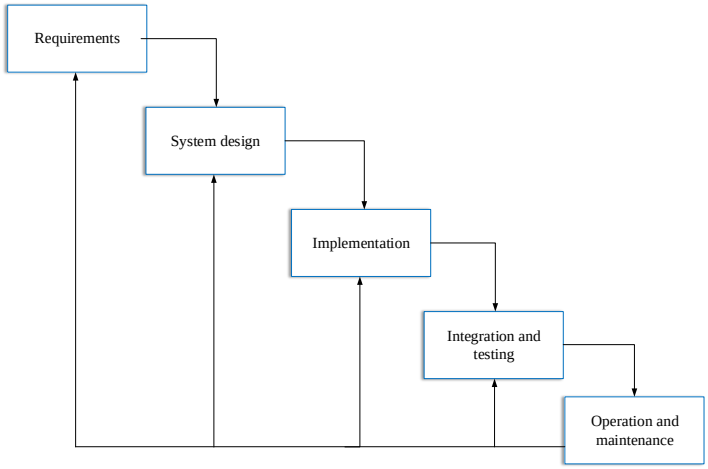
BAB III
METODOLOGI PENELITIAN

3.1 Arsitektur Aplikasi

Pada bab ini akan digambarkan rancangan aplikasi administrasi personalia. Rancangan aplikasi administrasi personalia dirancang menggunakan metode *waterfall.* Metode ini memungkinkan untuk departementalisasi dan kontrol. proses pengembangan model fase *one by one*, sehingga meminimalis kesalahan yang mungkin akan terjadi. Pengembangan bergerak dari konsep, yaitu melalui desain, implementasi, pengujian, instalasi, penyelesaian masalah, dan berakhir di operasi dan pemeliharaan. Tahapan metode *waterfall* dapat dilihat pada gambar 3.1.



Gambar 3.1 Tahapan metode *waterfall*

Dalam pengembangannya metode *waterfall* memiliki beberapa tahapan yang berurut yaitu: *requirement* (analisis kebutuhan), *design system* (desain sistem), *Coding* (pengkodean) *& Testing* (pengujian), Penerapan Program, pemeliharaan. Tahapan tahapan dari metode *waterfall* adalah sebagai berikut :

1. *Requirement* Analisis

   Pada tahap ini dilakukan analisa kebutuhan yang bertujuan untuk memahami kebutuhan pengguna dan batasan perangkat lunak tersebut. Informasi ini biasanya dapat diperoleh melalui wawancara, diskusi atau survei langsung. Informasi dianalisis untuk mendapatkan data yang

dibutuhkan oleh pengguna. Analisa kebutuhan akan dijelaskan pada sub bab 3.3.

2. *System design*

   System design digunakan untuk mengubah analisa kebutuhan menjadi representasi ke dalam bentuk blueprint *software* sebelum implementasi dimulai. Tahap *system design* menyangkut perancangan sistem dimana kita akana memberikan solusi dari masalah yang muncul pada tahap analisis. Perancangan menggunakan *Use case* diagram, *Activity* diagram, *ER* diagram, *Sequence* diagram, *Class* diagram dan perancangan antarmuka. Tahap *system design* akan dijelaskan pada sub bab 3.4.

3. *Implementation*

   Pada tahap ini, keseluruhan desain diubah menjadi kode-kode program. kode program yang dihasilkan masih berupa modul-modul yang selanjutnya akan di integrasikan menjadi program yang lengkap untuk meyakinkan bahwa persyaratan perangkat lunak telah dipenuhi. Tahap *implemetation* akan dijelaskan pada sub bab 4.1.

4. *Integration* & *Testing*

   Seluruh unit yang dikembangkan pada tahap implementasi akan diintegrasikan ke dalam sistem. Setelah integrasi, seluruh sistem diuji untuk mengecek setiap kegagalan maupun kesalahan. Tahap *testing* akan dijelaskan pada sub bab 4.2.

5. *Operation* & *Maintenance*

   Pada tahap ini perangkat lunak yang sudah jadi dijalankan serta dilakukan pemeliharaan. Pemeliharaan termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada tahap *testing* karena perangakt lunak harus menyesuaikan dengan sistem operasi dan perangkat baru. *Operation* dan *maintenance* dilakukan untuk perkembangan fungsionalitas.

3.2 Peralatan pendukung

Peralatan pendukung dalam pembuatan aplikasi administrasi personalia berbasis *android* di PT Indocement Tunggal Prakasa Tbk. terdiri dari perangkat

keras (*hardware*) dan perangkat lunak (*software*) untuk mendukung berjalannya perancangan dan pembuatan.

A. Perangkat keras

Perangkat keras yang dibutuhkan dalam merancang dan membuat *bussiness logic* dari perencanaan aplikasi administrasi personalia:

1. *Processor Intel® Core™ i5-5200U CPU @ 2.20GHz × 4*
2. *Memory 8GB RAM*
3. *Hardisk 500 GB*
4. *Monitor, Keyboard,* dan *Mouse*
5. *Smartphone android*

B. Perangkat lunak

Perangkat lunak yang dibutuhkan dalam merancang dan membuat *bussines logic* dari perancanaan aplikasi administrasi personalia :

1. Sistem Operasi *Ubuntu* 16.04 LTS
2. *NetBeans* 8.1
3. *Android Studio* 3.0.0
4. *Postman*
5. *Database Server : MySQL*
6. *Web server tomcat*

3.3 Analisis kebutuhan

Analisis kebutuhan didapat dari observasi dan wawancara. Berikut merupakan analisis kebutuhan pada aplikasi :

1. *User* melakukan registrasi untuk dapat membuat akun baru.
2. *User* melakukan *login* untuk dapat menggunakan aplikasi.
3. *User* dapat melihat informasi terbaru.
4. *User* dapat memberi komentar pada informasi.
5. *User* dapat memberikan *feedback* aplikasi
6. *User* dapat mengirimkan keluhan absensi.
7. *User* dapat mengirimkan keluhan cuti.

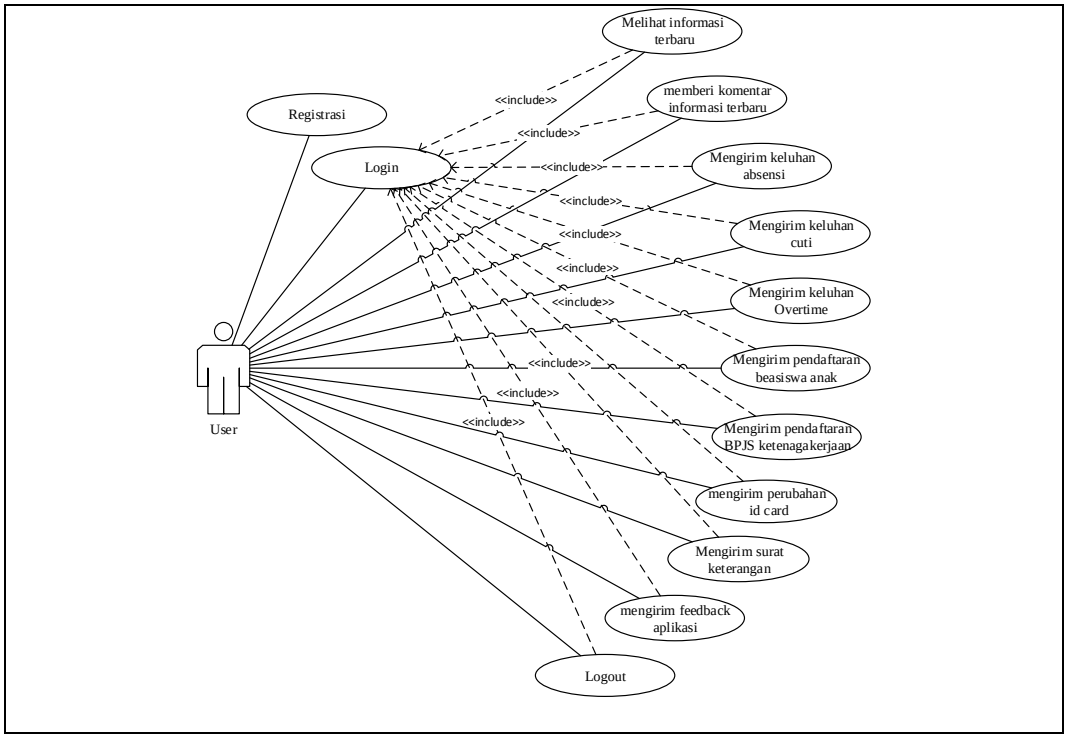
8. *User* dapat mengirimkan keluhan *overtime*.
9. *User* dapat mengirimkan pendaftaran beasiswa.
10. *User* dapat mengirimkan pendaftaran BPJS.
11. *User* dapat mengirimkan permintaan perubahan *id card*.
12. *User* dapat mengirimkan surat keterangan.
13. *User* dapat keluar dari aplikasi.

3.4 Rancangan

Dalam perancangan *logic* aplikasi ini menggunakan metode *United Markup Language* (UML). Model UML yang digunakan dalam pengembangan aplikasi administrasi personalia berbasis *android* yaitu *use case* diagram, *sequence* diagram, *activity* diagram dan *class* diagram.

3.4.1 Rancangan use case diagram

Berdasarkan analisis kebutuhan pada sub bab 3.2, maka dibuatlah *use case* diagram untuk membantu perancangan *logic* pada aplikasi. Pada aplikasi android terdapat 1 aktor yaitu *user*. Gambaran *use case* diagram dapat dilihat pada gambar 3.2.



Gambar 3.2 *Use case* diagram aplikasi

Berikut penjelasan tentang *use case* diagram pada gambar 3.2:

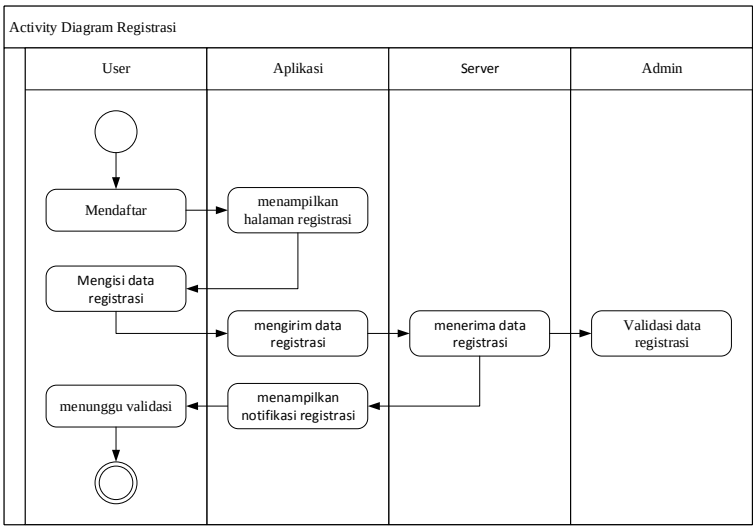
a. Registrasi : memungkinkan *user* baru untuk dapat melakukan registrasi agar dapat menggunakan aplikasi android.
b. *Login* : memungkinkan *user* untuk dapat menggunakan aplikasi android.
c. Melihat informasi terbaru : memungkinkan *user* untuk dapat melihat informasi terbaru.
d. Mengirim keluhan absensi : memungkinkan *user* untuk dapat mengirim keluhan mengenai absensi.
e. Mengirim keluhan cuti : memungkinkan *user* untuk dapat mengirim keluhan mengenai cuti.
f. Mengirim keluhan *overtime* : memungkinkan *user* untuk dapat mengirim keluhan mengenai *overtime.*
g. Mengirim pendaftaran beasiswa anak : memungkinkan *user* untuk dapat mengirimkan pendaftaran untuk beasiswa anak.
h. Mengirim pendaftaran BPJS : memungkinkan *user* untuk dapat mengirimkan pendaftaran untuk BPJS.
i. Mengirim perubahan *id card* : memungkinkan *user* untuk dapat mengirimkan permintaan perubahan *id card*.
j. Mengirim surat keterangan : memungkinkan *user* untuk dapat mengirimkan surat keterangan.
k. Mengirim *feedback* aplikasi : memungkinkan *user* untuk dapat mengirimkan *feedback* mengenai aplikasi administrasi personalia.
l. *Logout* : memungkinkan *user* untuk tidak lagi menggunakan fitur pada aplikasi administrasi personalia.

3.4.2 Rancangan activity diagram

Berdasarkan *use case* diagram pada gambar 3.2, maka diperoleh *activity* diagram berdasarkan aktor yang terlibat dalam *use case* diagram. *Activity* diagram dibagi menjadi 13 bagian yaitu :

A. Activity diagram registrasi

Rancangan a*ctivity* diagram registrasi dibuat berdasarkan *case* registrasi pada gambar 3.2. *User* akan memasukkan data nomor induk karyawan (NIK), nama *user*, nomor telepon, alamat *email*, *plant* pekerjaan, departemen pekerjaan, dan *password*. Rancangan *activity* diagram registrasi dapat dilihat pada gambar 3.3.



