

**PENGEMBANGAN SISTEM INFORMASI MANAJEMEN  
PEGAWAI BERBASIS MOBILE PADA PT XYZ  
MENGUNAKAN FRAMEWORK REACT NATIVE**

(Laporan Tugas Akhir Mahasiswa)

**Oleh**

**Irfandi Iqbal Abimanyu  
NPM 19753027**



**POLITEKNIK NEGERI LAMPUNG  
BANDAR LAMPUNG  
2023**

**PENGEMBANGAN SISTEM INFORMASI MANAJEMEN  
PEGAWAI BERBASIS MOBILE PADA PT XYZ  
MENGUNAKAN FRAMEWORK REACT NATIVE**

(Laporan Tugas Akhir Mahasiswa)

**Oleh**

**Irfandi Iqbal Abimanyu  
NPM 19753027**



**POLITEKNIK NEGERI LAMPUNG  
BANDAR LAMPUNG  
2023**

## KATA PENGANTAR

Puji dan syukur penulis panjatkan ke hadirat Allah SWT karena dengan limpahan rahmat dan hidayah-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul “Pengembangan Sistem Informasi Manajemen Pegawai Berbasis Mobile Pada PT XYZ Menggunakan Framework React Native”. Penulis mendapat bimbingan dan dukungan dari berbagai pihak, penulis menyampaikan rasa terima kasih kepada :

1. Arif Makhsun, S.E., M.S.Ak. Selaku ketua jurusan ekonomi dan bisnis Politeknik Negeri Lampung.
2. Dewi Kania Widyawati, S.Kom., M.Kom. Selaku ketua program studi manajemen informatika Politeknik Negeri Lampung sekaligus dosen penguji I yang telah memberikan arahan dan bimbingan dalam proses penulisan tugas akhir ini.
3. Imam Asrowardi, S.Kom., M.Kom. Selaku dosen pembimbing I yang telah memberikan bimbingan, saran, dan masukan kepada penulis sehingga tugas akhir ini dapat terselesaikan dengan baik.
4. Dr. Septafiansyah Dwi Putra, S.T., M.T. Selaku dosen pembimbing II yang telah memberikan ilmu, bimbingan, motivasi dan saran selama kuliah dan penyusunan tugas akhir ini.
5. Oki Arifin, S.Kom., M.Cs. selaku Dosen penguji II yang telah memberikan saran dalam penyusunan tugas akhir.
6. Seluruh dosen dan teknisi program studi manajemen informatika Politeknik Negeri Lampung yang telah memberikan ilmu dan motivasi selama kuliah.
7. Angga, Selaku pimpinan PT XYZ.
8. Fahma Abdurrahman Selaku pembimbing lapang PT XYZ.
9. Teman-teman angkatan 19 program studi manajemen informatika yang sudah banyak membantu dan saling memberikan dukungan.
10. Bapak Trimiadi dan Ibu Nur Handayani selaku orang tua penulis yang selalu memberikan dukungan dan doa demi keberhasilan anaknya.
11. Shafa dan Hanum selaku adik penulis yang selalu menjadi motivasi terbesar penulis dalam menyelesaikan tugas akhir ini.

12. Keluarga penulis yang lain yang selalu mendukung dan memberikan doanya kepada penulis.
13. Teman-teman penulis yang telah memberikan dukungan dalam menyelesaikan penulisan tugas akhir ini.

Pada penulisan tugas akhir ini penulis menyadari bahwa masih banyak kekurangan, oleh karena itu penulis mengharapkan kritik dan saran. Semoga penulisan tugas akhir ini bermanfaat bagi yang membacanya.

Bandar Lampung, 22 Februari 2023

Penulis

# DAFTAR ISI

	Halaman
<b>KATA PENGANTAR</b> .....	iii
<b>DAFTAR ISI</b> .....	v
<b>DAFTAR GAMBAR</b> .....	vii
<b>DAFTAR TABEL</b> .....	ix
<b>BAB I. PENDAHULUAN</b> .....	1
1.1. Latar Belakang .....	1
1.2. Tujuan .....	2
1.3. Kerangka Pemikiran.....	3
1.4. Kontribusi .....	4
<b>BAB II. TINJAUAN PUSTAKA</b> .....	5
2.1. Sistem Informasi Manajemen .....	5
2.2. Arsip.....	5
2.3. Aplikasi Mobile .....	7
2.3.1. JavaScript .....	9
2.3.2. React Native .....	9
2.3.3. NativeBase .....	10
2.3.4. Application Programming Interface.....	10
2.3.5. Representational State Transfer .....	11
2.4. Perancangan Sistem .....	12
2.4.1. Mapping Chart .....	13
2.4.2. Unified Modelling Language .....	14
2.5. Metode Pengembangan Sistem .....	16
2.5.1. Rapid Application Development (RAD).....	16
2.6. Pengujian Sistem.....	18
2.6.1. System Usability Scale.....	18
2.7. Jurnal terkait.....	19
<b>BAB III. METODOLOGI PELAKSANAAN</b> .....	21
3.1. Tempat dan Waktu .....	21
3.2. Alat dan Bahan.....	21
3.2.1. Perangkat Keras ( <i>Hardware</i> ) .....	21

3.2.2. Perangkat Lunak ( <i>Software</i> ).....	21
3.3. Metode Pengumpulan Data.....	22
3.4. Metode Pengembangan Sistem .....	22
3.4.1. Requirements Planning .....	22
3.4.2. User Design.....	22
3.4.3. Construction .....	22
3.4.4. Cutover.....	23
<b>BAB IV. HASIL DAN PEMBAHASAN .....</b>	<b>24</b>
4.1. Hasil dan Pembahasan .....	24
4.1.1. Requirements Planning .....	24
4.1.2. User Design.....	27
4.1.3. Construction .....	51
4.1.4. Cutover.....	51
<b>BAB V. KESIMPULAN DAN SARAN .....</b>	<b>60</b>
5.1. Kesimpulan .....	60
5.2. Saran .....	60
<b>DAFTAR PUSTAKA .....</b>	<b>61</b>
<b>LAMPIRAN.....</b>	<b>64</b>

## DAFTAR GAMBAR

Gambar	Halaman
Gambar 1. Kerangka pemikiran. ....	3
Gambar 2. Fase metode RAD. ....	17
Gambar 3. <i>Mapping chart</i> proses bisnis yang sedang berjalan.....	25
Gambar 4. <i>Mapping chart</i> proses bisnis yang diusulkan. ....	26
Gambar 5. Rancangan <i>use case</i> diagram.....	27
Gambar 6. <i>Activity diagram login</i> . ....	28
Gambar 7. <i>Activity diagram</i> data pribadi.....	29
Gambar 8. <i>Activity diagram</i> riwayat pendidikan.....	30
Gambar 9. <i>Activity diagram</i> riwayat mutasi. ....	31
Gambar 10. <i>Activity diagram</i> riwayat cuti.....	32
Gambar 11. <i>Activity diagram</i> data lamaran. ....	33
Gambar 12. <i>Activity diagram reset password</i> . ....	34
Gambar 13. <i>Activity diagram</i> pengajuan cuti. ....	35
Gambar 14. <i>Wireframe interface splash screen</i> . ....	36
Gambar 15. <i>Wireframe interface login screen</i> . ....	37
Gambar 16. <i>Wireframe interface home screen</i> . ....	37
Gambar 17. <i>Wireframe interface</i> data pribadi <i>screen</i> . ....	38
Gambar 18. <i>Wireframe interface</i> kontak & akun <i>screen</i> . ....	39
Gambar 19. <i>Wireframe interface reset password screen</i> . ....	39
Gambar 20. <i>Wireframe interface</i> riwayat pendidikan <i>screen</i> . ....	40
Gambar 21. <i>Wireframe interface</i> riwayat mutasi <i>screen</i> . ....	41
Gambar 22. <i>Wireframe interface</i> riwayat cuti <i>screen</i> . ....	41
Gambar 23. <i>Wireframe interface</i> pengajuan cuti <i>screen</i> .....	42
Gambar 24. <i>Wireframe interface</i> data lamaran <i>screen</i> . ....	43
Gambar 25. Desain <i>user interface splash screen</i> . ....	44
Gambar 26. Desain <i>user interface login screen</i> . ....	44
Gambar 27. Desain <i>user interface home screen</i> .....	45
Gambar 28. Desain <i>user interface</i> data pribadi <i>screen</i> . ....	46
Gambar 29. Desain <i>user interface</i> kontak & akun <i>screen</i> . ....	46

Gambar 30. Desain <i>user interface</i> reset password screen. ....	47
Gambar 31. Desain <i>user interface</i> riwayat pendidikan screen. ....	48
Gambar 32. Desain <i>user interface</i> riwayat mutasi screen.....	48
Gambar 33. Desain <i>user interface</i> riwayat cuti screen. ....	49
Gambar 34. Desain <i>user interface</i> pengajuan cuti screen.....	50
Gambar 35. Desain <i>user interface</i> pengajuan cuti screen.....	50
Gambar 36. Grafik persentase analisis aspek ketertarikan.....	52
Gambar 37. Grafik persentase analisis kompleksitas aplikasi. ....	52
Gambar 38. Grafik persentase analisis aspek efektivitas aplikasi.....	53
Gambar 39. Grafik persentase analisis aspek teknik aplikasi. ....	54
Gambar 40. Grafik analisis aspek fitur-fitur aplikasi.....	54
Gambar 41. Grafik analisis aspek keseharian aplikasi.....	55
Gambar 42. Grafik analisis aspek pemahaman aplikasi.....	56
Gambar 43. Grafik analisis aspek kejelasan sistem. ....	56
Gambar 44. Grafik analisis aspek kelancaran aplikasi.....	57
Gambar 45. Grafik analisis aspek penguasaan aplikasi. ....	58
Gambar 46. Hasil analisis keseluruhan. ....	59
Gambar 47. Skala SUS.....	59



## DAFTAR TABEL

<b>Tabel</b>	<b>Halaman</b>
Tabel 1. Metode HTTP. ....	12
Tabel 2. Simbol-simbol <i>mapping chart</i> . ....	13
Tabel 3. Simbol-simbol diagram <i>use case</i> . ....	14
Tabel 4. Lanjutan simbol-simbol <i>use case</i> . ....	15
Tabel 5. Simbol-simbol <i>activity diagram</i> . ....	15
Tabel 6. Lanjutan simbol-simbol <i>activity diagram</i> . ....	16

# **BAB I. PENDAHULUAN**

## **1.1. Latar Belakang**

Perkembangan teknologi informasi dewasa ini semakin pesat dan masif, kemajuan teknologi informasi ini memicu berbagai bidang untuk menerapkan teknologi informasi di setiap aktivitasnya. Hampir semua instansi baik pendidikan, pemerintahan, maupun swasta berlomba-lomba menerapkan teknologi informasi untuk memfasilitasi proses pelayanan sehingga mampu memberikan pelayanan yang terbaik bagi pelanggannya atau mempermudah pengelolaan internal instansinya. Salah satu bentuk penerapan teknologi informasi yang umum dijumpai di setiap instansi adalah sistem informasi manajemen pegawai. Sistem informasi manajemen merupakan sebuah sistem yang menyediakan fungsi manajemen seperti perencanaan, pengendalian, dan operasional pada sebuah instansi dengan cara yang efektif sehingga menghasilkan informasi yang dibutuhkan secara cepat, tepat dan akurat untuk membantu proses pengambilan keputusan dengan mudah (Sadikin & Wiranda, 2022). Penerapan sistem informasi manajemen kepegawaian dapat meningkatkan efektivitas pengelolaan sumber daya manusia. Sistem informasi manajemen kepegawaian tersebut dapat mempercepat dan mempermudah pengelolaan data pegawai, serta memudahkan dalam pengambilan keputusan terkait pengelolaan sumber daya manusia (A. Wibisono, 2020).

Pengelolaan arsip merupakan salah satu bagian dari sistem manajemen informasi yang penting dilakukan di sebuah instansi. Arsip yang dikelola dengan baik dan teratur mampu meningkatkan kinerja instansi dalam kegiatan administrasi dan pengambilan keputusan. Apabila arsip tidak dikelola dengan baik maka akan menghambat instansi dalam mengambil keputusan dan memperlambat proses administrasi (Hendriyani, 2021). Arsip yang dikelola di sebuah instansi meliputi semua arsip yang berkaitan, salah satunya yaitu arsip kepegawaian. Arsip kepegawaian merupakan salah satu jenis arsip yang berisi kumpulan data pegawai seperti daftar riwayat hidup, surat lamaran, surat keputusan pengangkatan, dll. Untuk itu, optimalisasi pengelolaan arsip kepegawaian menjadi penting dilakukan di sebuah instansi seperti PT XYZ.

PT XYZ merupakan Badan Usaha Milik Negara (BUMN) yang bertanggung jawab membangkitkan dan menyediakan listrik di Sumatera bagian tengah dan selatan. PT XYZ merupakan pusat administrasi pembangkitan, semua unit pembangkitan yang ada di provinsi Lampung direncanakan, dipelihara, dan diawasi oleh PT XYZ. Perusahaan ini memiliki beberapa divisi bagian yaitu : *Engineering*, *Ophar* (operasi dan pemeliharaan), dan *KSA* (logistik, akuntansi, SDM dan umum). Arsip kepegawaian PT XYZ dikelola di bagian SDM yang berisi data pribadi, keluarga, kenaikan pangkat, mutasi, cuti, dll. Pengelolaan arsip dari pegawai diterima sampai dengan pegawai pensiun masih dilakukan dengan cara manual. Pemberkasan arsip pegawai masih disimpan di dalam ordner map dan dikelompokkan berdasarkan tahun masuk pegawai. Proses perubahan data dan pencarian berkas pegawai yang lama menyebabkan proses administrasi pegawai yang dilakukan belum efektif dan efisien.

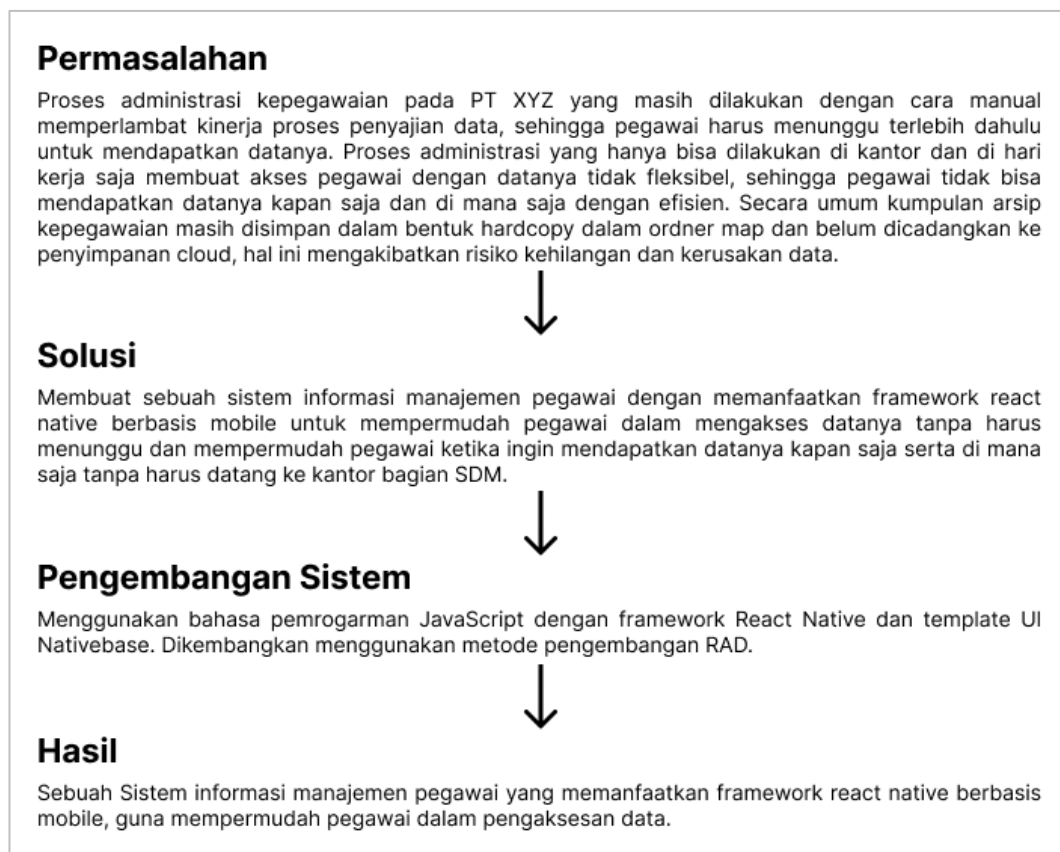
Permasalahan dalam proses administrasi tersebut dapat diselesaikan dengan mengembangkan sebuah sistem informasi manajemen pegawai berbasis *mobile* menggunakan *framework* react native. Sistem ini terdiri dari 2 bagian utama yaitu *web* dan *mobile*. Penulis hanya berfokus pada pengembangan sistem berbasis *mobile* yang menggunakan *backend* (REST API) yang sama dengan sistem *web*. Sistem ini dapat mempermudah pegawai ketika ingin mendapatkan data yang diinginkan tanpa harus menunggu proses pencarian di bagian SDM kapan saja dan dimana saja. Sistem akan dibangun menggunakan bahasa pemrograman JavaScript, dengan *framework* React Native dan NativeBase. Manfaat yang diharapkan dari tugas akhir ini adalah untuk memfasilitasi kebutuhan data pegawai dan membantu memfasilitasi perusahaan untuk melakukan pencadangan arsip kepegawaian.

## 1.2. Tujuan

Tujuan tugas akhir ini adalah mendesain dan membangun sistem informasi manajemen pegawai berbasis *mobile* untuk diterapkan di PT XYZ. Sistem informasi ini berfungsi untuk memfasilitasi pegawai dalam proses pencarian data dan memudahkan pegawai ketika ingin mendapatkan informasi kepegawaian yang berkaitan dengan dirinya. Data yang dapat disediakan sistem ini yaitu data pribadi pegawai, data riwayat pendidikan, surat keputusan mutasi, surat cuti, surat pensiun, dll.

### 1.3. Kerangka Pemikiran

Pengolahan arsip kepegawaian pada PT XYZ saat ini belum efektif dan efisien. Penyimpanan datanya masih disimpan di dalam lemari dan belum dicadangkan ke *cloud* sehingga berisiko hilang. Hal ini dapat menurunkan kinerja bagian SDM ketika melayani pegawai yang membutuhkan datanya. Selain itu, proses pelayanan yang hanya dapat dilakukan di hari kerja dapat mengurangi fleksibilitas akses pegawai terhadap datanya. Untuk mengatasi permasalahan tersebut perlu adanya sebuah sistem informasi yang dapat membantu pegawai untuk mengakses data yang diinginkan dengan cepat kapan saja dan di mana saja. Sebuah solusi yang muncul untuk menyelesaikan permasalahan tersebut adalah “Pengembangan Sistem Informasi Manajemen Pegawai Berbasis Mobile Pada PT XYZ Menggunakan Framework React Native” dengan menggunakan metode pengembangan RAD (*Rapid Application Development*). Tahapan kerangka pemikiran dalam pembuatan sistem ini disajikan pada Gambar 1.



Gambar 1. Kerangka pemikiran.

#### 1.4. Kontribusi

Pengembangan sistem informasi manajemen pegawai berbasis *mobile* menggunakan *framework* React Native diharapkan berhasil mengoptimalkan proses administrasi arsip kepegawaian pada bagian SDM di PT XYZ. Penerapan sistem informasi ini dapat memfasilitasi proses yang sebelumnya manual menjadi otomatis sehingga informasi yang disajikan untuk pegawai telah sesuai dengan data yang terbaru secara *real-time*. Selain itu, penggunaan sistem informasi ini dapat mempermudah pegawai dalam mengakses arsipnya melalui *smartphone* mereka tanpa harus mengurusnya langsung ke bagian SDM. Hal ini akan meningkatkan kinerja dan efisiensi proses administrasi perusahaan.

