**3. ANALISA DAN PEMBAHASAN**

* 1. **Identifikasi Masalah**

Kegiatan pengisian presensi dapat membantu seorang manajer dalam merekap jumlah gaji bagi seorang pegawai. Namun sistem presensi bagi pegawai yang bekerja di lapangan, terkadang susah untuk di realisasikan. Karena pegawai lapangan bekerja tidak menetap di satu lokasi, sehingga absensi terkadang di lakukan secara manual dan dipantau oleh seorang manajer secara manual.

Kelemahan dari sistem absensi manual adalah membuat pekerjaan manajer menjadi lebih banyak, karena manajer harus memproses data presensi secara manual, dan melakukan pemantauan secara manual juga.

Beberapa perusahaan menyadari jika sistem manual tersebut sangat membuang waktu dan tenaga manajer, sehingga disusun sebuah sistem presensi dengan sidik jari. Namun sistem presensi dengan sidik jari kurang relevan dengan pegawai yang bekerja turun langsung ke lapangan, karena pegawai tidak menetap di satu lokasi secara permanen.

Dari masalah tersebut penulis menyadari harus ada solusi yang lebih tepat untuk melakukan kegiatan presensi dengan lebih efisien bagi karyawan yang bekerja di luar kantor. Penulis menyadari bahwa hal utama dalam sistem presensi adalah bagaimana cara agar data yang diterima valid dan bukan laporan presensi palsu, kemudian bagaimana cara agar data yang diterima manajer adalah data yang telah diolah, sehingga memudahkan manajer dalam memproses data tersebut menjadi hal yang lebih berguna dan dapat diteruskan ke sebuah sistem informasi lainnya seperti sistem informasi penggajian karyawan.

Dan untuk masalah yang tak luput dari perhatian penulis adalah bagaimana cara agar sistem informasi ini dapat digunakan oleh semua perusahaan. Oleh karena itu penulis membuat sistem informasi ini, dengan fleksibel dan terintegrasi dengan tiga tingkatan level akses.

Tingkatan level 1 atau admin dapat mengatur semua hal dari sistem presensi *FreeSent* yang penulis buat. Sehingga admin dapat melakukan berbagai hal seperti mengganti judul dari sistem presensi, mengubah logo, roles, mengatur user, mengatur tingkatan level user, mengatur menu, mengatur nama tingkatan level dan lain sebagainya.

Tingkatan level 2 atau manajer dapat mengatur karyawan yang dimilikinya, mengakses rekapan presensi, mengunduh rekapan dalam bentuk excel, mengatur rekapan presensi jika dinilai ada yang mencurigakan atau kesalahan.

Tingkatan level 3 atau user/pegawai dapat melihat rekapan dari presensi yang user laporkan ke sistem.

* 1. **Analisa**

1. **Analisa Kebutuhan Sistem**

Dalam membuat aplikasi ini, penulis membutuhkan beberapa kebutuhan *software, framework*, *dependency package*, *hardware, server dan smartphone* sebagai berikut:

1. Kebutuhan *software* yang digunakan, antara lain:
2. Sistem Operasi Ubuntu 16.04 xenial
3. Text Editor Atom versi 1.24.1
4. Browser Google Chrome Versi 65.0.3325.181
5. Apache2 Server
6. Kebutuhan *framework* yang digunakan, antara lain:
7. Laravel versi 5.5
8. React-Native
9. Kebutuhan *dependency package* yang digunakan, antara lain:
10. barryvdh/laravel-cors
11. fideloper/proxy
12. spatie/laravel-fractal
13. yajra/laravel-datatables-oracle
14. maatwebsite/excel
15. react-native-vector-icons
16. react-navigation
17. native-base
18. Kebutuhan *hardware* yang digunakan, antara lain:
19. Intel Core i5-5200U CPU @ 2.7GHz
20. Kapasitas RAM 11929MB
21. Kapasitas Hardisk 500GB
22. Kebutuhan *Smartphone* yang digunakan, antara lain:

