data dalam bentuk chart yang bervariasi, interaktif, dan responsive. MySQL adalah system manajemen basis data yang

cepat, dan fleksibel yang dapat menyimpan dan mengolah data dalam skala besar.

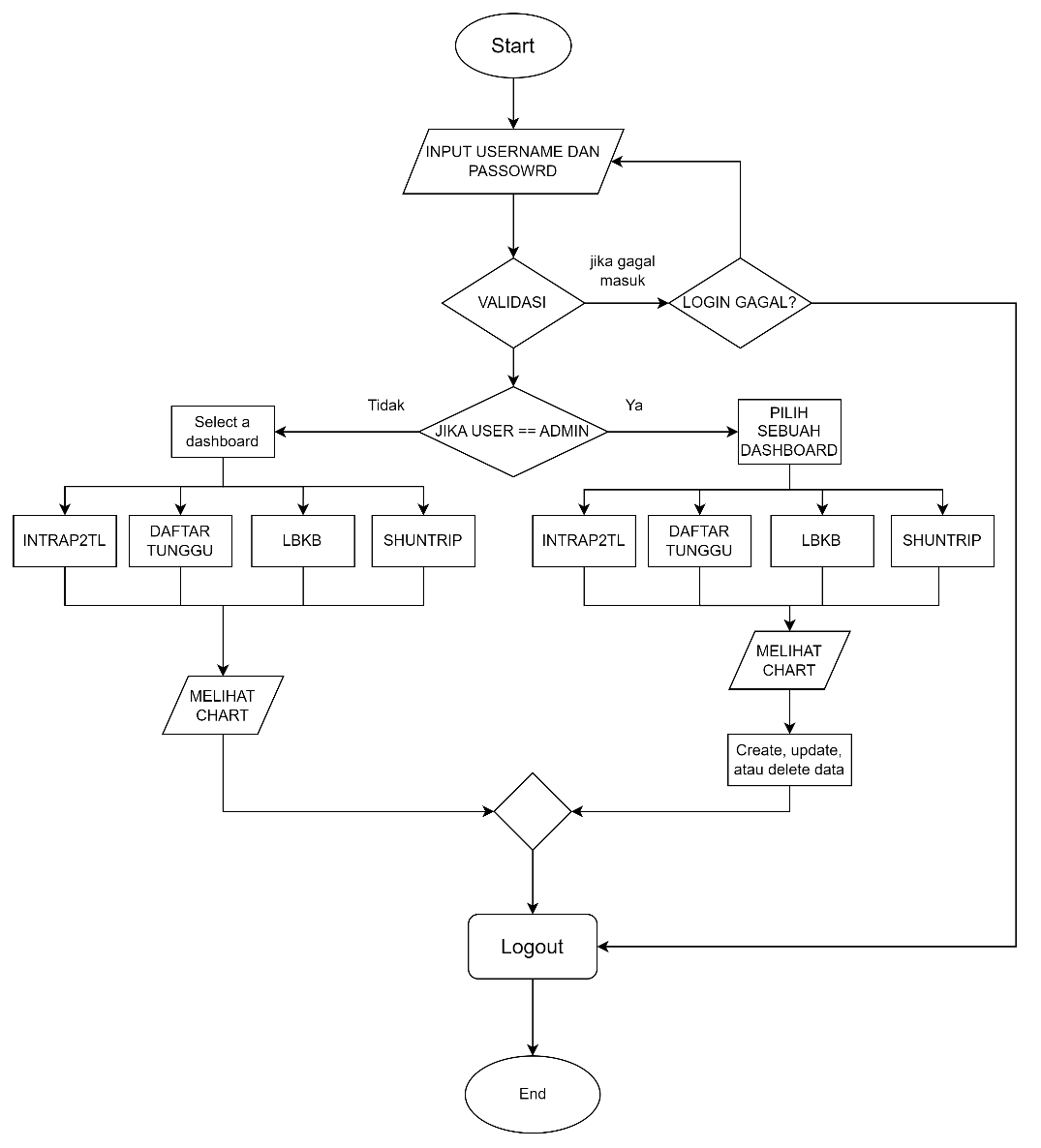
## 3.3 Desain Sistem

Adapun tahapan pembangunan sebuah website adalah menggunakan pendekatan model waterfall. Model waterfall adalah model klasik yang bersifat sistematis, berurutan dalam membangun software. Model ini melakukan pendekatan secara sistematis dan berurutan. Disebut dengan waterfall karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan.