## **BAB II. TINJAUAN PUSTAKA**

### **2.1. Sistem Informasi Manajemen**

Sistem informasi manajemen merupakan suatu metode pengorganisasian yang saling berkaitan dan saling berinteraksi antar komponen dalam sebuah kesatuan untuk menghasilkan informasi yang dibutuhkan manajer. Sistem informasi manajemen memfasilitasi proses pengambilan keputusan dalam menerapkan fungsi manajemen seperti perencanaan, pengendalian, dan pengorganisasian suatu instansi supaya dapat dilaksanakan dengan cepat dan efektif. Sistem informasi manajemen bertujuan untuk mengatasi berbagai permasalahan terkait dengan pengaturan suatu instansi dalam menjalankan proses bisnisnya seperti permasalahan layanan, biaya operasional, strategi bisnis, dan proses operasional instansi lainnya. Terdapat 5 komponen utama yang saling berkaitan dalam satu kesatuan dalam sistem informasi manajemen yaitu manusia (*brainware*), prosedur bisnis, data, perangkat keras (*hardware*) dan perangkat lunak (*software*) (Sadikin & Wiranda, 2022). Berikut ini manfaat penggunaan sistem informasi manajemen (Hutahaeen dkk., 2021).

1. Mempermudah dalam perencanaan sehingga lebih efektif dan efisien.
2. Menjadi sarana untuk menganalisis pelaksanaan dan keperluan.
3. Meningkatkan produktivitas dan kinerja instansi.
4. Mengurangi biaya operasional instansi.
5. Menghasilkan informasi yang aktual dan *real-time* bagi pengguna yang membutuhkan tanpa perantara.

### **2.2. Arsip**

Arsip adalah semua rekaman yang terekam dalam berbagai media baik tertulis, berupa gambar, atau berupa rekaman (dalam bentuk audio atau video) yang dihasilkan oleh suatu instansi. Suatu dokumen dapat dianggap sebagai arsip jika dalam dokumen tersebut mengandung informasi yang penting bagi sebuah instansi pada masa lalu, masa kini, maupun di masa yang akan datang. Pengelolaan arsip di sebuah instansi harusnya dikelola dengan baik dan teratur untuk menghindari

kesalahpahaman informasi yang disajikan sehingga pengambilan keputusan dapat dilakukan dengan benar (Rosalin, 2017).

Menurut Undang-Undang Republik Indonesia No. 43 Tahun 2009 tentang Kearsipan, Arsip dibagi menjadi beberapa jenis yaitu :

1. Arsip dinamis merupakan arsip yang dapat digunakan secara langsung dan dalam jangka waktu tertentu.
2. Arsip statis merupakan arsip yang memiliki nilai kesejarahan atau yang sudah habis masa retensinya.
3. Arsip vital merupakan arsip yang dipergunakan sebagai syarat dasar dalam proses operasional. Arsip ini tidak dapat diperbarui, dan keberadaannya tidak dapat digantikan apabila hilang atau rusak.
4. Arsip aktif merupakan arsip yang sering digunakan secara terus-menerus.
5. Arsip terjaga merupakan arsip yang berkaitan dengan negara yang keberadaannya mempengaruhi keberlangsungan hidup bangsa dan negara yang harus dilindungi, dijaga keamanannya, dan keutuhannya.
6. Arsip umum merupakan arsip yang berkategori selain arsip terjaga.

Berikut ini beberapa jenis arsip lainnya (Sugiarto & Wahyono, 2014).

1. Arsip berdasarkan subjeknya yaitu arsip kepegawaian, arsip pemasaran, arsip keuangan, arsip pendidikan, dan sebagainya.
2. Arsip berdasarkan bentuk medianya yaitu berkas surat, arsip digital, rekaman baik audio maupun video, berkas gambar, dan lain-lain.
3. Arsip berdasarkan nilai kegunaannya yaitu :
4. Arsip yang bernilai administrasi seperti prosedur kerja.
5. Arsip yang bernilai hukum seperti akta kelahiran, akta tanah, dll.
6. Arsip yang bernilai pendidikan seperti silabus, kurikulum, dll.
7. Arsip yang bernilai sejarah seperti laporan bulanan, tahunan, dll.
8. Arsip yang bernilai keuangan seperti kuitansi, nota pembayaran, dll.
9. Arsip yang bernilai informasi seperti pengumuman dan undangan.
10. Arsip yang bernilai ilmiah seperti laporan penelitian, skripsi, dll.
11. Arsip berdasarkan fungsinya yaitu arsip dinamis dan arsip statis.
12. Arsip berdasarkan tempat pengelolaannya yaitu arsip pusat dan arsip unit.

13. Arsip berdasarkan tingkatan keasliannya yaitu arsip asli, arsip tembusan, arsip salinan, dan arsip petikan.
14. Arsip berdasarkan kekuatan hukumnya yaitu arsip otentik, dan arsip tidak otentik.
15. Arsip berdasarkan kepentingannya yaitu :
  - a. Arsip yang tidak berguna seperti surat undangan.
  - b. Arsip yang berguna seperti surat cuti, surat izin, dan presensi pegawai.
  - c. Arsip yang penting seperti laporan keuangan, surat keputusan, dll.
  - d. Arsip yang vital seperti akta kelahiran, ijazah, sertifikat pelatihan, dll.

Salah satu jenis arsip berdasarkan subjeknya yaitu arsip kepegawaian. Arsip kepegawaian merupakan semua arsip yang ada hubungannya dengan masalah kepegawaian, seperti surat lamaran, daftar riwayat hidup, data pribadi, surat-surat keputusan pegawai, absensi pegawai, dan lain-lain (Suparman, 2020).

### 2.3. Aplikasi Mobile

Perangkat lunak (*software*) merupakan serangkaian detail instruksi yang ditulis menggunakan bahasa pemrograman untuk memberitahu komputer apa yang harus dilakukan guna mencapai sebuah tujuan tertentu. Keberadaan perangkat lunak berperan penting sebagai penghubung antara pengguna dengan perangkat keras (*hardware*) komputer, karena tanpa adanya perangkat lunak komputer hanyalah kumpulan perangkat keras yang tidak dapat dioperasikan (Fox, 2013). Secara umum perangkat lunak dibagi menjadi dua jenis yaitu:

#### 1. Perangkat lunak sistem

Perangkat lunak sistem merupakan kumpulan program atau instruksi yang dibuat untuk membentuk sebuah sistem operasi yang menghubungkan perangkat keras komputer dengan perangkat lunak aplikasi. Komponen inti dari sebuah perangkat lunak sistem disebut sebagai *kernel* yang dijalankan saat pertama kali komputer dihidupkan. *Kernel* memiliki beberapa tugas yaitu menangani manajemen proses, manajemen sumber daya, dan manajemen memori pada komputer. Tanpa adanya *kernel* perangkat lunak aplikasi tidak dapat dijalankan secara efisien atau bahkan tidak dapat dijalankan sama sekali. Beberapa contoh sistem operasi yang umum digunakan adalah *linux*, *windows*, dan *macOS*.



## 2. Perangkat lunak aplikasi

Perangkat lunak aplikasi adalah kumpulan program yang dijalankan pengguna untuk menyelesaikan beberapa tugas atau fungsi-fungsi lainnya yang berjalan di atas perangkat lunak sistem. Pengelompokan perangkat lunak aplikasi bergantung pada kegunaannya bagi pengguna.

- a. Aplikasi pendukung produktivitas yang digunakan semua orang mencakup aplikasi pengolah kata, aplikasi pengolah angka, aplikasi presentasi, sistem manajemen data, aplikasi kontak, aplikasi pengatur data dll.
- b. Aplikasi desain yang digunakan para desainer mencakup aplikasi menggambar, aplikasi edit gambar, aplikasi pengolah *vector*, aplikasi 3D, dan aplikasi pengolah video.
- c. Aplikasi pengolah suara yang digunakan para musisi.
- d. Aplikasi pengembangan yang digunakan para *programmer* mencakup aplikasi editor kode, aplikasi IDE, aplikasi pelacakan perubahan kode, *library* atau *framework*, aplikasi *debugging*, aplikasi visualisasi kode, dll.
- e. Aplikasi daring mencakup aplikasi yang menggunakan konektivitas internet ketika penggunaannya seperti aplikasi email, browser web, aplikasi FTP, aplikasi *remote*, dsb.
- f. Aplikasi lainnya mencakup aplikasi di luar kategori aplikasi yang dijelaskan di atas seperti aplikasi hiburan (pemutar musik dan video), dan permainan.

Perangkat lunak dapat berjalan di beberapa *platform* seperti komputer, perangkat *mobile*, perangkat jaringan, atau bahkan perangkat *embedded system*. Perangkat lunak yang dijalankan menggunakan perangkat *mobile* disebut aplikasi *mobile*. Aplikasi *mobile* merupakan kumpulan beberapa perangkat lunak atau program yang berjalan pada perangkat *mobile* yang mampu menjalankan tugas-tugas tertentu berdasarkan keinginan pengguna. Aplikasi *mobile* merupakan terobosan terbaru akibat berkembang pesatnya teknologi informasi dan komunikasi. Kemudahan penggunaan, tampilan yang *user-friendly*, mudah didapat, mudah diunduh, dan dapat dijalankan di berbagai perangkat *mobile* merupakan kelebihan aplikasi *mobile*. Penggunaannya yang luas mulai dari sarana komunikasi, menjelajah internet, jejaring sosial, pendidikan, bisnis, sampai dengan hiburan

mampu terpenuhi oleh aplikasi *mobile*. Jumlah pasar aplikasi *mobile* yang besar berbanding lurus dengan bertambahnya jumlah pengembang, penerbit, dan penyedia aplikasi menjadikan aplikasi *mobile* sebuah teknologi baru yang menguntungkan penggunanya (Islam dkk., 2010).

Aplikasi *mobile* saat ini dapat dikembangkan dengan berbagai bahasa pemrograman seperti Java, Kotlin, Dart, Objective-C, Swift, dan JavaScript. Selain itu para pengembang telah berhasil mengembangkan berbagai *framework* yang dapat digunakan untuk mengembangkan aplikasi *mobile* baik Android, IOS, maupun lintas *platform* dengan stabil dan lebih cepat. Salah satu *framework* yang bisa digunakan untuk mengembangkan aplikasi *mobile* yaitu React Native dengan bahasa pemrograman JavaScript.

### **2.3.1. JavaScript**

JavaScript merupakan bahasa pemrograman tingkat tinggi dinamis yang mampu dikembangkan dengan gaya pemrograman fungsional atau gaya pemrograman berorientasi objek. JavaScript merupakan bahasa pemrograman yang memerlukan interpreter untuk menjalankan kodenya. Dahulu JavaScript hanya dapat dijalankan di dalam lingkungan browser, namun satu dekade terakhir muncul *runtime-environment* yang memfasilitasi JavaScript supaya dapat berjalan di luar lingkungan browser yaitu Node.js. Keberhasilan Node.js membawa JavaScript keluar dari lingkungan browser membuat JavaScript sekarang menjadi bahasa pemrograman yang paling banyak digunakan di kalangan pengembang perangkat lunak (Flanagan, 2020). Selain pengembangan web, saat ini para pengembang JavaScript telah mengembangkan *framework* yang dapat digunakan untuk mengembangkan aplikasi *mobile*. *Framework* yang populer yaitu React Native dan Ionic.

### **2.3.2. React Native**

React Native adalah salah satu kerangka kerja (*framework*) JavaScript besutan Facebook yang digunakan untuk mengembangkan aplikasi baik aplikasi *mobile* (Android dan IOS) maupun aplikasi web. Kerangka kerja React Native dibuat berdasarkan *library* React. Dengan kata lain, React Native memungkinkan pengembang web untuk membuat aplikasi *mobile* yang mirip seperti aplikasi *mobile* yang dibuat secara *native* tanpa harus beralih ke kerangka kerja dan bahasa

pemrograman yang lain. Mirip seperti React, React Native ditulis menggunakan campuran bahasa pemrograman JavaScript dan *markup XML-esque* atau lebih dikenal sebagai JSX. React Native menghubungkan kode JSX yang ditulis dengan *API rendering* asli milik masing-masing *platform* sehingga aplikasi yang dibuat akan ditampilkan menggunakan komponen UI masing-masing *platform* dan akan terasa seperti aplikasi *mobile* lainnya (Eisenman, 2015).

### 2.3.3. NativeBase

*Library* komponen UI/UX (*User Interface/User Experience*) merupakan kumpulan serangkaian kode komponen yang siap digunakan pengguna untuk kebutuhan tertentu. Dengan menggunakan komponen yang sudah disediakan memungkinkan pengguna untuk menyusun tata letak tampilan dan pengalaman pengguna dengan cepat tanpa membuang waktu untuk merancanginya dari awal. Selain itu keuntungan menggunakan *library* komponen sering kali menghasilkan hasil yang lebih stabil dan konsisten baik dari segi UI maupun UX. Salah satu *library* yang populer digunakan dalam pengembangan aplikasi React Native adalah NativeBase (Boduch dkk., 2022).

### 2.3.4. Application Programming Interface

*Application Programming Interface* (API) merupakan kumpulan aturan yang ditentukan oleh pengembang guna memfasilitasi aplikasi untuk dapat berkomunikasi dengan aplikasi lain. API bertindak sebagai perantara yang memproses pertukaran data antar sistem atau aplikasi. Hal ini memungkinkan perusahaan membuka akses dan fungsionalitas aplikasinya untuk dapat dikembangkan atau dihubungkan dengan pengembang pihak ketiga, mitra bisnis, atau divisi internal perusahaan. Manfaat mengembangkan API adalah untuk membantu proses bisnis perusahaan dengan menghubungkan banyak aplikasi berbeda yang digunakan sehingga menghemat waktu pengerjaan dan memudahkan pengembang berkolaborasi dan berinovasi (IBM Cloud Education, 2022).

API menggunakan beberapa protokol yang sering dijumpai dalam pengembangan aplikasi yaitu:

1. SOAP (*Simple Object Access Protocol*)

API SOAP dibangun dengan menggunakan XML yang memungkinkan *endpoint* untuk mengirim dan menerima data melalui SMTP dan HTTP.

2. XML-RPC (*XML-Remote Procedure Call*)

Protokol XML-RPC menggunakan format XML tertentu untuk mentransfer datanya. XML-RPC dibuat sebelum adanya SOAP, akan tetapi jauh lebih sederhana, dan relatif ringan karena menggunakan *bandwidth* yang kecil.

3. JSON-RPC (*JSON-Remote Procedure Call*)

JSON-RPC hampir sama dengan XML-RPC, pembedanya hanya penggunaan JSON (*JavaScript Object Notation*) untuk mentransfer data dan tidak lagi menggunakan XML.

4. REST (*Representational State Transfer*)

REST adalah seperangkat prinsip pengembangan arsitektur web API.

### 2.3.5. Representational State Transfer

*Representational State Transfer* atau REST merupakan arsitektur web yang menyediakan proses pertukaran data antara *server* dan *client* dengan menggunakan API yang terhubung secara *point-to-point*. Arsitektur REST menggunakan prinsip *client-server*. Dengan kata lain *server* REST bertanggung jawab untuk menyediakan sumber daya yang diminta oleh *client* REST ketika melakukan pembuatan, pengambilan, pengubahan, dan penghapusan sumber daya. Arsitektur REST menggunakan format XML dan JSON (*JavaScript Object Notation*) sebagai media pertukaran datanya (Chatterjee & Mamatha, 2020). Ada beberapa prinsip yang perlu diperhatikan ketika mengembangkan REST API yaitu:

1. *Addressability*

Setiap sumber daya harus memiliki setidaknya satu URI (*Uniform Resource Identifier*) yang terkait. URI digunakan untuk menentukan sumber daya atau sekumpulan sumber daya.

## 2. *Statelessness*

Layanan REST adalah layanan yang independen. Setiap permintaan yang dikirimkan menggunakan protokol HTTP (*Hypertext Transfer Protocol*) tidak berhubungan dengan permintaan sebelumnya.

## 3. *Cacheable*

Sumber daya yang ditandai sebagai *cache* dapat disimpan sementara dalam sistem dan dapat digunakan kembali ketika permintaan menghasilkan hasil yang sama. Penandaan sumber daya dapat dilakukan ketika permintaan dimulai.

## 4. *Uniform Interface*

Layanan REST menggunakan kumpulan metode HTTP standar untuk menentukan permintaan yang dibuat. Berikut ini beberapa metode HTTP sederhana yang dapat digunakan dalam layanan REST. Berikut ini metode HTTP yang umum digunakan.

Tabel 1. Metode HTTP.

Metode	Penjelasan
GET	Digunakan untuk mengambil data ( <i>Read</i> ).
POST	Digunakan untuk <i>input</i> data ( <i>Create</i> ).
PUT	Digunakan untuk <i>input</i> data, apabila data sudah ada maka data tersebut akan diperbarui ( <i>Create/Update</i> ).
PATCH	Digunakan untuk memperbarui data ( <i>Update</i> )
DELETE	Digunakan untuk menghapus data ( <i>Delete</i> )

Sumber: (MDN Mozilla, 2022)

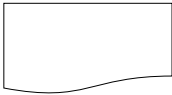



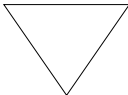
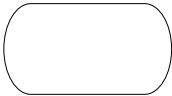

## 2.4. Perancangan Sistem

Perancangan sistem merupakan proses merinci pembuatan sistem menggunakan berbagai teknik, mulai dari deskripsi arsitektur pendukung dan detail komponen sampai dengan kendala pengerjaan yang bertujuan untuk memberikan gambaran yang jelas dalam memenuhi kebutuhan pengguna. Perancangan sistem bisanya digambarkan dalam bentuk diagram alir yang menampilkan proses secara runtut dan diagram-diagram pendukung lainnya (Aziz, N., Pribadi, G., & Nurcahya, 2020).

### 2.4.1. Mapping Chart

*Mapping chart* merupakan diagram alir yang memvisualisasikan langkah-langkah dalam bentuk simbol-simbol grafis yang digunakan untuk membantu proses analisis, penelitian, atau untuk memecahkan masalah tertentu. *Mapping chart* berfungsi untuk menyederhanakan proses agar lebih mudah dilihat dan dipahami menggunakan media simbol-simbol (Mukodimah dkk., 2019).

Tabel 2. Simbol-simbol *mapping chart*.

Simbol	Nama	Fungsi
	Dokumen	Menyatakan <i>input</i> dokumen yang dicetak
	<i>Connector</i>	Menyatakan sambungan satu dengan yang lainnya pada proses berikutnya
	Proses	Proses pengolahan yang ditugaskan oleh komputer
	<i>Display</i>	Menyatakan output yang digunakan
	Arsip	Mengarsipkan data di dalam program
	Terminal	Awalan dan akhiran pada program
	<i>Connecting Line</i>	Menghubungkan simbol dengan simbol lainnya dengan menyatakan suatu alur proses

Sumber: (Fitriyana & Susianto, 2018)



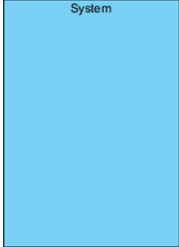

### 2.4.2. Unified Modelling Language

*Unified Modelling Language* atau UML merupakan pemodelan visual yang digunakan untuk mendefinisikan, memvisualisasikan, dan mendokumentasikan rancangan analisis dan desain aplikasi, serta memvisualisasikan arsitektur dalam OOP (*Object Oriented Programming*). UML menjadi sarana yang digunakan untuk memberikan gambaran jelas terkait analisis sistem sebelum dibangun baik secara struktural maupun secara fungsional (Putra & Andriani, 2019). Dalam UML terdapat beberapa diagram yang sering digunakan yaitu:

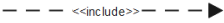

#### 1. *Use Case Diagram*

*Use case* diagram merupakan diagram yang memodelkan perlakuan (*behavior*) sistem yang akan dibuat kepada penggunanya. *Use case* digunakan untuk mendeskripsikan interaksi antara satu atau lebih pengguna dengan sebuah sistem. Melalui *use case* pengembang mendapat gambaran bagaimana interaksi pengguna dengan sistem dan fungsi apa saja yang ada pada sistem tersebut. Notasi *use case* dibagi menjadi beberapa bagian yaitu pengguna (*actor*), sistem atau sub-sistem (*use case*), dan hubungan (*relationship*).