Digunakan lima (3) tipe *Smartphone* Android untuk menguji aplikasi

Tabel 3.1 : Tabel Tipe *Smartphone* untuk Menguji Aplikasi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tipe** | **Merk** | **Ukuran Layar** | **RAM** | **Versi Android** |
| Asus Zenfone Max Plus M1 | Asus | 5.7” | 4 GB | 7.0 (Nougat) |
| Asus Zenfone 5 | Asus | 5.0” | 2 GB | 4.4 (Kitkat) |
| Samsung Galaxy J5 (2016) | Samsung | 5.2” | 2 GB | 6.0.1 (Marshmallow) |

1. **Analisa Kebutuhan Server**

Penulis memerlukan server agar end-point api yang ada dapat diakses bagi semua, serta dapat melakukan pengujian sistem dengan menggunakan *production build* lansgung di *server*.

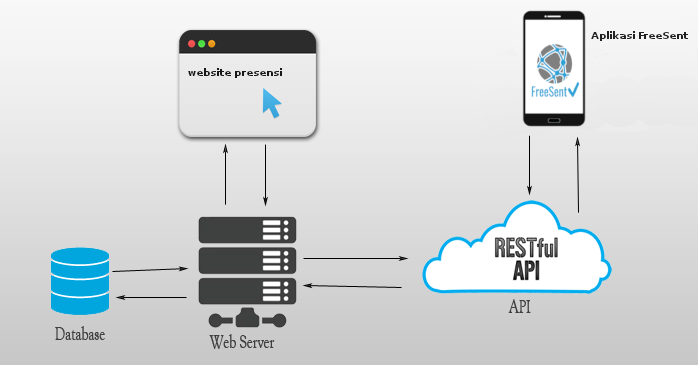
Kebutuhan *server* yang digunakan, antara lain:

1. Kapasitas Penyimpanan 500MB
2. Kapasitas Penyimpanan Database 318MB
3. Kapasitas Ram 1024MB
4. PHP 7.0
5. Akses git
6. Akses ssh
7. **Analisa Rest-Api**

Secara umum aplikasi sistem presensi yang dibuat akan menggunakan sistem *Rest-Api* untuk aplikasi android bagi user, sedangkan untuk website bagi manajer penulis tidak menggunakan sistem *Rest-Api*.

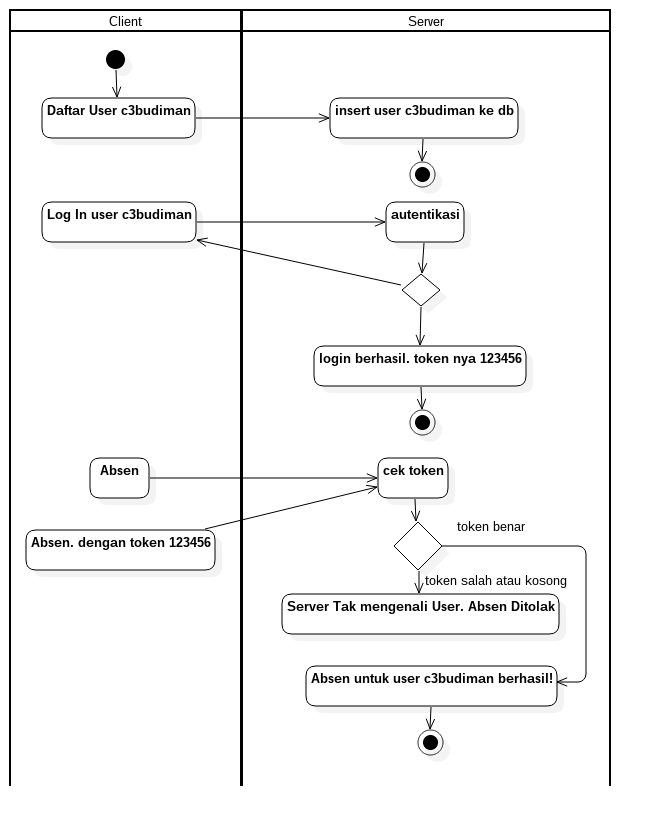
Bagi yang tidak familiar dengan sistem *Rest-Api*, penulis akan menjelaskan bagaimana client yang disini adalah sebuah aplikasi *Android*, berkomunikasi dengan end-point api yang ada di server.