Gambar 3.3 *Activity* diagram registrasi

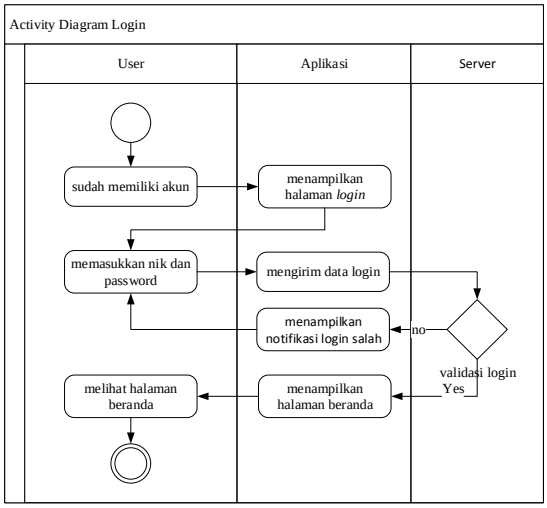
Berikut penjelasan *activity* diagram registrasi pada gambar 3.3

1. Alur dari kegiatan registrasi yang dilakukan oleh *user* baru yaitu : masuk ke halaman registrasi untuk mendaftarkan akun baru.
2. Aplikasi akan menampilkan halaman registrasi.
3. *User* akan memasukkan data-data yang dibutuhkan untuk registrasi akun.
4. Jika data sudah lengkap, aplikasi akan mengirimkan data registrasi ke *server*. Jika ada data kosong, aplikasi akan menampilkan informasi pengisian data tidak lengkap.
5. *Server* menerima data yang dikirimkan *user*.
6. *User* dapat menggunakan akun setelah validasi selesai dilakukan oleh admin.

B. Activity diagram login

Rancangan *activity* diagram *login* dibuat berdasarkan *case login* pada gambar 3.2. Untuk melakukan *login*, *user* akan menginputkan nomor induk

karyawan dan *password*. Rancangan *activity* diagram *login* dapat dilihat pada gambar 3.4



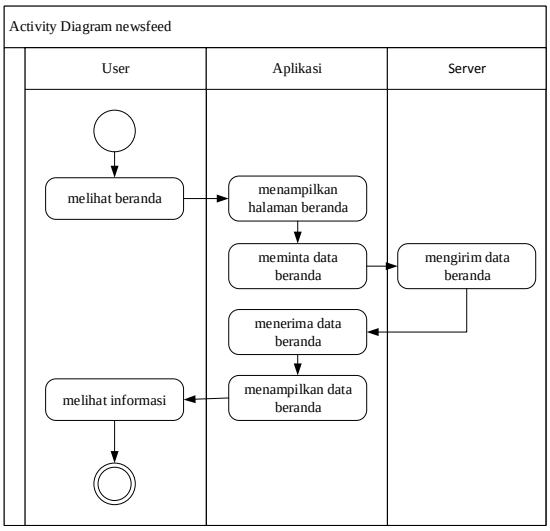
Gambar 3.4 *Activity* diagram *login*

Berikut penjelasan *activity* diagram *login* pada gambar 3.4:

1. Alur dari kegiatan *login* yang dilakukan oleh *user* yaitu membuka halaman *login* pada aplikasi.
2. *User* mengisi nik dan *password*, kemudian aplikasi akan mengirimkan data *login* ke *server* untuk melakukan validasi nik dan *password*. Ketika data yang dikirimkan benar, aplikasi akan menampilkan halaman beranda informasi. Namun apabila data yang dikirimkan salah, aplikasi akan menampilkan informasi bahwa data *login* salah.

C. Activity diagram melihat informasi terbaru

Rancangan *activity* diagram melihat informasi terbaru dibuat berdasarkan *case* melihat informasi terbaru pada gambar 3.2. Rancangan *activity* diagram melihat informasi terbaru dapat dilihat pada gambar 3.5.



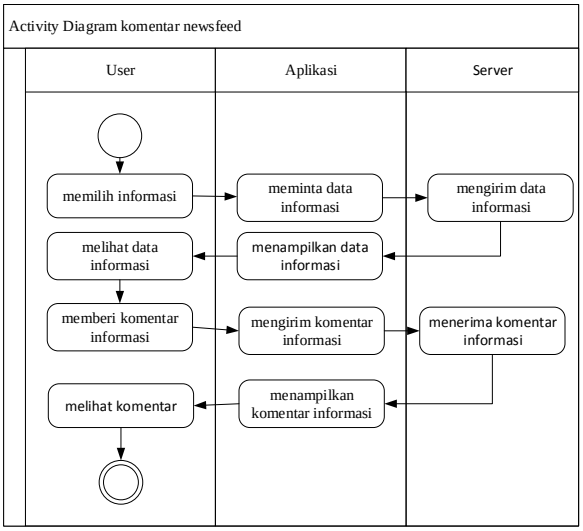
Gambar 3.5 *Activity* diagram melihat informasi terbaru

Berikut penjelasan *activity* diagram melihat informasi terbaru pada gambar 3.5 :

1. Halaman informasi terbaru ditampilkan setelah *user* melakukan *login* pada aplikasi.
2. Aplikasi menampilkan halaman informasi beranda dan mengirimkan permintaan data informasi dari *server*.
3. *Server* mengirimkan data informasi ke aplikasi.
4. Aplikasi menerima data informasi beranda dan menampilkan pada halaman beranda.
5. *User* melihat informasi beranda.

D. Activity diagram memberi komentar informasi terbaru

Rancangan *activity* diagram memberi komentar informasi terbaru dibuat berdasarkan *case* memberi komentar informasi terbaru pada gambar 3.2. Rancangan *activity* diagram memberi komentar informasi terbaru dapat dilihat pada gambar 3.6.



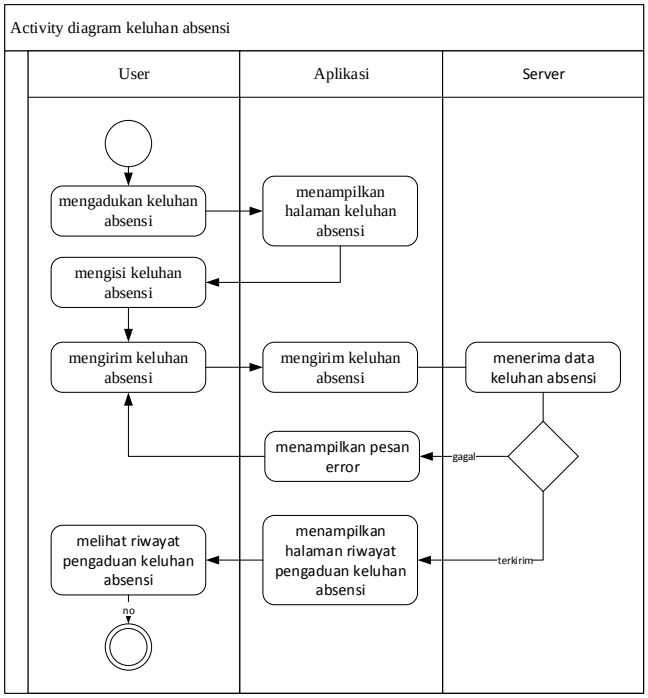
Gambar 3.6 *Activity* diagram memberi komentar terbaru

Berikut penjelasan *activity* diagram memberi komentar terbaru pada gambar 3.6 :

1. Alur kegiatan memberi komentar informasi dimulai dari *user* memilih informasi yang akan diberi komentar.
2. Setelah informasi dipilih, aplikasi akan meminta data informasi ke *server*. Aplikasi menampilkan data informasi yang dikirimkan oleh *server* dan menampilkan komentar-komentar yang diberikan oleh *user* lain.
3. *User* memberikan komentar pada informasi yang ditampilkan oleh aplikasi.
4. Setelah *user* menekan tombol kirim, aplikasi akan mengirimkan data komentar ke *server*. *Server* menerima komentar yang dikirimkan.
5. Aplikasi memuat ulang halaman informasi dan menampilkan kembali komentar yang diberikan pada informasi tersebut.

E. Activity diagram mengirim keluhan absensi

Rancangan *activity* diagram mengirim keluhan absensi dibuat berdasarkan *case* mengirim keluhan absensi pada gambar 3.2. Rancangan *activity* diagram mengirim keluhan absensi dapat dilihat pada gambar 3.7.



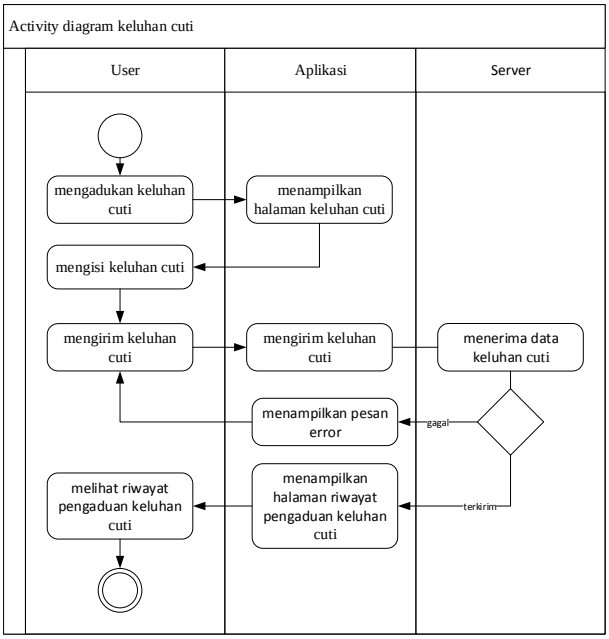
Gambar 3.7 *Activity* diagram mengirim keluhan absensi

Berikut penjelasan *activity* diagram mengirim keluhan absensi pada gambar 3.7 :

1. Alur kegiatan mengirim keluhan absensi dimulai dari memilih menu keluhan absensi pada aplikasi android. Aplikasi akan menampilkan halaman pengaduan keluhan absensi.
2. *User* mengisi keluhan mengenai absensi pada aplikasi.
3. Aplikasi mengirimkan data keluhan absensi ke *server*. Jika pengiriman data berhasil, aplikasi akan memberikan informasi pengiriman data berhasil dan aplikasi menampilkan halaman riwayat pengiriman keluhan absensi. Apabila pengiriman gagal, aplikasi akan memberikan informasi pengiriman data gagal dan *user* dapat mengirim kembali keluhan absensi.

F. Activity diagram mengirim keluhan cuti

Rancangan *activity* diagram mengirim keluhan cuti dibuat berdasarkan *case* mengirim keluhan cuti pada gambar 3.2. Rancangan *activity* diagram mengirim keluhan cuti dapat dilihat pada gambar 3.8.



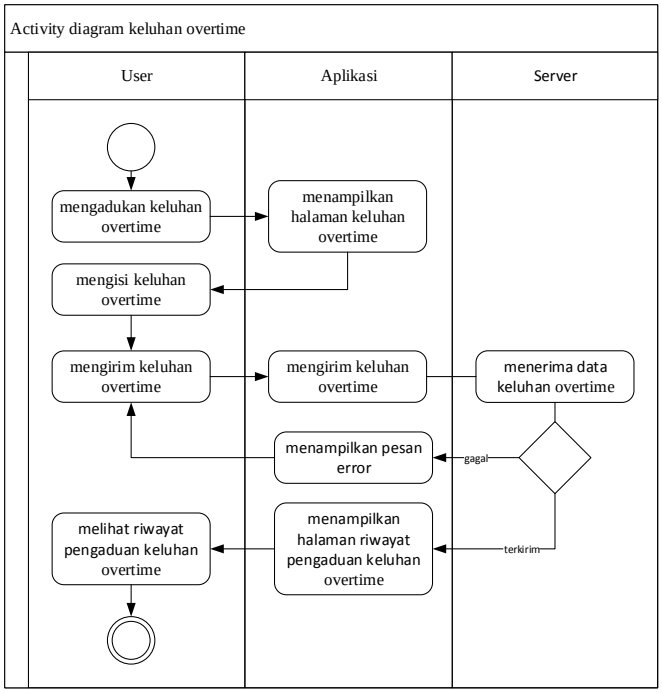
Gambar 3.8 *Activity* diagram mengirim keluhan cuti

Berikut penjelasan *activity* diagram mengirim keluhan cuti pada gambar 3.8:

1. Alur kegiatan mengirim keluhan cuti dimulai dari memilih menu keluhan cuti pada aplikasi android. Aplikasi akan menampilkan halaman pengaduan keluhan cuti.
2. *User* memasukkan keluhan mengenai cuti pada aplikasi.
3. Aplikasi mengirimkan data keluhan cuti ke *server*. Jika pengiriman data berhasil, aplikasi akan memberikan informasi pengiriman data berhasil dan aplikasi menampilkan halaman riwayat pengiriman keluhan cuti. Apabila pengiriman gagal, aplikasi akan memberikan informasi pengiriman data gagal dan *user* dapat mengirim kembali keluhan cuti.

G. Activity diagram mengirim keluhan overtime

Rancangan *activity* diagram mengirim keluhan *overtime* dibuat berdasarkan *case* mengirim keluhan *overtime* pada gambar 3.2. Rancangan *activity* diagram mengirim keluhan *overtime* dapat dilihat pada gambar 3.8.