Tahap pengembangan perangkat lunak dalam waterfall model adalah sebagai berikut:

* + 1. Analisis kebutuhan: Tahap ini melibatkan pengumpulan dan pendokumentasian kebutuhan fungsional dan non-fungsional dari perangkat lunak yang akan dibangun. Tujuan dari tahap ini adalah untuk mendapatkan spesifikasi kebutuhan yang jelas, lengkap, konsisten, dan dapat diverifikasi.
    2. Desain: Tahap ini melibatkan perancangan arsitektur, antarmuka, algoritma, dan struktur data dari perangkat lunak. Tujuan dari tahap ini adalah untuk mendapatkan desain yang efisien, andal, mudah dipelihara, dan dapat diuji.
    3. Implementasi: Tahap ini melibatkan pengkodean, pengujian unit, integrasi, dan debugging dari perangkat lunak. Tujuan dari tahap ini adalah untuk menghasilkan kode yang sesuai dengan desain dan memenuhi kebutuhan.
    4. Pengujian: Tahap ini melibatkan pengujian sistem, pengujian fungsional, pengujian non-fungsional, pengujian regresi, dan pengujian penerimaan dari perangkat lunak. Tujuan dari tahap ini adalah untuk memastikan bahwa perangkat lunak bebas dari kesalahan, sesuai dengan spesifikasi, dan memuaskan pelanggan.
    5. Pemeliharaan: Tahap ini melibatkan perbaikan bug, peningkatan kinerja, penambahan fitur, dan adaptasi terhadap perubahan lingkungan dari perangkat lunak. Tujuan dari tahap ini adalah untuk menjaga kualitas dan fungsionalitas perangkat lunak selama siklus hidupnya.