Rest Api akan membantu aplikasi android dalam penyimpanan dan pengaksesan data, seperti terlihat pada gambar 3.1 dibawah ini, Aplikasi FreeSent akan membuat request dalam protokol http yang berisi request/response dalam bentuk json ke dalam endpoint RESTful Api yang nantinya akan diteruskan ke web server untuk diolah dengan laravel yang kemudian akan memasukkan atau memanggil data yang ada di database.



Gambar 3.1 Deployment Diagram FreeSent

*Activity Diagram Rest-api* menggambarkan aliran kerja atau aktivitas client atau frontend dalam berinteraksi dengan server secara umum.



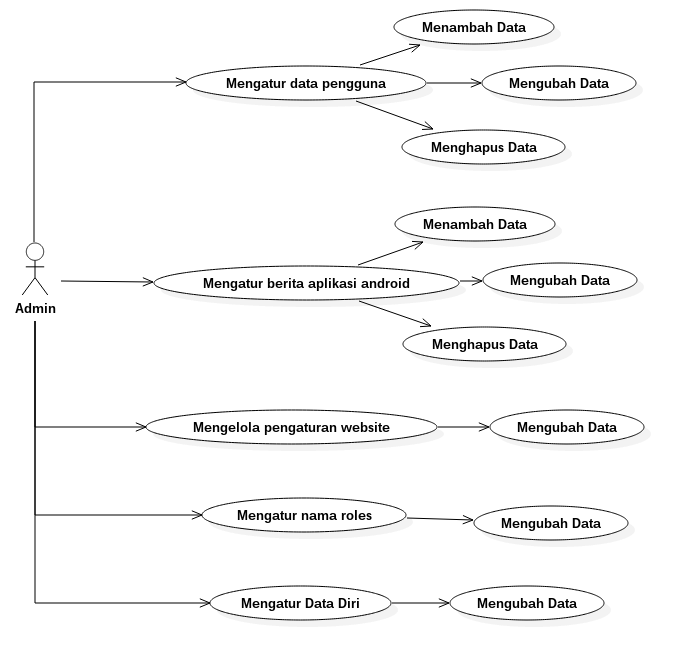
Gambar 3.2 Activity Diagram Interaksi Client-Server dengan konsep Rest-Api.

* 1. **Rancangan Struktur Sistem**

Dalam membangun sebuah sistem dibutuhkan rancangan untuk menggambarkan struktur sebuah sistem. UML menyediakan standar pada notasi dan diagram yang bisa digunakan untuk memodelkan suatu sistem. Diagram UML yang digunakan dalam sistem presensi *FreeSent*, yaitu:

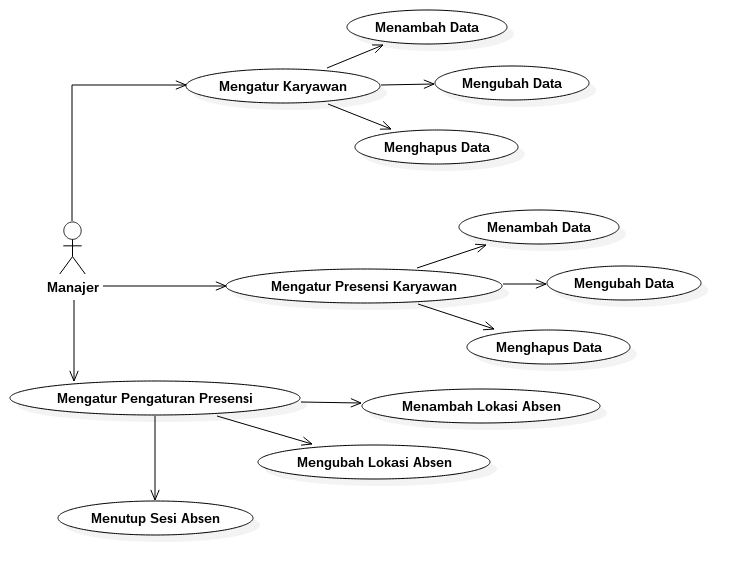
**1. Rancangan Use Case Diagram**

1. *Use Case Diagram Admin* digunakan untuk mengendalikan hal yang sangat penting di dalam sistem. Dalam *Use Case Diagram Admin* ini terdapat 1 aktor yaitu admin. Admin mempunyai akun untuk mendapat hak akses untuk mengelola roles, mengelola pengguna, berita yang muncul di aplikasi android, serta pengaturan website seperti pengaturan logo website dan aplikasi android.