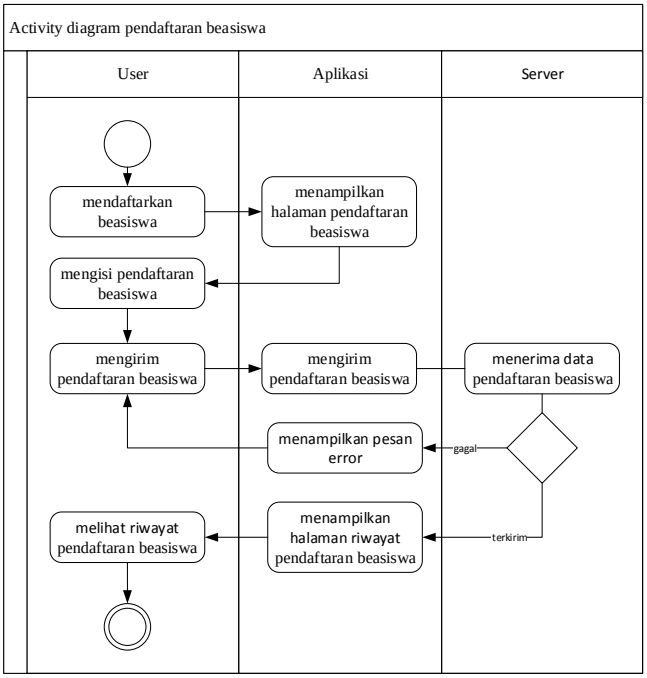
Gambar 3.9 *Activity* diagram mengirim keluhan *overtime*

Berikut penjelasan *activity* diagram mengirim keluhan *overtime* pada gambar 3.9 :

1. Alur kegiatan mengirim keluhan *overtime* dimulai dari memilih menu keluhan *overtime* pada aplikasi android. Aplikasi akan menampilkan halaman pengaduan keluhan *overtime*.
2. *User* memasukkan keluhan mengenai *overtime* pada aplikasi.
3. Aplikasi mengirimkan data keluhan *overtime* ke *server*. Jika pengiriman data berhasil, aplikasi akan memberikan informasi pengiriman data berhasil dan aplikasi menampilkan halaman riwayat pengiriman keluhan *overtime*. Apabila pengiriman gagal, aplikasi akan memberikan informasi pengiriman data gagal dan *user* dapat mengirim kembali keluhan *overtime*.

H. Activity diagram mengirim pendaftaran beasiswa anak

Rancangan *activity* diagram mengirim pendaftaran beasiswa anak berdasarkan *case* mengirim pendaftaran beasiswa anak pada gambar 3.2. Rancangan *activity* diagram mengirim pendaftaran beasiswa anak dapat dilihat pada gambar 3.10.



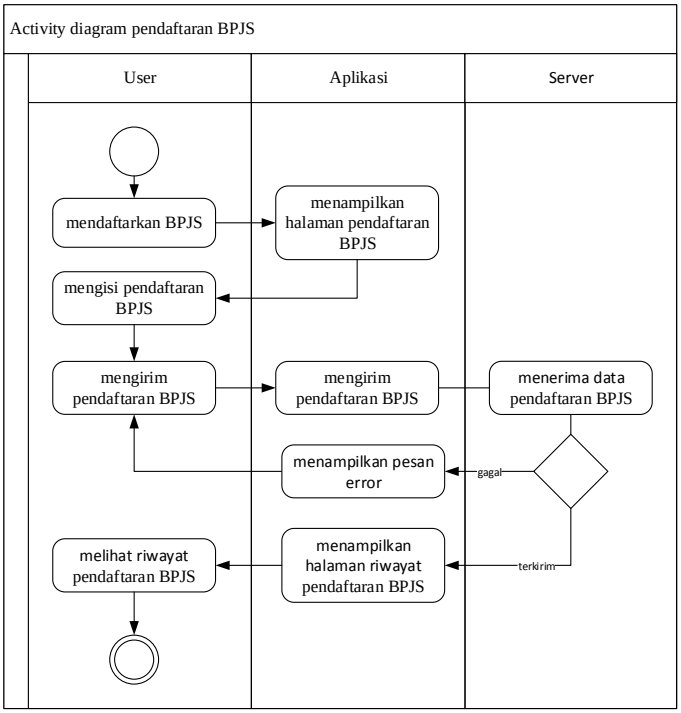
Gambar 3.10 *Activity* diagram mengirim pendafraran beasiswa anak

Berikut penjelasan *activity* diagram mengirim pendaftaran beasiswa anak pada gambar 3.10 :

1. Alur kegiatan mengirim pendaftaran beasiswa anak dimulai dari memilih menu pendaftaran beasiswa anak pada aplikasi android. Aplikasi akan menampilkan halaman pendaftaran beasiswa anak.
2. *User* memasukkan data pendaftaran beasiswa anak pada aplikasi.
3. Aplikasi mengirimkan data pendaftaran beasiswa anak ke *server*. Jika pengiriman data berhasil, aplikasi akan memberikan informasi pengiriman data berhasil dan aplikasi menampilkan halaman riwayat pengiriman pendaftaran beasiswa anak. Apabila pengiriman gagal, aplikasi akan memberikan informasi pengiriman data gagal dan *user* dapat mengirim kembali pendaftaran beasiswa anak.

I. Activity diagram mengirim pendaftaran BPJS

Rancangan *activity* diagram mengirim pendaftaran BPJS dibuat berdasarkan *case* mengirim pendaftaran BPJS pada gambar 3.2. Rancangan *activity* diagram mengirim pendaftaran BPJS dapat dilihat pada gambar 3.11.



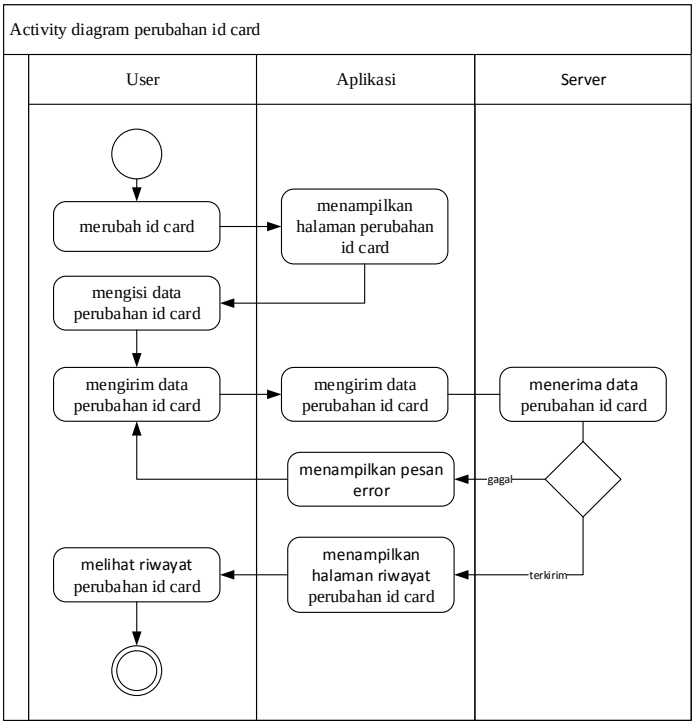
Gambar 3.11 *Activity* diagram mengirim pendaftaran BPJS

Berikut penjelasan *activity* diagram mengirim mengirim pendaftaran BPJS pada gambar 3.11 :

1. Alur kegiatan mengirim pendaftaran BPJS dimulai dari memilih menu pendaftaran BPJS pada aplikasi android. Aplikasi akan menampilkan halaman pengaduan pendaftaran BPJS.
2. *User* memasukkan data mengenai pendaftaran BPJS pada aplikasi.
3. Aplikasi mengirimkan data pendaftaran BPJS ke *server*. Jika pengiriman data berhasil, aplikasi akan memberikan informasi pengiriman data berhasil dan aplikasi menampilkan halaman riwayat pengiriman pendaftaran BPJS. Apabila pengiriman gagal, aplikasi akan memberikan informasi pengiriman data gagal dan *user* dapat mengirim kembali pendaftaran BPJS.

J. Activity diagram mengirim perubahan id card

Rancangan *activity* diagram mengirim perubahan *id card* dibuat berdasarkan *case* mengirim perubahan *id card* pada gambar 3.2. Rancangan *activity* diagram mengirim perubahan *id card* dapat dilihat pada gambar 3.12.



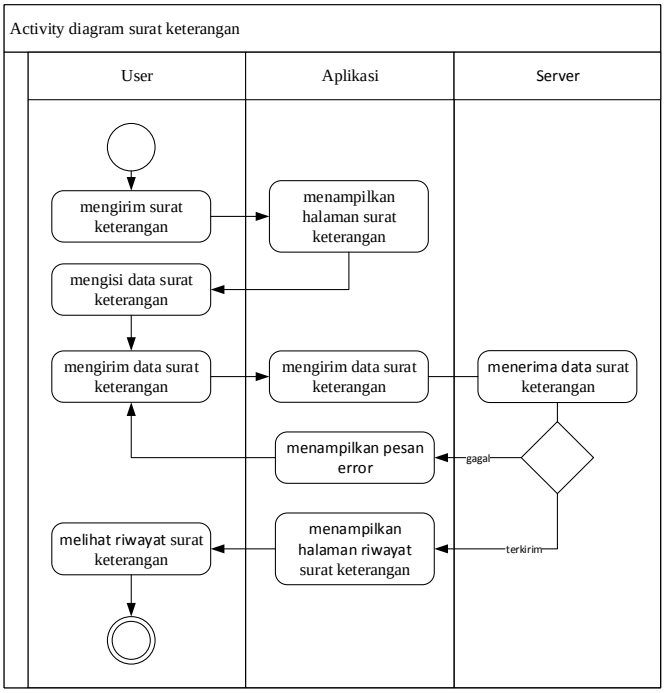
Gambar 3.12 *Activity* diagram mengirim perubahan *id card*

Berikut penjelasan *activity* diagram mengirim perubahan *id card* pada gambar 3.12 :

1. Alur kegiatan mengirim perubahan *id card* dimulai dari memilih menu perubahan *id card* pada aplikasi android. Aplikasi akan menampilkan halaman perubahan *id card*.
2. *User* memasukkan data mengenai perubahan *id card* pada aplikasi.
3. Aplikasi mengirimkan data perubahan *id card* ke *server*. Jika pengiriman data berhasil, aplikasi akan memberikan informasi pengiriman data berhasil dan aplikasi menampilkan halaman riwayat pengiriman perubahan *id card*. Apabila pengiriman gagal, aplikasi akan memberikan informasi pengiriman data gagal dan *user* dapat mengirim kembali perubahan *id card.*

K. Activity diagram mengirim surat keterangan

Rancangan *activity* diagram mengirim surat keterangan dibuat berdasarkan *case* mengirim surat keterangan pada gambar 3.2. Rancangan *activity* diagram mengirim surat keterangan dapat dilihat pada gambar 3.13.



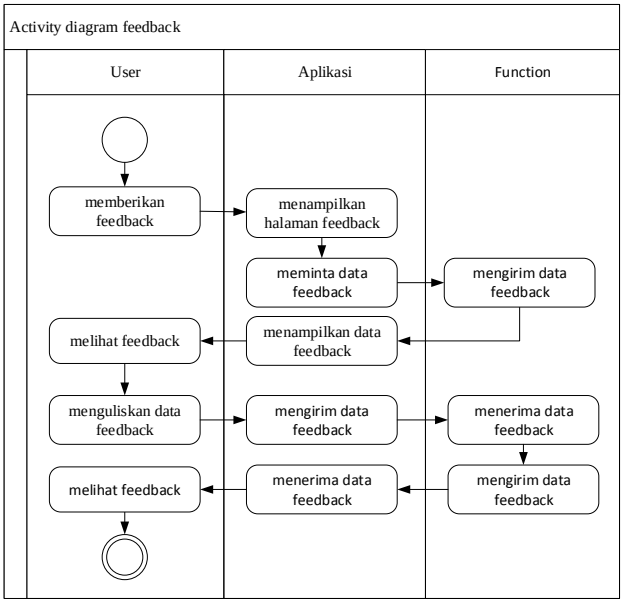
Gambar 3.13 *Activity* diagram mengirim surat keterangan

Berikut penjelasan *activity* diagram mengirim surat keterangan pada gambar 3.13 :

1. Alur kegiatan mengirim surat keterangan dimulai dari memilih menu surat keterangan pada aplikasi android. Aplikasi akan menampilkan halaman surat keterangan.
2. *User* memasukkan data mengenai surat keterangan pada aplikasi.
3. Aplikasi mengirimkan data surat keterangan ke *server*. Jika pengiriman data berhasil, aplikasi akan memberikan informasi pengiriman data berhasil dan aplikasi menampilkan halaman riwayat pengiriman surat keterangan. Apabila pengiriman gagal, aplikasi akan memberikan informasi pengiriman data gagal dan *user* dapat mengirim kembali surat keterangan.

L. Activity diagram mengirim feedback aplikasi

Rancangan *activity* diagram mengirim *feedback* aplikasi dibuat berdasarkan *case* mengirim *feedback* aplikasi pada gambar 3.2. Rancangan *activity* diagram mengirim *feedback* aplikasi dapat dilihat pada gambar 3.14.



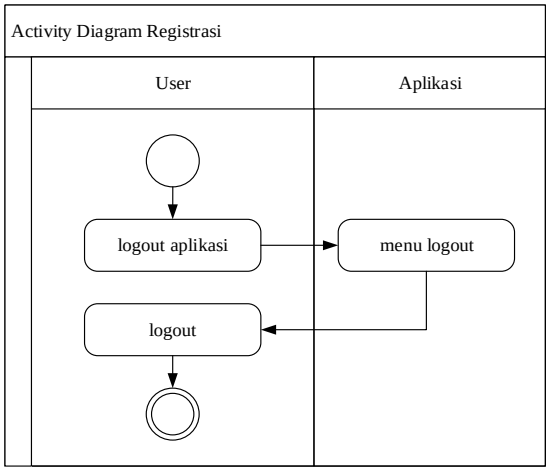
Gambar 3.14 *Activity* diagram mengirim *feedback* aplikasi

Berikut penjelasan *activity* diagram mengirim *feedback* aplikasi pada gambar 3.14 :

1. Alur kegiatan mengirim *feedback* aplikasi dimulai dari memilih menu *feedback* aplikasi pada aplikasi android. Aplikasi akan menampilkan halaman *feedback* aplikasi.
2. Aplikasi mengirimkan permintaan data *feedback* yang dikirim *user* lain, kemudian *server* mengirimkan data *feedback* ke aplikasi.
3. Aplikasi menampilkan *feedback* yang dikirim oleh *user* lain.
4. *User* memasukan *feedback* pada aplikasi dan aplikasi mengirimkan data *feedback* ke *server*.
5. Setelah *server* menerima data *feedback* yang dikirim, *server* akan mengirimkan kembali data-data *feedback* yang sudah dikirimkan. Kemudian aplikasi akan memuat ulang data *feedback* pada halaman *feedback.*

M. Activity diagram logout

Rancangan *activity* diagram *logout* dibuat berdasarkan *case logout* pada gambar 3.2. Rancangan *activity* diagram *logout* dapat dilihat pada gambar 3.15.