### 3.3.1 Flow Chart



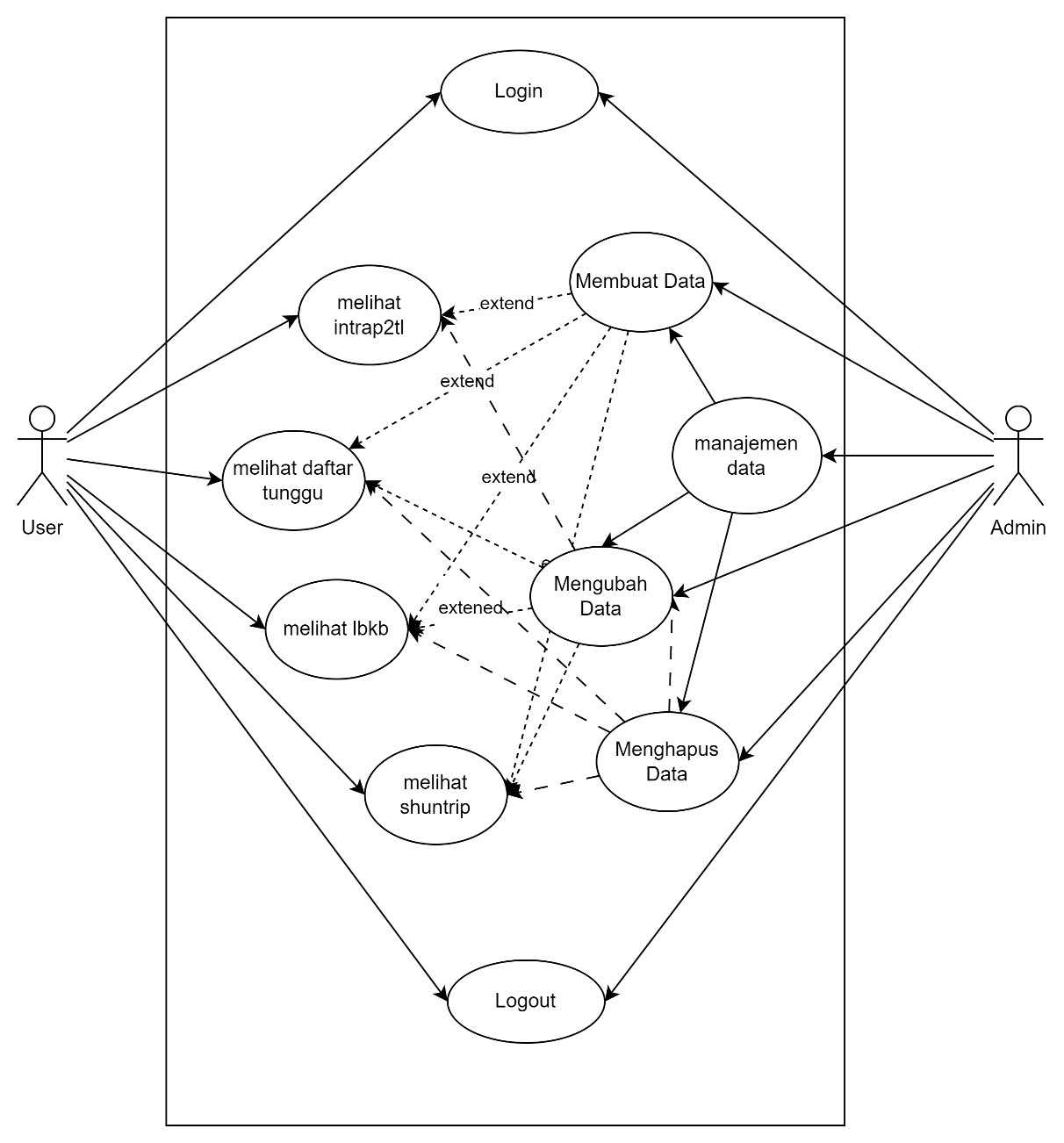
**Gambar 2.** Flowchart

Flowchart adalah sebuah diagram yang menggambarkan alur proses atau logika dari suatu sistem. Flowchart menggunakan simbol-simbol standar untuk menunjukkan aktivitas, kondisi, dan alur logika dari proses yang digambarkan. Flowchart bisa digunakan dalam berbagai bidang, seperti pengembangan perangkat lunak, perencanaan bisnis, manajemen proyek, dan lain-lain. Flowchart dapat membantu dalam menentukan alur proses, menemukan kesalahan atau kekurangan dalam sistem, dan meningkatkan efisiensi suatu proses. Flowchart juga dapat digunakan untuk menjelaskan bagaimana suatu sistem bekerja pada orang yang tidak familiar dengan sistem tersebut.

Flowchart diatas menjelaskan bagaimana proses pengguna dengan website Dimana jika pengguna adalah admin makai a dapat membuat, mengubah, dan menghapus data, sedangkan jika ia pengguna yang bukan admin maka hanya bisa melihar grafiknya saja.