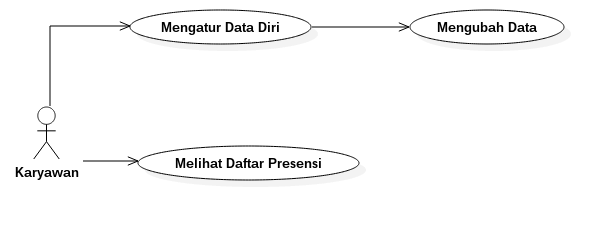
Gambar 3.3 Use Case Admin

1. *Use Case Diagram Manajer* digunakan untuk menjelaskan bagaimana manajer berperan dalam pengelolaan karyawan yang dimiliki oleh manajer itu sendiri. Dalam *Use Case Diagram Manajer* ini terdapat 1 aktor yaitu manajer. Manajer mempunyai akun untuk mendapat hak akses untuk mengelola akun karyawan, mengatur presensi dari karyawannya, dan mengatur pengaturan presensi bagi karyawan nya seperti membuka sesi, menambah koordinat lokasi, dan mengubah lokasi koordinat.



Gambar 3.4 Use Case Manajer

1. *Use Case Diagram Karyawan* digunakan untuk menunjukkan bagaimana karyawan dapat berinteraksi dengan website juga tidak hanya dengan aplikasi android. Dalam *Use Case Diagram Karyawan* ini terdapat 1 aktor yaitu manajer. Manajer mempunyai akun untuk mendapat hak akses untuk mengatur data dirinya, kemudian dapat melihat rekapan atau daftar presensi dari dirinya sendiri.



Gambar 3.5 Use Case Karyawan

**2. Rancangan Activity Diagram**

1. *Activity Diagram Karyawan*

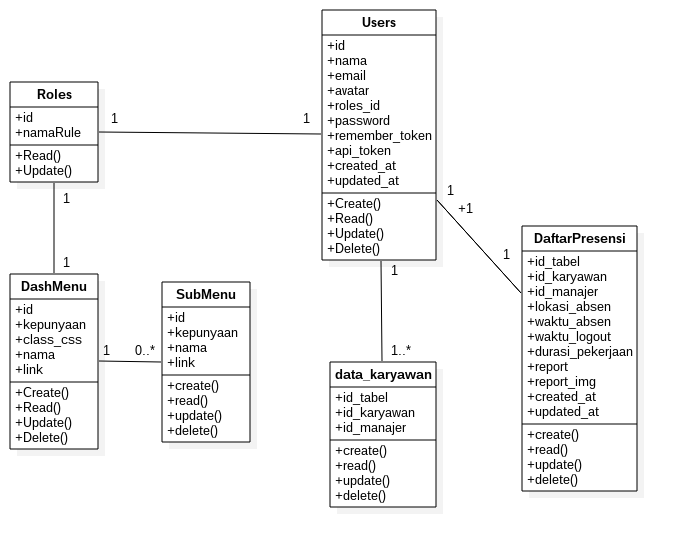
*Activity Diagram Karyawan* menggambarkan aliran kerja atau aktivitas *user* dalam sistem ini dalam hal ini dalam *website* dan aplikasi android. Diagram ini menggambarkan bagaimana alur awal karyawan, aktivitas apa saja yang ada, dan bagaimana akhir dari alur karyawan.

Gambar 3.5 Activity Diagram *Karyawan*

* 1. **Struktur Navigasi**
  2. **Perancangan Database**

**1. Rancangan Class Diagram**

Dalam perancangan database, hal yang pertama penulis pikirkan adalah struktur dan relasi dari database, karena itu yang akan menentukan bagaimana sistem dapat dibentuk. *Class Diagram* menggambarkan kelas yang ada pada sebuah sistem. Masing-masing kelas terdapat atribut dan metoda. Atribut akan menggambarkan keadaan suatu sistem, sedangkan metoda digunakan untuk memanipulasi data.



