BAB III
METODE PENELITIAN

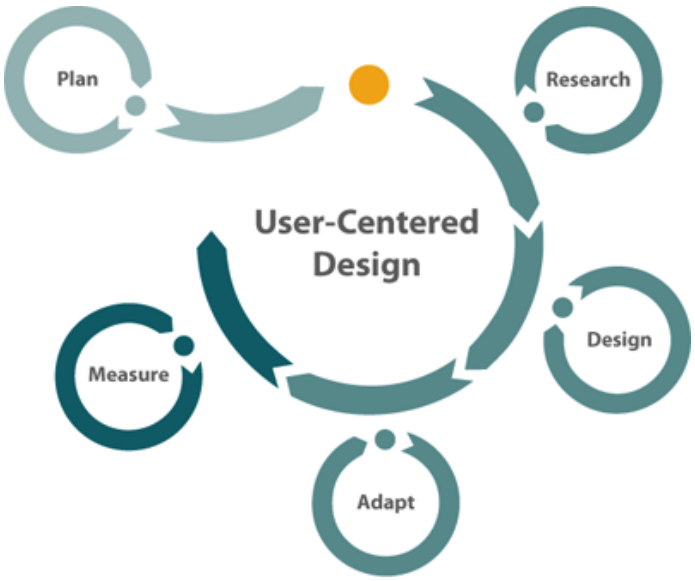
3.1 Tempat Penelitian

Penelitian dilaksanakan di PT. Masakini Mandiri tepatnya Divisi *Publishing* yang beralamat di Jln. Soekarno Hatta No.108, Rajabasa, Bandar Lampung. Penelitian dan penyusunan laporan dilakukan selama bulan April 2019 sampai bulan September 2019.

3.2 Metode Penelitian

Pada penelitian ini penulis menggunakan metode *User Centered Design* (UCD), metode yang digunakan dalam proses desain yang berfokus pada kebutuhan pengguna. Produk yang akan dikembangkan dengan pendekatan UCD, dioptimalkan untuk pengguna akhir atau *end-user* dan ditekankan tentang apa yang menjadi kebutuhan atau keinginan *end-user* terhadap penggunaan suatu produk. Desain harus dirancang dengan adaptasi terhadap *behavior* atau perilaku pengguna dalam menggunakan produk sehingga produk yang dikembangkan tidak memaksa pengguna untuk mengubah perilakunya ketika menggunakan produk tersebut. Tujuannya adalah agar produk yang dikembangkan dapat bermanfaat serta mudah digunakan bagi pengguna.

User Centered Design terfokus berdasarkan desain yang berpusat pada manusia dengan analisis *target audiens* yang lebih mendalam. Hal ini tidak hanya berkonsentrasi pada karakteristik dan persepsi manusia secara umum namun juga sifat dan fitur spesifik dari target pengguna. *User Centered Design* memperhatikan detail tentang target pengguna yang menjadi objek desain, seperti memperhitungkan usia, jenis kelamin, tingkat pendidikan potensial, latar belakang profesional, environment penggunaan produk, ciri emosional dan persepsi fisik serta tingkat kesadaran teknologi dan faktor-faktor lainnya.



Gambar III-1 Tahapan atau Aktivitas UCD

Label untuk setiap tahapan dari aktivitas UCD mungkin saja berbeda dalam setiap perusahaan atau agensi, namun secara umum setiap proses UCD dapat ditunjukkan seperti pada gambar III.1.

Dalam UCD, ada beberapa prinsip yang harus diperhatikan, yaitu:

1. Fokus pada pengguna

   Berhubungan dengan pengguna atau calon pengguna dalam sebuah perancangan merupakan hal yang harus dilakukan agar karakteristik *anthropometric*, sikap, dan kognisi pengguna dapat dipahami.

2. Perancangan terintegrasi

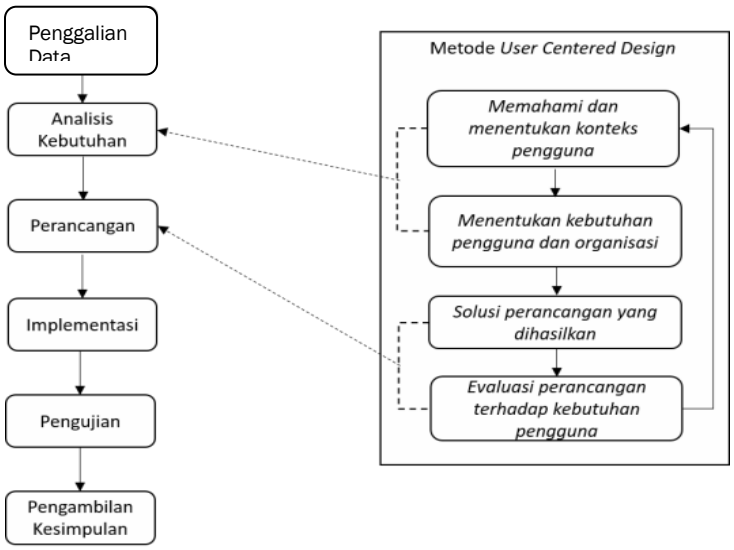
   Sistem bantuan, dukungan teknis, antarmuka pengguna, prosedur instalasi dan konfigurasi harus dimasukkan dalam perancangan.

3. Pengujian pengguna

   Perlu adanya observasi terhadap perilaku pengguna, wawasan terhadap *problem solving*, evaluasi terhadap umpan balik, dan motivasi yang kuat jika ingin ada perubahan terhadap rancangan.

4. Perancangan interaktif

   Berdasarkan hasil pengujian yang tadi sudah dilakukan, untuk mencapai perancangan yang interaktif, sistem harus dideskripsikan, dirancang, dan diuji berkali-kali.



Gambar III-2 Kerangka Metode UCD

3.3 Tahapan Penelitian

Sebelum memulai penelitian perlu dibuat langkah-langkah penelitian, dimana langkah- langkah penelitian tersebut adalah sebagai berikut:

3.3.1 Pengumpulan Data

Kegiatan pengumpulan data dilakukan dengan cara melakukan studi literatur, wawancara, dan kuisioner kepada calon pengguna *(end user)*. Hal ini dilakukan untuk mendapatkan konsep yang 'matang' dan mengenal perilaku calon pengguna sehingga informasi yang dibutuhkan untuk membangun suatu sistem dapat terpenuhi.

3.3.2 Analisis Kebutuhan

Kegiatan analisis kebutuhan adalah tahapan untuk mengidentifikasi kebutuhan dari sistem informasi ini. Pada penelitian ini menggunakan metode *User Centered Design* dimana ada dua alur yaitu *specify the context of use* dan *specify user and organizational requirement.*

1. S*pecify the context of use*, dilakukan identifikasi siapa saja calon pengguna yang akan menggunakan produk ini.
2. *Specify user and organizational requirement* yaitu mengidentifikasi kebutuhan calon pengguna.

Berdasarkan hasil penelitian dan pengamatan penulis, diperlukan sebuah gambaran tentang kebutuhan pengguna terhadap satu aplikasi yang menarik dan mudah untuk digunakan, khususnya bagi pembaca dan penerbit.

3.3.3 Perancangan

Membuat perancangan antarmuka sebagai *design solution* berdasarkan dari hasil analisis kebutuhan. Pada tahap ini sangatlah penting dan harus diperhatikan karena mempunyai tujuan untuk memberikan kemudahan dalam penggunaannya. Perancangan antarmuka menggunakan *wireframe* dimana hanya sebatas *User Interface* dan beberapa fitur yang menggambarkan secara umum kerja sistem.

3.3.3.1 Kebutuhan Fungsional Pengguna

Berikut kebutuhan fungsional bagi pengguna *(user)* yang menjadi fokus dalam perancangan desain antarmuka aplikasi *e-Publishing "Pustakala*" yang berbasis Android:

1. *User* dapat melakukan insatalasi aplikasi dengan mudah dibeberapa perangkat Android dengan spesifikasi yang berbeda.
2. *User* masuk kedalam aplikasi *e-Publishing* Pustakala, kemudian aplikasi menampilkan Beranda *(home)*
3. *User* dapat mencari dan melihat buku-buku dengan mudah berdasarkan Kategori dan Rating.
4. *User* dapat memilih, dan memasukkan buku kedalam daftar pesanan.
5. *User* dapat melakukan transaksi pemesanan buku setelah memiliki akun.
6. *User* dapat membuat akun menggunakan email.