### 3.3.2 Diagram

#### 3.3.2.1 Use Case Diagram



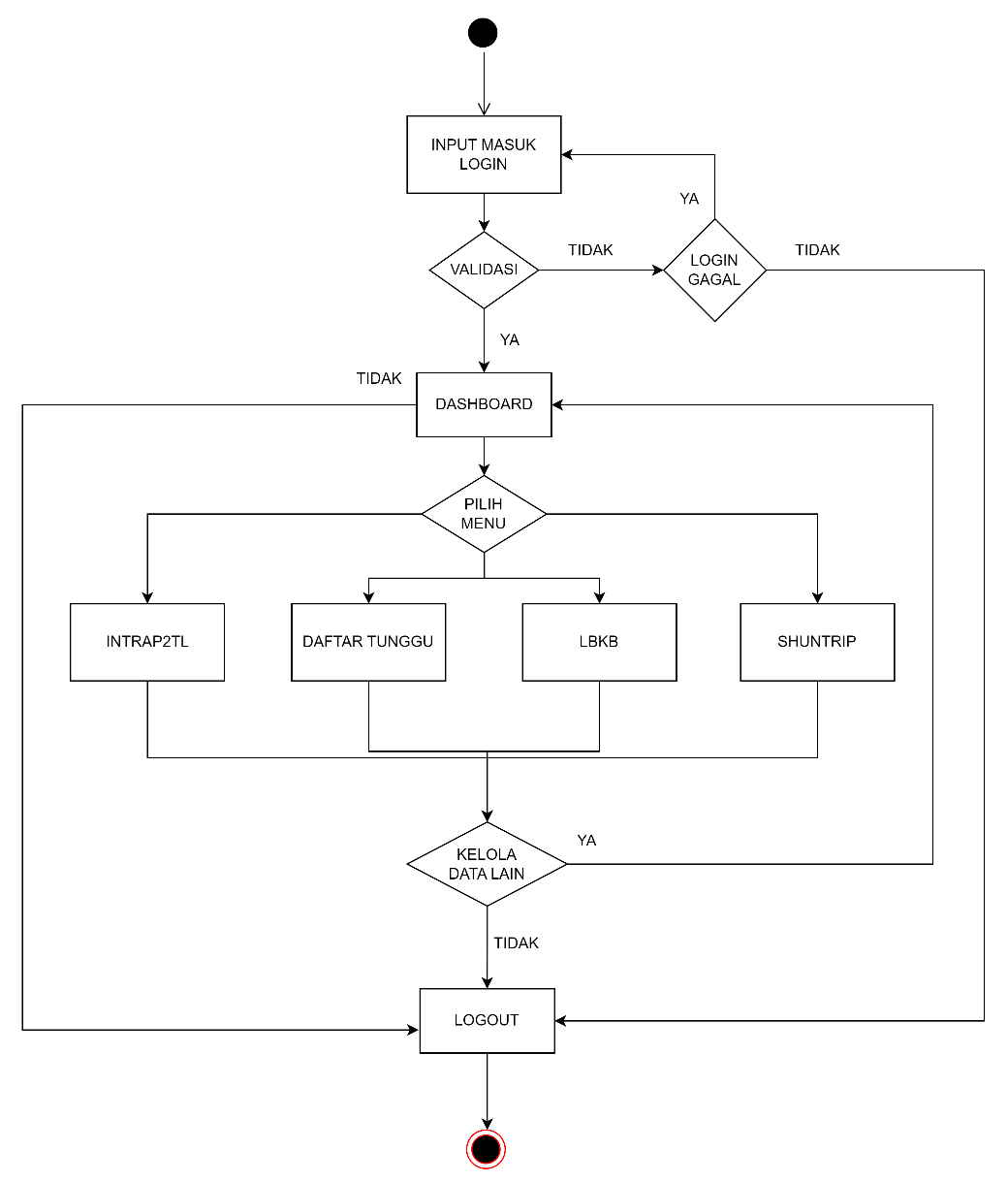
**Gambar 4.** Use case Diagram

Use case diagram adalah jenis diagram UML yang menggambarkan fungsi, ruang lingkup, dan interaksi pengguna dengan sistem. Use case diagram memvisualisasikan interaksi antara pengguna (aktor) dan sistem (use case), serta tindakan apa saja yang dapat dilakukan aktor terhadap use case secara rinci. Use case diagram dapat digunakan untuk merancang sistem dari sisi fungsionalitas, persyaratan, atau skenario.

Beberapa tujuan dan fungsi dari use case diagram adalah:

* Menjelaskan tujuan dan manfaat dari sistem bagi pengguna
* Mendefinisikan dan mengorganisir persyaratan fungsional dari sistem
* Menentukan ruang lingkup dan batasan dari sistem
* Memodelkan alur dasar dan alternatif dari use case
* Mengkomunikasikan kebutuhan pengguna dan stakeholder kepada tim pengembang

#### 3.3.2.2 Activity Diagram



**Gambar 5.** Activity Diagram

Activity diagram adalah sebuah diagram yang menggambarkan alur aktivitas atau proses yang terjadi pada sebuah sistem. Activity diagram menggunakan simbol-simbol standar untuk menunjukkan aktivitas, kondisi, dan alur logika dari proses yang digambarkan. Activity diagram bisa digunakan dalam berbagai bidang, seperti pengembangan perangkat lunak, perencanaan bisnis, manajemen proyek, dan lain-lain. Activity diagram dapat membantu dalam menentukan alur proses, menemukan kesalahan atau kekurangan dalam sistem, dan meningkatkan efisiensi suatu proses. Activity diagram juga dapat digunakan untuk menjelaskan bagaimana suatu sistem bekerja pada orang yang tidak familiar dengan sistem tersebut.