Gambar 3.3 Class Diagram *FreeSent*

**2. Rancangan File MigrasiRancangan**

Dalam perancangan database, penulis menggunakan mysql dengan bantuan laravel-migration, sehingga seandainya akan mengganti database dari mysql ke oracle atau yang lainnya. Dapat dengan mudah menyesuaikan dan tidak harus membuatnya lagi.

Selain tabel, file migrasi juga dapat digunakan untuk mengisi record ke dalam tabel, sehingga data yang harus ada seperti menu navigasi bisa diletakkan secara langsung.

Namun dalam pembuatannya penulis harus melakukan pengkodean dalam bahasa laravel. Setiap tabel dan atribut nya akan dimasukkan ke dalam kode laravel, yang disebut schema.

Untuk membuat schema dan model dengan otomatis dapat menggunakan “php artisan make:model ‘nama\_tabel’ -m”. Kemudian terdapat file migration yang diciptakan secara otomatis di bagian database->migrations. Untuk memasukkan hasil pengkodean ke database dapat menggunakan command “php artisan migrate”, jika terdapat kesalahan dapat menggunakan command “php artisan migrate:rollback” yang akan mengosongkan database.

A. Tabel users

|  |
| --- |
| Schema::create('users', function (Blueprint $table) {  $table->increments('id');  $table->string('nama');  $table->string('email')->unique();  $table->unsignedInteger('roles\_id')->nullable();  $table->string('avatar');  $table->string('password');  $table->string('api\_token');  $table->rememberToken();  $table->timestamps();  }); |

B. Tabel roles

|  |
| --- |
| Schema::create('roles', function(Blueprint $kolom) {  $kolom->increments('id');  $kolom->string('namaRule');  }); |

C. Relasi antara user dan roles

|  |
| --- |
| Schema::table('users', function(Blueprint $kolom){  $kolom->foreign('roles\_id')->references('id')->on('roles')->onDelete('cascade')->onUpdate('cascade');  }); |

D. Tabel dashmenu

|  |
| --- |
| Schema::create('dashmenu', function (Blueprint $table) {  $table->increments('id');  $table->unsignedInteger('kepunyaan')->nullable();  $table->string('class\_css');  $table->string('nama');  $table->string('link');  }); |

E. Tabel submenu

|  |
| --- |
| Schema::create('submenu', function (Blueprint $table) {  $table->increments('id');  $table->unsignedInteger('kepunyaan')->nullable();  $table->string('nama');  $table->string('link');  }); |

F. Relasi antara dashmenu dan submenu

|  |
| --- |
| Schema::table('submenu', function(Blueprint $kolom){  $kolom->foreign('kepunyaan')->references('id')->on('dashmenu')->onDelete('cascade')->onUpdate('cascade');  }); |

G. Tabel setting\_situses

|  |
| --- |
| Schema::create('setting\_situses', function (Blueprint $table) {  $table->increments('id');  $table->string('namaSitus');  $table->string('slogan');  $table->string('favicon');  $table->string('logo');  $table->string('alamatSitus');  $table->timestamps();  }); |

H. Tabel setting\_situses

|  |
| --- |
| Schema::create('setting\_situses', function (Blueprint $table) {  $table->increments('id');  $table->string('namaSitus');  $table->string('slogan');  $table->string('favicon');  $table->string('logo');  $table->string('alamatSitus');  $table->timestamps();  }); |

I. Tabel data\_karyawan

|  |
| --- |
| Schema::create('data\_karyawan', function (Blueprint $table) {  $table->unsignedInteger('id\_karyawan')->nullable();  $table->unsignedInteger('id\_manajer')->nullable();  $table->string('nama\_karyawan');  }); |