Tabel 3. Simbol-simbol diagram *use case*.

Simbol	Nama	Fungsi
(1)	(2)	(3)
	Aktor	Menggambarkan pengguna yang berinteraksi dan berperan dalam sistem bisnis.
	<i>Use Case</i>	Menggambarkan fungsi yang dapat dilakukan sebuah sistem bisnis.
	Batas Sistem	Menggambarkan batasan sistem bisnis untuk kasus penggunaan khusus.
	<i>Association</i>	Relasi asosiasi antara <i>use case</i> dengan pengguna.

Tabel 4. Lanjutan simbol-simbol *use case*.


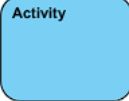
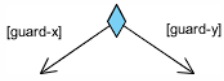
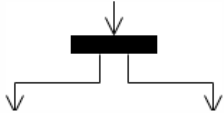
Simbol	Nama	Fungsi
(1)	(2)	(3)
	<i>Includes</i>	Relasi ketika sebuah <i>use case</i> digambarkan menggunakan fungsionalitas dari <i>use case</i> yang lain.
	<i>Extends</i>	Relasi yang digunakan untuk menyertakan perilaku opsional dari <i>use case</i> yang diperluas.

Sumber: (Visual Paradigm, 2022b)

## 2. Activity Diagram

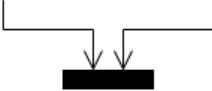


*Activity* diagram merupakan diagram yang memvisualisasikan alir kerja (*workflow*) atau aktivitas dari sebuah sistem yang menjalankan proses bisnis atau menu yang ada pada perangkat lunak. Dengan kata lain *activity* diagram hanya menggambarkan aktivitas yang berjalan pada sistem bukan aktivitas yang dilakukan pengguna (*actor*).

Tabel 5. Simbol-simbol *activity diagram*.

Simbol	Nama	Fungsi
(1)	(2)	(3)
	<i>Initial Node</i>	Menggambarkan titik awal dari serangkaian tindakan atau kegiatan.
	<i>Activity</i>	Menggambarkan serangkaian tindakan atau aktivitas sistem.
	<i>Decision Node</i>	Menggambarkan seleksi kondisi untuk memastikan bahwa aliran kontrol atau aliran aktivitas hanya turun satu jalur.
	<i>Fork Node</i>	Membagi aktivitas sistem menjadi serangkaian aktivitas (atau tindakan) paralel atau bersamaan.



Tabel 6. Lanjutan simbol-simbol *activity diagram*.

Simbol	Nama	Fungsi
(1)	(2)	(3)
	<i>Join Node</i>	Menyatukan kembali aktivitas atau tindakan sistem yang paralel atau bersamaan.
	<i>Swimlane and Partition</i>	Mengelompokkan aktivitas yang dilakukan oleh aktor yang sama atau pengelompokan aktivitas dalam satu partisi.
	<i>Activity Final Node</i>	Menghentikan semua aliran kontrol dan aliran aktivitas atau tindakan

Sumber: (Visual Paradigm, 2022a)

## 2.5. Metode Pengembangan Sistem

### 2.5.1. Rapid Application Development (RAD)

*Rapid Application Development* atau RAD merupakan salah satu metode pengembangan sistem perangkat lunak yang bersifat bertahap (*incremental*). Dalam penerapannya, metode RAD mementingkan waktu siklus pengembangan yang relatif singkat, cepat, pendek, dan berulang (*iterative*) (Kaban dkk., 2022). Metode RAD menggunakan pendekatan yang berfokus pada tahap perancangan dan pembuatan prototipe dengan tujuan mendapatkan umpan balik pengguna secara instan. Perulangan yang konstan dalam mengembangkan sistem berdasarkan umpan balik pengguna dan pembaruan yang cepat membantu mencapai hasil yang sesuai dengan keinginan pengguna. Berikut ini keuntungan menggunakan metode RAD (Glaschenko, 2021).

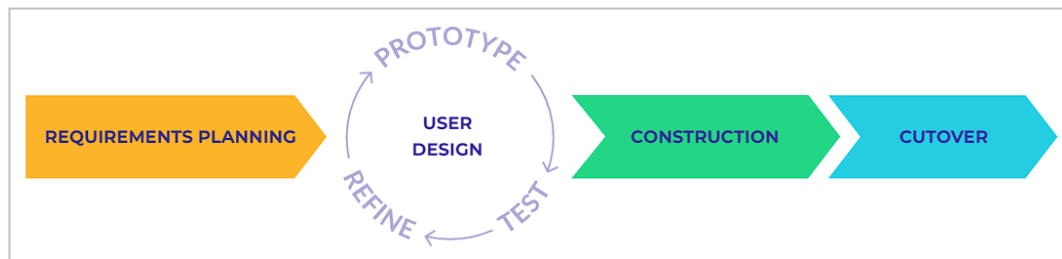
#### 1. Menghasilkan kualitas sistem yang tinggi

Keterlibatan pengguna dalam tahap pembuatan prototipe, sistem yang dihasilkan akan lebih relevan dengan keinginan dan ekspektasi pengguna.

#### 2. Meminimalisir biaya dan risiko pengembangan

Ketika menggunakan metode *waterfall*, pengguna hanya dapat melihat hasil dan memberikan umpan balik ketika proyek diluncurkan. Perubahan yang

terjadi pada proses ini memakan biaya yang mahal dan memakan lebih banyak waktu. Dengan metode pengembangan aplikasi yang cepat, risiko penulisan ulang dari proyek yang telah diluncurkan menjadi lebih minim.



Gambar 2. Fase metode RAD.

Metode RAD memiliki beberapa fase dalam pengerjaannya. Berikut ini fase yang harus dilalui ketika mengembangkan sebuah sistem menggunakan metode RAD.

### 1. *Requirements Planning*

Pada fase ini, pengguna dan tim proyek mengidentifikasi tujuan sistem yang diinginkan. Pengembangan nantinya berfokus untuk mencapai tujuan bisnis dengan kebutuhan yang telah ditetapkan. Kebutuhan yang sebelumnya sudah ditetapkan dapat diubah dan disesuaikan kembali pada tahap pembuatan prototipe.

### 2. *User Design*

*User design* merupakan fase penting dari metode RAD. Pada fase ini, pengembang dimulai dengan mengerjakan prototipe. Tujuan membuat prototipe adalah untuk mendemonstrasikan sebuah gagasan tentang sistem yang akan dibuat kepada klien dengan cepat. Tim mengumpulkan semua umpan balik dari pengguna berdasarkan prototipe yang telah disampaikan. Pada proses inilah kebutuhan awal dapat disesuaikan kembali berdasarkan keinginan pengguna. Dengan umpan balik yang telah dikumpulkan, pengembang akan mengulang langkah pembuatan prototipe, hingga pengguna puas dengan hasilnya.

### 3. *Construction*

Pada fase ini, pengembangan dan pengujian dilakukan untuk mempersiapkan sistem menuju ke tahap produksi. Dalam pengerjaannya fokus utamanya

adalah pada kualitas, skalabilitas, pemeliharaan, dll. Namun, pengguna terus berpartisipasi pada fase ini dan terus memberikan umpan balik saat fitur diimplementasikan. Sedikit penyempurnaan dimungkinkan pada fase ini.

#### **4. *Cutover***

*Cutover* merupakan fase terakhir. Dalam fase ini mencakup *acceptance testing*, peluncuran (*rollout*) dan pelatihan pengguna.

### **2.6. Pengujian Sistem**

Pengujian sistem merupakan proses mencari kesalahan pada setiap komponen sistem, melakukan pencatatan, mengevaluasi semua aspek dan fitur-fitur sistem yang sedang dikembangkan. Berikut ini beberapa aturan dalam pengujian sistem (W. Wibisono & Baskoro, 2002).

1. Pengujian dilakukan dengan mengeksekusi program dengan tujuan menemukan kesalahan
2. Sebuah kasus pengujian dikatakan baik jika dalam proses pengujiannya berpotensi menemukan kesalahan lebih tinggi.
3. Pengujian dikatakan berhasil jika menemukan kesalahan.

#### **2.6.1. System Usability Scale**

*System Usability Scale* merupakan metode evaluasi sebuah aplikasi untuk menilai dan mengukur tingkat kegunaannya dengan menggunakan sebuah kuesioner sederhana yang memiliki sepuluh pernyataan. Pernyataan metode SUS yang bernomor ganjil menggunakan kalimat positif dan pernyataan yang bernomor genap menggunakan kalimat negatif. Selanjutnya untuk menilai hasil kuesioner tersebut cukup menggunakan skala *likert* yang telah ditentukan (Muhammad Nur Fauzi dkk., 2022). Kelebihan menggunakan SUS adalah:

1. SUS mudah digunakan karena tidak menggunakan perhitungan yang rumit.
2. Rentang skor SUS bernilai 0-100.
3. Akurat walau menggunakan sampel yang kecil.
4. Gratis tanpa biaya.

## 2.7. Jurnal terkait

Berikut ini beberapa jurnal terkait yang menjadi bahan pertimbangan dan menjadi bahan referensi pengembangan sistem.

1. Adam dkk. (2022). Dengan judul “Aplikasi Jasa Titip Belanja Berbasis Mobile di Minahasa Utara”. Karya ilmiah ini bertujuan untuk membangun sebuah aplikasi jasa titip belanja supermarket di daerah Minahasa Utara. Aplikasi yang akan dibangun akan tersedia di platform Android untuk Pelanggan dan Kurir serta *website* untuk admin dan juga toko, dengan sisi *front-end* dikembangkan dengan bahasa pemrograman Javascript menggunakan *framework* React Native, React JS, dan sisi *back-end* dikembangkan lewat bahasa pemrograman Javascript dengan memanfaatkan teknologi firebase *real-time database*.
2. Karim & Adriansyah (2022). Dengan judul “Analisis dan Perancangan Aplikasi Mobile untuk Donasi menggunakan Metode Hybrid berbasis React Native”. Karya ilmiah ini dibuat untuk memudahkan pengguna untuk berdonasi barang, dimana pun dan kapan pun. Aplikasi dibangun menggunakan *framework* React Native dengan RESTFul API dan MySQL sebagai bahasa pemrograman *database*. Pengujian yang dilakukan yaitu pengujian fungsional sistem dengan menggunakan teknik *black-box testing* dan *User Acceptance Test* (UAT) serta pengujian dengan menggunakan kuesioner.
3. Nursaid dkk. (2020). Dengan judul “Pengembangan Sistem Informasi Pengelolaan Persediaan Barang Dengan ReactJS Dan React Native Menggunakan Prototype (Studi Kasus : Toko Uda Fajri)”. Karya ilmiah ini dibuat untuk mempermudah pedagang yang memiliki toko Uda Fajri tanpa harus mencatat barang masuk dan keluar dengan menggunakan kertas lagi, selain itu juga dapat melakukan transaksi barang dengan menggunakan telepon genggam yang sudah terpasang aplikasi sistem pengelolaan barang.
4. Mukodimah dkk. (2019). Dengan Judul “Aplikasi Penentuan Bengkel TSM Berkualitas Untuk UKK Siswa SMK Kabupaten Pringsewu Berbasis Mobile”. *Website* untuk mengukur kelayakan lab/bengkel uji kompetensi kejuruan TSM dapat dijadikan sebagai solusi dalam menyelesaikan permasalahan pengukuran kelayakan bengkel TSM SMK secara tepat dan akurat. Dengan menerapkan 5 kriteria yang ada seperti area kerja mesin otomotif, area kerja kelistrikan, area

kerja *chasis* dan pemindah tenaga, ruang penyimpanan dan instruktur dan kelengkapan peralatan. Proses pengukuran kelayakan dapat dilakukan dengan lebih tepat, akurat dan efisien.

5. Prastio & Ani (2018). Dengan judul “Aplikasi Self Service Menu Menggunakan Metode Scrum Berbasis Android (Case Study : Warkobar Café Cikarang)”. Sebuah aplikasi yang diharapkan membantu pelanggan *café*, pelayan, kasir, dapur, dan pemilik *café*. Perancangan sistem akan dibuat dengan tampilan sederhana dan mudah dipahami oleh pelanggan. Dengan begitu pemesanan menu bisa berjalan dengan cepat dan praktis, tanpa melalui proses panjang serta mengurangi kemungkinan terjadinya kesalahan komunikasi antara pelayan dan pelanggan. Sistem ini juga akan meminimalisir adanya antrean tempat duduk dalam *café*, dengan adanya fungsi reservasi dengan pemilihan tempat dan waktu yang sudah ditentukan oleh pelanggan. Sistem ini juga akan menggunakan OS Mobile populer saat ini, yaitu Android.

## **BAB III. METODOLOGI PELAKSANAAN**

### **3.1. Tempat dan Waktu**

Tugas akhir yang berjudul “Pengembangan Sistem Informasi Manajemen Pegawai Berbasis Mobile Pada PT XYZ Menggunakan Framework React Native” ini dilaksanakan pada bulan Juni 2022 hingga bulan Februari 2023 di Politeknik Negeri Lampung. Tugas akhir ini mengambil permasalahan dari tempat Praktik Kerja Lapangan (PKL) pada semester sebelumnya di PT XYZ.

### **3.2. Alat dan Bahan**

Alat dan bahan merupakan kebutuhan wajib yang perlu disiapkan sebelumnya untuk membangun atau membuat sebuah produk. Pada saat penulisan tugas akhir ini alat dan bahan yang diperlukan untuk membangun sebuah produk sistem informasi manajemen pegawai berbasis *mobile* adalah sebagai berikut:

#### **3.2.1. Perangkat Keras (*Hardware*)**

Perangkat keras yang dibutuhkan dalam penulisan dan pengembangan tugas akhir ini yaitu:

1. Laptop/komputer dan kabel *charger*.
2. Tetikus (*mouse*).
3. Papan ketik (*keyboard*).

#### **3.2.2. Perangkat Lunak (*Software*)**

Sedangkan perangkat lunak yang dibutuhkan dalam penulisan dan pengembangan tugas akhir ini yaitu:

1. Sistem operasi (Windows 11).
2. Aplikasi pengolah kata (Microsoft Word 2021).
3. Aplikasi pembuat diagram (Figma/FigJam/EdrawMax)
4. Aplikasi sitasi daring (Mendeley).
5. Aplikasi browser web (Google Chrome).
6. Aplikasi editor kode (Visual Studio Code).
7. Kerangka kerja aplikasi (React Native).
8. Pustaka komponen UI (NativeBase).
9. Aplikasi REST *endpoint testing* (Postman/Insomnia).

### 3.3. Metode Pengumpulan Data

Pengumpulan data yang dilakukan penulis untuk “Pengembangan Sistem Informasi Manajemen Pegawai Berbasis Mobile Pada PT XYZ Menggunakan Framework React Native” menggunakan metode wawancara secara langsung kepada pihak PT XYZ yang bersangkutan yaitu Ibu Dita selaku pegawai divisi SDM PT XYZ, Bapak Angga selaku manajer PT XYZ dan berkonsultasi dengan Bapak Fahma selaku pegawai divisi Engineering sekaligus pembimbing lapangan penulis ketika PKL. Informasi yang didapatkan sangat berguna dan membantu dalam pengembangan sistem tugas akhir ini.

### 3.4. Metode Pengembangan Sistem

Metode pengembangan sistem yang digunakan untuk “Pengembangan Sistem Informasi Manajemen Pegawai Berbasis Mobile Pada PT XYZ Menggunakan Framework React Native” menggunakan metode RAD. Berikut ini runtutan fase pengembangan menggunakan metode RAD:

#### 3.4.1. Requirements Planning

Saat merancang sebuah sistem menggunakan metode RAD, hal pertama yang harus dilakukan adalah menyusun rencana kebutuhan sistem melalui pengumpulan data dan observasi pada proses bisnis yang sedang berjalan. Data atau informasi yang telah didapatkan dari pengguna akan digunakan sebagai referensi awal kebutuhan sistem.

#### 3.4.2. User Design

Pada fase *user design*, pengembang akan membuat sebuah prototipe terkait sistem yang akan dibuat. Pembuatan prototipe dimulai dari UML (*use case* dan *activity diagram*), sampai dengan rancangan UI/UX (*user interface/user experience*). Prototipe ini akan terus dikonsultasikan dengan pengguna untuk menyesuaikan kebutuhan sampai pengguna merasa cukup.

#### 3.4.3. Construction

Pada fase *construction*, pengembang akan membangun sistem yang diinginkan pengguna berdasarkan hasil *user design*. Sistem dikembangkan dengan melakukan pengetikan kode menggunakan bahasa pemrograman (*coding*), pengujian fitur-fitur, dan perbaikan terkait *error* atau *bugs* yang terjadi.

#### **3.4.4. Cutover**

Setelah sistem selesai dikembangkan pada fase *construction*, sistem akan diulas kembali oleh PT XYZ menggunakan SUS. Apabila masih ada kekurangan maka kekurangan tersebut akan diperbaiki lagi pada fase *user design* dan tahapan akan diulang kembali sampai aplikasi sesuai dengan keinginan pengguna dan lolos pengujian.



## **BAB IV. HASIL DAN PEMBAHASAN**

### **4.1. Hasil dan Pembahasan**

Pengerjaan sistem manajemen pegawai berbasis *mobile* yang dibangun menggunakan metode RAD melalui beberapa fase yang harus dilalui sebelum aplikasi dinyatakan jadi dan dapat digunakan.

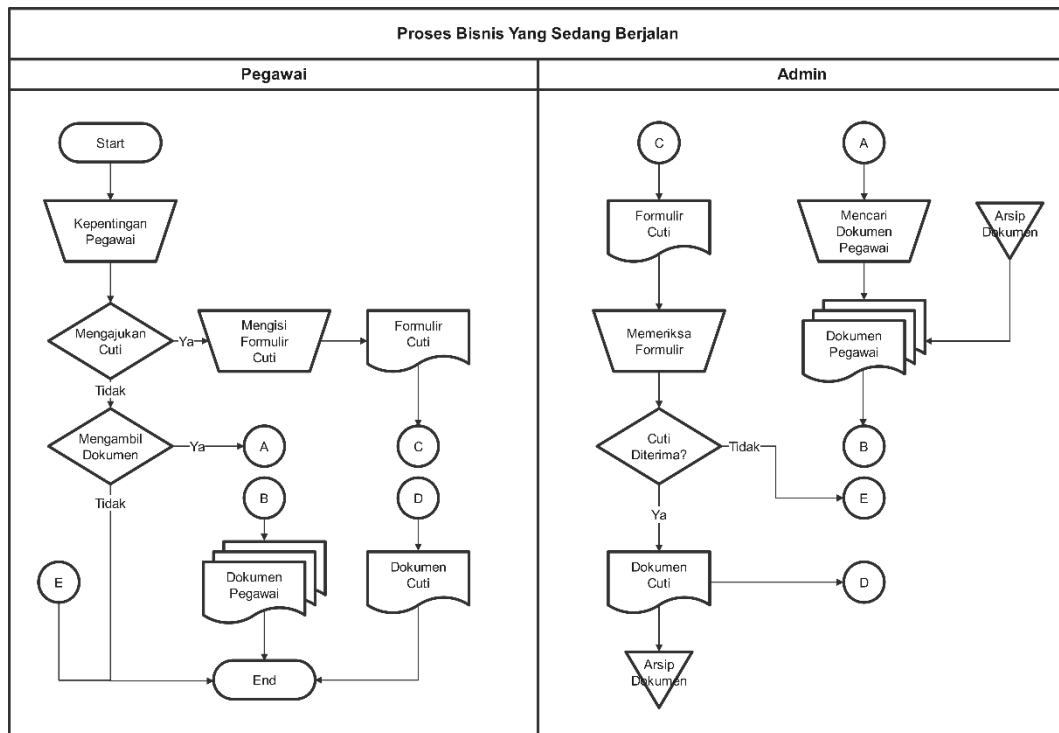
#### **4.1.1. Requirements Planning**

Pada fase *requirements planning*, semua kebutuhan dalam pengembangan sistem secara teknis dikumpulkan dan dipelajari sebagai bahan pertimbangan pengembangan sistem yang diinginkan pengguna, mulai dari proses bisnis yang saat ini sedang berjalan sampai dengan usulan proses bisnis yang baru sesuai dengan kebutuhan pengguna.

