Tugas Besar 1 - Android

IF3210 Pengembangan Aplikasi pada Platform Khusus

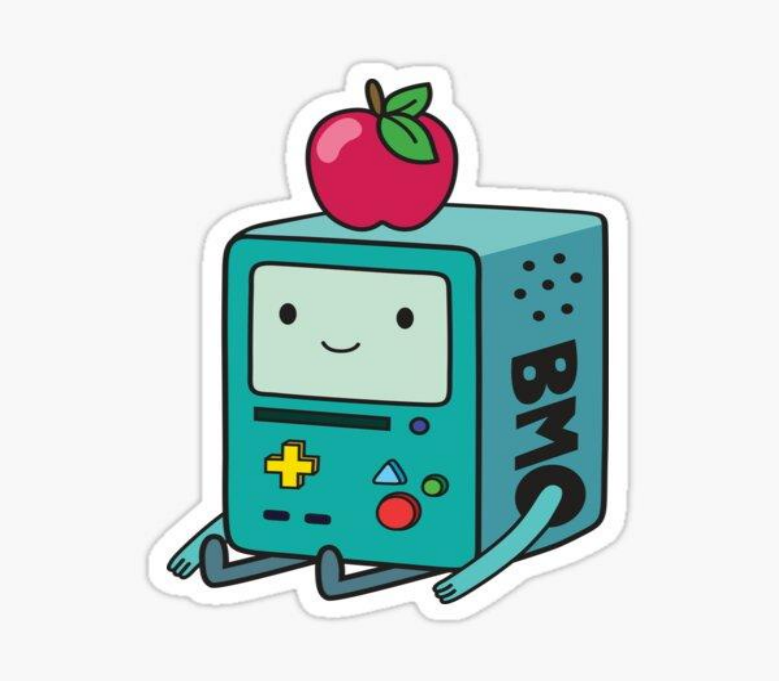
Aplikasi Majika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2023

Latar Belakang



"BNMO pusing karena harga buah muuuahallllll"

Doni sekarang sudah memasuki semester 6. Konon katanya semester ini merupakan semester yang mengerikan. Namun, Doni bukanlah mahasiswa pada umumnya. Ia malah merasa haus dan lapar akan tubes-tubes yang akan datang.

Semangat Doni hanya bisa ditambah jika mulutnya merasakan semua jenis rasa, mulai dari gurih, manis, asam, asin bahkan pahit. Oleh karena itu, untuk membantu temannya yang sedang dimabukkan oleh tubes, BNMO pun ingin membuat aplikasi pembelian makanan berbasis android.

Akan tetapi, BNMO tidak terlalu paham bahasa Java dan turunannya ~~karena bersifat jawasentris dan BNMO bukan orang jawa~~ karena BNMO sering bolos kelas Pemrograman Berorientasi Objek. Apalagi Doni hanya punya android karena harga buah-buahan mahal ~~(selain buah beri hitam, contoh buah mahal: apel)~~. *Yuk* dibantu *yuk* BNMO-nya untuk buat aplikasi ini!

Tujuan

Berikut adalah tujuan dari tugas besar ini:

1. Mahasiswa memahami prinsip dasar dalam platform Android
2. Mahasiswa mampu bekerja dalam tim pada konteks pengembangan aplikasi Android
3. Mahasiswa mampu mempresentasikan perangkat lunak yang sudah dibangun.

Spesifikasi Aplikasi

Berikut adalah spesifikasi yang harus dipenuhi oleh **Aplikasi Majika**:

1. **Header dan Navbar**
   - Terdapat empat *navigation menu*, yaitu Menu Twibbon, Menu Cabang Restoran Terdekat, Menu Daftar Makanan dan Minuman, dan Menu Keranjang.
   - Jika berada di menu tertentu, berikan *cue* pada *icon* menu tersebut, serta *header*-nya berubah menyesuaikan menu tersebut (kecuali menu twibbon).
   - Referensi dapat dilihat pada Lampiran.