Beberapa tujuan dan fungsi dari activity diagram adalah:

* Menjelaskan urutan aktivitas dalam suatu proses.
* Dapat digunakan untuk modeling (memperlihatkan urutan proses bisnis).
* Mudah dalam memahami proses yang ada dalam sistem secara keseluruhan.
* Merupakan metode perancangan yang terstruktur, mirip dengan flowchart maupun data flow diagram.
* Mengetahui aktivitas aktor/pengguna berdasarkan use case/diagram yang dibuat sebelumnya

#### 3.3.2.3 Class diagram



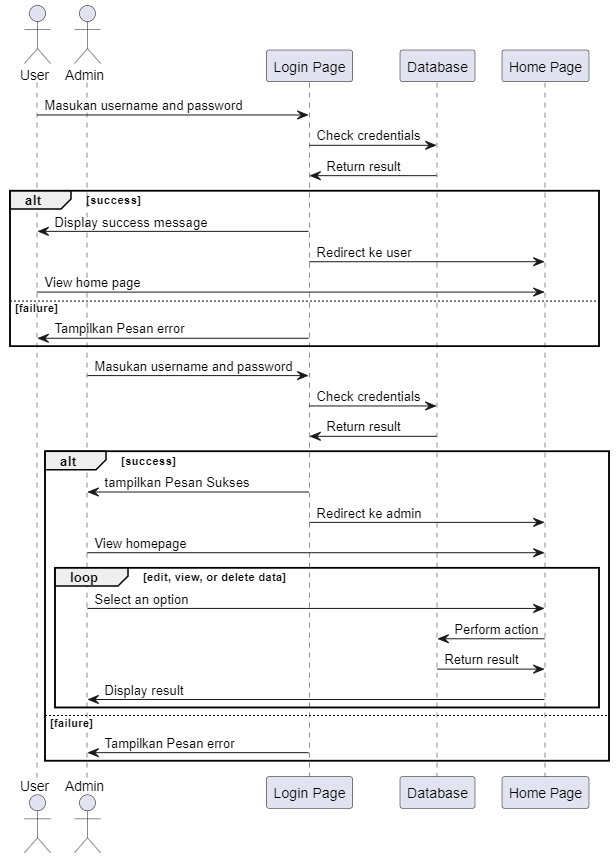
**Gambar 6.** Class Diagram

Class diagram adalah sebuah diagram yang menggambarkan struktur dan hubungan dari setiap objek dalam sebuah sistem. Class diagram menggunakan simbol-simbol standar untuk menunjukkan kelas, atribut, operasi, dan hubungan antara kelas. Class diagram dapat digunakan untuk memodelkan sistem dari sisi struktur, perilaku, atau interaksi.

Tujuan dari class diagram adalah untuk menunjukkan struktur statis dari sebuah sistem, yaitu apa yang ada di dalam sistem dan bagaimana mereka berhubungan. Class diagram dapat membantu dalam mendefinisikan spesifikasi, dokumentasi, dan implementasi dari sebuah sistem. Class diagram juga dapat digunakan untuk analisis, desain, dan pengujian sistem.

Fungsi dari class diagram adalah untuk menggambarkan kelas-kelas yang ada di dalam sistem, atribut yang dimiliki oleh setiap kelas, operasi-operasi yang dapat dilakukan oleh setiap kelas, dan hubungan-hubungan antara kelas-kelas, seperti asosiasi, agregasi, komposisi, atau generalisasi.