##### **4.1.1.1. Proses Bisnis Yang Sedang Berjalan**

Pada tahap *requirements planning* dalam mengembangkan sistem manajemen pegawai berbasis *mobile*, dimulai dengan mempelajari proses bisnis yang saat ini sedang berjalan. Dengan mempelajari proses bisnis tersebut, pengembang dapat menilai bagian mana saja yang dapat dipermudah dan digantikan dengan suatu sistem. Sehingga proses bisnis menjadi lebih cepat, efektif, dan efisien.

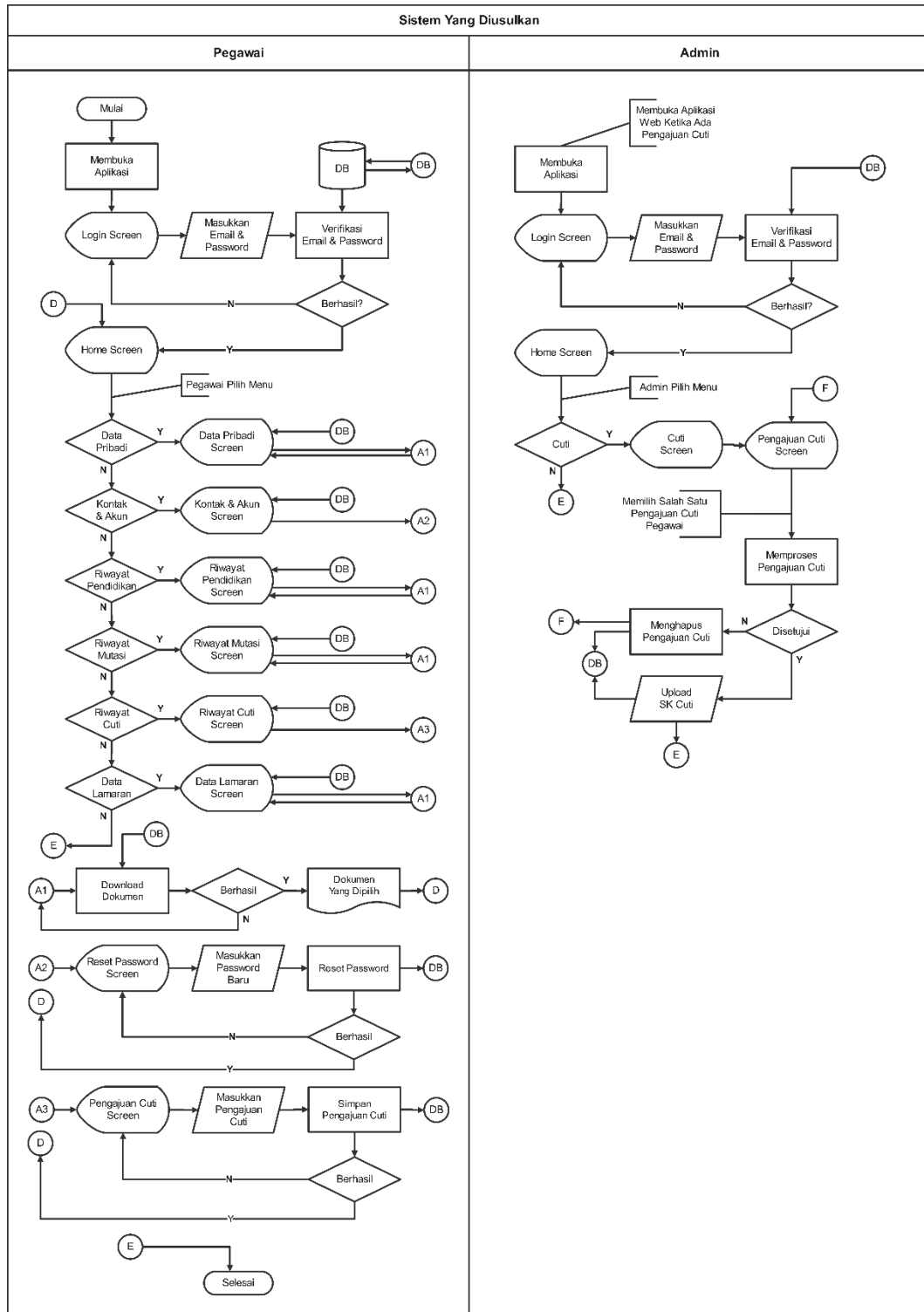
Proses bisnis yang sedang berjalan diawali ketika pegawai memiliki kepentingan tertentu. Jika pegawai ingin mengambil dokumen kepegawaiannya, pegawai masih harus menemui admin di bagian sumber daya manusia. Proses pencarian dokumen pegawai oleh admin masih dilakukan secara manual sehingga memerlukan proses yang cukup lama. Setelah dokumen ditemukan maka dokumen akan diberikan ke pegawai. Jika pegawai ingin mengajukan cuti, pegawai masih harus menemui admin di bagian sumber daya manusia untuk mengisi formulir pengajuan cuti dan formulir tersebut akan diberikan kembali ke admin untuk diperiksa apakah cuti disetujui atau tidak. Apabila cuti disetujui, admin akan menyediakan dan menyimpan dokumen cuti, lalu dokumen cuti tersebut diberikan kembali ke pegawai. Apabila cuti tidak disetujui maka pegawai akan mendapat pemberitahuan. Berikut ini *mapping chart* proses bisnis yang sedang berjalan.



Gambar 3. *Mapping chart* proses bisnis yang sedang berjalan.

#### 4.1.1.2. Proses Bisnis Yang Diusulkan

Proses bisnis yang baru menggunakan sebuah sistem manajemen pegawai berbasis *mobile* sehingga proses bisnis menjadi lebih cepat, efektif, efisien, dan dapat dilakukan dimana saja dan kapan saja. Proses bisnis yang diusulkan memuat semua kebutuhan pegawai mulai dari *login*, data pribadi, kontak & akun, riwayat pendidikan, riwayat mutasi, riwayat cuti, pengajuan cuti, *reset password*, dan data lamaran. Semua dokumen yang berkaitan dengan kepegawaian dapat diunduh oleh pegawai berdasarkan akun yang telah mereka miliki. Selain itu, untuk proses pengajuan cuti pegawai mampu menambahkan pengajuan cuti secara mandiri melalui sistem. Ketika pengajuan cuti ditambahkan, maka pengajuan tersebut akan diperiksa oleh admin melalui sistem web. Apabila pengajuan cuti diterima, admin akan mengunggah dokumen yang berkaitan dengan pengajuan cuti tersebut lalu dokumen dan riwayat cuti akan ditampilkan di sisi pegawai. Apabila pengajuan cuti tidak diterima, pengajuan cuti tersebut akan dihapus oleh sistem secara otomatis. Berikut ini *mapping chart* proses bisnis yang diusulkan.



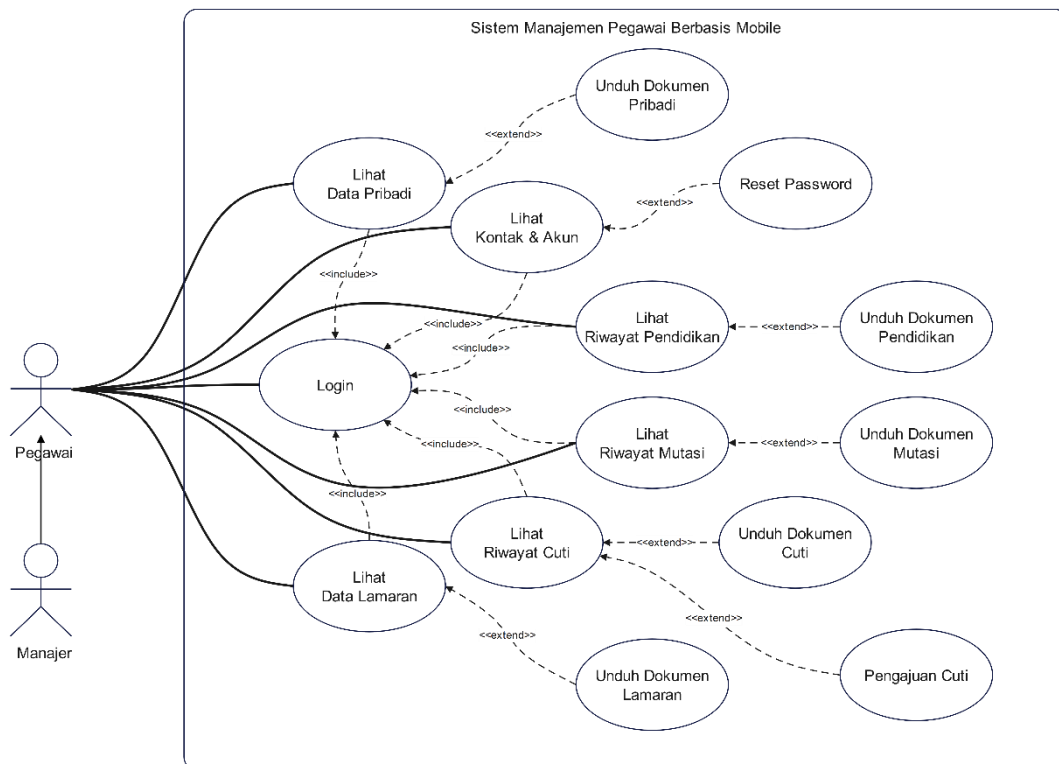
Gambar 4. Mapping chart proses bisnis yang diusulkan.

#### 4.1.2. User Design

Pada fase ini, prototipe akan di kembangkan berdasarkan analisa pada proses bisnis yang sedang berjalan. Prototipe dikembangkan mulai dari UML sampai rancangan desain UI/UX.

##### 4.1.2.1. Use Case Diagram

Perancangan diagram *use case* dimaksudkan untuk memberikan gambaran hubungan antara satu atau lebih aktor dengan sistem yang dibangun. Sehingga pengembang dapat memahami fungsi yang ada di dalam sistem dan memahami aktor mana saja yang dapat mengakses fungsi tersebut. Dalam pengembangan sistem manajemen pegawai berbasis *mobile* ini hanya terdapat aktor pegawai yang berinteraksi dengan sistem. Berikut ini *use case* diagram yang dibuat.



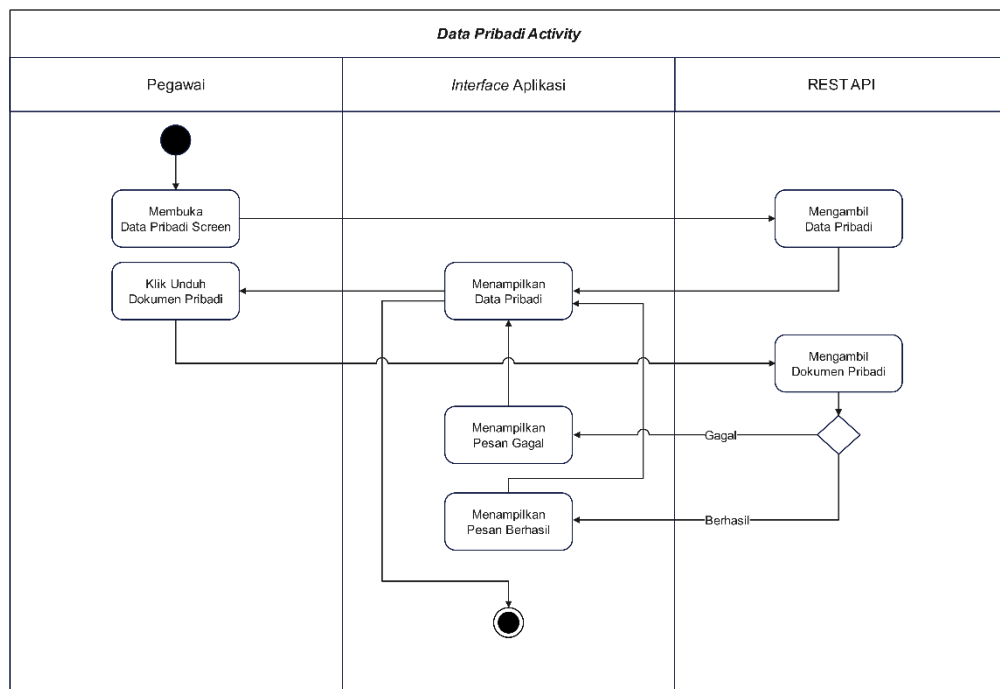
Gambar 5. Rancangan *use case* diagram.

Dalam rancangan *use case* diagram pada Gambar 5, hanya aktor pegawai yang dapat berinteraksi dengan sistem. Semua jabatan pegawai dapat berinteraksi dengan sistem mulai dari pegawai biasa sampai dengan manajer. Pegawai dapat melakukan semua fungsi yang ada di dalam sistem manajemen pegawai berbasis *mobile*. Pegawai dapat melihat *login*, data pribadi, kontak & akun, riwayat pendidikan, riwayat mutasi, riwayat cuti, dan data lamaran. Selain itu, pegawai juga



pegawai gagal diverifikasi, maka REST API akan mengirimkan respons gagal ke *interface* aplikasi untuk ditampilkan ke hadapan pegawai dan pegawai akan diminta memasukkan kembali data *login* yang benar. Apabila *email* dan *password* berhasil diverifikasi, maka REST API akan mengirimkan respons berhasil bersamaan dengan *session* ke *interface* aplikasi. Dengan *session* tersebut *interface* aplikasi akan mengalihkan pegawai menuju ke halaman *home* dan proses *login* pun berhasil.

## 2. Activity diagram data pribadi

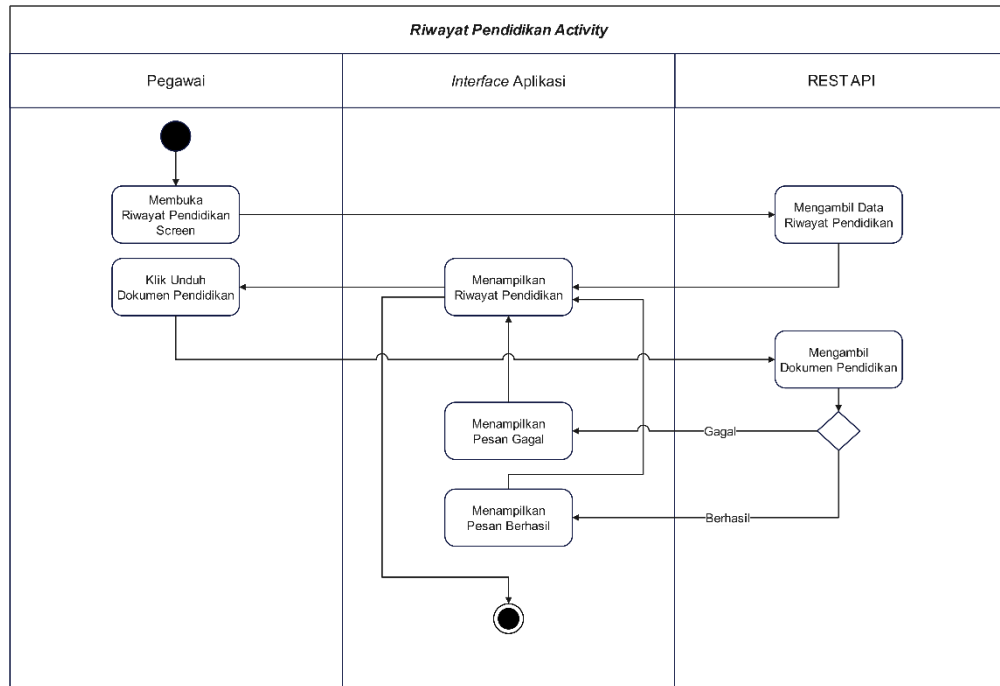


Gambar 7. Activity diagram data pribadi.

Pada *activity* diagram ini, proses akan dimulai ketika pegawai membuka halaman data pribadi. Sebelum data pribadi ditampilkan pada *interface* aplikasi, REST API mendapat permintaan untuk menyediakan data pribadi berdasarkan pegawai. Data yang disediakan REST API selanjutnya akan diteruskan dan ditampilkan *interface* aplikasi ke hadapan pegawai. Jika pegawai ingin mengunduh dokumen pribadi dengan menekan *button* unduh, maka permintaan tersebut diteruskan ke REST API. Berdasarkan permintaan pegawai tersebut REST API akan menyediakan dokumen pribadi. Jika dokumen yang diminta gagal disiapkan, maka respons gagal akan diteruskan ke *interface* aplikasi untuk ditampilkan ke hadapan pegawai. Jika dokumen

yang diminta berhasil disiapkan, maka respons berhasil akan diteruskan ke *interface* aplikasi bersamaan dengan dokumen yang diminta sehingga pegawai dapat menyimpan dokumen tersebut lalu proses pun berakhir.

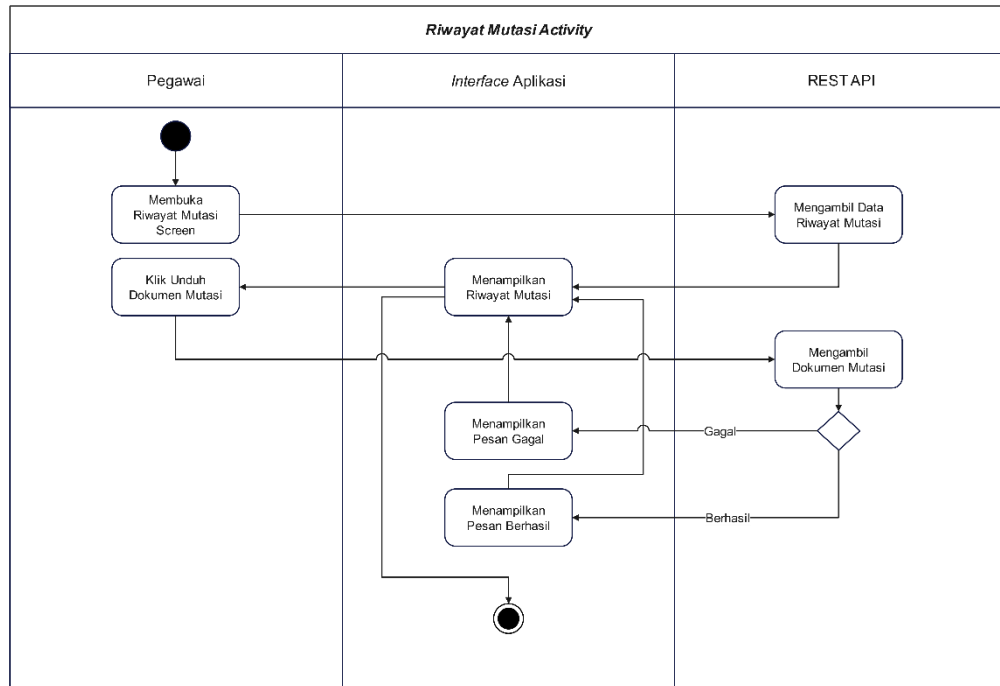
### 3. Activity diagram riwayat pendidikan



Gambar 8. Activity diagram riwayat pendidikan.