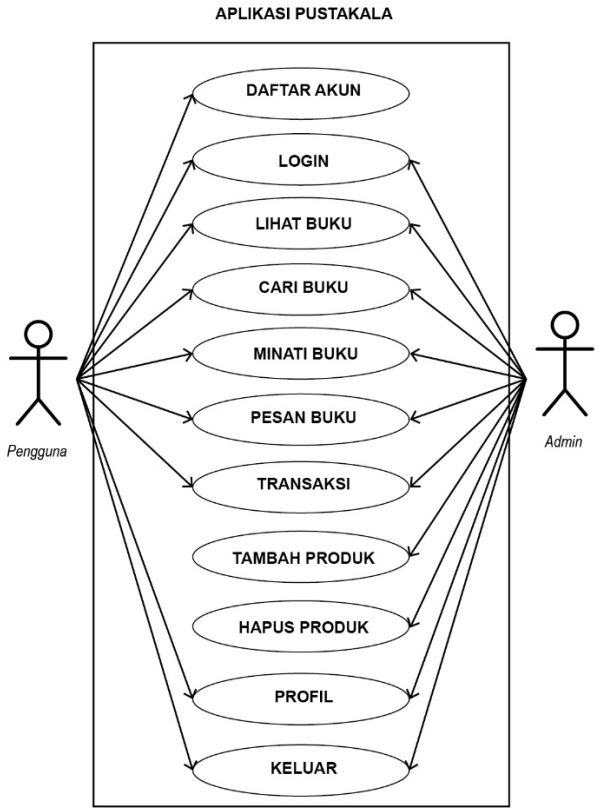
7. *User* dapat keluar dari aplikasi dengan memilih tombol keluar atau kembali, kemudian sistem menutup aplikasi.

3.3.3.2 Rancangan *Desain Logic*

Pada rancangan desain logic berisi tentang pemodelan *Unified Modeling Language* (UML) menggunakan *use case diagram*, *class diagram* dan *activity diagram*. Berikut penjelasannya:

1. *Use Case Diagram*

Use case diagram merupakan *prototyping* atau pemodelan untuk perilaku sistem yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dengan kata lain, *use case* digunakan untuk mengetahui alur dan fungsi yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Berikut adalah *use case* untuk aplikasi Pustakala:



Gambar III-3 *Use Case Diagram* Aplikasi Pustakala

a. Definisi Aktor

Berikut adalah tabel definisi aktor dapat dilihat pada tabel III.1. sebagai berikut:

Tabel III-1 Tabel Definisi Aktor

No	Aktor	Deskripsi
1	*Admin*	Orang yang mengelola data pengguna, melihat aktivitas, menambah buku, menghapus buku, menhapus dan membuat akun.
2	*Pengguna*	Orang yang bisa melakukan aktivitas, daftar, *login,* melihat, mencari, memilih, dan meminati buku, membuat pesanan, melakukan transaksi.

b. Definisi *Use Case*

Berikut adalah tabel *definisi use case* dapat dilihat pada tabel III.2. sebagai berikut:

Tabel III-2 Tabel Definisi *Use Case*

No	*Use Case*	Deskripsi
1	Daftar Akun *(Signup)*	Merupakan aktivitas untuk melakukan pendaftaran akun.
2	*Login*	Merupakan aktivitas untuk masuk ke aplikasi bagi pengguna terdaftar.
3	Lihat Buku	Merupakan aktivitas yang dapat dilakukan semua pengguna di beranda.
4	Cari Buku	Merupakan aktivitas yang dilakukan semua pengguna di

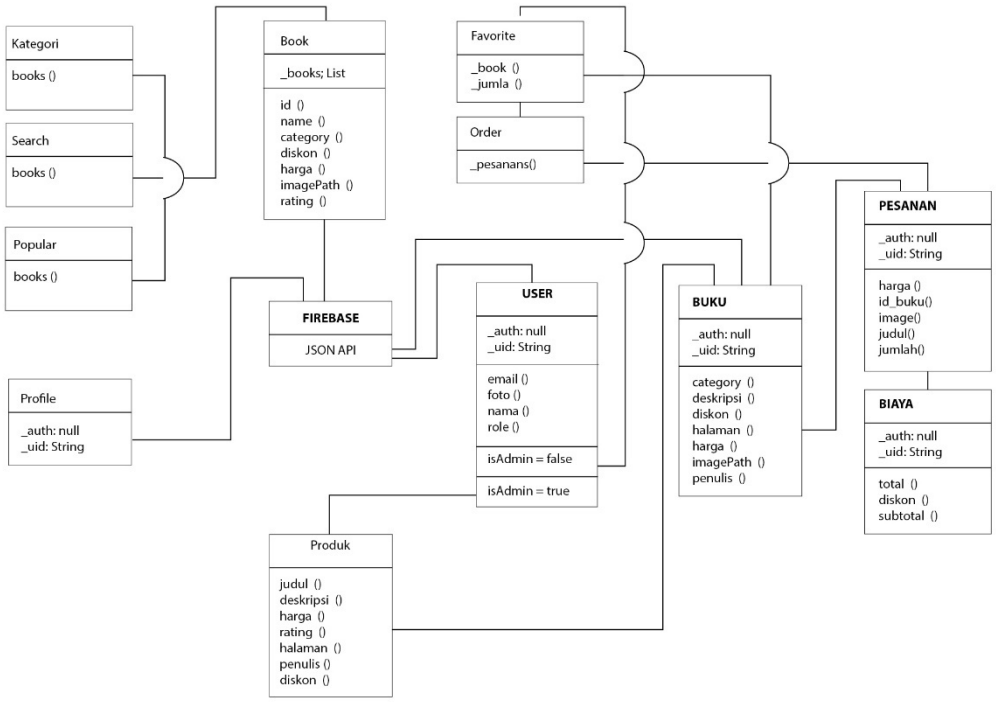
		beranda, berdasarkan kata kunci tertentu.
5	Minati Buku	Merupakan aktivitas yang dapat dilakukan semua pengguna untuk memilih dan memasukkan buku kedalam daftar minat.
6	Pesan Buku	Merupakan aktivitas untuk membuat pesanan buku yang diminati.
7	Transaksi	Merupakan aktivitas untuk melakukan transaksi pembelian dan hanya bisa dilakukan oleh akun terdaftar.
8	Tambah Produk	Merupakan aktivitas untuk menambah produk buku dan hanya bisa dilakukan oleh admin.
9	Hapus Produk	Merupakan aktivitas untuk menambah produk buku dan hanya bisa dilakukan oleh admin.
10	Profil	Merupakan aktivitas untuk melihat profil pengguna terdaftar.
11	Keluar	Merupakan halaman aktivitas untuk keluar dari aplikasi bagi semua pengguna terdaftar.

2. *Class Diagram*

Class Diagram menginterprestasikan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sebuah sistem. Kelas memiliki apa yang disebut atribut maupun metode atau

operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.