Gambar 3.15 *Activity* diagram *logout*

Berikut penjelasan *activity* diagram *logout* pada gambar 3.15:

1. Alur kegiatan *logout* dimulai dari *user* memilih menu *logout* pada aplikasi.
2. Setelah *user* melakukan *logout* aplikasi akan menampilkan halaman *login.*

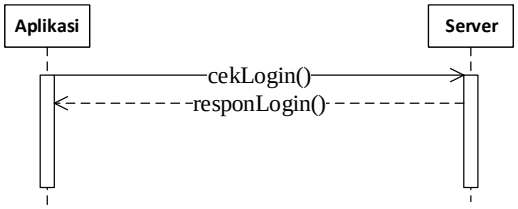
3.4.3 Rancangan sequence diagram

Berdasarkan aktivitas pada *activity* diagram pada sub bab 3.3.2 maka dirancanglah *sequence* diagram. *Sequence* diagram menggambarkan proses yang dilakukan oleh aplikasi dengan *server*. Aplikasi dan *server* melakukan proses mengirim dan menerima data. Pada aplikasi administrasi personalia terdapat 2 proses yaitu *request* dan *response*. Proses *request* yaitu aplikasi melakukan pengiriman dan permintaan data ke *server*. Proses *response* yaitu respon yang dikirimkan *server* terhadap pengiriman dan permintaan data dari aplikasi. Dalam proses *request* dan *response* data antara aplikasi dan *server*, bentuk data yang digunakan adalah *JSON*.

Rancangan *sequence* diagram dibagi menjadi 12 bagian yaitu sebagai berikut.

A. Sequence diagram login

Sequence diagram *login* menggambarkan rancangan arsitektur untuk proses *login*. *Sequence* diagram *login* dibuat berdasarkan *activity* diagram *login* pada gambar 3.4. Rancangan *sequence* diagram *login* dapat dilihat pada gambar 3.16.



Gambar 3.16 *Sequence* diagram proses *login*

Berikut penjelasan *sequence* diagram *login* pada gambar 3.16 :

1. Dalam proses *login*, aplikasi mengirimkan data nik dan password sebagai validasi *user*. Bentuk data *JSON* nik dan password dapat dilihat pada *list* 3.1.

```
{
   "nik": "int",
   "password": "string"
}
```

***List* 3.1** Data *JSON* nik dan *password*

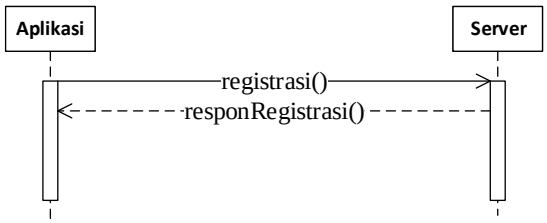
2. Aplikasi mengirimkan *request* cekLogin() data nik dan *password user* ke *server* untuk validasi data user.
3. *Server* memberikan *response* responLogin() ke aplikasi. Bentuk data *JSON* yang dikirim *server* dapat dilihat pada *list* 3.2

```
{
     "status": boolean,
     "message": "string"
}
```

***List* 3.2** Data *JSON response* responLogin()

B. Sequence diagram registrasi

Sequence diagram registrasi menggambarkan rancangan arsitektur untuk proses registrasi. *Sequence* diagram registrasi dibuat berdasarkan rancangan *activity* diagram pada gambar 3.3. Rancangan *sequence* diagram registrasi dapat dilihat pada gambar 3.17.



Gambar 3.17 *Sequence* diagram registrasi

Berikut penjelasan untuk *sequence* diagram registrasi pada gambar 3.17 :

1. Dalam proses registrasi, bentuk data *JSON* yang dikirim dapat dilihat pada *list* 3.3.

```
{
    "nik": int,
    "nama": "string",
    "nomor telepon": "string",
    "alamat email": "string",
    "plant": "string",
    "departemen": "string",
    "password": "string",
}
```

***List* 3.3** Data *JSON request* registrasi

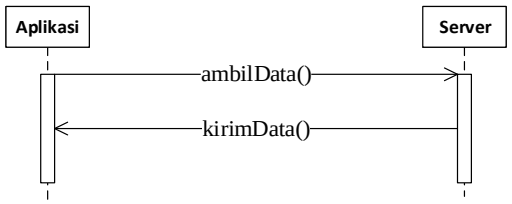
2. Aplikasi mengirimkan *request* registrasi() ke server.
3. *Server* memberikan *response* responRegistrasi() ke aplikasi. Bentuk data *JSON* yang dikirimkan *server* dapat dilihat pada *list* 3.4

```
{
   "status": boolean,
   "message": "string"
}
```

***List* 3.4** Data *JSON response* registrasi

C. Sequence diagram beranda

Berdasarkan *activity* diagram melihat komentar beranda pada gambar 3.5, maka dirancanglah *sequence* diagram beranda yang akan menggambarkan arsitektur aplikasi dengan *server* untuk proses melihat komentar beranda. *Sequence* diagram beranda dapat dilihat pada gambar 3.18.



Gambar 3.18 *Sequence* diagram beranda

Berikut penjelasan *sequence* diagram beranda pada gambar 3.18

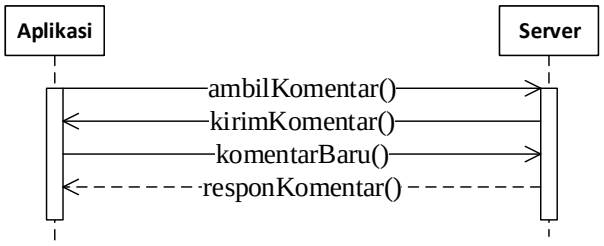
1. Dalam proses mengambil data beranda, aplikasi mengirimkan *request* ambilData() untuk mengambil data beranda dari *server*.
2. *Server* akan mengirimkan *response* kirimData() ke aplikasi dan mengirim data beranda. *Response* data yang dikirimkan ke aplikasi dapat dilihat pada list 3.5.

```
[
  {
    "isi beranda": "string",
    "tanggal": "string",
    "pengirim":{
                  "nik": int,
                  "nama": "string",
                  "nomor telepon": "string",
                  "alamat email": "string",
                  "plant": "string",
                  "departemen": "string"
               }
  },
]
```

***List* 3.5** *Response JSON* data beranda

D. Sequence diagram komentar beranda

Sequence diagram komentar beranda dibuat berdasarkan *activity* diagram pada gambar 3.6. *Sequence* diagram komentar beranda menggambarkan proses arsitektur aplikasi dengan *server* untuk proses aktifitas komentar pada informasi beranda. *Sequence* diagram komentar beranda dapat dilihat pada gambar 3.19.



Gambar 3.19 *Sequence* diagram komentar beranda

Berikut penjelasan *sequence* diagram komentar beranda pada gambar 3.19

1. Aplikasi mengirimkan *request* ambilKomentar() ke *server* untuk mengambil data komentar dari informasi yang dipilih.

2. Setelah *request* diterima oleh *server*, *server* mengirimkan *response* kirimKomentar() berisi data komentar pada informasi yang dipilih dalam bentuk *JSON*. *Response* data informasi dapat dilihat pada *list* 3.6.

```
[
  {
       "isi komentar": "string",
       "tanggal": "string",
       "id informasi": int,
       "pengirim":{
                    "nik": int,
                    "nama": "string",
                    "nomor telepon": "string",
                    "alamat email": "string",
                    "plant": "string",
                    "departemen": "string",
                    "password": "string"
                  }
  },
]
```

***List* 3.6** *Response* data JSON komentar informasi beranda

3. Untuk mengirimkan komentar aplikasi mengirimkan *request* komentarBaru() ke *server* dalam bentuk *JSON*, *request* mengirim komentar beranda dapat dilihat pada *list* 3.7.

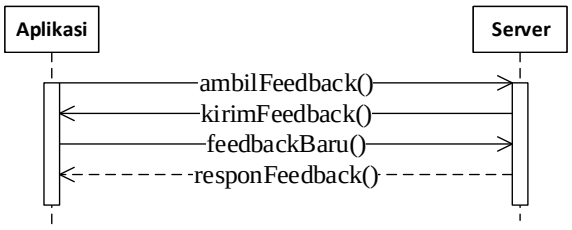
```
  {
    "isi komentar": "string",
    "tanggal": "string",
    "id informasi": int,
    "pengirim":
               {
                  "nik": int,
                  "nama": "string",
                  "nomor telepon": "string",
                  "alamat email": "string",
                  "plant": "string",
                  "departemen": "string",
                  "password": "string"
               }
  }
```

***List* 3.7** *Request* mengirim komentar beranda

4. Setelah *request* komentar baru terkirim, *server* memberikan *response* responKomentar() ke aplikasi dan akan aplikasi akan menampilkan data komentar seperti pada *list* 3.6.

E. Sequence diagram feedback

Sequence diagram *feedback* dibuat berdasarkan *activity* diagram pada gambar 3.14. *Sequence* diagram *feedback* menggambarkan proses arsitektur aplikasi dengan *server* untuk proses aktifitas *feedback*. *Sequence* diagram *feedback* dapat dilihat pada gambar 3.20.



Gambar 3.20 *Sequence* diagram *feedback*

Berikut penjelasan *sequence* diagram *feedback* pada gambar 3.20

1. Aplikasi mengirimkan *request* ambilFeedback() untuk mengambil data *feedback* yang ada pada *server*.
2. Setelah *request* diterima oleh *server*, *server* mengirimkan *response* kirimFeedback() berisi data *feedback* dalam bentuk *JSON*. *Response* data *feedback* dapat dilihat pada *list* 3.8.

```
{
      "isi feedback": "string",
      "tanggal": "string",
      "pengirim":{ "nik": int,
                   "nama": "string",
                   "nomor telepon": "string",
                   "alamat email": "string",
                   "plant": "string",
                   "departemen": "string",
                   "password": "string"
                 }
}
```

***List* 3.8** *Response* data *feedback*

3. Untuk mengirimkan *feedback* baru, aplikasi mengirimkan *request* feedbackBaru() ke *server* dalam bentuk *JSON*, *request* mengirim *feedback* dapat dilihat pada *list* 3.9.

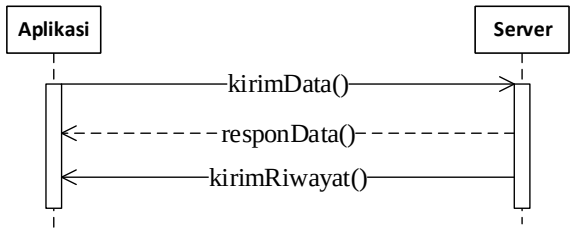
```
{
  "isi feedback": "string",
  "tanggal": "string",
  "pengirim":
             {
                "nik": int,
                "nama": "string",
                "nomor telepon": "string",
                "alamat email": "string",
                "plant": "string",
                "departemen": "string",
                "password": "string"
             }
}
```

***List* 3.9** *Request* mengirim *feedback*

4. Setelah *request feedback* baru terkirim, *server* memberikan *response* responFeedback() ke aplikasi dan aplikasi akan menampilkan data *feedback* seperti pada *list* 3.8.

F. Sequence diagram keluhan absensi

Sequence diagram keluhan absensi dibuat berdasarkan *activity* diagram pada gambar 3.7. *Sequence* diagram keluhan absensi menggambarkan proses arsitektur aplikasi dengan *server* untuk proses aktifitas keluhan absensi. *Sequence* diagram keluhan absensi dapat dilihat pada gambar 3.21



Gambar 3.21 *Sequence* diagram keluhan absensi

Berikut penjelasan *sequence* diagram keluhan absensi pada gambar 3.21

1. Aplikasi mengirimkan *request* kirimData() untuk mengirim data keluhan absensi. *Request* data keluhan absensi dapat dilihat pada *list* 3.10

```
{
        "isi keluhan": "string",
        "tanggal": "string",
        "pengirim":{
                        "nik": int,
                        "nama": "string",
                        "nomor telepon": "string",
                        "alamat email": "string",
                        "plant": "string",
                        "departemen": "string",
                        "password": "string"
        }
}
```

***List* 3.10** *Request* keluhan absensi

2. Setelah *request* diterima oleh *server*, *server* memberikan *response* responData() untuk status pengiriman keluhan absensi
3. *Server* memberikan *response* kirimRiwayat() berisi data pengiriman keluhan absensi yang dapat dilihat pada *list* 3.11

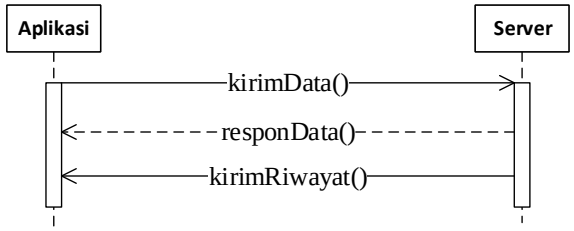
```
  {
        “isi keluhan”: “string”,
        “tanggal”: “string”,
```

```
        "pengirim":{
                    "nik": int,
                    "nama": "string",
                    "nomor telepon": "string",
                    "alamat email": "string",
                    "plant": "string",
                    "departemen": "string",
                    "password": "string"
                   }
   }
```