Pada *activity* diagram ini, proses akan dimulai ketika pegawai membuka halaman riwayat pendidikan. Sebelum data riwayat pendidikan ditampilkan pada *interface* aplikasi, REST API mendapat permintaan untuk menyediakan data riwayat pendidikan berdasarkan pegawai. Data yang disediakan REST API selanjutnya akan diteruskan dan ditampilkan *interface* aplikasi ke hadapan pegawai. Jika pegawai ingin mengunduh dokumen pendidikan dengan menekan *button* unduh, maka permintaan tersebut diteruskan ke REST API. Berdasarkan permintaan pegawai tersebut REST API akan menyediakan dokumen pendidikan. Jika dokumen yang diminta gagal disiapkan, maka respons gagal akan diteruskan ke *interface* aplikasi untuk ditampilkan ke hadapan pegawai. Jika dokumen yang diminta berhasil disiapkan, maka respons berhasil akan diteruskan ke *interface* aplikasi bersamaan dengan dokumen yang diminta sehingga pegawai dapat menyimpan dokumen tersebut lalu proses pun berakhir.

#### 4. Activity diagram riwayat mutasi

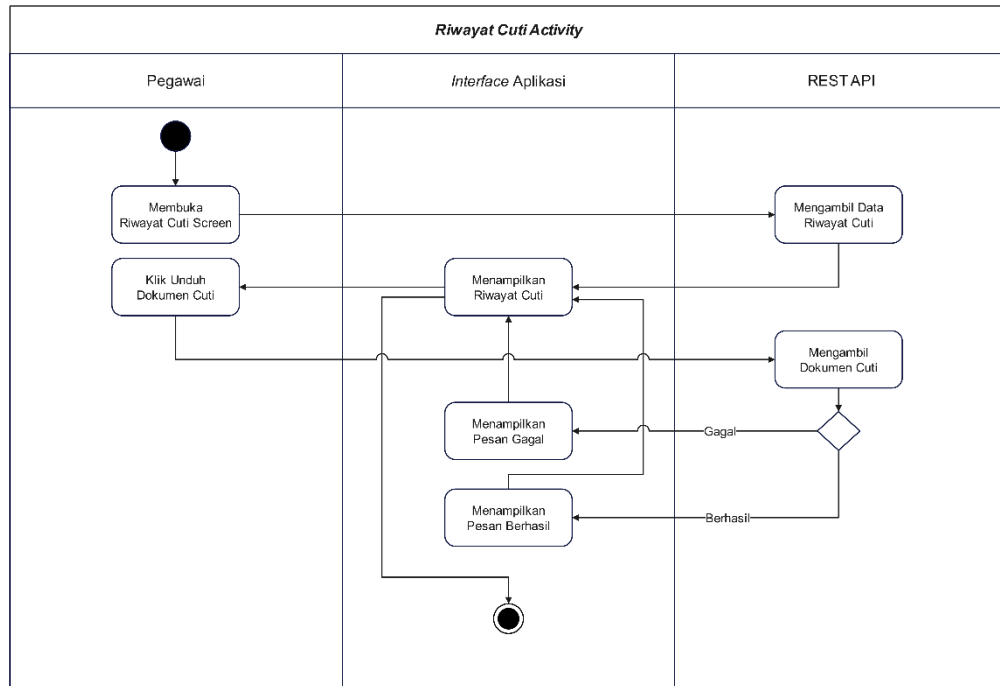


Gambar 9. Activity diagram riwayat mutasi.

Pada *activity* diagram ini, proses akan dimulai ketika pegawai membuka halaman riwayat mutasi. Sebelum data riwayat mutasi ditampilkan pada *interface* aplikasi, REST API mendapat permintaan untuk menyediakan data riwayat mutasi berdasarkan pegawai. Data yang disediakan REST API selanjutnya akan diteruskan dan ditampilkan *interface* aplikasi ke hadapan pegawai. Jika pegawai ingin mengunduh dokumen mutasi dengan menekan *button* unduh, maka permintaan tersebut diteruskan ke REST API. Berdasarkan permintaan pegawai tersebut REST API akan menyediakan dokumen mutasi. Jika dokumen yang diminta gagal disiapkan, maka respons gagal akan diteruskan ke *interface* aplikasi untuk ditampilkan ke hadapan pegawai. Jika dokumen yang diminta berhasil disiapkan, maka respons berhasil akan diteruskan ke *interface* aplikasi bersamaan dengan dokumen yang diminta sehingga pegawai dapat menyimpan dokumen tersebut lalu proses pun berakhir.



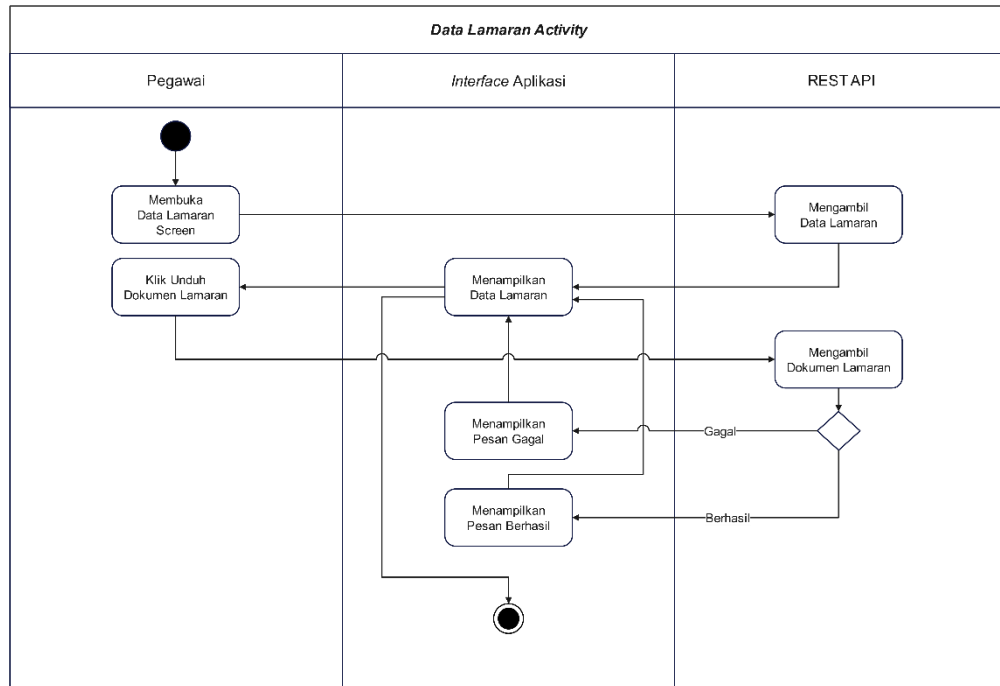
## 5. Activity diagram riwayat cuti



Gambar 10. Activity diagram riwayat cuti.

Pada *activity* diagram ini, proses akan dimulai ketika pegawai membuka halaman riwayat cuti. Sebelum data riwayat cuti ditampilkan pada *interface* aplikasi, REST API mendapat permintaan untuk menyediakan data riwayat cuti berdasarkan pegawai. Data yang disediakan REST API selanjutnya akan diteruskan dan ditampilkan *interface* aplikasi ke hadapan pegawai. Jika pegawai ingin mengunduh dokumen cuti dengan menekan *button* unduh, maka permintaan tersebut diteruskan ke REST API. Berdasarkan permintaan pegawai tersebut REST API akan menyediakan dokumen cuti. Jika dokumen yang diminta gagal disiapkan, maka respons gagal akan diteruskan ke *interface* aplikasi untuk ditampilkan ke hadapan pegawai. Jika dokumen yang diminta berhasil disiapkan, maka respons berhasil akan diteruskan ke *interface* aplikasi bersamaan dengan dokumen yang diminta sehingga pegawai dapat menyimpan dokumen tersebut lalu proses pun berakhir.

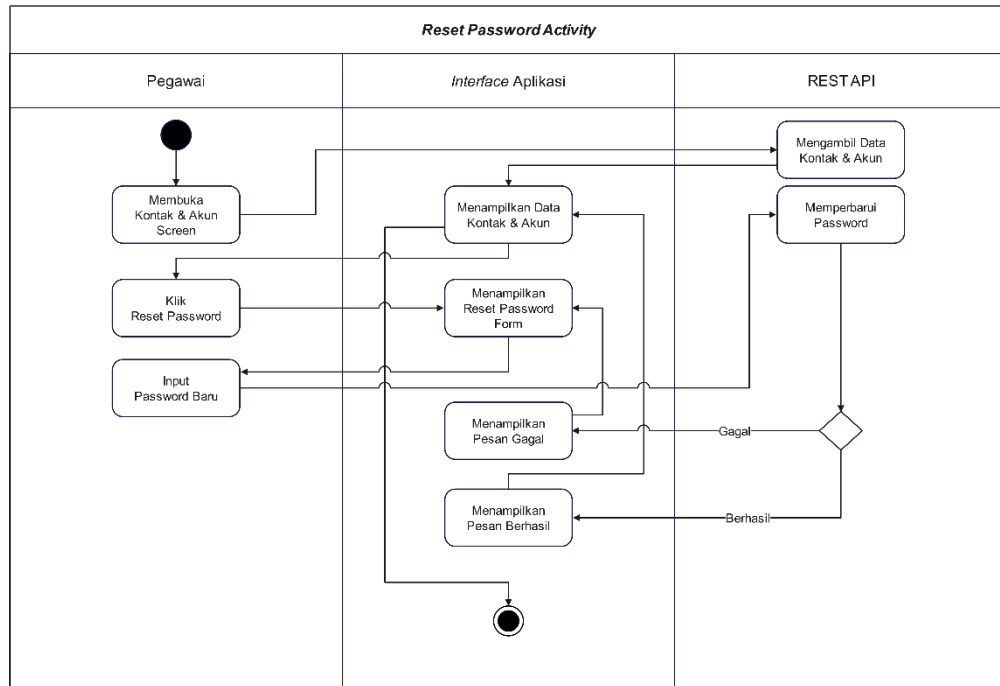
## 6. Activity diagram data lamaran



Gambar 11. Activity diagram data lamaran.

Pada *activity* diagram ini, proses akan dimulai ketika pegawai membuka halaman data lamaran. Sebelum data lamaran ditampilkan pada *interface* aplikasi, REST API mendapat permintaan untuk menyediakan data lamaran berdasarkan pegawai. Data yang disediakan REST API selanjutnya akan diteruskan dan ditampilkan *interface* aplikasi ke hadapan pegawai. Jika pegawai ingin mengunduh dokumen lamaran dengan menekan *button* unduh, maka permintaan tersebut diteruskan ke REST API. Berdasarkan permintaan pegawai tersebut REST API akan menyediakan dokumen lamaran. Jika dokumen yang diminta gagal disiapkan, maka respons gagal akan diteruskan ke *interface* aplikasi untuk ditampilkan ke hadapan pegawai. Jika dokumen yang diminta berhasil disiapkan, maka respons berhasil akan diteruskan ke *interface* aplikasi bersamaan dengan dokumen yang diminta sehingga pegawai dapat menyimpan dokumen tersebut lalu proses pun berakhir.

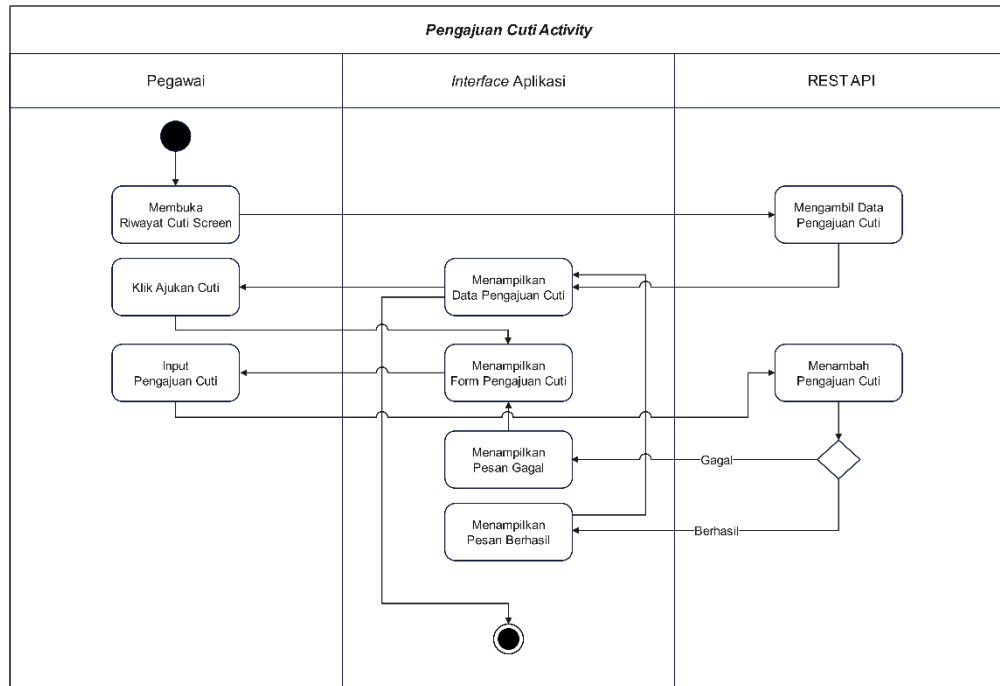
## 7. Activity diagram reset password



Gambar 12. Activity diagram reset password.

Pada *activity* diagram ini, proses akan dimulai ketika pegawai membuka halaman kontak & akun. Sebelum data kontak & akun ditampilkan pada *interface* aplikasi, REST API mendapat permintaan untuk menyediakan data kontak & akun berdasarkan pegawai. Data yang disediakan REST API selanjutnya akan diteruskan dan ditampilkan *interface* aplikasi ke hadapan pegawai. Jika ingin melakukan *reset password* dengan menekan *button reset password*, maka *interface* aplikasi akan menampilkan *reset password form* dan pegawai diminta untuk memasukkan *password* yang baru. Data *password* baru tersebut diteruskan ke REST API untuk diproses. Jika *password* baru gagal diproses, maka REST API akan meneruskan respons gagal ke *interface* aplikasi untuk ditampilkan ke hadapan pegawai dan pegawai akan diminta memasukkan kembali data *password* baru. Jika *password* baru tersebut berhasil diproses, maka REST API akan meneruskan respons berhasil ke *interface* aplikasi untuk ditampilkan ke hadapan pegawai dan proses pun berakhir.

## 8. Activity diagram pengajuan cuti



Gambar 13. Activity diagram pengajuan cuti.

Pada *activity* diagram ini, proses akan dimulai ketika pegawai membuka halaman riwayat cuti. Sebelum data pengajuan cuti ditampilkan pada *interface* aplikasi, REST API mendapat permintaan untuk menyediakan data pengajuan cuti berdasarkan pegawai. Data yang disediakan REST API selanjutnya akan diteruskan dan ditampilkan *interface* aplikasi ke hadapan pegawai. Jika ingin melakukan pengajuan cuti baru dengan menekan *button* ajukan cuti, maka *interface* aplikasi akan menampilkan pengajuan cuti *form* dan pegawai diminta untuk memasukkan data pengajuan cuti baru. Data pengajuan cuti baru tersebut diteruskan ke REST API untuk diproses. Jika data pengajuan cuti baru gagal diproses, maka REST API akan meneruskan respons gagal ke *interface* aplikasi untuk ditampilkan ke hadapan pegawai dan pegawai akan diminta memasukkan kembali data pengajuan cuti baru. Jika data pengajuan cuti baru berhasil diproses, maka REST API akan meneruskan respons berhasil ke *interface* aplikasi untuk ditampilkan ke hadapan pegawai dan proses pun berakhir.

#### 4.1.2.3. Desain Tampilan Aplikasi

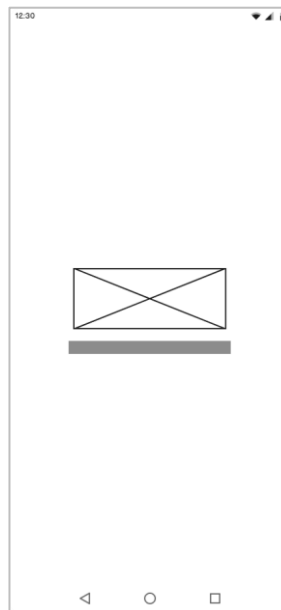
Rancangan desain tampilan aplikasi dilakukan melalui dua tahapan yaitu *wireframing*, dan desain *user interface*.

##### 1. Wireframing

Aplikasi didesain dengan mementingkan elemen-elemen dasar seperti *card*, *button*, *text*, *textfield*, *dsb.* tanpa mementingkan pewarnaan dan gaya rumit lainnya. Berikut ini daftar *item* pengerjaan dalam *sprint* desain tampilan aplikasi (*wireframing*) beserta lampiran hasilnya.

###### a. Wireframe interface splash screen

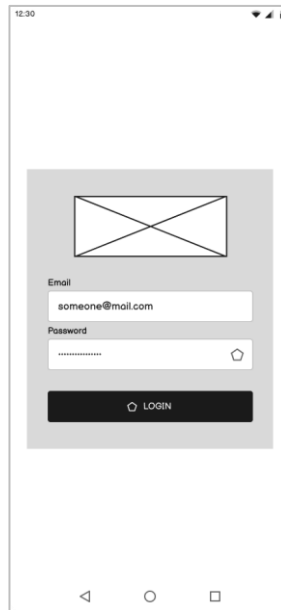
Halaman ini ditampilkan pertama kali saat aplikasi baru dibuka dan sebelum menampilkan halaman *login* atau halaman *home*. Pada halaman ini ditampilkan *brand logo* sebagai identitas aplikasi. Berikut ini tampilan *wireframe interface splash screen*.



Gambar 14. Wireframe interface splash screen.

###### b. Wireframe interface login screen

Halaman ini ditampilkan setelah halaman *splash* dan ditampilkan apabila pegawai belum memiliki *login session*. Pada halaman ini terdapat *brand logo*, *textfield email* dan *password*, serta *button login*. Berikut ini tampilan *wireframe interface login screen*.



Gambar 15. *Wireframe interface login screen.*

c. *Wireframe interface home screen*

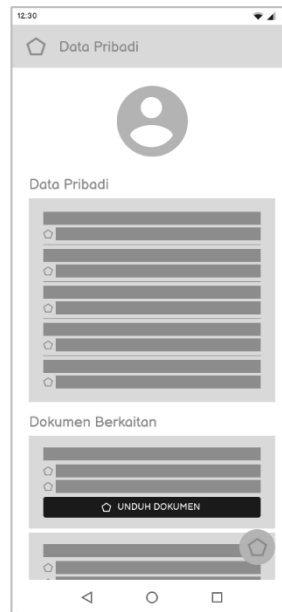
Halaman ini ditampilkan setelah halaman *splash* apabila pegawai memiliki *login session* dan ditampilkan setelah halaman *login* apabila pegawai belum memiliki *login session*. Pada halaman ini terdapat beberapa komponen utama seperti *card* profil, *list* menu, *card* mutasi terbaru, *card* cuti terbaru, *card* dokumen terbaru, dan *fab*. Berikut ini tampilan *wireframe interface home screen*.



Gambar 16. *Wireframe interface home screen.*

d. *Wireframe interface data pribadi screen*

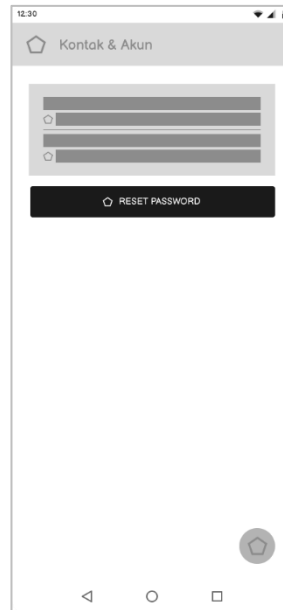
Halaman ini ditampilkan apabila pegawai menekan menu data pribadi pada komponen *list* menu yang ada di halaman *home*. Pada halaman terdapat beberapa komponen yaitu foto profil, *card* detail data pribadi, dan *list* dokumen data pribadi. Berikut ini tampilan *wireframe interface* data pribadi *screen*.