J. Relasi data\_karyawan dan users

|  |
| --- |
| Schema::table('data\_karyawan', function(Blueprint $kolom){  $kolom->foreign('id\_karyawan')->references('id')->on('users')->onDelete('cascade')->onUpdate('cascade');  $kolom->foreign('id\_manajer')->references('id')->on('users')->onDelete('cascade')->onUpdate('cascade');  }); |

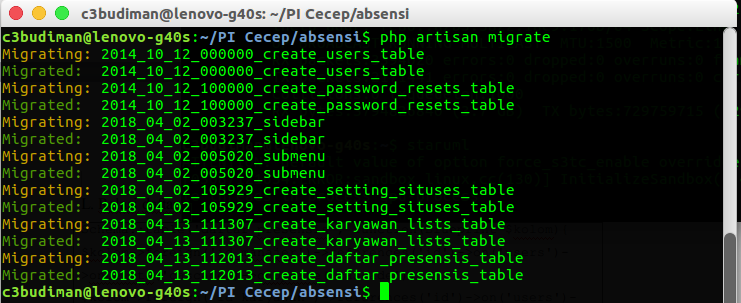
K. Tabel Daftar\_presensis

|  |
| --- |
| Schema::create('daftar\_presensis', function (Blueprint $table) {  $table->increments('id');  $table->unsignedInteger('id\_manajer')->nullable();  $table->unsignedInteger('id\_karyawan')->nullable();  $table->string('lokasi\_absen');  $table->timestamp('waktu\_absen')->nullable();  $table->timestamp('waktu\_logout')->nullable();  $table->time('durasi\_pekerjaan')->nullable();  $table->string('report');  $table->string('report\_img');  $table->timestamps();  }); |

L. Relasi daftar\_presensis dan user

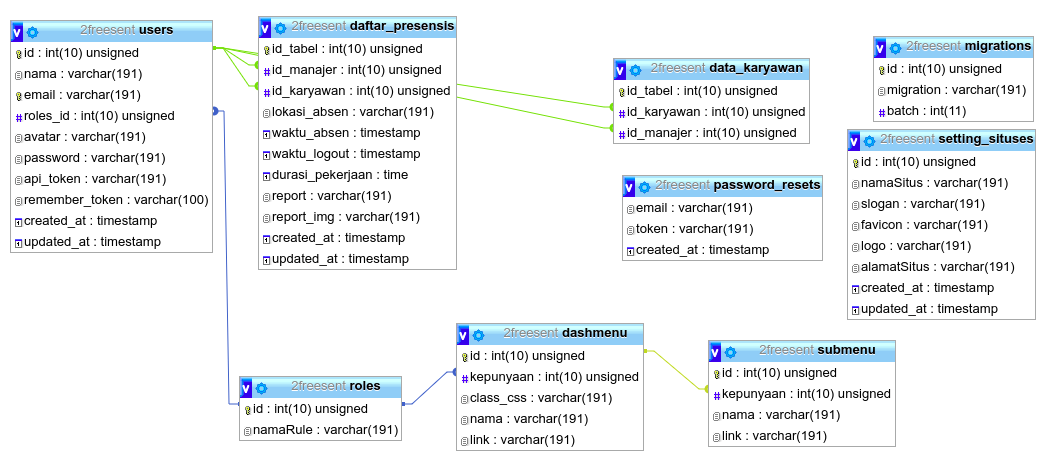
|  |
| --- |
| Schema::table('daftar\_presensis', function(Blueprint $kolom){  $kolom->foreign('id\_manajer')->references('id')->on('users')->onDelete('cascade')->onUpdate('cascade');  $kolom->foreign('id\_karyawan')->references('id')->on('users')->onDelete('cascade')->onUpdate('cascade');  }); |

Menjalankan migrasi :



Gambar 3.4 Proses Migrasi

Hasil migrasi :



Gambar 3.5 Database *Freesent*

1. **Tabel Database**

Berdasarkan dari rancangan yang telah dibuat, *Platform Freesent* membutuhkan *database* berikut, yakni:

1. Tabel users

Tabel users, digunakan untuk menyimpan data-data pengguna dari situs, pengguna termasuk admin, manajer, dan karyawan. Pada tabel ini terdapat 10 *field* yang terdiri dari id, nama, email, roles\_id, avatar, password, remember\_token, api\_token, created\_at, dan updated\_at. Strukturnya dijabarkan pada Tabel 3.1