***List* 3.11** *Response* data keluhan absensi

G. *Sequence* diagram keluhan cuti

Sequence diagram keluhan cuti dibuat berdasarkan *activity* diagram pada gambar 3.8. *Sequence* diagram keluhan cuti menggambarkan proses arsitektur aplikasi dengan *server* untuk proses aktifitas keluhan cuti. *Sequence* diagram keluhan cuti dapat dilihat pada gambar 3.22



Gambar 3.22 *Sequence* diagram keluhan cuti

Berikut penjelasan *sequence* diagram keluhan cuti pada gambar 3.22

1. Aplikasi mengirimkan *request* untuk mengirim data keluhan cuti. Data keluhan cuti dikirim dalam bentuk *JSON*. *Request* keluhan cuti dapat dilihat pada *list* 3.12

```
{
        "isi keluhan": "string",
        "tanggal": "string",
        "pengirim":{  "nik": int,
                      "nama": "string",
                      "nomor telepon": "string",
                      "alamat email": "string",
                      "plant": "string",
                      "departemen": "string",
                      "password": "string"
        }
}
```

***List* 3.12** *Request* keluhan cuti

2. Setelah *request* diterima oleh *server*, *server* akan memberikan *response body* untuk status pengiriman keluhan cuti

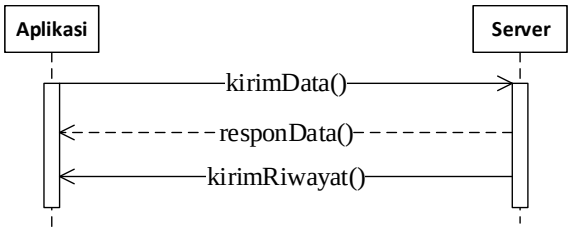
3. Setelah *request* diterima dan *server* memberikan *response*, *server* kemudian mengirimkan data pengiriman keluhan cuti seperti pada *list* 3.13

```
[
  {
       "isi keluhan": "string",
       "tanggal": "string",
       "pengirim":{ "nik": int,
                    "nama": "string",
                    "nomor telepon": "string",
                    "alamat email": "string",
                    "plant": "string",
                    "departemen": "string",
                    "password": "string"
                  }
  },
]
```

***List* 3.13** *Response* data keluhan cuti

H. Sequence diagram keluhan *overtime*

Sequence diagram keluhan *overtime* dibuat berdasarkan *activity* diagram pada gambar 3.9. *Sequence* diagram keluhan *overtime* menggambarkan proses arsitektur aplikasi dengan *server* untuk proses aktifitas keluhan *overtime*. *Sequence* diagram keluhan *overtime* dapat dilihat pada gambar 3.23



Gambar 3.23 *Sequence* diagram keluhan *overtime*

Berikut penjelasan *sequence* diagram keluhan *overtime* pada gambar 3.23

1. Aplikasi mengirimkan *request* untuk mengirim data keluhan *overtime*. Data keluhan *overtime* dikirim dalam bentuk *JSON*. *Request* keluhan *overtime* dapat dilihat pada *list* 3.14

```
{
       "isi keluhan": "string",
       "tanggal": "string",
       "pengirim":{
                    "nik": int,
                    "nama": "string",
                    "nomor telepon": "string",
                    "alamat email": "string",
                    "plant": "string",
                    "departemen": "string",
                    "password": "string"
```

```
        }
}
```

***List* 3.14** *Request* keluhan *overtime*

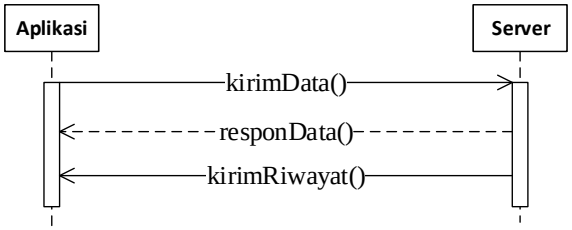
2. Setelah *request* diterima oleh *server*, *server* akan memberikan *response body* untuk status pengiriman keluhan *overtime*
3. Setelah *request* diterima dan *server* memberikan *response*, *server* kemudian mengirimkan data pengiriman keluhan *overtime* seperti pada *list* 3.15

```
[
  {
       "isi keluhan": "string",
       "tanggal": "string",
       "pengirim":{
                    "nik": int,
                    "nama": "string",
                    "nomor telepon": "string",
                    "alamat email": "string",
                    "plant": "string",
                    "departemen": "string",
                    "password": "string"
                  }
  },
]
```

***List* 3.15** *Response* data keluhan *overtime*

I. *Sequence* diagram pendaftaran beasiswa

Sequence diagram pendaftaran beasiswa dibuat berdasarkan *activity* diagram pada gambar 3.10. *Sequence* diagram pendaftaran beasiswa menggambarkan proses arsitektur aplikasi dengan *server* untuk proses aktifitas pendaftaran beasiswa. *Sequence* diagram pendaftaran beasiswa dapat dilihat pada gambar 3.24



Gambar 3.24 *Sequence* diagram pendaftaran beasiswa

Berikut penjelasan *sequence* diagram pendaftaran beasiswa pada gambar 3.24

1. Aplikasi mengirimkan *request* untuk mengirim data pendaftaran beasiswa. Data pendaftaran beasiswa dikirim dalam bentuk *JSON*. *Request* pendaftaran beasiswa dapat dilihat pada *list* 3.16

```
{
        "nama anak": "string",
        "tanggal lahir": "string",
        "dokumen lampiran": "string",
        "tanggal": "string",
        "pengirim":{
                        "nik": int,
                        "nama": "string",
                        "nomor telepon": "string",
                        "alamat email": "string",
                        "plant": "string",
                        "departemen": "string",
                        "password": "string"
        }
}
```

***List* 3.16** *Request* pendaftaran beasiswa

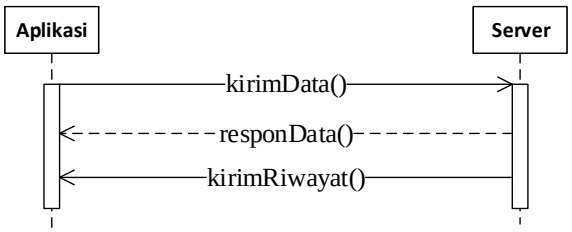
2. Setelah *request* diterima oleh *server*, *server* akan memberikan *response body* untuk status pengiriman pendaftaran beasiswa
3. Setelah *request* diterima dan *server* memberikan *response*, *server* kemudian mengirimkan data pengiriman pendaftaran beasiswa seperti pada *list* 3.17

```
[
   {
        "nama anak": "string",
        "tanggal lahir": "string",
        "dokumen lampiran": "string",
        "tanggal": "string",
        "pengirim":{
                        "nik": int,
                        "nama": "string",
                        "nomor telepon": "string",
                        "alamat email": "string",
                        "plant": "string",
                        "departemen": "string",
                        "password": "string"
        }
   },
]
```

***List* 3.17** *Response* data pendaftaran beasiswa

J. *Sequence* diagram pendafraran BPJS

Sequence diagram pendaftaran BPJS dibuat berdasarkan *activity* diagram pada gambar 3.11. *Sequence* diagram pendaftaran BPJS menggambarkan proses arsitektur aplikasi dengan *server* untuk proses aktifitas pendaftaran BPJS. *Sequence* diagram pendaftaran BPJS dapat dilihat pada gambar 3.25.



Gambar 3.25 *Sequence* diagram pendaftaran BPJS

Berikut penjelasan *sequence* diagram pendaftaran BPJS pada gambar 3.25

1. Aplikasi mengirimkan *request* untuk mengirim data pendaftaran BPJS. Data pendaftaran BPJS dikirim dalam bentuk *JSON*. *Request* pendaftaran BPJS dapat dilihat pada *list* 3.18

```
{
        "nomor ktp": "string",
        "nomor kk": "string",
        "dokumen lampiran": "string",
        "tanggal": "string",
        "pengirim":{
                        "nik": int,
                        "nama": "string",
                        "nomor telepon": "string",
                        "alamat email": "string",
                        "plant": "string",
                        "departemen": "string",
                        "password": "string"
        }
}
```

***List* 3.18** *Request* pendaftaran BPJS

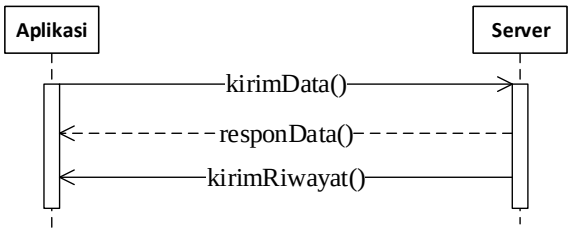
2. Setelah *request* diterima oleh *server*, *server* akan memberikan *response body* untuk status pengiriman pendaftaran BPJS
3. Setelah *request* diterima dan *server* memberikan *response*, *server* kemudian mengirimkan data pengiriman pendaftaran BPJS seperti pada *list* 3.19

```
[ {
        "nama anak": "string",
        "tanggal lahir": "string",
        "dokumen lampiran": "string",
        "tanggal": "string",
        "pengirim":{ "nik": int,
                        "nama": "string",
                        "nomor telepon": "string",
                        "alamat email": "string",
                        "plant": "string",
                        "departemen": "string",
                        "password": "string"
        }},]
```

***List* 3.19** *Response* data pendaftaran BPJS

K. Sequence diagram perubahan *id card*

Sequence diagram perubahan *id card* dibuat berdasarkan *activity* diagram pada gambar 3.12. *Sequence* diagram perubahan *id card* menggambarkan proses arsitektur aplikasi dengan *server* untuk proses aktifitas perubahan *id card*. *Sequence* diagram perubahan *id card* dapat dilihat pada gambar 3.26.



Gambar 3.26 *Sequence* diagram perubahan *id card*

Berikut penjelasan *sequence* diagram perubahan *id card* pada gambar 3.26

1. Aplikasi mengirimkan *request* untuk mengirim data perubahan *id card*. Data perubahan *id card* dikirim dalam bentuk *JSON*. *Request* perubahan *id card* dapat dilihat pada *list* 3.20

```
{
        "alasan perubahan": "string",
        "dokumen lampiran": "string",
        "tanggal": "string",
        "pengirim":{
                        "nik": int,
                        "nama": "string",
                        "nomor telepon": "string",
                        "alamat email": "string",
                        "plant": "string",
                        "departemen": "string",
                        "password": "string"
        }
}
```

***List* 3.20** *Request* perubahan *id card*

2. Setelah *request* diterima oleh *server*, *server* akan memberikan *response body* untuk status pengiriman perubahan *id card*
3. Setelah *request* diterima dan *server* memberikan *response*, *server* kemudian mengirimkan data pengiriman perubahan *id card* seperti pada *list* 3.21

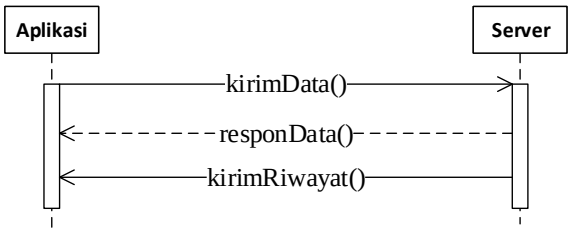
```
[ {
        "alasan perubahan": "string",
        "dokumen lampiran": "string",
        "tanggal": "string",
        "pengirim":{ "nik": int,
                        "nama": "string",
                        "nomor telepon": "string",
```

```
                    "alamat email": "string",
                    "plant": "string",
                    "departemen": "string",
                    "password": "string"
        }},]
```

***List* 3.21** *Response* data perubahan *id card*

L. *Sequence* diagram surat keterangan

Sequence diagram surat keterangan dibuat berdasarkan *activity* diagram pada gambar 3.13. *Sequence* diagram surat keterangan menggambarkan proses arsitektur aplikasi dengan *server* untuk proses aktifitas surat keterangan. *Sequence* diagram surat keterangan dapat dilihat pada gambar 3.27.



Gambar 3.27 *Sequence* diagram surat keterangan

Berikut penjelasan *sequence* diagram surat keterangan pada gambar 3.27

1. Aplikasi mengirimkan *request* untuk mengirim data surat keterangan. Data surat keterangan dikirim dalam bentuk *JSON*. *Request* surat keterangan dapat dilihat pada *list* 3.22.

```
{
        "nomor surat": "string",
        "jenis surat": "string",
        "dokumen lampiran": "string",
        "tanggal": "string",
        "pengirim":{
                    "nik": int,
                    "nama": "string",
                    "nomor telepon": "string",
                    "alamat email": "string",
                    "plant": "string",
                    "departemen": "string",
                    "password": "string"
        }
}
```

***List* 3.22** *Request* surat keterangan

2. Setelah *request* diterima oleh *server*, *server* akan memberikan *response body* untuk status pengiriman surat keterangan

3. Setelah *request* diterima dan *server* memberikan *response*, *server* kemudian mengirimkan data pengiriman surat keterangan seperti pada *list* 3.23

```
[
    {
        "nomor surat": "string",
        "jenis surat": "string",
        "dokumen lampiran": "string",
        "tanggal": "string",
        "pengirim":{ "nik": int,
                     "nama": "string",
                     "nomor telepon": "string",
                     "alamat email": "string",
                     "plant": "string",
                     "departemen": "string",
                     "password": "string"
        }
    },
]
```

***List* 3.23** *Response* data surat keterangan

3.4.4 Rancangan *class* Diagram

Berdasarkan proses pada *sequence* diagram maka dirancanglah *class* diagram. *Class* diagram menggambarkan rancangan *class* dalam pembangunan aplikasi android.

Class Diagram yang digunakan dapat dilihat pada gambar 3.28.