Gambar 17. *Wireframe interface* data pribadi *screen*.

e. *Wireframe interface kontak & akun screen*

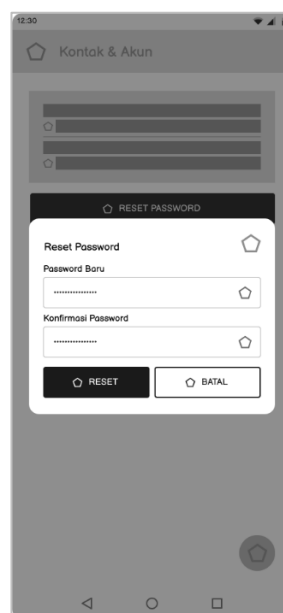
Halaman ini ditampilkan apabila pegawai menekan menu kontak & akun pada komponen *list* menu yang ada di halaman *home*. Pada halaman ini terdapat beberapa komponen yaitu *card* detail kontak & akun, dan *button* *reset password*. Berikut ini tampilan *wireframe interface* kontak & akun *screen*.



Gambar 18. *Wireframe interface kontak & akun screen.*

f. *Wireframe interface reset password screen*

Halaman ini ditampilkan apabila pegawai menekan *button reset password* yang ada di halaman kontak & akun. Pada halaman ini komponen akan ditampilkan di atas *modal card* yang berisi *textfield password* baru dan konfirmasi *password*, serta *button reset* dan *batal*. Berikut ini tampilan *wireframe interface reset password screen*.

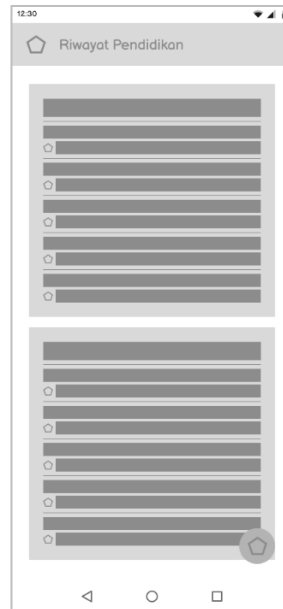


Gambar 19. *Wireframe interface reset password screen.*



g. *Wireframe interface riwayat pendidikan screen*

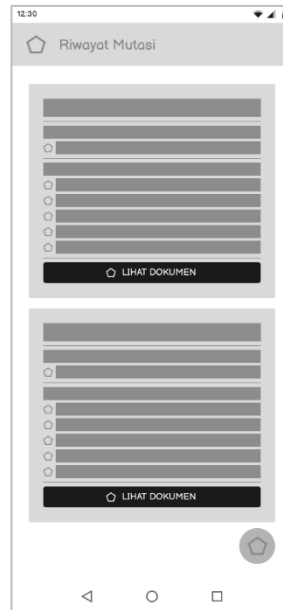
Halaman ini ditampilkan apabila pegawai menekan menu riwayat pendidikan pada komponen *list* menu yang ada di halaman *home*. Pada halaman ini terdapat komponen *card* detail riwayat pendidikan. Berikut ini tampilan *wireframe interface* riwayat pendidikan *screen*.



Gambar 20. *Wireframe interface* riwayat pendidikan *screen*.

h. *Wireframe interface riwayat mutasi screen*

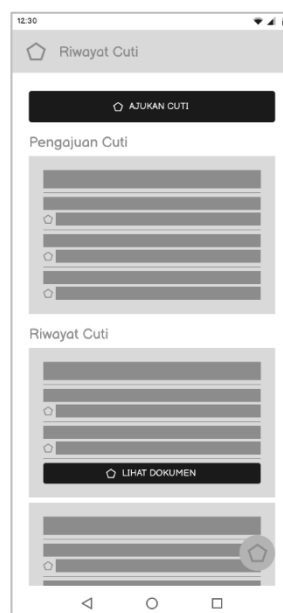
Halaman ini ditampilkan apabila pegawai menekan menu riwayat mutasi pada komponen *list* menu yang ada di halaman *home*. Pada halaman ini terdapat komponen *card* detail riwayat mutasi. Berikut ini tampilan *wireframe interface* riwayat mutasi *screen*.



Gambar 21. *Wireframe interface riwayat mutasi screen.*

i. *Wireframe interface riwayat cuti screen*

Halaman ini ditampilkan apabila pegawai menekan menu riwayat cuti pada komponen *list* menu yang ada di halaman *home*. Pada halaman ini terdapat beberapa komponen yaitu *button* ajukan cuti, *card* detail pengajuan cuti, *card*, dan detail riwayat cuti. Berikut ini tampilan *wireframe interface* riwayat cuti screen.



Gambar 22. *Wireframe interface riwayat cuti screen.*

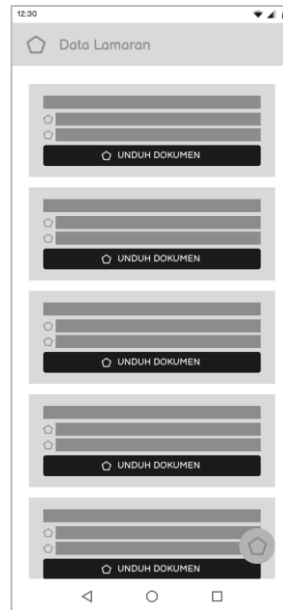
j. *Wireframe interface pengajuan cuti screen*

Halaman ini ditampilkan apabila pegawai menekan *button* ajukan cuti yang ada di halaman riwayat cuti. Pada halaman ini terdapat beberapa komponen yaitu *datefield* tanggal mulai dan tanggal selesai, *textarea* keterangan, dan *button* ajukan. Berikut ini tampilan *wireframe interface* pengajuan cuti *screen*.

Gambar 23. *Wireframe interface* pengajuan cuti *screen*.

k. *Wireframe interface data lamaran screen*

Halaman ini ditampilkan apabila pegawai menekan menu data lamaran pada komponen *list* menu yang ada di halaman *home*. Pada halaman ini terdapat komponen *card* detail data lamaran. Berikut ini tampilan *wireframe interface* data lamaran *screen*.



Gambar 24. Wireframe interface data lamaran screen.

## 2. Desain user interface (UI)

Desain *user interface* aplikasi dilakukan dengan panduan *wireframing* yang sebelumnya sudah dibuat. Pada tahap ini warna, *icon*, *typography*, ilustrasi, ukuran, dan gaya rumit lainnya sangat diperhatikan sehingga menghasilkan tampilan yang menarik ketika dipandang pegawai. Berikut ini daftar *item* pengerjaan dalam *sprint* desain tampilan aplikasi (*user interface*) beserta lampiran hasilnya.

### a. Desain user interface splash screen

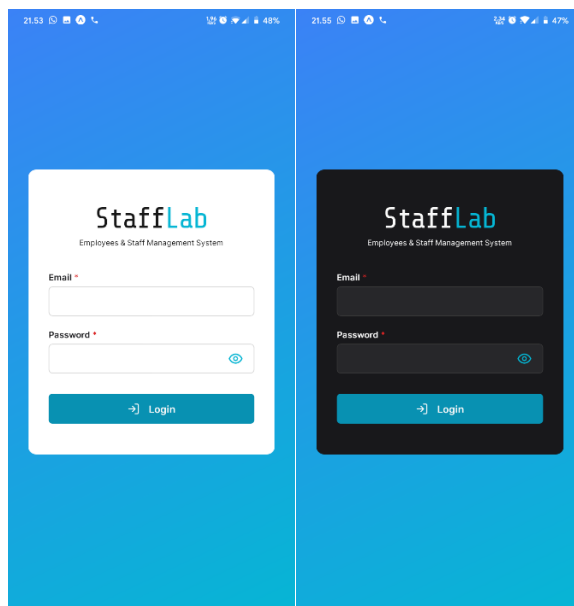
Halaman ini ditampilkan pertama kali saat aplikasi baru dibuka dan sebelum menampilkan halaman *login* atau halaman *home*. Pada halaman ini ditampilkan *brand logo* sebagai identitas aplikasi. Berikut ini tampilan desain *user interface splash screen*.



Gambar 25. Desain *user interface splash screen*.

b. Desain *user interface login screen*

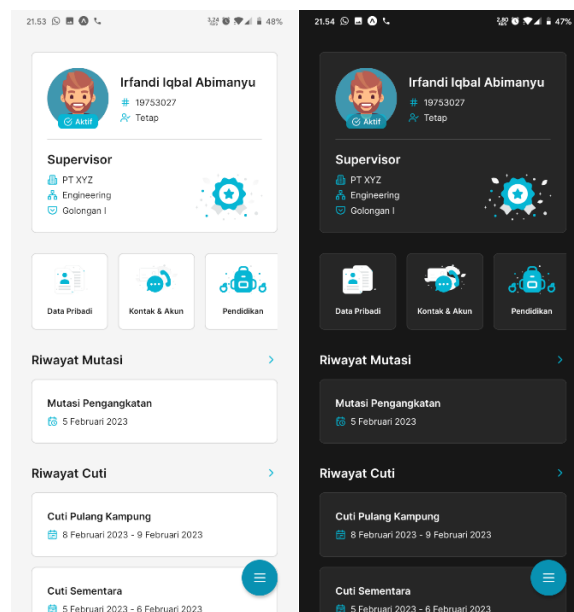
Halaman ini ditampilkan setelah halaman *splash* dan ditampilkan apabila pegawai belum memiliki *login session*. Pada halaman ini terdapat *brand logo*, *textfield email* dan *password*, serta *button login*. Berikut ini tampilan desain *user interface login screen* dalam dua tema aplikasi.



Gambar 26. Desain *user interface login screen*.

c. Desain *user interface home screen*

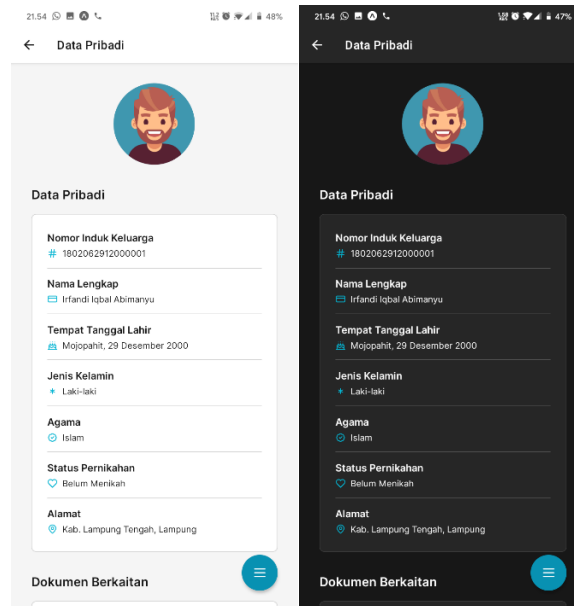
Halaman ini ditampilkan setelah halaman *splash* apabila pegawai memiliki *login session* dan ditampilkan setelah halaman *login* apabila pegawai belum memiliki *login session*. Pada halaman ini terdapat beberapa komponen utama seperti *card* profil, *list* menu, *card* mutasi terbaru, *card* cuti terbaru, *card* dokumen terbaru, dan *fab*. Berikut ini tampilan desain *user interface home screen* dalam dua tema aplikasi.



Gambar 27. Desain *user interface home screen*.

d. Desain *user interface data pribadi screen*

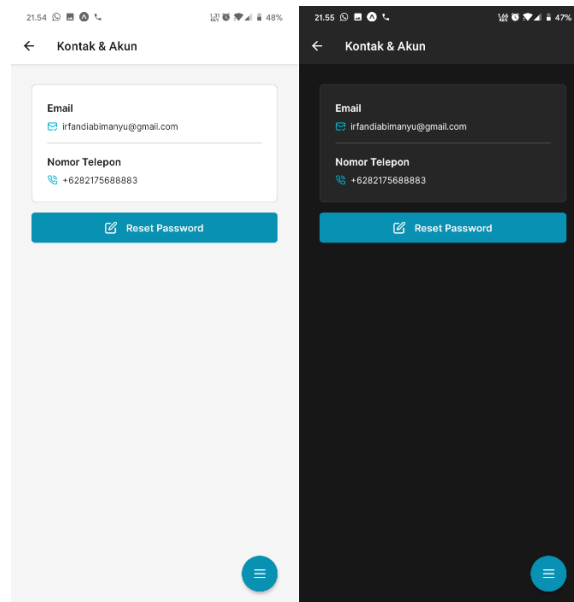
Halaman ini ditampilkan apabila pegawai menekan menu data pribadi pada komponen *list* menu yang ada di halaman *home*. Pada halaman terdapat beberapa komponen yaitu foto profil, *card* detail data pribadi, dan *list* dokumen data pribadi. Berikut ini tampilan desain *user interface data pribadi screen* dalam dua tema aplikasi.



Gambar 28. Desain *user interface* data pribadi screen.

e. Desain *user interface* kontak & akun screen

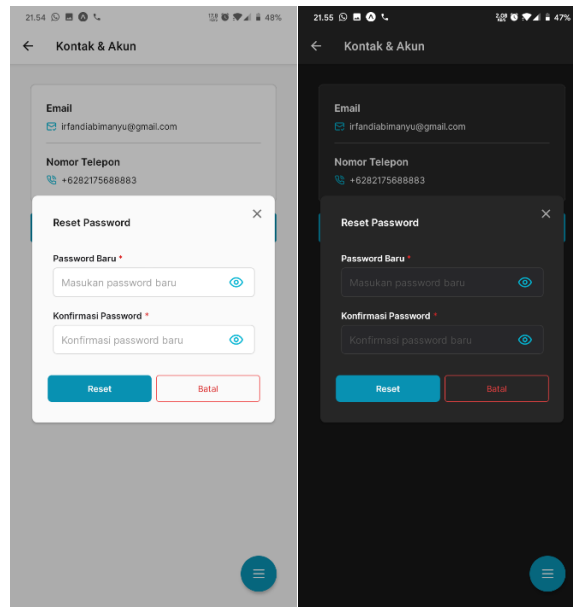
Halaman ini ditampilkan apabila pegawai menekan menu kontak & akun pada komponen *list* menu yang ada di halaman *home*. Pada halaman ini terdapat beberapa komponen yaitu *card* detail kontak & akun, dan *button* *reset password*. Berikut ini tampilan desain *user interface* kontak & akun screen dalam dua tema aplikasi.



Gambar 29. Desain *user interface* kontak & akun screen.

f. Desain *user interface reset password screen*

Halaman ini ditampilkan apabila pegawai menekan *button reset password* yang ada di halaman kontak & akun. Pada halaman ini komponen akan ditampilkan di atas *modal card* yang berisi *textfield password* baru dan konfirmasi *password*, serta *button reset* dan *batal*. Berikut ini tampilan desain *user interface reset password screen* dalam dua tema aplikasi.

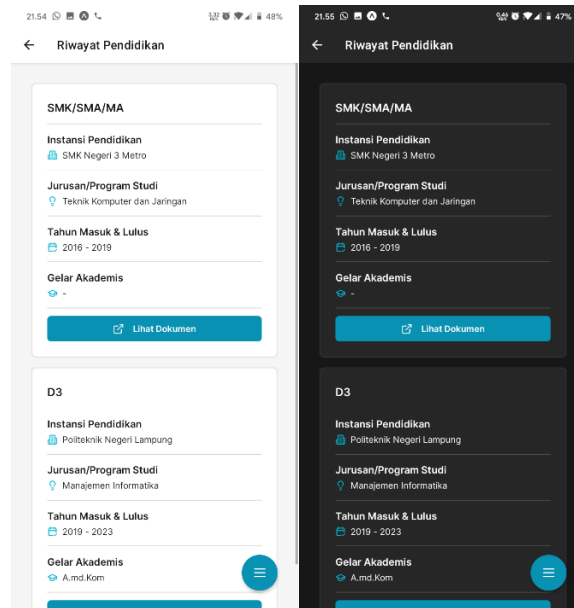


Gambar 30. Desain *user interface reset password screen*.

g. Desain *user interface riwayat pendidikan screen*

Halaman ini ditampilkan apabila pegawai menekan menu riwayat pendidikan pada komponen *list* menu yang ada di halaman *home*. Pada halaman ini terdapat komponen *card* detail riwayat pendidikan. Berikut ini tampilan desain *user interface riwayat pendidikan screen* dalam dua tema aplikasi.

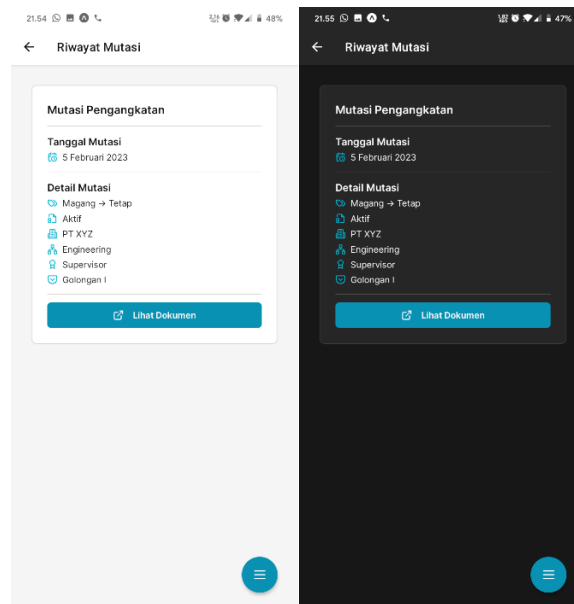




Gambar 31. Desain *user interface* riwayat pendidikan *screen*.

h. Desain *user interface* riwayat mutasi *screen*

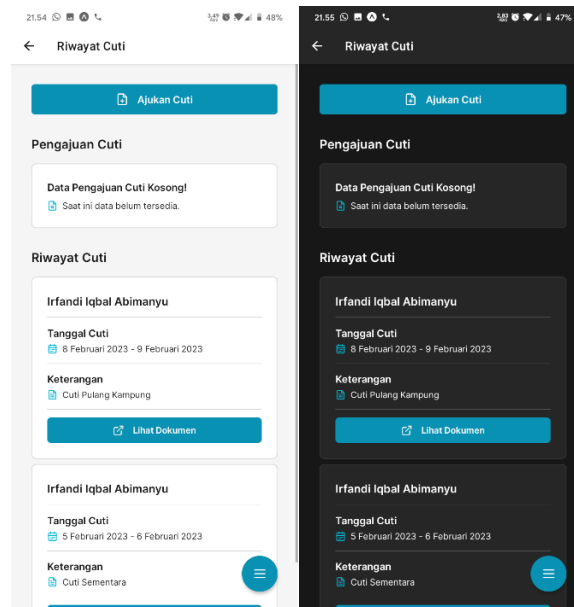
Halaman ini ditampilkan apabila pegawai menekan menu riwayat mutasi pada komponen *list* menu yang ada di halaman *home*. Pada halaman ini terdapat komponen *card* detail riwayat mutasi. Berikut ini tampilan desain *user interface* riwayat mutasi *screen* dalam dua tema aplikasi.



Gambar 32. Desain *user interface* riwayat mutasi *screen*

i. Desain *user interface* riwayat cuti *screen*

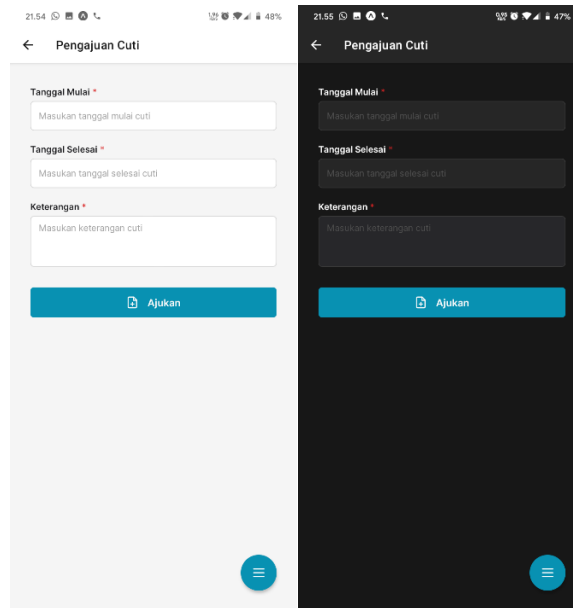
Halaman ini ditampilkan apabila pegawai menekan menu riwayat cuti pada komponen *list* menu yang ada di halaman *home*. Pada halaman ini terdapat beberapa komponen yaitu *button* ajukan cuti, *card* detail pengajuan cuti, *card*, dan detail riwayat cuti. Berikut ini tampilan desain *user interface* riwayat cuti *screen* dalam dua tema aplikasi.