Tabel 3.1 Tabel users

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Tipe Data** | **Panjang** | **Keterangan** |
| id | Integer | 10 | Primary Key |
| nama | Varchar | 191 |  |
| email | Varchar | 191 |  |
| roles\_id | int | 10 |  |
| avatar | Varchar | 191 |  |
| password | Varchar | 191 |  |
| api\_token | Varchar | 191 |  |
| remember\_token | Varchar | 100 |  |
| created\_at | timestamp |  |  |
| updated\_at | timestamp |  |  |

1. Tabel roles

Tabel roles, digunakan untuk menyimpan data tingkatan user/roles dari user. Tabel roles tidak berguna jika tidak digunakan oleh tabel user. Pada tabel ini terdapat 2 *field* yang terdiri dari id, dan namaRule. Strukturnya dijabarkan pada Tabel 3.2

Tabel 3.2 Tabel roles

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Tipe Data** | **Panjang** | **Keterangan** |
| id | Integer | 10 | Primary Key |
| namaRule | Varchar | 191 |  |

1. Tabel dashmenu

Tabel dahsmenu, digunakan untuk menyimpan struktur navigasi pada semua tingkatan user. Pada tabel ini terdapat 5 *field* yang terdiri dari id, kepunyaan, class\_css, nama, dan link. Strukturnya dijabarkan pada Tabel 3.3

Tabel 3.3 Tabel roles

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Tipe Data** | **Panjang** | **Keterangan** |
| id | Integer | 11 | Primary Key |
| kepunyaan | Integer | 11 |  |
| class\_css | Varchar | 191 |  |
| nama | Varchar | 191 |  |
| link | Varchar | 191 |  |

1. Tabel submenu

Tabel submenu, digunakan untuk menyimpan struktur navigasi yang detail dari dashmenu pada semua tingkatan user. Pada tabel ini terdapat 4 *field* yang terdiri dari id, kepunyaan, nama dan link. Strukturnya dijabarkan pada Tabel 3.4

Tabel 3.4 Tabel roles

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Tipe Data** | **Panjang** | **Keterangan** |
| id | Integer | 11 | Primary Key |
| kepunyaan | Integer | 11 |  |
| nama | Varchar | 191 |  |
| link | Varchar | 191 |  |

1. Tabel data\_karyawan

Tabel data\_karyawan, digunakan untuk menjelaskan hubungan antara manajer dan karyawan. Pada tabel ini terdapat 2 *field* yang terdiri dari id, id\_manajer, dan id\_karyawan. Strukturnya dijabarkan pada Tabel 3.5

Tabel 3.5 Tabel data\_karyawan

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Tipe Data** | **Panjang** | **Keterangan** |
| id\_tabel | Integer | 11 | Primary Key |
| id\_manajer | Integer | 11 |  |
| id\_karyawan | Integer | 11 |  |

1. Tabel daftar\_presensi

Tabel daftar\_presensi, digunakan untuk menyimpan data kehadiran dan rekapan jam kerja dari para karyawan. Pada tabel ini terdapat 11 *field* yang terdiri dari id, id\_manajer, id\_karyawan, lokasi\_absen, waktu\_absen, waktu\_logout, durasi\_pekerjaan, report, report\_img, created\_at dan updated\_at. Strukturnya dijabarkan pada Tabel 3.1

Tabel 3.1 Tabel users

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Tipe Data** | **Panjang** | **Keterangan** |
| Id\_tabel | Integer | 10 | Primary Key |
| id\_manajer | Integer | 10 |  |
| id\_karyawan | Integer | 10 |  |
| lokasi\_absen | Varchar | 191 |  |
| waktu\_absen | Timestamp |  |  |
| waktu\_logout | Timestamp |  |  |
| durasi\_pekerjaan | Time | 100 |  |
| report | Varchar | 191 |  |
| report\_img | Varchar | 191 |  |
| created\_at | Timestamp |  |  |
| updated\_at | Timestamp |  |  |