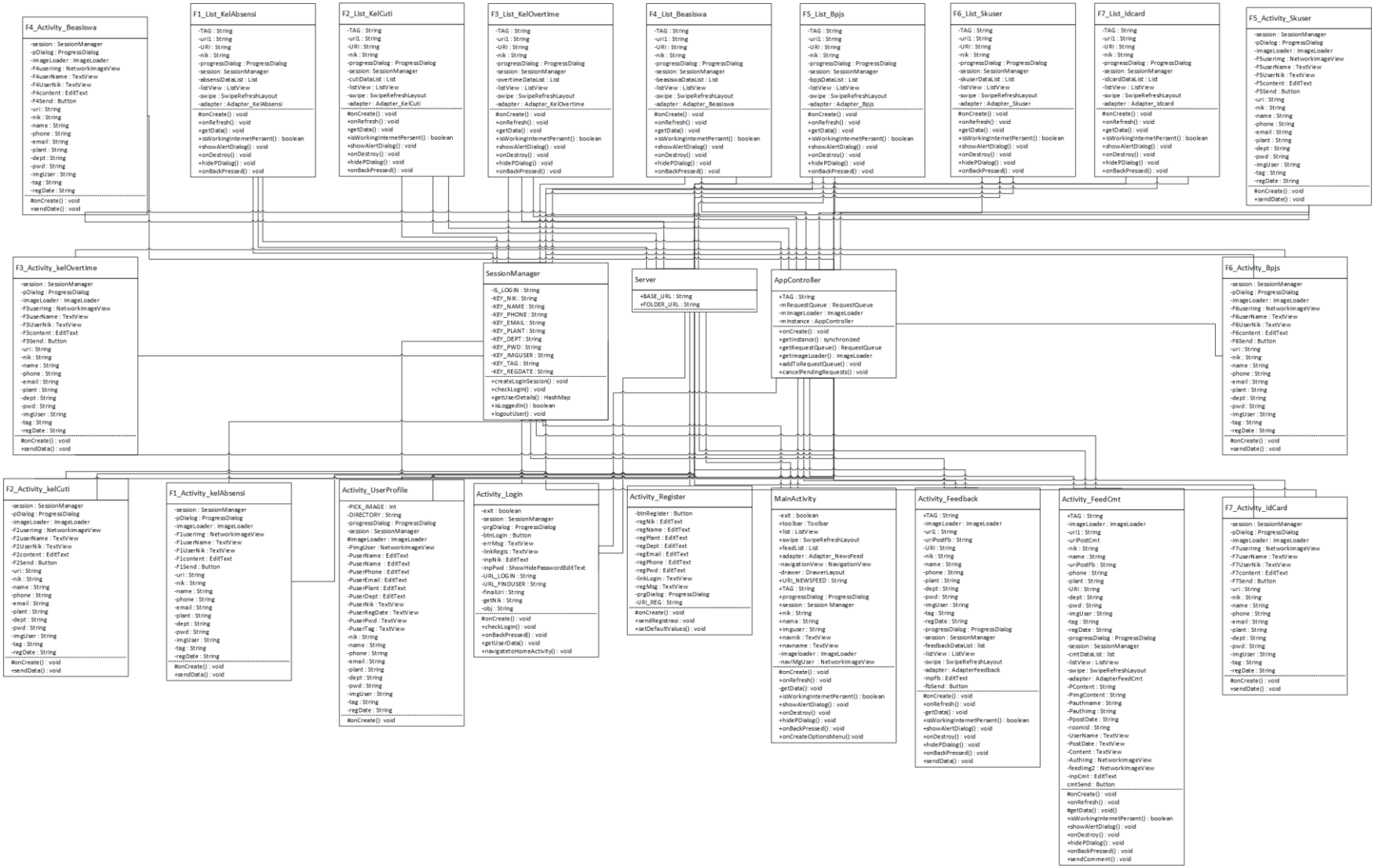
Berikut penjelasan *class* diagram pada gambar 3.28 :

1. Pada *class Session Manager* terdapat fungsi *check Login* yang akan melakukan pengecekan apakah *User* sudah dalam keadaan *login* atau belum, jika sudah maka aplikasi akan langsung menampilkan halaman beranda. Terdapat fungsi *get User Details* yang akan menyimpan data *User* pada aplikasi.
2. Pada class *App Controller* terdapat fungsi *get Image Loader* yang berfungsi menampilkan gambar pada *field* aplikasi. Terdapat fungsi *add Request To Queue* yang akan melakukan proses pengiriman data dari aplikasi ke *server*.
3. Pada class *Activity* Registrasi terdapat fungsi *send Registrasi* sehingga *User* baru dapat membuat akun. Terdapat fungsi *set Default Values* untuk mengkosongkan *field* pada halaman registrasi setelah data terkirim.
4. Pada *class Activity login* terdapat fungsi *on Check Login* yang akan melakukan pengecekan apakah *User* sudah melakukan *login* atau belum. Jika

User sudah melakukan *login* maka aplikasi akan langsung menampilkan halaman beranda. Terdapat fungsi *get User Data* yang akan mengambil data *User* setelah *User* melakukan *login*.

5. Pada *class Main Activity* terdapat fungsi *is Working Internet* yang akan melakukan pengecekan apakah *smartphone* terhubung ke jaringan internet atau tidak, jika *smartphone* tidak terhubung ke jaringan internet maka aplikasi akan menampilkan informasi dan aplikasi akan menutup. Terdapat fungsi *get Data* yang mengambil data *newsfeed* dari *database*. Terdapat fungsi *on Refresh* yang akan memperbaharui data *newsfeed* jika *User* mengusap layar atau melakukan swap pada layar *smartphone*.
6. Pada *class Activity feedback* terdapat fungsi *get Data* yang akan menampilkan data *feedback* yang sudah tersimpan di *database*. Terdapat fungsi *send Data* yang akan mengirimkan data *feedback* ke *database*.
7. Pada *class Activity Feedcmt* terdapat fungsi *get Data* yang akan menampilkan data kometar dari *database*. Terdapat fungsi *send Comment* yang akan mengirimkan data komentar ke *database*.
8. Pada *class Activity* kelAbsensi terdapat fungsi *send Data* yang akan mengirim data keluhan absensi ke *database*
9. Pada *class Activity* kelCuti terdapat fungsi *send Data* yang akan mengirim data keluhan cuti ke *database*
10. Pada *class Activity* kel*Overtime* terdapat fungsi *send Data* yang akan mengirim data keluhan *overtime* ke *database*
11. Pada class *Activity* Beasiswa terdapat fungsi *send Data* yang akan mengirim data pendaftaran beasiswa ke database
12. Pada *class Activity* Bpjs terdapat fungsi *send Data* yang akan mengirim data pendaftaran BPJS ke *database*
13. Pada *class Activity* Sk*User* terdapat fungsi *send Data* yang akan mengirim data pengajuan SK ke *database*
14. Pada *class Activity idCard* terdapat fungsi *send Data* yang akan mengirim data perubahan *id card* ke *database*

15. Pada *class List* kelAbsensi terdapat fungsi *get Data* yang akan mengambil data keluhan absensi dari *database.*
16. Pada *class List* kelCuti terdapat fungsi *get Data* yang akan mengambil data keluhan cuti dari *database.*
17. Pada *class List* kel*Overtime* terdapat fungsi *get Data* yang akan mengambil data keluhan *overtime* dari *database.*
18. Pada *class List* Beasiswa terdapat fungsi *get Data* yang akan mengambil data pendaftaran beasiswa dari *database.*
19. Pada *class List* Bpjs terdapat fungsi *get Data* yang akan mengambil data pendaftaran BPJS dari *database.*
20. Pada *class List* Sk*User* terdapat fungsi *get Data* yang akan mengambil data pengajuan Sk dari *database.*
21. Pada *class List idCard* terdapat fungsi *get Data* yang akan mengambil data perubahan *id card* dari *database.*



Gambar 3.28 *Class* Diagram

3.4.5 Perancangan Sistem

Pada tahap ini dilakukan perancangan arsitektur sistem yang akan digunakan. Tahap perancangan sistem akan dibagi menjadi 3 bagian yaitu :

A. Perancangan Model Data

Berdasarkan rancangan *sequence* diagram pada sub bab 3.3.3, maka dirancanglah model data. Model data dirancang berdasarkan kebutuhan data *JSON* pada *sequence* diagram.

Rancangan model data dibagi menjadi 11 bagian yaitu sebagai berikut:

1. Model data user

Model data *user* dirancang berdasarkan data *JSON* yang dikirimkan pada *sequence* diagram registrasi pada *list* 3.3. Model data *user* dapat dilihat pada *list* 3.24.

```
public class m_user {

    private String dept,email,imgUser,name,phone,plant,pwd,regDate,tag;
    private Integer nik;

    public m_user(String dept, String email, String imguser, String name,
                  Integer nik, String phone,String plant, String pwd, String
                  regDate, String tag) {
        super();
        this.dept = dept;
        this.email = email;
        this.imgUser = imgUser;
        this.name = name;
        this.nik = nik;
        this.phone = phone;
        this.plant = plant;
        this.pwd = pwd;
        this.regDate = regDate;
        this.tag = tag;
    }
```

***List* 3.24** Model data *user*

2. Model data informasi beranda

Model data informasi beranda dirancang berdasarkan data *JSON* yang pada *sequence* diagram beranda pada *list* 3.5. Model data informasi beranda dapat dilihat pada list 3.25.

```
public class m_NewsFeed {

    private String content,imgCOntent, postDate, dept, email, imgUser, name,
                   phone, plant,pwd,regDate,tag;
    private Integer nik, roomId;
    private String count;
```

```
    public m_NewsFeed(String content, String imgContent, String posdate, Integer
                      roomId,String count, String dept, String email, String
                      imgUser, String name, Integer nik, String phone, String
                      plant, String pwd, String regDate, String tag) {
        super();
        this.content = content;
        this.imgContent = imgContent;
        this.posdate = posdate;
        this.roomId = roomId;
        this.count = count;
        this.dept = dept;
        this.email = email;
        this.imgUser = imgUser;
        this.name = name;
        this.nik = nik;
        this.phone = phone;
        this.plant = plant;
        this.pwd = pwd;
        this.regDate = regDate;
        this.tag = tag;
    }
```

***List* 3.25** Model data informasi beranda

3. Model data komentar beranda

Model data informasi komentar beranda dirancang berdasarkan data *JSON* yang pada *sequence* diagram komentar beranda pada *list* 3.7. Model data komentar beranda dapat dilihat pada list 3.26.

```
public class m_FeedComment {

    private String cmtContent,cmtPostDate, dept, email, imgUser, name,
                   phone, plant,pwd,regDate,tag;
    private Integer id, roomId, nik;
    private String cmtposdate;

    public m_FeedComment(String cmtcontent, String cmtposdate, Integer id, Integer
                         roomId, String dept, String email, String imgUser, String
                         name, Integer nik, String phone, String plant, String
                         pwd, String regDate, String tag) {
        super();
        this.cmtcontent = cmtcontent;
        this.cmtposdate = cmtposdate;
        this.id = id;
        this.roomId = roomId;
        this.dept = dept;
        this.email = email;
        this.imgUser = imgUser;
        this.name = name;
        this.nik = nik;
        this.phone = phone;
        this.plant = plant;
        this.pwd = pwd;
        this.regDate = regDate;
        this.tag = tag;
    }
```

***List* 3.26** Model data komentar beranda

4. Model data feedback

Model data *feedback* dirancang berdasarkan data *JSON* yang pada *sequence* diagram *feedback* pada *list* 3.8. Model data *feedback* dapat dilihat pada list 3.27.

```
public class m_feedback {

    private String content, postDate,  dept, email, imgUser, name, phone, plant,
                   pwd, regDate, tag;
    private Integer id, nik;

    public m_feedback( String content, Integer id, String posdate,String dept,
                       String email, String imgUser, String name, Integer nik,
                       String phone, String plant, String pwd, String regDate,
                       String tag ) {
        super();
        this.content = content;
        this.id = id;
        this.posdate = posdate;
        this.dept = dept;
        this.email = email;
        this.imgUser = imgUser;
        this.name = name;
        this.nik = nik;
        this.phone = phone;
        this.plant = plant;
        this.pwd = pwd;
        this.regDate = regDate;
        this.tag = tag;
    }
```

***List* 3.27** Model data *feedback*

5. Model data keluhan absensi

Model data keluhan absensi dirancang berdasarkan data *JSON* yang pada *sequence* diagram keluhan absensi pada *list* 3.11. Model data keluhan absensi dapat dilihat pada list 3.28.

```
public class m_KelAbsensi {

    private String content, date, status, dept, email, imgUser, name, phone,
                   plant, pwd, regDate, tag;
    private Integer id, nik;

    public m_KelAbsensi(String content, String date, String status, Integer id,
                        String dept, String email, String img User, String name,
                        Integer nik, String phone, String plant, String pwd,
                        String regDate, String tag) {
        super();
        this.content = content;
        this.date = date;
        this.status = status;
        this.id = id;
        this.dept = dept;
        this.email = email;
        this.img User = img User;
        this.name = name;
        this.nik = nik;
        this.phone = phone;
        this.plant = plant;
        this.pwd = pwd;
        this.regDate = regDate;
        this.tag = tag;
    }
```