#### 3.3.2.4 Sequence diagram



**Gambar 7.** Sequence Diagram

Sequence diagram adalah sebuah diagram yang menggambarkan interaksi antara objek-objek dalam sebuah sistem secara urutan waktu. Sequence diagram menggunakan simbol-simbol standar untuk menunjukkan objek-objek, pesan-pesan, dan alur logika dari interaksi yang digambarkan. Sequence diagram dapat digunakan untuk memodelkan sistem dari sisi perilaku, interaksi, atau skenario.

Tujuan dari sequence diagram adalah untuk menunjukkan urutan kejadian yang dapat menghasilkan output yang diinginkan. Sequence diagram dapat membantu dalam menentukan alur proses, menemukan kesalahan atau kekurangan dalam sistem, dan meningkatkan efisiensi suatu proses. Sequence diagram juga dapat digunakan untuk menjelaskan bagaimana suatu sistem bekerja pada orang yang tidak familiar dengan sistem tersebut.

Fungsi dari sequence diagram adalah untuk menggambarkan objek-objek yang berpartisipasi dalam interaksi, pesan-pesan yang dikirim dan diterima oleh objek-objek, dan urutan dari pesan-pesan tersebut. Objek-objek dapat direpresentasikan dengan nama atau alias, dan diberi tanda titik dua di depannya. Pesan-pesan dapat berupa sinkron, asinkron, balasan, atau penciptaan. Urutan dari pesan-pesan dapat ditunjukkan dengan nomor atau garis putus-putus.

#### 3.3.2.5 Rancangan Database

Struktur database terdiri dari 9 table yang dimana menyimpan data dari tiap-tiap halaman yang ada. Nama dari tiap database ini adalah sebagai berikut:

1. **shuntrip.**

**Tabel 2.** Tabel shuntrip

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama Field | Tipe | Panjang | Keterangan |
| 1 | Id | int | 11 | Primary key, auto\_increment |
| 2 | bulan | varchar | 20 | - |
| 3 | target | int | 10 | - |
| 4 | realisasi | int | 10 | - |

1. **transaksienergi.**

**Tabel 3.** transaksienergi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama Field | Tipe | Panjang | Keterangan |
| 1 | Id | int | 11 | Primary key, auto\_increment |
| 2 | bulan | varchar | 20 | - |
| 3 | target | int | 10 | - |
| 4 | realisasi | int | 10 | - |

1. **lbkb.**

**Tabel 4.** LBKB

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama Field | Tipe | Panjang | Keterangan |
| 1 | Id | int | 11 | Primary key, auto\_increment |
| 2 | bulan | varchar | 20 | - |
| 3 | target | int | 10 | - |
| 4 | realisasi | int | 10 | - |

1. **users.**

**Tabel 5.** Users

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama Field | Tipe | Panjang | Keterangan |
| 1 | Id | int | 20 | Primary key, auto\_increment |
| 2 | Username | varchar | 255 | - |
| 3 | password | int | 255 | - |

1. **daftung.**

**Tabel 6.** Daftung (daftar tunggu)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama Field | Tipe | Panjang | Keterangan |
| 1 | Id | int | 10 | Primary key, auto\_increment |
| 2 | tanggal | date | - | - |
| 3 | Permohonan | varchar | 10 | - |
| 4 | tigahs | int | 10 | - |
| 5 | ltigahs | int | 10 | - |
| 6 | kumulatif | float | - | - |
| 7 | target | int | 10 | - |
| 8 | persen | int | 10 | - |

1. **daftungbulanberjalan.**

**Tabel 7.** Daftung bulan berjalan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama Field | Tipe | Panjang | Keterangan |
| 1 | id | int | 20 | Primary key, auto\_increment |
| 2 | tanggal | date | - | - |
| 3 | wopk | int | 30 | - |
| 4 | pb | int | 30 | - |
| 5 | pd | int | 30 | - |
| 6 | paskem | int | 30 | - |
| 7 | layanant | varchar | 50 | - |
| 8 | plgt | int | 30 | - |
| 9 | layananr | int | 30 | - |
| 10 | pcl | int | 30 | - |
| 11 | limahari | int | 30 | - |
| 12 | limabelashari | int | 30 | - |
| 13 | dualimahari | int | 30 | - |
| 14 | status | int | 30 | - |