Sedangkan operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas. Berikut adalah *class diagram* Aplikasi Pustakala dengan bahasa pemrograman *Dart*:



Gambar III-4 *Class Diagram* Aplikasi Pustakala

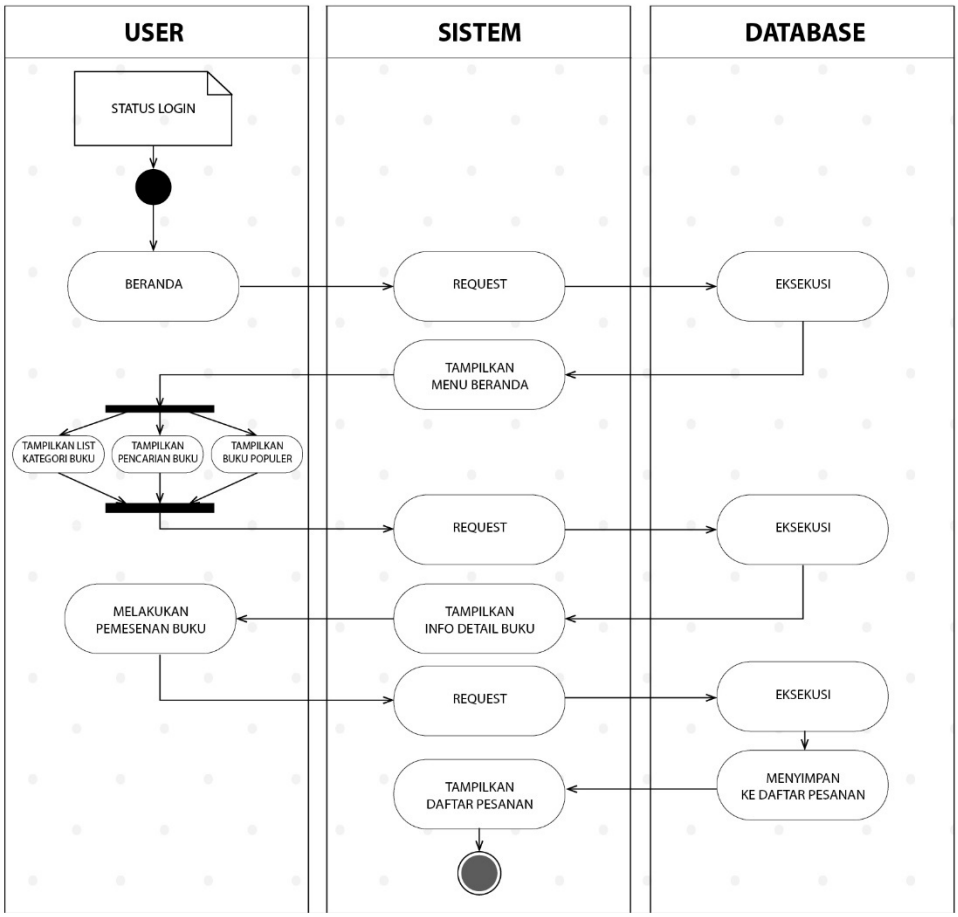
Class Diagram pada Aplikasi Pustakala terdapat 11 kelas yang terdiri dari; empat class pada *Firebase* yaitu, *Class User*, *Class* Buku, *Class* Pesanan dan *Class* Biaya, *class* ini juga berperan sebagai *backend* pada rancangan sistem yang akan dibuat. Pada sisi *fronted*, terdapat delapan class sebagai e*ntity*, yaitu *Class* Kategori, *Class* Search, *Class* Populer, *Class* Profil, *Class book*, *Class Favorite*, *Class Order*, dan *Class* Produk.

3. *Activity Diagram*

Activity Diagram menggambarkan aliran kerja (*workflow)* atau aktivitas pada sebuah proses bisnis atau sistem. Pada prinsipnya diagram

aktivitas menggambarkan aktivitas suatu sistem, bukan apa yang dilakukan aktor pada suatu sistem yang sedang berjalan atau dikembangkan.

Berikut adalah diagram aktivitas pada rancangan sistem aplikasi pustakala yang berbasis *online;*



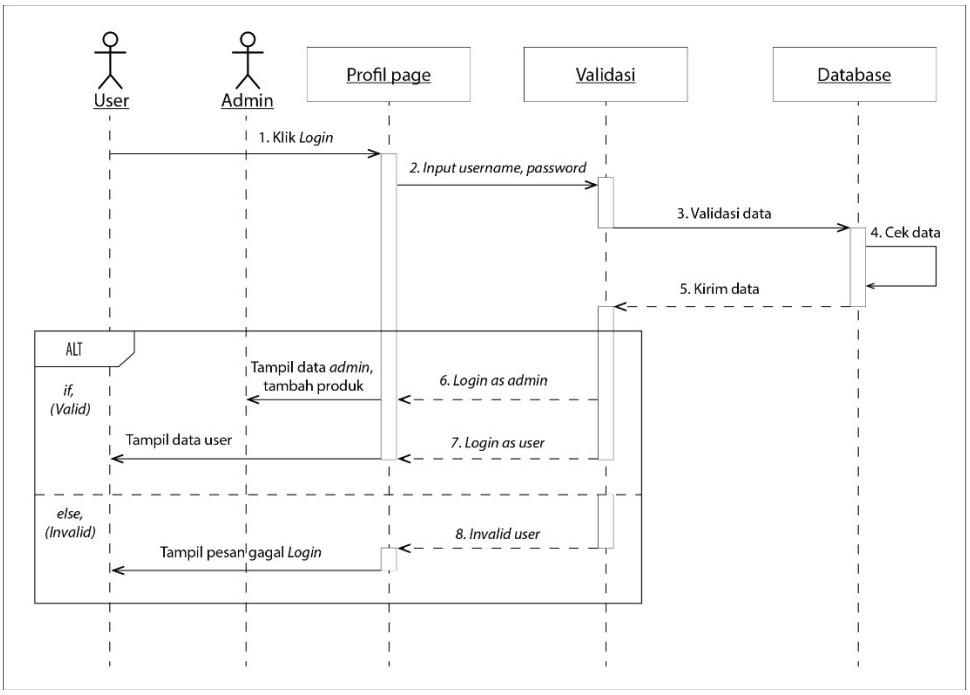
Gambar III-5 *Activity Diagram* Pemesanan Buku

4. *Sequence Diagram*

Salah satu diagram pada perancangan sistem adalah *Squence Diagram*, diagram ini menggambarkan kolaborasi atau hubungan sejumlah elemen yang dinamis untuk menunjukkan pesan yang dikirim dari elemen yang satu ke elemen yang lainnya, *Squence Diagram* adalah tentang bagaimana elemen-elemen tersebut melakukan interaksi dan kerjasama untuk mencapai suatu hasil dalam sebuah sistem.

Squence Diagram digunakan sebagai sebuah model untuk menjelaskan skenario dari sebuah kasus pengunaan sistem dengan melibatkan *object* dan aktor didalamnya.

Berikut adalah *Squence Diagram* pada rancangan sistem aplikasi pustakala yang berbasis *online*;



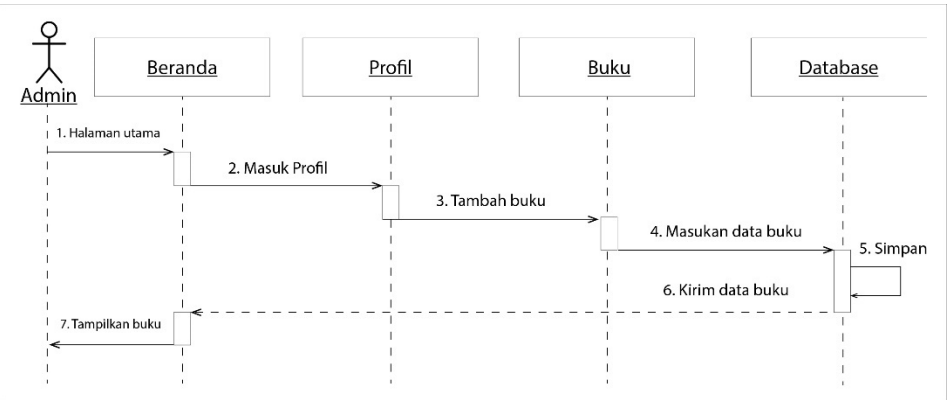
Gambar III-6 *Squence Diagram Login*

Pada gambar III-6 menjelaskan *Sequence Diagram* untuk proses *login* dengan dua tipe pengguna, yaitu *User* dan *Admin*.