2. **Menampilkan Daftar Makanan dan Minuman — Halaman Menu**
   - Pada awalnya aplikasi akan berada pada halaman ini.
   - Halaman ini terdiri dari **dua *section*** yaitu *section* makanan dan *section* minuman.
   - Ketika pengguna memilih suatu makanan atau minuman, makanan atau minuman tersebut secara otomatis masuk ke daftar keranjang.
   - Daftar semua makanan dan minuman didapatkan dengan melakukan HTTP request ke API *server* Majika.
   - Berdasarkan API *response* yang diterima, aplikasi akan menempatkan makanan dan minuman pada *section*-nya masing-masing.
   - Pengguna juga dapat mencari makanan atau minuman dengan ***sticky search bar*** yang disediakan pada Halaman Menu.
   - API *server* tidak menyediakan *interface* untuk melakukan *filtering search query*, sehingga *filtering* hanya dilakukan dari **sisi aplikasi/*frontend***.
   - Halaman ini dapat menampilkan **informasi suhu ruangan**, menggunakan **sensor temperatur** pada Android, dan dapat dilihat pada sebelah kanan atas.
   - Halaman ini diwajibkan untuk *responsive*.
   - Data bisa diambil dari backend dengan menembak *endpoint* berikut.
     - **{{baseUrl}}/v1/menu** untuk semua menu tanpa difilter
     - **{{baseUrl}}/v1/menu/food** untuk semua menu makanan
     - **{{baseUrl}}/v1/menu/drink** untuk semua menu minuman

3. **Menampilkan Keranjang — Halaman Keranjang**
   - Daftar keranjang disimpan dalam **Room**.
   - Halaman ini akan menampilkan seluruh isi dari daftar keranjang.
   - Pengguna dapat mengetahui nama, harga, dan jumlah pada daftar keranjang untuk masing-masing makanan dan minuman.
   - Pengguna dapat mengurangi dan/atau menambahkan jumlah makanan/minuman dengan adanya tombol “-” dan “+”.
   - Total harga ditampilkan, beserta tombol bayar yang akan mengarah ke Pembayaran.

4. **Pembayaran — Halaman Pembayaran**
   - Pengguna yang sudah memilih makanan melakukan pembayaran dengan melakukan scan QR Code.
   - QR Code yang dapat di-*scan* tersedia pada API Backend. Diakses melalui:

    - **{{baseUrl}}/v1/payment/success** untuk QR Code pembayaran yang sukses
    - **{{baseUrl}}/v1/payment/failed** untuk QR Code pembayaran yang gagal

- Isi data QR Code adalah *transaction_id*. Transaction ID dari *qrcode* adalah string dengan panjang 32.
  (contoh: avdppxqhrgukavhtwumoatshsguumvou)
- Setelah di-*scan*, aplikasi memanggil API Backend server yang dapat diakses di API Majika melalui **{{baseUrl}}/v1/payment/{{transaction_id}}**.
- Kemudian aplikasi menampilkan hasil pembayaran berdasarkan *response* yang diterima.
- Jika status = “SUCCESS”, pembayaran berhasil, tampilkan pesan pembayaran berhasil selama 5 detik, dan pengguna di-*redirect* ke Halaman Menu.
- Jika status = “FAILED”, pembayaran gagal. Ulang kembali sampai sukses.

5. Mencari Cabang Restoran — Halaman Cabang Restoran

- Pengguna dapat mencari cabang restoran dengan memanfaatkan API Majika.
- Terdapat tombol “Maps” untuk masing-masing cabang restoran, yang akan mengarah pengguna ke **Google Maps** sesuai dengan lokasi restoran yang ditekan.
- Daftar restoran diurutkan berdasarkan abjad nama cabang (a-z).
- **Catatan**:
    - Untuk mendapatkan *list* restoran
      -> **{{baseUrl}}/v1/branch**
    - Untuk mendapatkan restoran terdekat dapat menggunakan informasi jarak *latitude* dan *longitude* pengguna dan data restoran.