***List* 3.28** Model data keluhan absensi

6. Model data keluhan cuti

Model data keluhan cuti dirancang berdasarkan data *JSON* yang pada *sequence* diagram keluhan cuti pada *list* 3.13. Model data keluhan cuti dapat dilihat pada list 3.29.

```
public class m_KelCuti {

    private String content, date, status, dept, email, imgUser, name, phone,
                   plant, pwd, regDate, tag;
    private Integer id, nik;

    public m_KelCuti(String content, String date, String status, Integer id,
                     String dept, String email, String imgUser, String name,
                     Integer nik, String phone, String plant, String pwd, String
                     regDate, String tag) {
        super();
        this.content = content;
        this.date = date;
        this.status = status;
        this.id = id;
        this.dept = dept;
        this.email = email;
        this.imgUser = imgUser;
        this.name = name;
        this.nik = nik;
        this.phone = phone;
        this.plant = plant;
        this.pwd = pwd;
        this.regDate = regDate;
        this.tag = tag;
    }
```

***List* 3.29** Model data keluhan cuti

7. Model data keluhan *overtime*

Model data keluhan *overtime* dirancang berdasarkan data *JSON* yang pada *sequence* diagram keluhan *overtime* pada *list* 3.15. Model data keluhan *overtime* dapat dilihat pada list 3.30.

```
public class m_KelOvertime {

    private String content, date, status, dept, email, imgUser, name, phone,
                   plant, pwd, regDate, tag;
    private Integer id, nik;

    public m_KelOvertime(String content, String date, String status, Integer
                         id,String dept, String email, String imgUser, String
                         name, Integer nik, String phone, String plant, String
                         pwd, String regDate, String tag) {
        super();
        this.content = content;
        this.date = date;
        this.status = status;
        this.id = id;
        this.dept = dept;
        this.email = email;
        this.imgUser = imgUser;
        this.name = name;
        this.nik = nik;
        this.phone = phone;
        this.plant = plant;
```

```
        this.pwd = pwd;
        this.regDate = regDate;
        this.tag = tag;
    }
```

***List* 3.30** Model data keluhan *overtime*

8. Model data pendaftaran beasiswa

Model data pendaftaran beasiswa dirancang berdasarkan data *JSON* yang pada *sequence* diagram pendaftaran beasiswa pada *list* 3.16. Model data pendaftaran beasiswa dapat dilihat pada list 3.31.

```
public class m_Beasiswa {

    private String doc, nmAnak, ttl, date, status, dept, email, imgUser, name,
                   phone, plant, pwd, regDate, tag;
    private Integer id, nik;

    public m_Beasiswa(String date, String doc, Integer id, String nmAnak, String
                      status, String ttl,String dept, String email, String
                      imgUser, String name, Integer nik, String phone, String
                      plant, String pwd, String regDate, String tag) {
        super();
        this.date = date;
        this.doc = doc;
        this.id = id;
        this.nmAnak = nmAnak;
        this.status = status;
        this.ttl = ttl;
        this.dept = dept;
        this.email = email;
        this.imgUser = imgUser;
        this.name = name;
        this.nik = nik;
        this.phone = phone;
        this.plant = plant;
        this.pwd = pwd;
        this.regDate = regDate;
        this.tag = tag;
    }
```

***List* 3.31** Model data pendaftaran beasiswa

9. Model data pendaftaran BPJS

Model data pendaftaran BPJS dirancang berdasarkan data *JSON* yang pada *sequence* diagram pendaftaran BPJS pada *list* 3.18. Model data pendaftaran BPJS dapat dilihat pada list 3.32.

```
public class m_Bpjs {

    private String doc, noKK, noKtp, date, status, dept, email, imgUser, name,
                   phone, plant, pwd, regDate, tag;
    private Integer id, nik;

    public m_Bpjs(String date, String doc, Integer id, String nokk, String noktp,
                  String status,String dept, String email, String imgUser, String
                  name,Integer nik, String phone, String plant, String pwd, String
                  regDate,String tag) {
        super();
        this.date = date;
        this.doc = doc;
```

```
        this.id = id;
        this.nokk = nokk;
        this.noktp = noktp;
        this.status = status;
        this.dept = dept;
        this.email = email;
        this.imgUser = imgUser;
        this.name = name;
        this.nik = nik;
        this.phone = phone;
        this.plant = plant;
        this.pwd = pwd;
        this.regDate = regDate;
        this.tag = tag;
    }
```

***List* 3.32** Model data pendaftaran BPJS

10. Model data perubahan *id card*

Model data perubahan *id card* dirancang berdasarkan data *JSON* yang pada *sequence* diagram perubahan *id card* pada *list* 3.20. Model data perubahan *id card* dapat dilihat pada list 3.33.

```
public class m_IdCard {

    private String doc, content, date, status, dept, email, imgUser, name,
                   phone, plant, pwd, regDate, tag;
    private Integer id, nik;

    public m_IdCard(String content, String date, String doc, Integer id, String
                    status, String dept, String email, String imgUser, String
                    name, Integer nik, String phone, String plant, String pwd,
                    String regDate,String tag) {
        super();
        this.content = content;
        this.date = date;
        this.doc = doc;
        this.id = id;
        this.status = status;
        this.dept = dept;
        this.email = email;
        this.imgUser = imgUser;
        this.name = name;
        this.nik = nik;
        this.phone = phone;
        this.plant = plant;
        this.pwd = pwd;
        this.regDate = regDate;
        this.tag = tag;
    }
```

***List* 3.33** Model data perubahan *id card*

11. Model data surat keterangan

Model data surat keterangan dirancang berdasarkan data *JSON* yang pada *sequence* diagram surat keterangan pada *list* 3.22. Model data surat keterangan dapat dilihat pada list 3.34.

```
public class m_SkUser {

    private String doc, noSk, jenisSk, date, status, dept, email, imgUser, name,
```

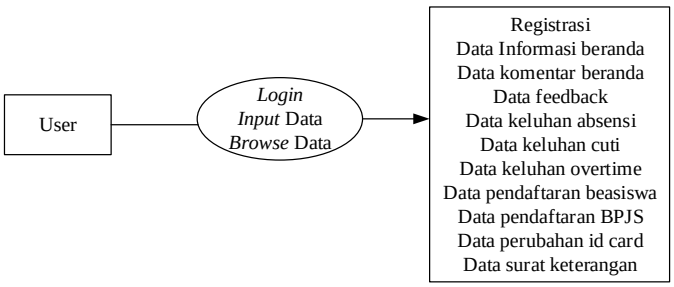
```
                phone, plant, pwd, regDate, tag;
    private Integer id, nik;

    public m_SkUser(Integer id, String date, String doc, String noSk, String
                    status, String dept, String email, String imgUser, String
                    name, String jenisSk, Integer nik, String phone, String plant,
                    String pwd, String regDate, String tag) {
        super();
        this.id = id;
        this.date = date;
        this.doc = doc;
        this.noSk = noSk;
        this.jenisSk = jenisSk;
        this.status = status;
        this.dept = dept;
        this.email = email;
        this.imgUser = imgUser;
        this.name = name;
        this.nik = nik;
        this.phone = phone;
        this.plant = plant;
        this.pwd = pwd;
        this.regDate = regDate;
        this.tag = tag;
    }
```

***List* 3.34** Model data surat keterangan

B. Perancangan *web service*

Pada *web service* akan menyediakan fungsi *login*, *input* data, dan *browse* data. Rancangan fungi pada *web service* dapat dilihat pada gambar 3.29.



Gambar 3.29 Rancangan *web service*

Berdasarkan rancangan *server web service* pada gambar 3.29 maka dibuat daftar *web API* yang akan di implementasikan, diantaranya :

1. *Web API* dengan kegunaan *input* data
    a. Registrasi
    b. Tambah data komentar beranda
    c. Tambah data *feedback*
    d. Tambah data keluhan absensi
    e. Tambah data keluhan cuti

f. Tambah data keluhan *overtime*
g. Tambah data pendaftaran beasiswa
h. Tambah data pendaftaran BPJS
i. Tambah data perubahan *id card*
j. Tambah data surat keterangan

2. *Web API* dengan kegunaan *browse* data
   a. Tampil data beranda
   b. Tampil data komentar beranda
   c. Tampil data *feedback*
   d. Tampil data keluhan absensi
   e. Tampil data keluhan *overtime*
   f. Tampil data pendaftaran beasiswa
   g. Tampil data pandaftaran BPJS
   h. Tampil data perubahan *id card*
   i. Tampil data surat keterangan
3. *Web API login* dengan kegunaan memberi akses *user* untuk dapat menggunakan aplikasi.

Berikut daftar *API* yang akan digunakan pada aplikasi administrasi personalia di PT Indocement Tunggal Prakasa Tbk.

Tabel 3.1 Daftar *API* pada *server web service*

No	Fungsi	*URL*	Metode
1	*Login*	api/login/dologin	*@POST*
2	Registrasi	api/register/doregister	*@POST*
3	Mengambil data *user* berdasarkan nik	api/user/find/nik/{nik}	*@GET*
4	Mengambil data *newsfeed*	api/newsfeed/list	*@GET*
5	Mengambil data komentar pada *newsfeed*	api/feedcomment/find/byroom/{roomid}	*@GET*

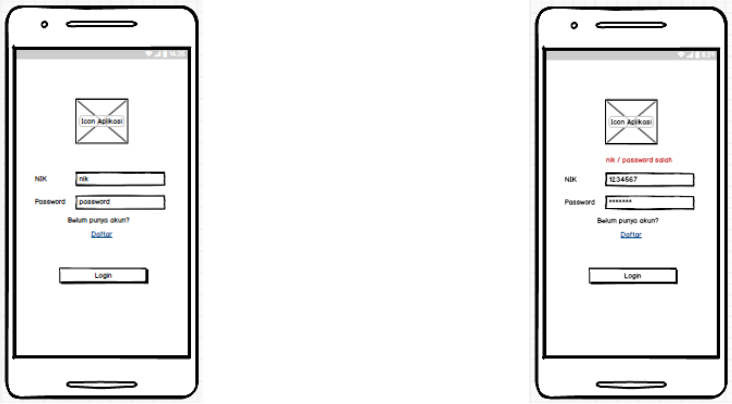
6	Mengirim komentar pada *newsfeed*	api/feedcomment/create	*@POST*
7	Mengambil data *feedback*	api/feedback/list	*@GET*
8	Mengirim data *feedback*	api/feedback/create	*@POST*
9	Mengirim data keluhan absensi	api/absensi/create	*@POST*
10	Mengambil data keluhan absensi	api/absensi/find/nik/{nik}	*@GET*
11	Mengirim data keluhan cuti	api/cuti/create	*@POST*
12	Mengambil data keluhan cuti	api/cuti/find/bynik/{nik}	*@GET*
13	Mengirim data keluhan *overtime*	api/overtime/create	*@POST*
14	Mengambil data keluhan *overtime*	api/overtime/find/bynik/{nik}	*@GET*
15	Mengirim pendaftaran beasiswa	api/beasiswa/create	*@POST*
16	Mengambil data pendaftaran beasiswa	api/beasiswa/find/bynik/{nik}	*@GET*
17	Mengirim data perubahan *id card*	api/idcard/create	*@POST*
18	Mengambil data perubahan *id card*	api/idcard/find/bynik/{nik}	*@GET*
19	Mengirim data surat keterangan	api/skuser/create	*@POST*
20	Mengambil data surat keterangan	api/skuser/find/bynik/{nik}	*@GET*

C. Perancangan antarmuka android

Tahap perancangan antarmuka dibagi menjadi 13 bagian yaitu :

1. Perancangan antarmuka halaman login

Berdasarkan *activity* diagram login pada gambar 3.4 maka dibuat rancangan antarmuka halaman *login*. Rancangan antarmuka halaman *login* menampilkan *icon* aplikasi, 2 *textbox*, teks, dan 1 tombol. Untuk melakukan *login user* menggunakan NIK (Nomor Induk Karyawan) dan *password* yang telah dibuat oleh *user*. Jika *user* belum memiliki akun aplikasi, *user* dapat membuat akun dengan menekan pilihan "Daftar". Jika *user* salah memasukkan nik atau *password* akan muncul peringatan berupa teks di bagian atas *field* nik seperti pada gambar B. Rancangan halaman login dapat dilihat pada gambar 3.30.

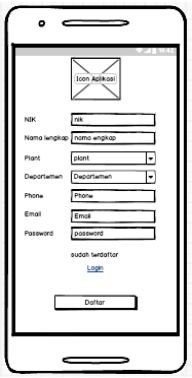


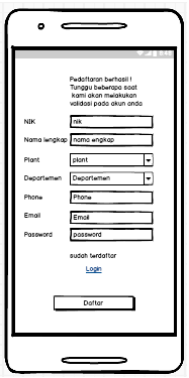
Gambar A Gambar B

Gambar 3.30 Rancangan halaman *login*

2. Perancangan antarmuka halaman registrasi

Berdasarkan *activity* diagram registrasi pada gambar 3.3 maka dibuat rancangan antarmuka halaman registrasi. Rancangan antarmuka halaman registrasi terdapat *icon* aplikasi, 5 *textbox,* 2 pilihan dan 1 tombol. Data-data yang harus diisi saat melakukan registrasi yaitu NIK, nama lengkap, nomot telepon, alamat email, *plant* pekerjaan, departemen pekerjaan, dan *password.* Setelah melakukan registrasi, akan muncul teks di bagian atas seperti pada gambar B. Rancangan halaman registrasi dapat dilihat pada gambar 3.31.