Keterangan;

1. *User* dan *Admin* masuk kehalaman Profil, untuk melakukan *Login.*
2. *User* dan *Admin* melakukan *Login* menggunakan *username* dan *password.*
3. Sistem melakukan validasi berdasarkan *username* dan *password*. Dalam kasus ini validasi menggunakan *Firebase Auth.*
4. Database melakukan cek data *username* dan *password.*

5. Sistem mengirimkan hasil cek database dengan tiga kondisi alternatif (ALT).
6. ALT *"if"*, pengguna *login* sebagai admin, halaman profil menampilkan data pengguna dan menu "Tambah Produk".
7. ALT *"if"*, pengguna *login* sebagai *user*, halaman profil menampilkan data pengguna.
8. ALT *"else"*, pengguna tidak dikenali atau *invalid*, menampilkan pesan "Gagal *Login*".

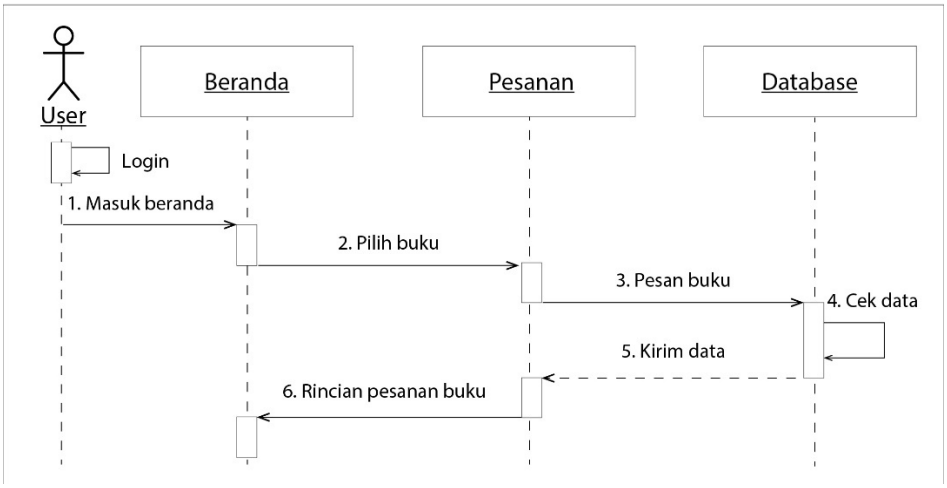


Gambar III-7 *Sequence Diagram* Tambah Produk

Pada gambar III-7 menjelaskan *Sequence Diagram* untuk proses menambahkan produk buku oleh pengguna dengan Level *Admin*.

Keterangan;

1. *Admin* membuka halaman Beranda.
2. *Admin* masuk ke halaman Profil.
3. *Admin* masuk menu Tambah Produk.
4. *Admin* memasukkan data buku.
5. Data buku disimpan.
6. Sistem mengirmkan data buku ke halaman Beranda.
7. Data buku ditampilkan di halaman Beranda.



Gambar III-8 *Sequence Diagram* Pemesanan Buku

Pada gambar III-8 menjelaskan *Sequence Diagram* untuk proses pemesanan buku oleh pengguna dengan Level *User*.

Keterangan;

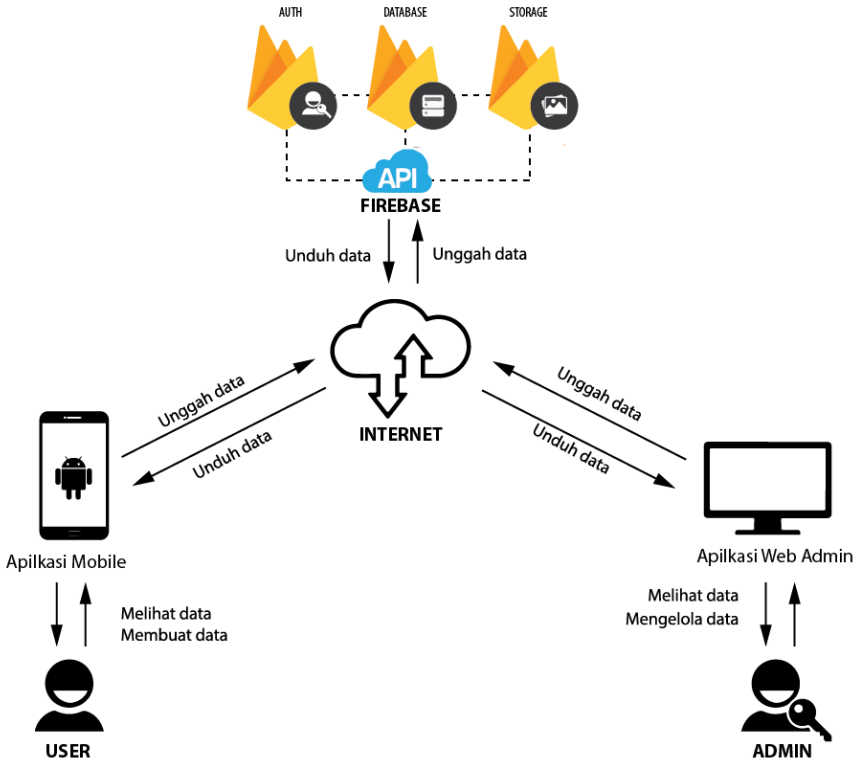
1. *User* dengan status *Login* membuka halaman Beranda.
2. *User* memilih buku.
3. *User* melakukan pemesanan buku.
4. Cek data buku dan pemesanan pada *database*.
5. Data pemesanan buku dikirim dari *database*.
6. Rincian data pemesanan buku ditampilkan di halaman Pesanan.

3.3.3.3 Rancangan Arsitektur Sistem

Arsitektur sistem yang akan dibuat dalam perancangan aplikasi Pustakala ini terdiri dari aplikasi Pustakala itu sendiri sebagai *Frontend. Frontend* adalah komponen visualisasi yang terdapat dalam suatu sistem atau aplikasi yang berinteraksi dengan *user*, termasuk didalamnya UX dan UI.

Komponen utama kedua dalam rancangan aplikasi ini adalah *Backend*, dalam penelitian ini penulis memanfaatkan layanan *Firebase*

Backend Service, dari *Google* untuk mengelola *database, authentication process, dan storage* yang disajikan dan diproses secara *realtime.* Pada Gambar III-9 secara umum digambarkan arsitektur sistem menggunakan *Firebase Realtime Database* (F-RDB)



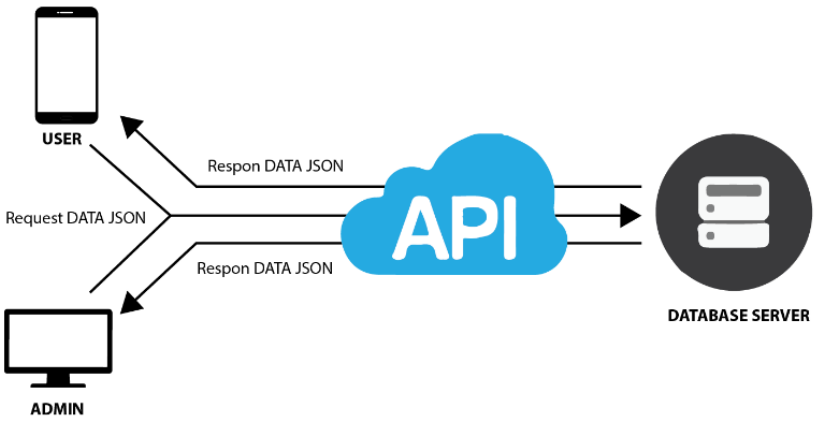
Gambar III-9 Arsitektur Sistem Menggunakan Firebase RDB

Keterangan Gambar III-9;

1. Pengguna melakukan aktivitas *Signup* dan *Signin* melalui aplikasi *mobile (frontend)* yang dibangun dengan bahasa pemrograman *Dart*.
2. Data pengguna yang telah didaftarkan di validasi melalui *Firebase Authentication*, dan kemudian disimpan dalam *Firebase Database*.
3. Admin dapat melihat dan mengelola data pengguna melalui *Console Dashboard Firebase* menggunakan aplikasi berbasis *(backend)*.