Gambar 33. Desain *user interface* riwayat cuti *screen*.

j. Desain *user interface* pengajuan cuti *screen*

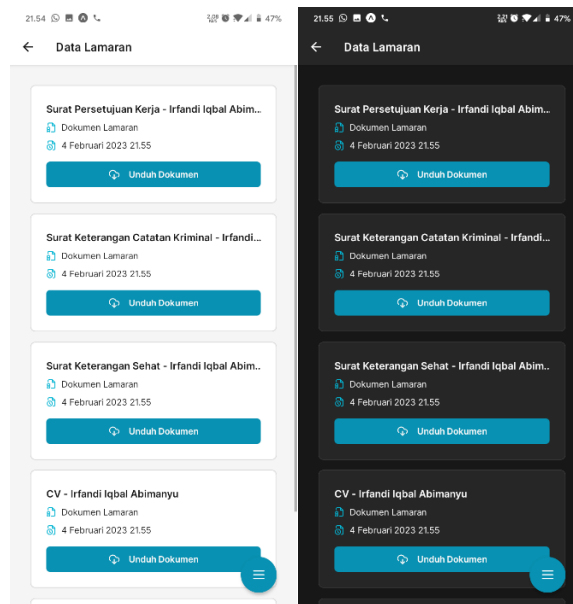
Halaman ini ditampilkan apabila pegawai menekan *button* ajukan cuti yang ada di halaman riwayat cuti. Pada halaman ini terdapat beberapa komponen yaitu *datefield* tanggal mulai dan tanggal selesai, *textarea* keterangan, dan *button* ajukan. Berikut ini tampilan desain *user interface* pengajuan cuti *screen* dalam dua tema aplikasi.



Gambar 34. Desain *user interface* pengajuan cuti screen.

k. Desain *user interface* data lamaran screen

Halaman ini ditampilkan apabila pegawai menekan menu data lamaran pada komponen *list* menu yang ada di halaman *home*. Pada halaman ini terdapat komponen *card* detail data lamaran. Berikut ini tampilan desain *user interface* data lamaran screen dalam dua tema aplikasi.



Gambar 35. Desain *user interface* pengajuan cuti screen.

### **4.1.3. Construction**

Pada fase ini, pengembang mulai mengembangkan aplikasi. Proses pengembangan tersebut dilakukan dengan *coding* dan pengujian fitur yang baru saja dikembangkan. Dalam fase ini, *error* atau *bugs* sering terjadi. Oleh karena itu, proses ini memakan waktu yang cukup lama dan memerlukan tingkat ketelitian yang tinggi.

#### **4.1.3.1. Pengembangan Aplikasi**

Pengembangan aplikasi dilakukan dengan mengimplementasikan desain menjadi sebuah aplikasi dan menyediakan fitur-fitur yang diinginkan pengguna. Proses pengembangan dilakukan dengan mengetikkan kode menggunakan kode editor (*coding*) lalu hasil sementara aplikasi di tampilkan sebagai bentuk pengujian kode yang baru saja diketik.

#### **4.1.3.2. Pengujian Fitur Aplikasi**

Pengujian fitur aplikasi dilakukan guna menguji apakah hasil pengembangan aplikasi telah berfungsi secara tampilan maupun fungsionalitasnya. Dalam *sprint* pengujian fitur aplikasi, setiap kekurangan baik dalam segi tampilan maupun fungsionalitas dilihat dan langsung diperbaiki kembali sebelum aplikasi di ulas oleh PT XYZ.

### **4.1.4. Cutover**

Setelah pengembangan sistem manajemen pegawai berbasis *mobile* melalui fase *construction* dan berhasil melalui pengetesan fitur, selanjutnya sistem akan melalui fase *cutover*. Pada tahap ini ulasan dan *feedback* pengguna dikumpulkan dan dipertimbangkan apakah sistem layak digunakan atau perlu penambahan fitur baru. Pengumpulan ulasan dilakukan menggunakan metode SUS dengan jumlah responden 23 orang dengan 10 pernyataan yang bernilai 1-5 yaitu sangat tidak setuju, tidak setuju, ragu-ragu, setuju, dan sangat setuju.

#### **4.1.4.1. Analisis Aspek Ketertarikan**

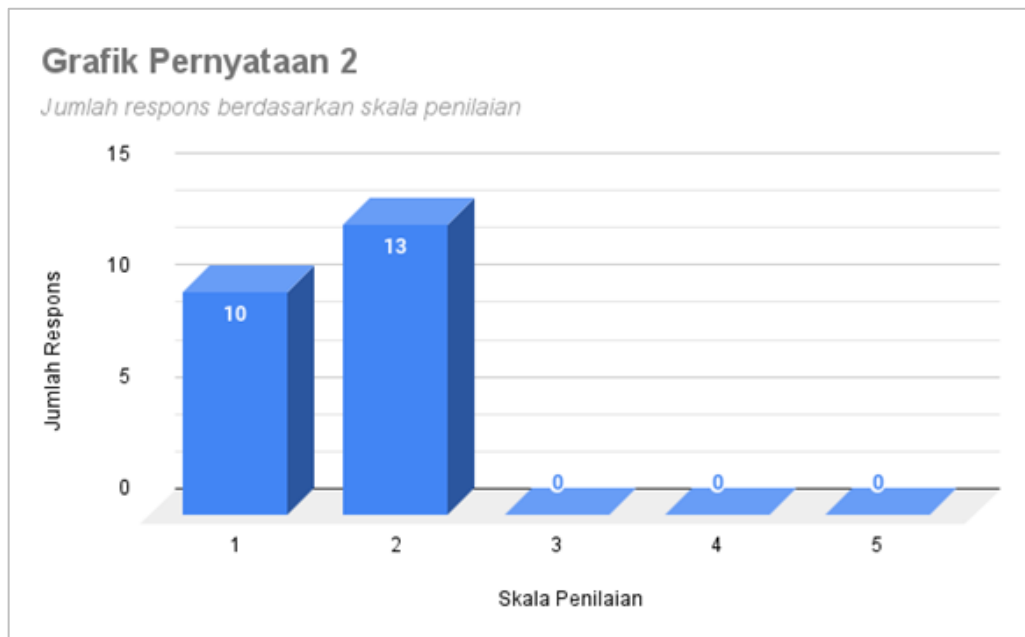
Berdasarkan hasil analisis aspek ketertarikan, terdapat 15 responden dengan persentase 65,22% menyatakan sangat setuju dan 8 responden lainnya dengan persentase 34,78% menyatakan setuju. Hal ini mengindikasikan bahwa responden tertarik dalam menggunakan aplikasi ini. Berikut ini grafik persentase analisis aspek ketertarikan.



Gambar 36. Grafik persentase analisis aspek ketertarikan.

#### 4.1.4.2. Analisis Kompleksitas Aplikasi

Berdasarkan analisis kompleksitas aplikasi, terdapat 13 responden dengan persentase 56,52% menyatakan tidak setuju dan 10 responden lainnya dengan persentase 43,48% menyatakan sangat tidak setuju. Hal ini mengindikasikan aplikasi ini tidak terlalu rumit untuk digunakan. Berikut ini grafik persentase analisis kompleksitas aplikasi.



Gambar 37. Grafik persentase analisis kompleksitas aplikasi.

#### 4.1.4.3. Analisis Aspek Efektivitas Aplikasi

Berdasarkan analisis aspek efektivitas aplikasi, terdapat 13 responden dengan persentase 56,52% menyatakan sangat setuju dan 10 responden lainnya dengan persentase 43,48% menyatakan setuju. Hal ini mengindikasikan aplikasi yang ini sudah efektif dan mudah digunakan. Berikut ini grafik persentase analisis aspek efektivitas aplikasi.



Gambar 38. Grafik persentase analisis aspek efektivitas aplikasi.

#### 4.1.4.4. Analisis Aspek Teknis Aplikasi

Berdasarkan analisis aspek teknis aplikasi, terdapat 13 responden dengan persentase 56,52% memilih tidak setuju, 6 responden dengan persentase 26,09% menyatakan sangat tidak setuju, dan 4 responden lainnya dengan persentase 17,39% menyatakan ragu-ragu. Dengan demikian sebagian responden tidak memerlukan bantuan teknisi atau orang lain ketika menjalankan aplikasi ini. Berikut ini grafik persentase analisis aspek teknik aplikasi.



Gambar 39. Grafik persentase analisis aspek teknik aplikasi.

#### 4.1.4.5. Analisis Aspek Fitur-fitur Aplikasi

Berdasarkan analisis aspek fitur-fitur aplikasi, terdapat 13 responden dengan persentase 56,52% menyatakan sangat setuju dan 10 responden lainnya dengan persentase 43,48% menyatakan setuju. Hal ini mengindikasikan bahwa fitur-fitur yang tersedia dalam aplikasi ini berjalan semestinya. Berikut ini grafik analisis aspek fitur-fitur aplikasi.



Gambar 40. Grafik analisis aspek fitur-fitur aplikasi.

#### 4.1.4.6. Analisis Aspek Penggunaan Keseharian Aplikasi

Berdasarkan analisis aspek penggunaan keseharian aplikasi, terdapat 12 responden dengan persentase 52,17% menyatakan tidak setuju dan 11 responden lainnya dengan persentase 47,83% menyatakan sangat tidak setuju. Hal ini mengindikasikan aplikasi ini sudah serasi dan konsisten ketika digunakan. Berikut ini grafik analisis aspek keseharian aplikasi.

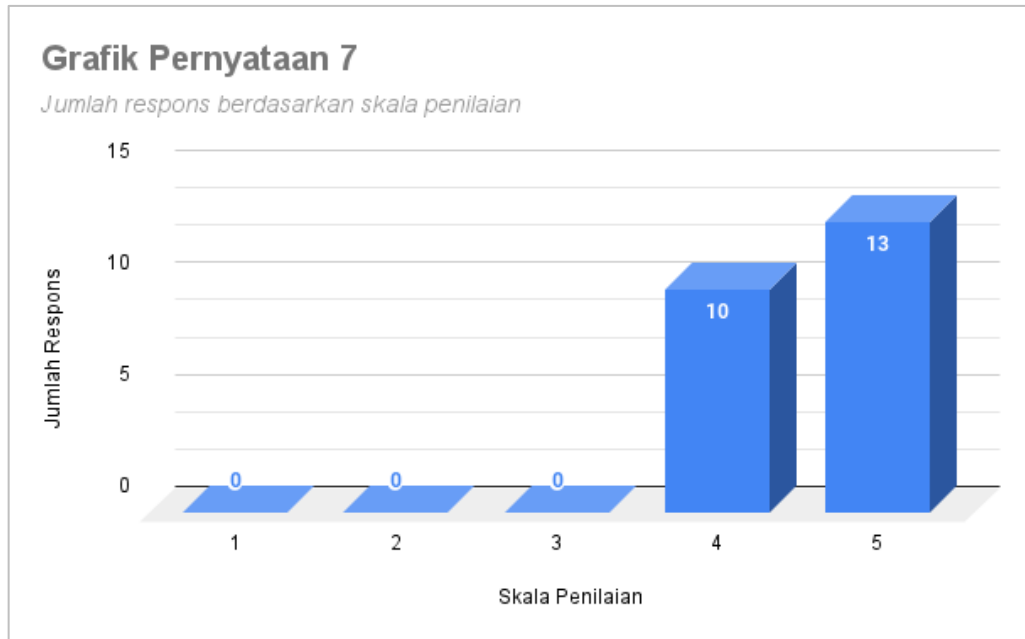


Gambar 41. Grafik analisis aspek keseharian aplikasi.

#### 4.1.4.7. Analisis Aspek Pemahaman Pengguna

Berdasarkan analisis aspek pemahaman pengguna, terdapat 13 responden dengan persentase 56,52% menyatakan sangat setuju dan 10 responden lainnya dengan persentase 43,48% menyatakan setuju. Hal ini mengindikasikan bahwa pengguna mudah memahami cara penggunaan aplikasi ini dengan cepat. Berikut ini grafik analisis aspek pemahaman aplikasi.





Gambar 42. Grafik analisis aspek pemahaman aplikasi.

#### 4.1.4.8. Analisis Aspek Kejelasan Sistem

Berdasarkan analisis aspek kejelasan sistem, terdapat 13 responden dengan persentase 56,52% menyatakan sangat tidak setuju dan 10 responden lainnya dengan persentase 43,48% menyatakan tidak setuju. Hal ini mengindikasikan aplikasi ini sangat jelas dan tidak membingungkan pengguna. Berikut ini grafik analisis aspek kejelasan sistem.



Gambar 43. Grafik analisis aspek kejelasan sistem.

#### 4.1.4.9. Analisis Aspek Kelancaran Aplikasi

Berdasarkan analisis aspek kelancaran aplikasi, terdapat 14 responden dengan persentase 60,87% menyatakan sangat setuju dan 9 responden lainnya dengan persentase 39,13% menyatakan setuju karena dalam menjalankan aplikasi tidak ada hambatan dalam penggunaannya. Hal ini mengindikasikan bahwa dalam penggunaannya, aplikasi ini sudah berjalan dengan lancar tanpa adanya hambatan. Berikut ini grafik analisis aspek kelancaran aplikasi.



Gambar 44. Grafik analisis aspek kelancaran aplikasi.

#### 4.1.4.10. Analisis Aspek Penguasaan Aplikasi

Berdasarkan analisis aspek penguasaan aplikasi, terdapat 10 responden dengan persentase 43,48% menyatakan sangat tidak setuju, 10 responden dengan persentase 43,48% menyatakan tidak setuju, dan 3 responden lainnya dengan persentase 13,04% menyatakan ragu-ragu. Hal ini mengindikasikan bahwa aplikasi ini mudah digunakan sehingga pengguna tidak memerlukan waktu yang lama untuk membiasakan diri. Berikut ini grafik analisis aspek penguasaan aplikasi.



Gambar 45. Grafik analisis aspek penguasaan aplikasi.

#### 4.1.4.11. Hasil Analisis Keseluruhan

Berdasarkan analisis yang dilakukan oleh 23 responden yang berpartisipasi, Langkah terakhir yang perlu dilakukan untuk mengetahui hasil analisis keseluruhan yaitu dengan melakukan perhitungan setiap nilai respons kuesioner di setiap aspek dengan aturan perhitungan metode SUS. Berikut ini aturan perhitungan skor kuesioner dalam metode SUS.

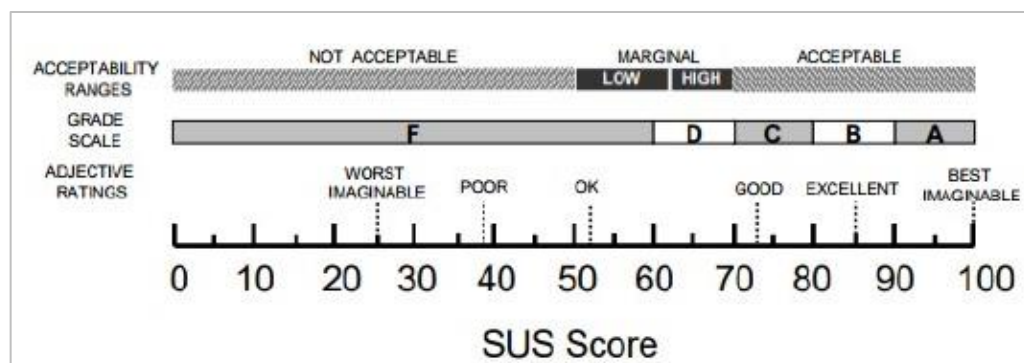
1. Skor dari respons pernyataan dengan nomor ganjil akan dikurangi 1 ( $x-1$ ).
2. Skor dari respons pernyataan dengan nomor genap akan digunakan untuk mengurangkan 5 ( $5-x$ ).
3. Skor yang diperoleh setiap pernyataan akan dijumlahkan seluruhnya dan akan dikalikan 2,5.
4. Skor akhir SUS didapatkan dengan cara mencari nilai rata-rata dari hasil perhitungan keseluruhan pada aturan nomor 3.

Berikut merupakan hasil perhitungan akhir analisis keseluruhan kuesioner dari 23 responden dapat dilihat pada Gambar 46.

User	P1	SUS#	P2	SUS#	P3	SUS#	P4	SUS#	P5	SUS#	P6	SUS#	P7	SUS#	P8	SUS#	P9	SUS#	P10	SUS#	Skor SUS
1	4	3	2	3	4	3	2	3	4	3	2	3	4	3	2	3	4	3	2	3	75,0
2	4	3	2	3	5	4	1	4	5	4	2	3	4	3	1	4	5	4	3	2	85,0
3	4	3	1	4	4	3	1	4	4	3	2	3	5	4	2	3	4	3	1	4	85,0
4	4	3	1	4	4	3	3	2	5	4	2	3	5	4	2	3	4	3	3	2	77,5
5	5	4	1	4	5	4	1	4	5	4	1	4	5	4	1	4	5	4	1	4	100,0
6	5	4	2	3	5	4	3	2	5	4	2	3	5	4	2	3	5	4	3	2	82,5
7	5	4	2	3	4	3	2	3	4	3	2	3	5	4	2	3	5	4	1	4	85,0
8	5	4	1	4	5	4	2	3	5	4	1	4	5	4	1	4	5	4	2	3	95,0
9	5	4	1	4	4	3	2	3	4	3	1	4	4	3	1	4	4	3	2	3	85,0
10	5	4	1	4	5	4	1	4	4	3	1	4	4	3	1	4	5	4	1	4	95,0
11	5	4	2	3	4	3	3	2	5	4	1	4	4	3	2	3	5	4	1	4	85,0
12	5	4	2	3	5	4	2	3	4	3	1	4	4	3	1	4	5	4	2	3	87,5
13	5	4	2	3	5	4	2	3	4	3	1	4	4	3	1	4	5	4	2	3	87,5
14	5	4	1	4	5	4	2	3	4	3	1	4	5	4	1	4	4	3	1	4	92,5
15	5	4	2	3	4	3	2	3	5	4	2	3	5	4	1	4	4	3	1	4	87,5
16	4	3	1	4	5	4	1	4	5	4	1	4	5	4	1	4	5	4	1	4	97,5
17	5	4	1	4	5	4	1	4	5	4	1	4	5	4	1	4	5	4	1	4	100,0
18	4	3	2	3	5	4	2	3	5	4	2	3	4	3	2	3	4	3	2	3	80,0
19	5	4	2	3	4	3	3	2	5	4	2	3	5	4	2	3	5	4	2	3	82,5
20	5	4	1	4	5	4	2	3	4	3	1	4	5	4	1	4	5	4	1	4	95,0
21	4	3	2	3	4	3	2	3	5	4	2	3	5	4	2	3	5	4	2	3	82,5
22	5	4	2	3	5	4	2	3	5	4	2	3	4	3	1	4	4	3	2	3	85,0
23	4	3	2	3	4	3	2	3	4	3	2	3	4	3	2	3	4	3	2	3	75,0
Rata-rata																					87,1

Gambar 46. Hasil analisis keseluruhan.

Cara untuk menentukan peringkat skala dari hasil perhitungan dengan metode SUS dapat dilihat pada Gambar 47. Hasil perhitungan di atas menunjukkan skor akhir 87,1. Sistem ini mendapatkan predikat *Acceptable* dengan *grade B* dan mendapatkan *adjective rating excellent*. Dengan demikian sistem ini sudah layak untuk digunakan.



Gambar 47. Skala SUS.