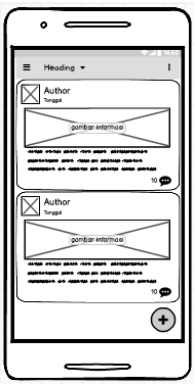


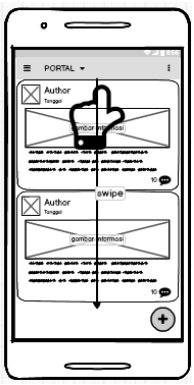
Gambar A Gambar B

Gambar 3.31 Rancangan halaman registrasi

3. Perancangan antarmuka halaman beranda

Berdasarkan *activity* diagram pada gambar 3.5 maka dibuat rancangan antarmuka halaman beranda. Rancangan halaman beranda berisi tentang informasi-informasi terbaru seputar kegiatan administrasi personalia. Setiap informasi yang dikirim akan menampilkan nama pengirim dan tanggal informasi dikirim. Informasi yang dimuat dapat berupa gambar dan teks atau hanya berupa teks. Pada setiap informasi terdapat jumlah banyaknya komentar yang diberikan pada informasi tersebut. Untuk memperbaharui informasi yang tampil dapat dilakukan dengan cara mengusap atau *swipe* pada layar seperti pada gambar B. Rancangan halaman beranda dapat dilihat pada gambar 3.32.



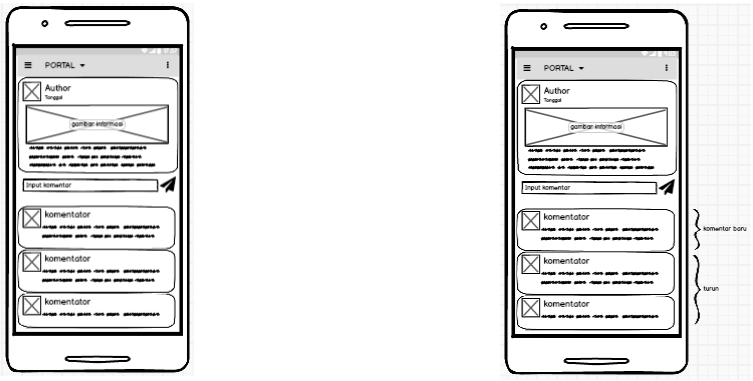


Gambar A Gambar B

Gambar 3.32 Rancangan halaman beranda

4. Perancangan antarmuka halaman komentar beranda

Berdasarkan *activity* diagram pada gambar 3.6 maka dibuat rancangan antarmuka halaman komentar beranda. Rancangan halaman komentar akan menampilkan informasi yang akan dikomentari dan komentar-komentar yang diberikan oleh *User* lain. Setelah *user* mengirimkan komentar, maka komentar lebih lama akan menurun dan komentar baru berapda pada atas komentar lama seperti pada gambar B. Rancangan halaman komentar dapat dilihat pada gambar 3.33.

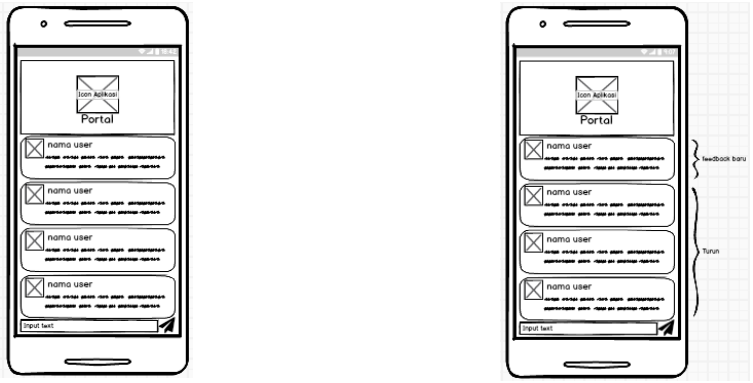


Gambar A Gambar B

Gambar 3.33 Rancangan halaman komentar

5. Perancangan antarmuka halaman feedback

Berdasarkan *activity* diagram pada gambar 3.14 maka dibuat rancangan antarmuka halaman *feedback*. Rancangan halaman *feedback* akan menampilkan *icon* dari aplikasi dan *feedback* yang diberikan oleh *User* lain. Setelah *feedback* terkirim data *feedback* lama akan menurun dan *feedback* yang dikirim akan berada di atas *feedback* yang lebih lama seperti pada gambar B. Rancangan halaman *feedback* dapat dilihat pada gambar 3.34.

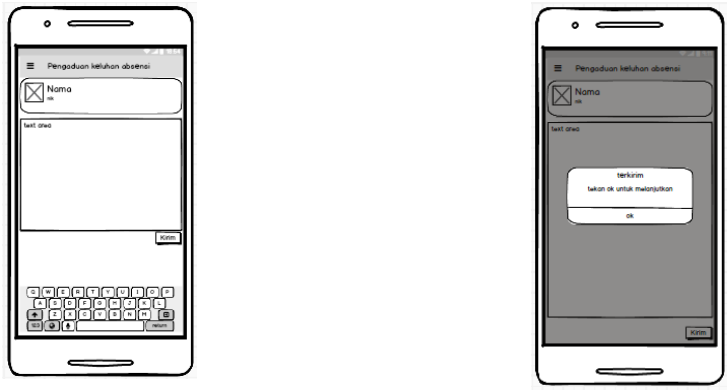


Gambar A Gambar B

Gambar 3.34 Rancangan halaman *feedback*

6. Perancangan antarmuka halaman pengaduan keluhan absensi

Berdasarkan *activity* diagram pada gambar 3.7 maka dibuat rancangan antarmuka halaman keluhan absensi. Rancangan halaman pengaduan keluhan absensi berisi nama dan nik pengguna. Keluhan dapat di *input*kan ke kolom yang telah disediakan dan untuk mengirim pengaduan *User* harus menekan tombol kirim. Setelah data terkirim akan muncul notifikasi seperti pada gambar B. Rancangan halaman pengaduan keluhan absensi dapat dilihat pada gambar 3.35.



Gambar A Gambar B

Gambar 3.35 Rancangan halaman Pengaduan keluhan absensi

7. Perancangan antarmuka halaman pengaduan keluhan cuti

Berdasarkan *activity* diagram pada gambar 3.8 maka dibuat rancangan antarmuka halaman keluhan cuti. Rancangan halaman pengaduan keluhan cuti berisi nama dan nik pengguna. Pengaduan keluhan dapat di *input*kan ke kolom yang

telah disediakan dan untuk mengirim pengaduan *User* harus menekan tombol kirim. Setelah data terkirim akan muncul notifikasi seperti pada gambar B. Rancangan halaman pengaduan keluhan cuti dapat dilihat pada gambar 3.36.

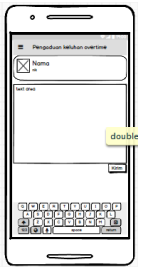
Gambar A

Gambar B

Gambar 3.36 Rancangan halaman Pengaduan keluhan cuti

8. Perancangan antarmuka halaman pengaduan keluhan overtime

Berdasarkan *activity* diagram pada gambar 3.9 maka dibuat rancangan antarmuka halaman keluhan *overtime*. Rancangan halaman pengaduan keluhan *overtime* berisi nama dan nik pengguna. Pengaduan keluhan dapat di *input*kan ke kolom yang telah disediakan dan untuk mengirim keluhan *User* harus menekan tombol kirim. Setelah data terkirim akan muncul notifikasi seperti pada gambar B. Rancangan halaman pengaduan keluhan *overtime* dapat dilihat pada gambar 3.37.



Gambar A

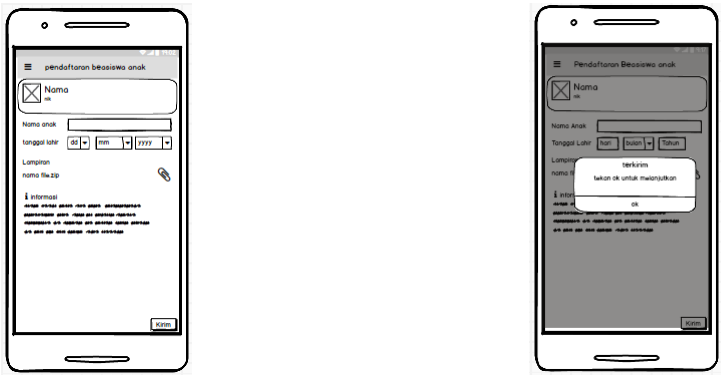
Gambar B

Gambar 3.37 Rancangan halaman Pengaduan keluhan *overtime*

9. Perancangan antarmuka halaman pendaftaran beasiswa anak

Berdasarkan *activity* diagram pada gambar 3.10 maka dibuat rancangan antarmuka halaman pendaftaran beasiswa. Rancangan halaman pendaftaran beasiswa anak menapilkan nama dan nik pengguna. Data-data yang harus di

*input*kan oleh pengguna adalah nama dan tanggal lahir anak. Untuk pendaftaran beasiswa pengguna harus melampirkan dokumen yang diperlukan kedalam 1 dokumen dengan format yang telah ditentukan oleh aplikasi. Informasi mengenai lampiran dokumen di muat pada bagian informasi pada halaman pendaftaran beasiswa anak. Setelah data terkirim akan muncul notifikasi seperti pada gambar B. Rancangan halaman pendaftaran beasiswa anak dapat dilihat pada gambat 3.38.

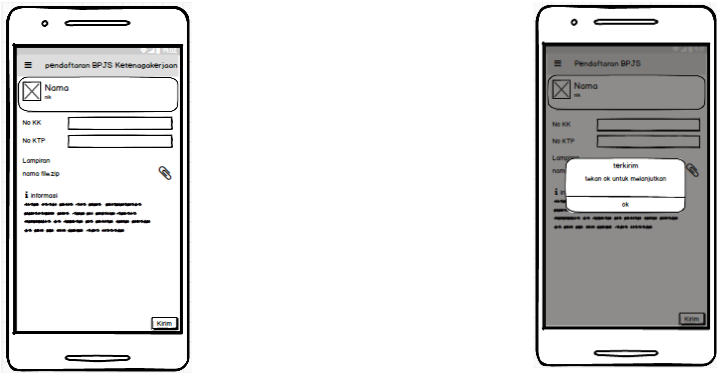


Gambar A Gambar B

Gambar 3.38 Rancangan halaman pendaftaran beasiswa anak

10. Perancangan antarmuka halaman pendaftaran BPJS

Berdasarkan *activity* diagram pada gambar 3.11 maka dibuat rancangan antarmuka halaman pendaftaran BPJS. Rancangan halaman pendaftaran BPJS, *User* harus memasukkan nomor KTP dan nomor Kartu Keluarga. Dokumen yang diperlukan dilampirkan dalam 1 dokumen dengan format yang telah ditentukan pada aplikasi. Informasi tentang lampiran dokumen ditampilkan pada informasi halaman pendaftaran BPJS. Setelah data terkirim akan muncul notifikasi seperti pada gambar B. Rancangan halaman pendaftaran BPJS dapat dilihat pada gambar 3.39



Gambar A Gambar B

Gambar 3.39 Rancangan halaman pendaftaran BPJS

11. Perancangan antarmuka halaman perubahan data id card

Berdasarkan *activity* diagram pada gambar 3.12 maka dibuat rancangan antarmuka halaman perubahan *Id card*. Rancangan halaman perubahan data *id card*, *User* harus meng*input*kan alasan perubahan *id card* dan juga melampirkan *id card* lama kedalam 1 file dokumen yang telah ditentukan pada aplikasi. Informasi tentang lampiran dokumen perubahan *id card* ditampilkan pada bagian informasi. Setelah data terkirim akan muncul notifikasi seperti pada gambar B. Rancangan halaman perubahan data *id card* dapat dilihat pada gambar 3.40.

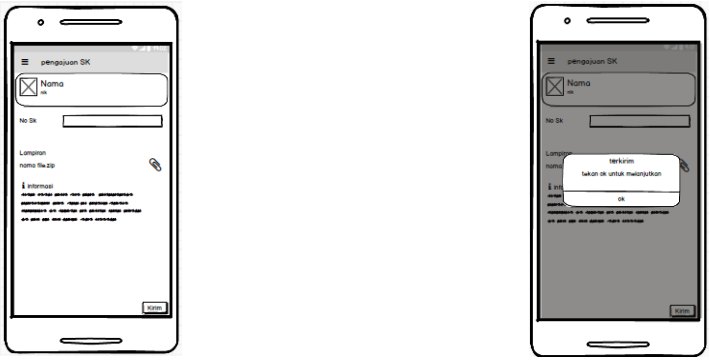
Gambar A Gambar B

Gambar 3.40 Rancangan halaman perubahan data *id card*

12. Perancangan antarmuka halaman surat keterangan

Berdasarkan *activity* diagram pada gambar 3.13 maka dibuat rancangan antarmuka halaman surat keterangan. Rancangan halaman surat ketarangan, pengguna harus mengisi nomor surat dan jenis surat yang akan di lampirkan.

Lampiran dokumen surat keputusan dilampirkan dalam 1 dokumen yang telah ditentukan pada aplikasi. Setelah data terkirim akan muncul notifikasi seperti pada gambar B. Rancangan halaman pengajuan surat keputusan dapat dilihat pada gambar 3.41.

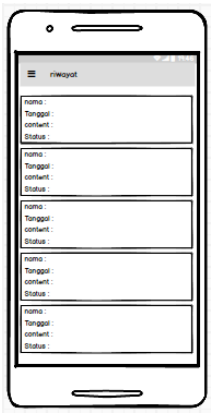


Gambar A Gambar B

Gambar 3.41 Rancangan halaman pengajuan surat keterangan

13. Perancangan antarmuka halaman riwayat pengiriman data

Rancangan halaman riwayat pengiriman data berisi tentang data-data yang telah dikirimkan oleh *User*, data yang dikirim oleh *User* memiliki 2 status yaitu status menunggu dan diterima. Status menunggu berarti data yang dikirim belum diproses oleh admin, sedangkan status diterima berarti data telah di terima dan diproses oleh admin. Rancangan halaman riwayat pengiriman data dapat dilihat pada gambar 3.42.



Gambar 3.42 Rancangan halaman riwayat pengiriman data