Selain menjadi *Backend Service*, *Firebase* juga menyediakan layananpengembangan aplikasi API *(Application Program Interface)*. API adalah service yang menyediakan data dari server yang di *request* oleh *client* agar dapat bekerja pada *platform* yang berbeda dan disajikan dalam format JSON *(Javascript Object Notation)*.

API ini adalah bagian bagian penting dalam sebuah arsitektur sistem aplikasi perangkat bergerak atau *mobile -frontend.*



Gambar III-10 *Basic* Arsitektur JSON API

Keterangan Gambar III-10;

1. *Client*, dalam hal ini *user* dan *admin* melakukan *request* data, misalnya *Signup, Signin* dan lainnya.
2. Rest API kemudian meneruskan *request* tersebut dengan sebuah *query (ex:get, put, post,delete)* ke *Database Server*.
3. *Request* data dari Rest API kemudian dieksekusi oleh *database* dan direspon kembali ke Rest API, lalu diteruskan ke *Client.*

3.3.3.4 Struktur *Firebase*

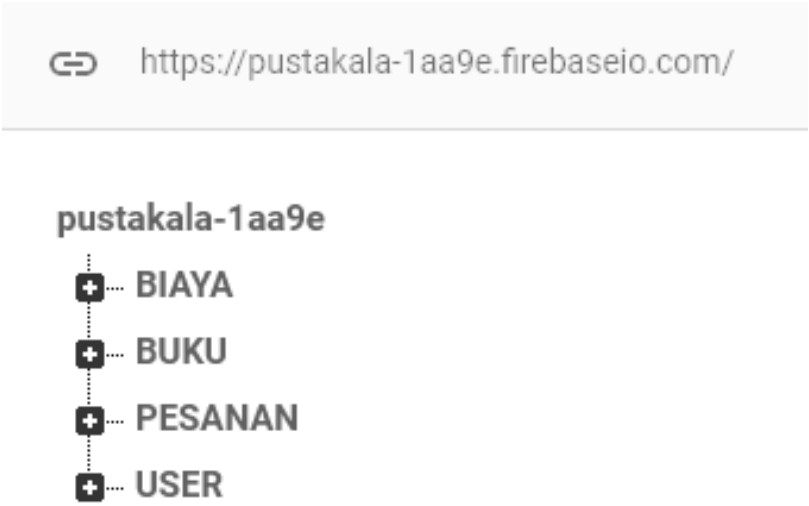
Firebase adalah sebuah *Cloud Service* yang diberikan Google untuk para pengembang aplikasi agar lebih fokus pada pengembangan aplikasi *mobile* dan *web*. Pengembang tidak perlu lagi membuat fitur-fitur yang biasanya dibangun pada sebuah backend saat membuat sebuah infrastruktur, sehingga pengembang bisa fokus pada untuk

mengembangkan aplikasi yang akan memberikan dampak yang luas pada masyarakat.

Pada rancangan aplikasi Pustakala ini menggunakan tiga fitur yang terdapat pada *Firebase*, yaitu; *Firebase Realtime Database*, *Firebase Authentication*, dan *Firebase Cloud Storage*.

1. *Firebase* Realtime Database

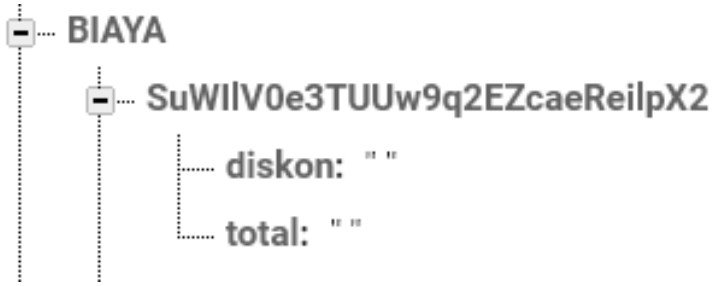
Salah satu fitur unggulan pada *Firebase* adalah *Realtime Database* yang merupakan sebuah *Cloud Hosted Database* yang artinya *database* tersebut dapat menyimpan dan mensinkronasi data secara langsung untuk semua *Client* yang terhubung. Ketika pengguna memperbaharui data, data tersebut akan di simpan di *Cloud*, dan saat itu juga memberikan notifikasi kepada setiap *Client* yang terhubung.



Gambar III-11 Struktur *Firebase Database*

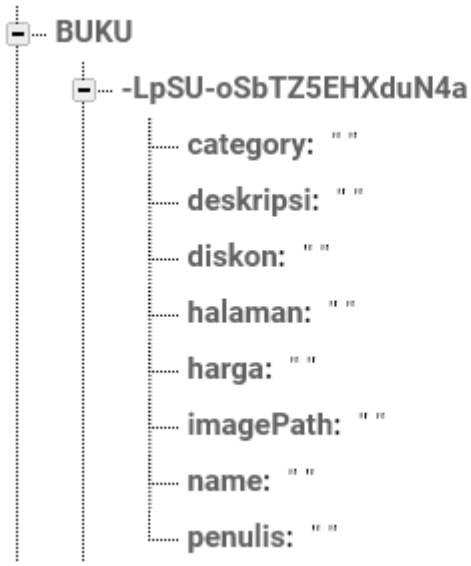
Data pada *Firebase* RDB disimpan sebagai objek JSON, *Firebase* adalah *Database Cloud* NoSQL yang struktur datanya direpresentasikan dengan sebuah Pohon (JSON *Tree*) yang di*hosting* di *cloud*. Tidak seperti MySQL yang menggunakan tabel, ketika kita menambahkan data pada pohon JSON struktur data pada F-RDB akan membentuk sebuah *node*.

Terdapat empat *Parent* pada struktur *database* JSON untuk Aplikasi Pustakala, yaitu Biaya, Buku, Pesanan dan *User*.Pada masing-masing *parent* terdapat *node*, atau *sub class* yang disebut *child.*



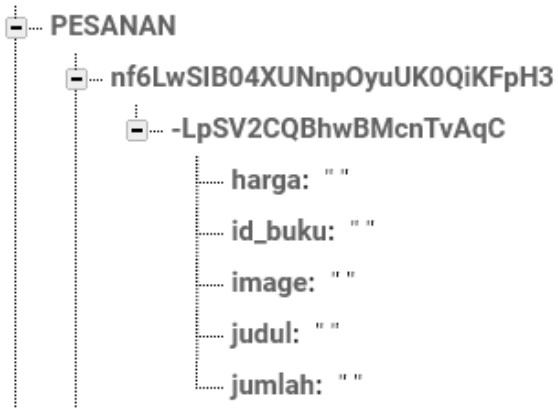
Gambar III-12 *Parent* Biaya

Pada Gambar III-12 terlihat pada *parent* BIAYA terdapat 1 *child* dengan dua atribut "diskon" dan "total". *Parent* ini berisikan data diskon dan total transaksi yang ditampilkan pada *form* pesanan pengguna.



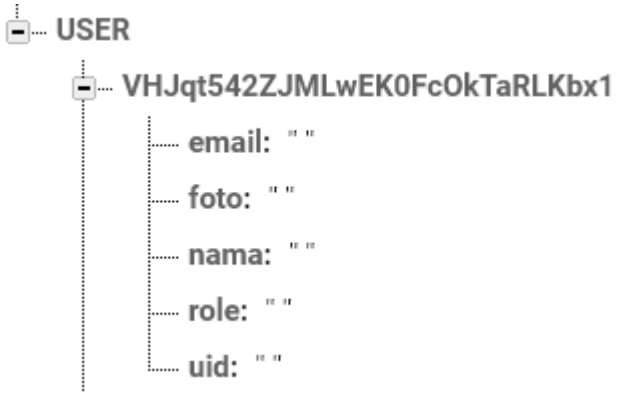
Gambar III-13 *Parent* Buku

Pada Gambar III-13 terlihat pada *parent* BUKU terdapat 1 *child* dengan delapan atribut yaitu "*category*", "deskripsi", "diskon", "halaman","harga", "*imagePath*", "*name*" dan "penulis". *Parent* ini berisikan data buku yang ditampilkan pada *form* detail buku.