## **BAB V. KESIMPULAN DAN SARAN**

### **5.1. Kesimpulan**

Kesimpulan yang bisa didapat dari tugas akhir ini adalah telah berhasil mendesain dan membangun sebuah sistem informasi manajemen pegawai berbasis *mobile* pada PT XYZ. Sistem informasi ini bertujuan untuk memfasilitasi pegawai dalam proses pencarian data dan memudahkan pegawai ketika ingin mendapatkan informasi kepegawaian yang berkaitan dengan dirinya secara *real-time*. Hal ini akan meningkatkan kinerja dan efisiensi proses administrasi perusahaan.

### **5.2. Saran**

Saran ini ditujukan untuk pengembang selanjutnya untuk mengembangkan sistem informasi manajemen pegawai ini menjadi sistem yang utuh dan lengkap dengan berbagai fitur manajemen pegawai seperti pengaturan gaji, pemantauan kinerja, dan lain-lain.

## DAFTAR PUSTAKA

- Adam, S. I., Mononutu, M. J., & Damping, G. (2022). Aplikasi Jasa Titip Belanja Berbasis Mobile di Minahasa Utara. *CogITO Smart Journal*, 8(2), 434–445. <https://doi.org/10.31154/cogito.v8i2.422.434-445>
- Aziz, N., Pribadi, G., & Nurcahya, M. S. (2020). Analisa dan Perancangan Aplikasi Pembelajaran Bahasa Inggris Dasar Berbasis Android. *Jurnal IKRAITH-INFORMATIKA*, 1(3).
- Boduch, A., Derks, R., & Sakhniuk, M. (2022). *React and React Native* (H. Edwards, Ed.; 4 ed.). Packt Publishing.
- Chatterjee, S., & Mamatha, T. (2020). A comparative study on SOAP and RESTful web services. *IRJET*, 7(5).
- Eisenman, B. (2015). *Learning React Native: Building Mobile Applications with JavaScript* (M. Foley, Ed.; 1 ed.). O'Reilly Media, Inc.
- Fitriyana, I., & Susianto, D. (2018). Aplikasi Akuntansi Piutang Jasa Service Pada PT. AUX Indonesia Bandar Lampung. *Jurnal JUSINTA*, 1(1).
- Flanagan, D. (2020). *JavaScript: The Definitive Guide* (J. Pollock, Ed.; 7 ed.). O'Reilly Media, Inc.
- Fox, R. (2013). *Information Technology: An Introduction for Today's Digital World* (1 ed.). Chapman and Hall/CRC.
- Glaschenko, A. (2021, Oktober 25). *What is Rapid Application Development (RAD)?* Jmix Blog. <https://www.jmix.io/rapid-application-development>
- Hendriyani, M. (2021). Pemberkasan Arsip Dinamis Aktif di Subbagian Persuratan Dan Arsip Aktif pada Arsip Nasional Republik Indonesia (ANRI). *Kompleksitas*, 10(1). <https://doi.org/10.56486/kompleksitas.vol10no01.80>
- Hutahaean, J., Purba, R. A., Siagian, Y., Heriyani, N., Amina, H. U. st., Syah, A. Z., Ardiana, D. P. Y., & Simarmata, J. (2021). *Pengantar Sistem Informasi Manajemen* (A. Rikki, Ed.; 1 ed.). Yayasan Kita Menulis.
- IBM Cloud Education. (2022). *What is an Application Programming Interface (API)*. IBM. <https://www.ibm.com/topics/api>
- Islam, Md. R., Islam, Md. R., & Mazumder, T. A. (2010). Mobile Application and Its Global Impact. *IJET-IJENS*, 10(06).
- Kaban, R., Danur, S. R., & Zuliaty, R. (2022). Penerapan Metode Rapid Application Development (RAD) dalam Perancangan Sistem Informasi

- Penjualan Berbasis Web. *Jurnal Informatika Dan Perancangan Sistem (JIPS)*, 4(2). <https://jurnal.itbi.ac.id/index.php/journalinformatika/article/view/36>
- Karim, M. A., & Adriansyah, A. R. (2022). Analisis dan Perancangan Aplikasi Mobile untuk Donasi menggunakan Metode Hybrid berbasis React Native. *Jurnal Informatika Terpadu*, 8(1), 26–34. <https://doi.org/10.54914/jit.v8i1.394>
- MDN Mozilla. (2022). *HTTP request methods*. Mozilla Foundation. <https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods>
- Muhammad Nur Fauzi, A., Triayudi, A., & Diana Sholihati, I. (2022). MENGUKUR TINGKAT KEPUASAN PENGGUNA APLIKASI KEARSIPAN MENGGUNAKAN SYSTEM USABILITY SCALE DAN PIECES FRAMEWORK. *JUPI*, 7(1), 231–239.
- Mukodimah, S., Muslihudin, M., & Trisnawati. (2019). APLIKASI PENENTUAN BENGKEL TSM BERKUALITAS UNTUK UKK SISWA SMK KABUPATEN PRINGSEWU BERBASIS MOBILE. *SINTAK*, 3.
- Nursaid, F. F., Hendra Brata, A., & Kharisma, A. P. (2020). Pengembangan Sistem Informasi Pengelolaan Persediaan Barang Dengan ReactJS Dan React Native Menggunakan Prototipe (Studi Kasus : Toko Uda Fajri). *J-Ptiik.Ub.Ac.Id*, 4(1).
- Prastio, C. E., & Ani, N. (2018). Aplikasi Self Service Menu Menggunakan Metode Scrum Berbasis Android (Case Study : Warkobar Café Cikarang). *PETIR*, 11(2), 203–220. <https://doi.org/10.33322/petir.v11i2.255>
- Putra, D. W. T., & Andriani, R. (2019). Unified Modelling Language (UML) dalam Perancangan Sistem Informasi Permohonan Pembayaran Restitusi SPPD. *Jurnal TeknoIf*, 7(1), 32. <https://doi.org/10.21063/jtif.2019.V7.1.32-39>
- Rosalin, S. (2017). *Manajemen Arsip Dinamis* (1 ed.). UB Press.
- Sadikin, A., & Wiranda, N. (2022). *Sistem Informasi Manajemen* (I. Mirsa, Ed.; 1 ed.). K-Media. <http://digilib.iain-palangkaraya.ac.id/3890/>
- Sugiarto, A., & Wahyono, T. (2014). *Manajemen Kearsipan Elektronik* (1 ed.). Gava Media.
- Suparman. (2020). Pelaksanaan Manajemen Arsip dalam Meningkatkan Efektivitas Kerja Pegawai pada Kantor Camat Kecamatan Tanjung Lago, Kabupaten Banyuasin. *Jurnal Studia Administrasi*, 2(2), 42–57. <https://doi.org/10.47995/jian.v2i2.13>
- Visual Paradigm. (2022a). *What is Activity Diagram?* <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/>

- Visual Paradigm. (2022b). *What is Use Case Diagram?* <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/>
- Wibisono, A. (2020). Sistem Informasi Manajemen Kepegawaian Untuk Meningkatkan Efektivitas Pengelolaan Sumber Daya Manusia. *Jurnal Ilmu Komputer dan Informatika*, 13(1), 1–10.
- Wibisono, W., & Baskoro, F. (2002). PENGUJIAN PERANGKAT LUNAK DENGAN MENGGUNAKAN MODEL BEHAVIOUR UML. *JUTI: Jurnal Ilmiah Teknologi Informasi*, 1(1). <https://doi.org/10.12962/j24068535.v1i1.a95>



## **LAMPIRAN**

## Lampiran 1. Kode Program

## PENULISAN KODE PROGRAM SISTEM INFORMASI MANAJEMEN PEGAWAI BERBASIS MOBILE PADA PT XYZ MENGGUNAKAN FRAMEWORK REACT NATIVE

### 1. Koneksi REST API

```
import AsyncStorage from "@react-native-async-storage/async-storage";
import { createClient } from "@supabase/supabase-js";
import "react-native-url-polyfill/auto";

const Supabase = createClient(process.env.SUPABASE_URL,
process.env.SUPABASE_ANON_KEY, {
  auth: {
    storage: AsyncStorage,
    autoRefreshToken: true,
    persistSession: true,
    detectSessionInUrl: false,
  },
});

export default Supabase;
```

### 2. Halaman Login

```
import { Keyboard } from "react-native";

// Styles & Icons
import { Flex, Pressable, VStack, useColorModeValue } from "native-base";

// Components
import BrandLogo from "components/BrandLogo";
import LoginForm from "components/forms/LoginForm";

export default function Login() {
  // Color Mode
  const bgGradient = { colors: ["blue.500", "cyan.500"], start: [0, 0],
end: [1, 1] };
  const bgWrapper = useColorModeValue("white", "gray.900");

  return (
    <Pressable onPress={() => Keyboard.dismiss()}>
      <Flex justify='center' bg={{ linearGradient: bgGradient
}} align='center' h='full' px={8}>
        <VStack bg={bgWrapper} py={12} space={8}
rounded='xl' shadow='md' w='full'>
          <BrandLogo />
          <LoginForm />
        </VStack>
      </Flex>
    </Pressable>
  );
}
```

### 3. Halaman Home

```
// Styles & Icons
import { Flex, ScrollView, VStack, ZStack } from "native-base";

// Components
import ProfilCard from "components/cards/ProfilCard";
import RecentCutiCard from "components/cards/RecentCutiCard";
import RecentDokumenCard from "components/cards/RecentDokumenCard";
import RecentMutasiCard from "components/cards/RecentMutasiCard";
import MenuList from "components/lists/MenuList";
import BaseStagger from "components/staggers/BaseStagger";

export default function Home() {
  return (
    <Flex bg='trueGray.100' h='full' safeAreaTop _dark={{ bg:
"trueGray.900" }}>
      <ZStack justifyContent='flex-end' h='full'>
        <ScrollView h='full' w='full'>
          <VStack h='full' space={8} p={8}>
            <ProfilCard />
            <MenuList />
            <RecentMutasiCard />
            <RecentCutiCard />
            <RecentDokumenCard />
          </VStack>
        </ScrollView>
        <BaseStagger />
      </ZStack>
    </Flex>
  );
}
```

### 4. Halaman Data Pribadi

```
// Styles & Icons
import { Flex, ScrollView, ZStack } from "native-base";

// Components
import DataPegawaiDetail from
"components/details/dataPegawai/DataPegawaiDetail";
import BaseStagger from "components/staggers/BaseStagger";

export default function DataPribadi() {
  return (
    <Flex bg='trueGray.100' h='full' _dark={{ bg: "trueGray.900"
}}>
      <ZStack justifyContent='flex-end' h='full'>
        <ScrollView h='full' w='full'>
          <DataPegawaiDetail />
        </ScrollView>
        <BaseStagger />
      </ZStack>
    </Flex>
  );
}
```

## 5. Halaman Kontak & Akun

```
// Styles & Icons
import { Flex, ScrollView, ZStack } from "native-base";

// Components
import DataKontakAkunDetail from
"components/details/kontakAkun/DataKontakAkunDetail";
import BaseStagger from "components/staggers/BaseStagger";

export default function KontakAkun() {
  return (
    <Flex bg='trueGray.100' h='full' _dark={{ bg: "trueGray.900"
    }}>
      <ZStack justifyContent='flex-end' h='full'>
        <ScrollView h='full' w='full'>
          <DataKontakAkunDetail />
        </ScrollView>
        <BaseStagger />
      </ZStack>
    </Flex>
  );
}
```

## 6. Halaman Reset Password

```
import { yupResolver } from "@hookform/resolvers/yup";
import { ResetPasswordSchema } from "helpers/Validations";
import { updateUser } from "helpers/api/FunctionApi";
import { useState } from "react";
import { FormProvider, useForm } from "react-hook-form";
import { Keyboard } from "react-native";
import { useSelector } from "react-redux";
import { PegawaiSelector } from "states/slices/PegawaiSlice";

// Styles & Icons
import { Button, HStack, KeyboardAvoidingView, Modal, useToast } from "native-
base";

// Components
import BaseAlert from "components/alerts/BaseAlert";
import ResetPasswordForm from "components/forms/ResetPasswordForm";

export default function ResetPasswordModal({ disclosure }) {
  const [loading, setLoading] = useState(false);
  const { pegawai } = useSelector(PegawaiSelector.pegawai);
  const { isOpen, onClose } = disclosure;
  const toast = useToast();

  // Form
  const resolver = yupResolver(ResetPasswordSchema);
  const mainForm = useForm({ resolver, mode: "onChange" });

  const onReset = async ({ password }) => {
    Keyboard.dismiss();
    setLoading(true);
    try {
      await updateUser({ password }, pegawai?.uuidUser);
      toast.show({
        placement: "top",
        duration: 3000,
        render: ({ id }) => (
          <BaseAlert
            props={{
              toast,
            }}
          />
        )
      });
    } catch (error) {
      // Handle error
    }
  };
}
```

```

Berhasil!",
telah diganti.",
    id,
    status: "success",
    variant: "left-accent",
    title: "Reset Password",
    description: "Password Anda",
    isCloseable: true,
  }
  </BaseAlert>
  </toast>
  </>
  ),
  });
  mainForm.reset();
  onClose();
} catch (err) {
  toast.show({
    placement: "top",
    duration: 3000,
    render: ({ id }) => (
      <BaseAlert
        props={{
          toast,
          id,
          status: "error",
          variant: "left-accent",
          title: "Reset Password",
          description: "Gagal",
          isCloseable: true,
        }}
      </BaseAlert>
    ),
  });
}
setLoading(false);
};

return (
  <Modal isOpen={isOpen} onClose={onClose}>
    <KeyboardAvoidingView behavior='padding' enabled
      keyboardVerticalOffset={120}>
      <Modal.Content p={4}>
        <Modal.Header
          borderColor='transparent'>Reset Password</Modal.Header>
        <Modal.CloseButton />
        <Modal.Body>
          <FormProvider {...mainForm}>
            <ResetPasswordForm />
          </FormProvider>
        </Modal.Body>
        <Modal.Footer borderColor='transparent'>
          <HStack space={2}>
            <Button
              isLoading={loading}
              isLoadingText='Memproses'
              colorScheme='cyan'
              rounded='md'
              w='50%'
              _text={{
                fontWeight: "semibold"
              }}
              _spinner={{ height: 5 }}
              onPress={mainForm.handleSubmit(onReset)}
            >

```

```

                                Reset
                                </Button>
                                <Button colorScheme='red'
variant='outline' borderColor='red.500' w='50%'>
                                Batal
                                </Button>
                                </HStack>
                                </Modal.Footer>
                                </Modal.Content>
                                </KeyboardAvoidingView>
                                </Modal>
    );
}

```

## 7. Halaman Riwayat Pendidikan

```

// Styles & Icons
import { Flex, ScrollView, ZStack } from "native-base";

// Components
import DataPendidikanDetail from
"components/details/pendidikan/DataPendidikanDetail";
import BaseStagger from "components/staggers/BaseStagger";

export default function Pendidikan() {
    return (
        <Flex bg='trueGray.100' h='full' _dark={{ bg: "trueGray.900"
}}>
            <ZStack justifyContent='flex-end' h='full'>
                <ScrollView h='full' w='full'>
                    <DataPendidikanDetail />
                </ScrollView>
                <BaseStagger />
            </ZStack>
        </Flex>
    );
}

```

## 8. Halaman Riwayat Mutasi

```

// Styles & Icons
import { Flex, ScrollView, ZStack } from "native-base";

// Components
import DataMutasiDetail from "components/details/mutasi/DataMutasiDetail";
import BaseStagger from "components/staggers/BaseStagger";

export default function Mutasi() {
    return (
        <Flex bg='trueGray.100' h='full' _dark={{ bg: "trueGray.900"
}}>
            <ZStack justifyContent='flex-end' h='full'>
                <ScrollView h='full' w='full'>
                    <DataMutasiDetail />
                </ScrollView>
                <BaseStagger />
            </ZStack>
        </Flex>
    );
}

```

## 9. Halaman Riwayat Cuti

```
// Styles & Icons
import { Flex, ScrollView, ZStack } from "native-base";

// Components
import DataCutiDetail from "components/details/cuti/DataCutiDetail";
import BaseStagger from "components/staggers/BaseStagger";

export default function Cuti() {
  return (
    <Flex bg='trueGray.100' h='full' _dark={{ bg: "trueGray.900" }}>
      <ZStack justifyContent='flex-end' h='full'>
        <ScrollView h='full' w='full'>
          <DataCutiDetail />
        </ScrollView>
        <BaseStagger />
      </ZStack>
    </Flex>
  );
}
```

## 10. Halaman Pengajuan Cuti

```
import { yupResolver } from "@hookform/resolvers/yup";
import { useNavigation } from "@react-navigation/native";
import { PengajuanCutiSchema } from "helpers/Validations";
import { postPengajuanCuti } from "helpers/api/PegawaiApi";
import { useState } from "react";
import { FormProvider, useForm } from "react-hook-form";
import { Keyboard } from "react-native";
import { useSelector } from "react-redux";
import { PegawaiSelector } from "states/slices/PegawaiSlice";

// Styles & Icons
import { FilePlus } from "lucide-react-native";
import { Button, Flex, Icon, ScrollView, VStack, ZStack, useToast } from
"native-base";

// Components
import BaseAlert from "components/alerts/BaseAlert";
import PengajuanCutiForm from "components/forms/PengajuanCutiForm";
import BaseStagger from "components/staggers/BaseStagger";

export default function PengajuanCuti() {
  const [loading, setLoading] = useState(false);
  const { pegawai } = useSelector(PegawaiSelector.pegawai);
  const toast = useToast();
  const navigation = useNavigation();

  // Form
  const resolver = yupResolver(PengajuanCutiSchema);
  const mainForm = useForm({ resolver, mode: "onChange" });

  const onAjukan = async (data) => {
    Keyboard.dismiss();
    setLoading(true);
    clearTimeout();
    try {
      await postPengajuanCuti({ ...data, nipPegawai:
pegawai?.nip });
      toast.show({
        placement: "top",
        duration: 3000,
      });
    }
  };
}
```

```

        render: ({ id }) => (
          <BaseAlert
            props={{
              toast,
              id,
              status: "success",
              variant: "left-accent",
              title: "Pengajuan Cuti
Berhasil!",
              description: "Cuti telah
diajukan.",
              isCloseable: true,
            }}
          />
        ),
      );
      setTimeout(() => {
        mainForm.reset();
        navigation.goBack();
      }, 3000);
    } catch (err) {
      toast.show({
        placement: "top",
        duration: 3000,
        render: ({ id }) => (
          <BaseAlert
            props={{
              toast,
              id,
              status: "error",
              variant: "left-accent",
              title: "Pengajuan Cuti
Gagal!",
              description: "Cuti gagal
diajukan.",
              isCloseable: true,
            }}
          />
        ),
      });
      setloading(false);
    }
  };

  return (
    <Flex bg='trueGray.100' h='full' _dark={{ bg: "trueGray.900"
}}>
      <ZStack justifyContent='flex-end' h='full'>
        <ScrollView h='full' w='full'>
          <VStack h='full' space={8} p={8}>
            <FormProvider {...mainForm}>
              <PengajuanCutiForm />
            </FormProvider>
            <Button
              isLoading={loading}
              isLoadingText='Memproses'
              size='lg'
              colorScheme='cyan'
              leftIcon={<Icon
as={<FilePlus size={20} />} mr={2} />}
              rounded='md'
              _text={{ fontWeight:
"semibold" }}
              _spinner={{ height: 5 }}
              onPress={mainForm.handleSubmit(onAjukan)}
            >

```



```

                                Ajukan
                                </Button>
                            </VStack>
                        </ScrollView>
                    <BaseStagger />
                </ZStack>
            </Flex>
        );
    }

```

## 11. Halaman Data Lamaran

```

// Styles & Icons
import { Flex, ScrollView, ZStack } from "native-base";

// Components
import LamaranList from "components/lists/LamaranList";
import BaseStagger from "components/staggers/BaseStagger";

export default function Lamaran() {
    return (
        <Flex bg='trueGray.100' h='full' _dark={{ bg: "trueGray.900" }}>
            <ZStack justifyContent='flex-end' h='full'>
                <ScrollView h='full' w='full'>
                    <LamaranList />
                </ScrollView>
                <BaseStagger />
            </ZStack>
        </Flex>
    );
}

```