Gambar III-14 *Parent* Pesanan

Pada Gambar III-14 terlihat pada parent PESANAN terdapat 2 *child*, *child*1 adalah sebuah *"uid"* yang mengidentifikasikan pengguna, *child*2 adlah data pemesanan yang dilakukan *child*1 dengan lima atribut yaitu "harga", "id_buku", "image", "judul", dan "jumlah". *Parent* ini berisikan data buku yang dipesan oleh pengguna yang akan ditampilkan ditampilkan pada *form* Pesanan.



Gambar III-15 *Parent User*

Pada Gambar III-15 terlihat pada parent *USER* terdpat 1*child*, dengan lima atribut yaitu "*email*", "foto", "nama", "*role*", dan "*uid*". *Parent* ini berisikan data pengguna yang ditampilkan ditampilkan pada *form* Profil.

2. *Firebase Authentication*

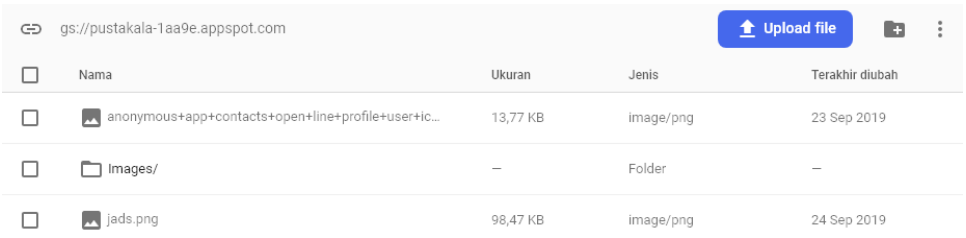
Firebase Authentication (auth) adalah layanan *backend* yang digunakan untuk mengautentikasi pengguna pada sebuah aplikasi dengan menggunakan sandi, nomor telepon, akun sosial media dan lainnya.

Gambar III-16 *Dashboard Firebase Authentication*

Metode autentikasi pengguna pada aplikasi pustakala terlihat seperti pada Gambar III-16, yaitu menggunakan kata sandi dengan alamat *email* yang terdaftar. Pada *dashboard* ini menampilkan informasi antara lain; pengenal, penyedia, tanggal dibuatnya akun, *history* masuk akun, dan sebuah *unique id (uid)* yang akan dipakai untuk mengidentifikasi pengguna pada *database* termasuk level pengguna mengunakan sebuah *"role"*.

3. *Firebase Storage*

Firebase Storage sejatinya adalah layanan dukungan untuk penyimpanan objek berupa gambar, audio, video atau konten lainnya disertai dengan dukungan *server* yang handal saat melakukan aktivitas unggah atau unduh data. *Firebase Storage* juga dilengkapi model kemanan deklaratif dengan mengatur perizinan hak akses berdasarkan nama file, ukuran, jenis, dan metadata lainnya.



Gambar III-17 *Dashboard Firebase Storage*

3.3.3.5 Rancangan Desain Fisik

Rancangan desain fisik adalah sebuah konsep estetika untuk menciptakan desain antar muka sebuah aplikasi dengan tidak mengesampingkan fitur pada aplikasi sehingga menjadi indah dan mudah diterima pengguna, antara lain; tampilan pada halaman Beranda (*home*), tampilan pada halaman Pesanan, tampilan pada halaman Profil dan seterusnya. Berikut konsep dan penjelasan dari masing masing *form*:

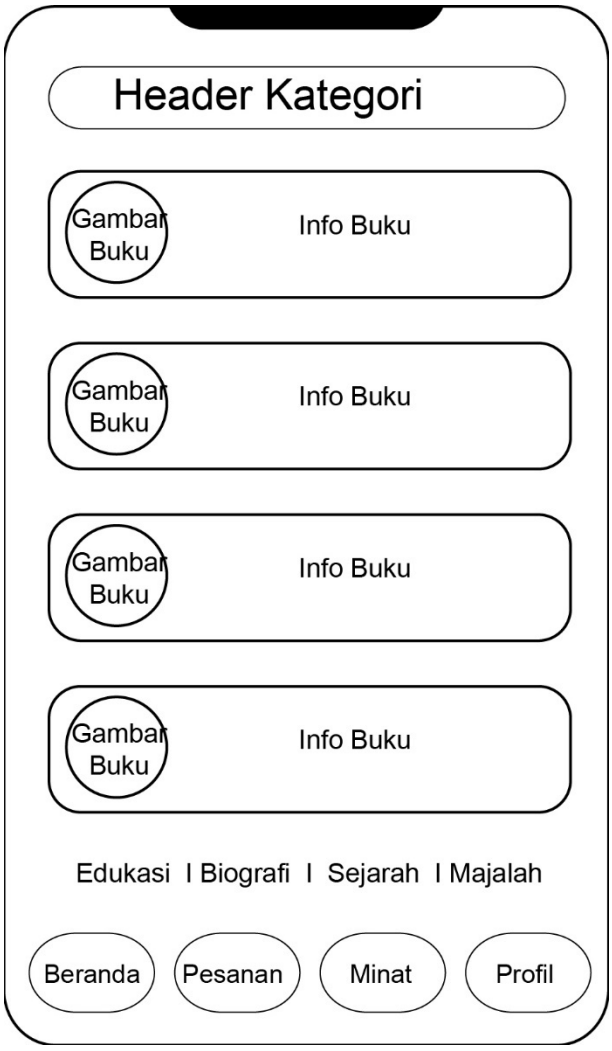
1. Rancangan Desain *Form* Beranda

Pada halaman Beranda, *form* ini terdiri dari *Header*, Notifikasi, Kategori, Kolom Pencarian, daftar buku berdasarkan Popularitas, dan terdapat empat Navigation Bar terdiri dari *Home*, Pesanan, Diminati dan Profil.

Gambar III-18 Rancangan *Form* Halaman Utama

2. Rancangan Desain *Form* Kategori

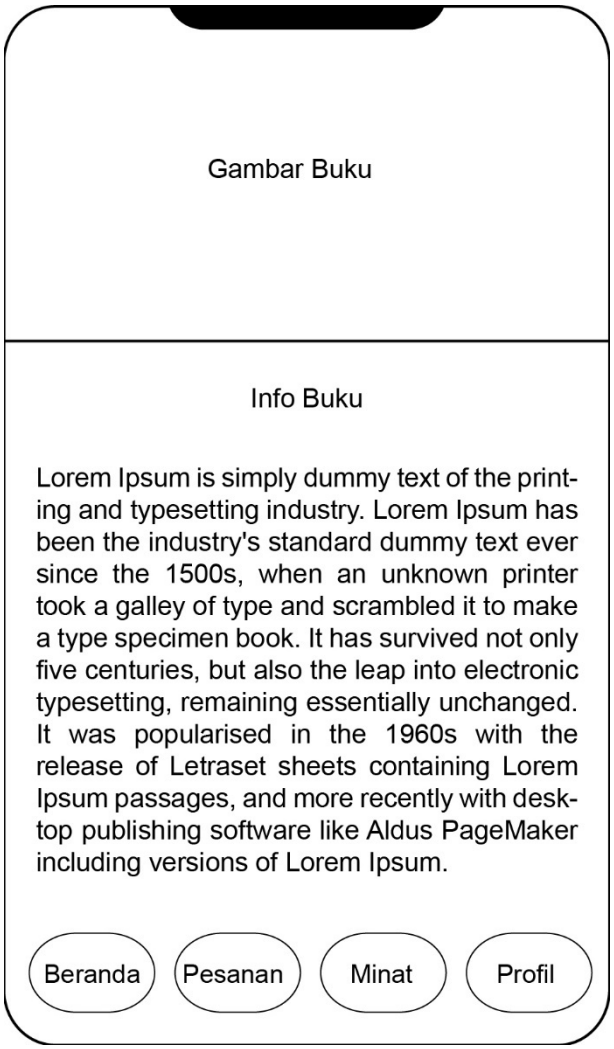
Pada halaman Kategori, *form* ini berisikan info buku dengan menggunakan *ListView* sesuai dengan kategori yang dipilih oleh pengguna *(user*). Kategori buku tersebut antara lain; Edukasi, Biografi, Sejarah dan Budaya, Majalah dan Surat Kabar.



Gambar III-19 Rancangan *Form* Halaman Kategori

3. Rancangan Desain *Form* Info Buku

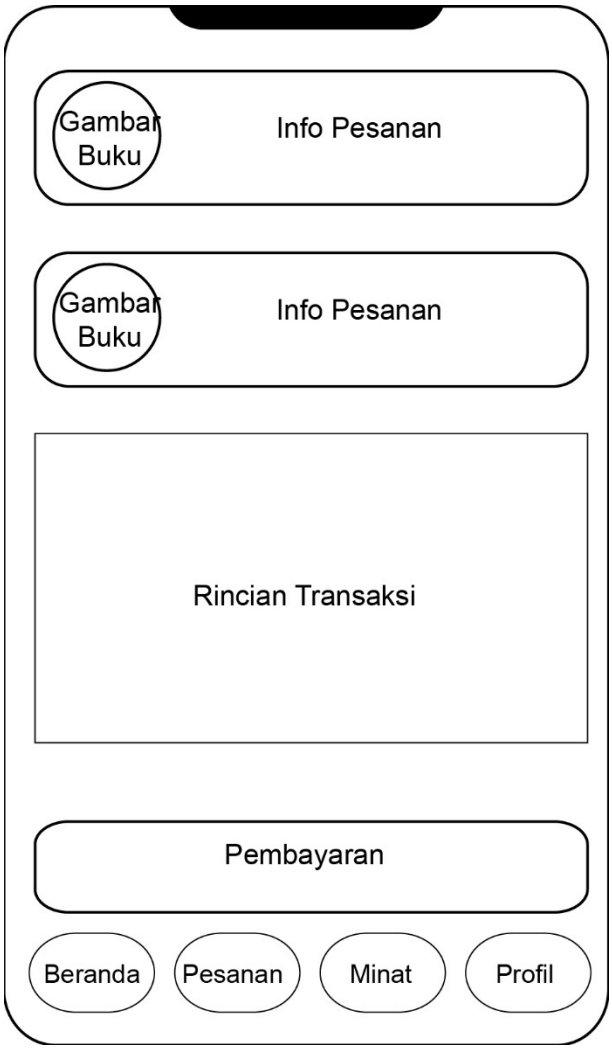
Pada halaman Info Buku, *form* ini berisikan info buku mencakup; gambar buku, harga buku, resensi buku, jumlah Bab, jumlah halaman, dan Penerbit.



Gambar III-20 Rancangan *Form* Info Buku

4. Rancangan *Form* Pesanan

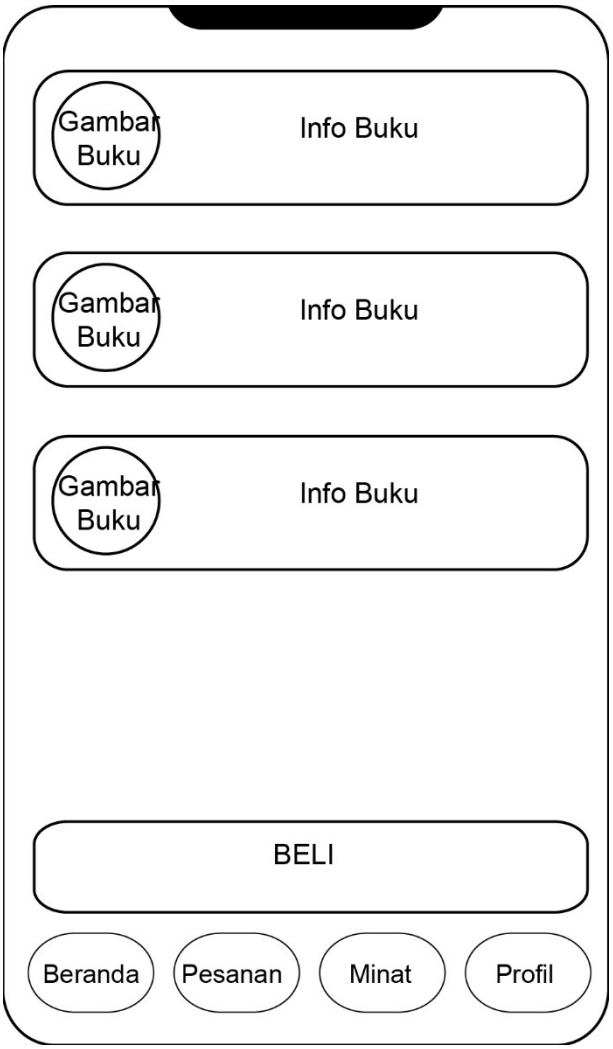
Pada halaman Pesanan, *form* ini menampilkan detail pesanan buku mencakup; Info buku, jumlah pesanan, nilai transaksi, dan tombol "PEMBAYARAN".



Gambar III-21 Rancangan *Form* Pesanan

5. Rancangan *Form* Minat

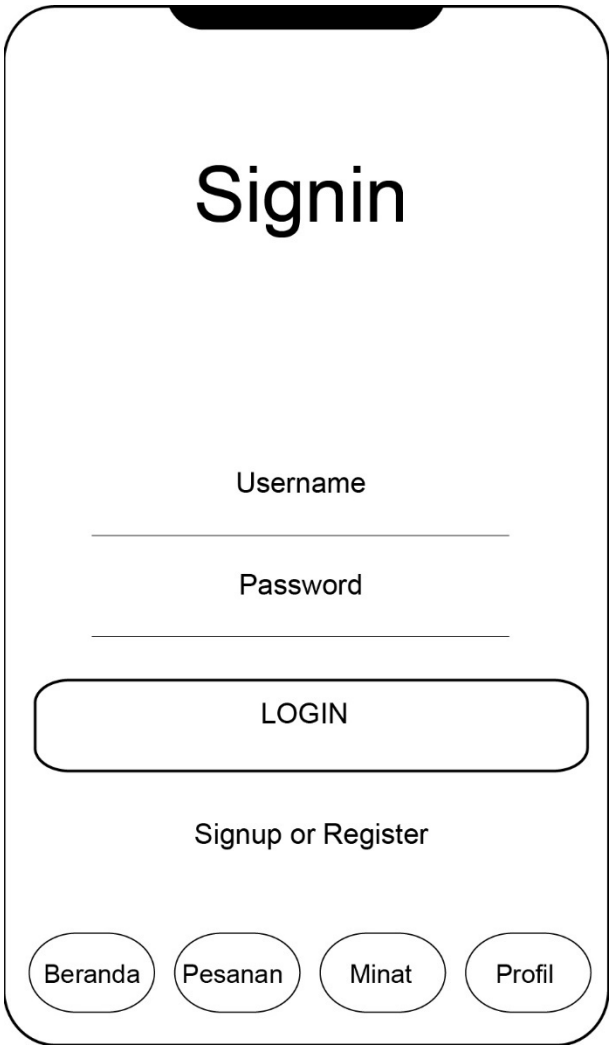
Pada halaman Pesanan, *form* ini menampilkan detail buku yang diminati oleh pengguna mencakup; judul buku, harga buku, dan tombol "BELI".



Gambar III-22 Rancangan *Form* Diminati

6. Rancangan *Form Login*

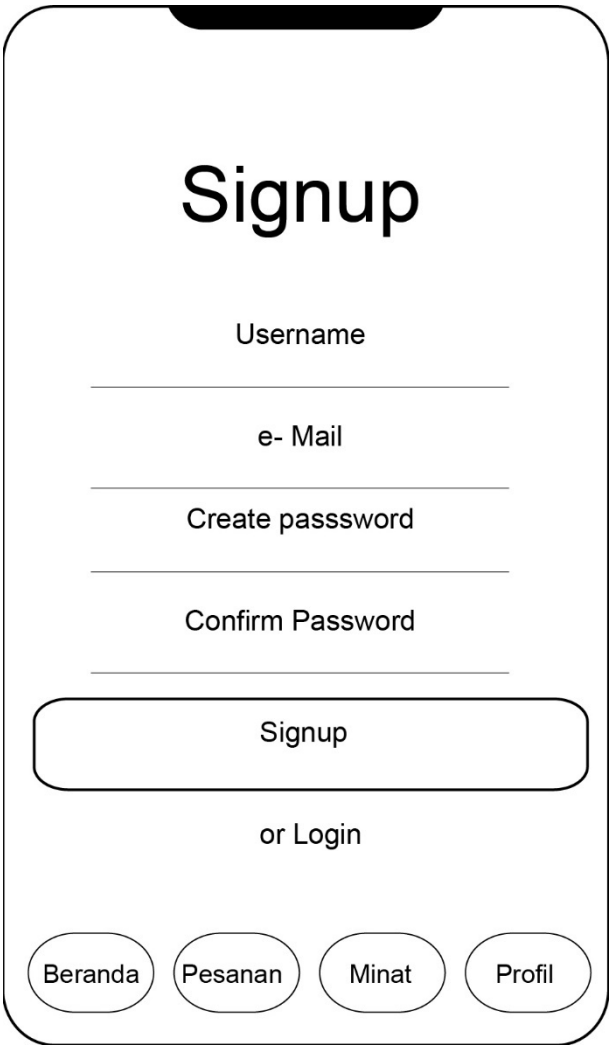
Pada halaman Login, *form* ini menampilkan kolom verifikasi berupa *"username"* dan *"password"* yang harus diinput bagi pengguna yang sudah terdaftar.



Gambar III-23 Rancangan *Form Login*

7. Rancangan *Form Signup*

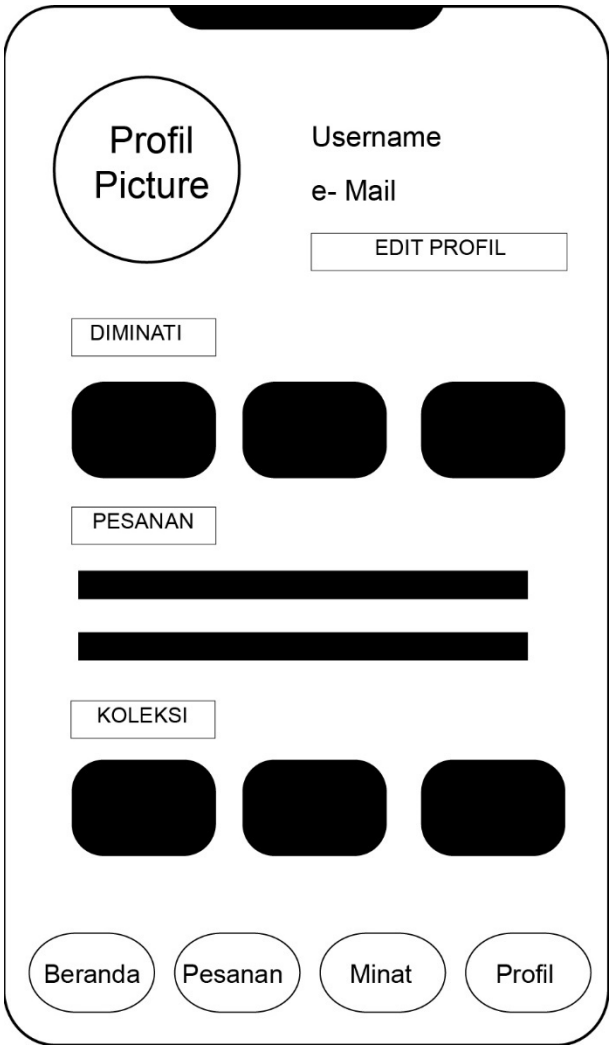
Pada halaman *Signup*, *form* ini menampilkan kolom input data untuk melakukan registerasi bagi pengguna agar bisa melakukan transaksi kolom tersebut antaralain: *"username"*, *"email"* dan *"create password"* dan *"confirm password"*.



Gambar III-24 Rancangan *Form* Pendaftaran *(Signup)*

1. Rancangan *Form Profile*

Pada halaman *Profile*, *form* ini hanya bisa dibuat atau diakses bagi pengguna terdaftar, menampilkan informasipengguna, antara lain; *username*, *email*, info buku yang diminati, *history* transaksi dan informasi dasar lainnya.



Gambar III-25 Rancangan *Form User Profile*

3.3.4 Implementasi

Setelah melakukan evaluasi terhadap *design solution* langkah selanjutnya adalah implementasi. Tahapan ini merupakan tahapan dimana perancangan sistem diubah menjadi bahasa pemrograman untuk menghasilkan suatu bentuk rancangan sistem yang dapat digunakan oleh pengguna.

Dalam penelitian ini alat tersebut berupa perangkat keras *(hardware)* maupun perangkat lunak *(software),* yaitu:

1. Perangkat Keras

Spesifikasi Perangkat keras *(hardware)* yang digunakan dalam perancangan sistem adalah sebagai berikut:

a. PC dengan spesifikasi processor Intel(R) Core i3, 2,20 Ghz.

b. *Memory* (RAM) 4 GB.

c. *Harddisk* 1 TB.

d. Monitor 19 Inch.

e. Keyboard dan Mouse.

2. Perangkat Lunak

Perangkat lunak *(software)* yang dapat digunakan dalam perancangan sistem adalah sebagai berikut:

a. Sistem operasi *Microsoft Windows 10 Pro.*

b. Android *Studio*

c. *Java SDK*

d. Visual Studio Code

e. *Flutter Framework*

f. *Adobe Illustrator*

3.3.5 Pengujian

Pengujian dilakukan untuk mengetahui apakah implementasi sistem sudah sesuai dengan perancangan dan analisis kebutuhan, juga untuk mengetahui

apakah ada *error* atau *bug* pada sistem. Didalam proses pengujian ada beberapa metode yang digunakan yaitu pengujian unit, pengujian *integrase*, pengujian validasi dan pengujian *usability* serta pengujian *compatibility.*

Pengujian unit, digunakan untuk menguji beberapa metode yang ada pada *class* yang didapatkan dari hasil perancangan. Pengujian ini dilakukan menggunakan metode *white box testing*. Jenis pengujian yang dilakukan adalah dengan menggunakan basis *path testing* untuk menguji kode program berdasarkan algoritma dari setiap metode.

Pengujian validasi, merupakan pengujian yang digunakan untuk menguji seluruh kebutuhan fungsional sistem apakah sudah sesuai dengan kebutuhan yang didapatkan sebelumnya. Pengujian ini dilakukan dengan metode *black box.*

Pengujian *usability,* merupakan pengujian yang digunakan untuk mengukur tingkat pengalaman pengguna ketika menggunakan sistem ini. Pengujian yang akan dilakukan adalah efektivitas, efisiensi dan tingkat kepuasan pengguna ketika menggunakan sistem ini. Untuk mendapatkan data, maka dilakukan pembagian kuisioner kepada calon pengguna. Untuk soal dan hasil kuisioner dari pengujian *usability* terlampir pada halaman lampiran.

Pengujian *compatibility,* merupakan pengujian yang digunakan untuk mengetahui apakah aplikasi sudah berjalan dengan baik di berbagai *device* pengguna dengan spesifikasi yang berbeda. Pengujian dilakukan pada lima perangkat dengan parameter sistem operasi Android yang berbeda, resolusi layar dan fitur sensor *gyroscope*. analisis dilakukan dengan ujicoba menjalankan aplikasi pada perangkat Android. Selanjutnya dilakukan perhitungan skor presentase hasil pengujian dan dicocokan dengan akala penilaian untuk mengetahui tingkat kualitas aspek *compatibility* aplikasi. (Liu, et al., 2014)

Dengan dilakukannya analisis pengujian maka akan didapatkan apa saja kekurangan dari sistem dan mengetahui apa saja yang harus diperbaiki dalam sistem, sehingga hasil dari analisis pengujian akan dijadikan sebagai isi dari kesimpulan dan saran.