6. Fitur Twibbon — Halaman Twibbon

1. Di halaman ini, aplikasi membuka kamera dan menampilkan gambar dari kamera yang sudah dipasang twibbon. Cukup gunakan **satu** jenis twibbon saja.
2. Terdapat tombol **capture**. Fungsinya untuk menangkap foto pada saat itu tombol ditekan. Setelah itu, aplikasi menampilkan foto yang beserta twibbon. Foto tidak perlu di-*save* di perangkat, cukup ditampilkan saja. Setelah itu, pengguna dapat mengambil foto ulang.
3. Gambar Twibbon dibebaskan.
4. Gambar Twibbon disimpan ke storage Android (*files* ataupun *media*/*photos*).
5. Boleh menyelesaikan permasalahan dengan pendekatan apapun. Contohnya:
    a. Menggunakan konversi gambar ke bitmap lalu digabungkan. Keyword: *bitmap merging in kotlin*.
    b. Gambar dan *twibbon* disusun pada page tertentu lalu di *screenshot programmatically* (**bukan di screenshot manual**), lalu disimpan sebagai gambar. *Menyimpan gambar di kotlin juga via bitmap, jadi sama saja, hanya saja tidak perlu menggabungkan dua buah bitmap, kalau kalian depressed gak bisa gabungin bitmap mungkin bisa dicoba cara ini.*
    c. Cara kreatif lain jika ada.

Bonus

Berikut adalah fitur yang dapat ditambahkan pada **Aplikasi Majika**:

1. **Mengirimkan *Feedback***
    - Pengguna dapat mengirimkan *feedback* ke restoran. *Feedback* berupa pesan, kritik, saran, dan berapa bintang (dari skala 1-5) yang ingin diberikan kepada restoran.
    - *Feedback* yang diberikan oleh pengguna pada aplikasi nantinya akan dapat dilihat pada *email*.
    - *Email* pengirim dan pengguna dibebaskan.
    - Implementasi *Interface* dibebaskan.
2. **Multiple Twibbon**
    - Untuk Halaman Twibbon, terdapat **lebih dari 1** twibbon yang dapat digunakan.

Daftar Halaman

Untuk melihat contoh daftar halaman, dapat dilihat pada Lampiran.
Atau dapat juga dilihat dari Figma.
~~Cerita/Asset Gambar hanya fiktif belaka. Jika ada kesamaan nama tokoh, tempat kejadian ataupun cerita, itu adalah kebetulan semata dan tidak ada unsur kesengajaan.~~
Adapun yang harus diperhatikan:

- Tampilan pada halaman-halaman tersebut hanya spesifikasi minimum. Silakan berkreasi (*menambah*) tanpa batas.
- Diperbolehkan untuk menambahkan jenis informasi lain yang tersedia dari API namun tidak ada pada *figma* dan mengubah susunan selama memenuhi spesifikasi minimum tersebut.
- Diharuskan memuat semua informasi minimum yang ada pada *figma*. Tidak boleh mengurangi.
- Nilai bonus akan dipertimbangkan untuk diberikan apabila dapat memberikan tampilan yang menarik, indah untuk dilihat, dan halaman memuat jenis informasi yang lebih banyak dibanding minimum.
- Nilai bonus akan dipertimbangkan untuk diberikan jika menambahkan fitur-fitur yang meningkatkan *user experience* (terutama kemudahan saat Demo Tugas Besar dengan Asisten).
    - Contoh: filter category, halaman baru untuk …, menampilkan modal untuk … dan sebagainya.
    - Contoh yang merusak *User Experience*: Random Crash, Random Behavior (kadang ada data, kadang data kosong), harus ubah-ubah kode untuk pindah halaman, dsb.
- Akan ada penalti jika terdapat hal yang mengurangi *user experience* ataupun menyulitkan *testing* saat Demo Tugas Besar dengan Asisten.

Batasan Aplikasi

Berikut adalah batasan dalam implementasi aplikasi ini:

1. **Grid**
   Gunakan *RecyclerView* untuk menampilkan list makanan pada Halaman Menu, Halaman Cabang Restoran, dan Halaman Keranjang.
2. **Database**
   Sistem penyimpanan *cart* daftar makanan disimpan pada SQLite dengan memanfaatkan Room.
3. **Responsive Layout**
   Pada Halaman Menu, aplikasi menampilkan *search bar* di atas dan daftar makanan/minuman di bawah saat berada pada orientasi *portrait*. Kemudian, *search bar*

pada sebelah kiri, dan daftar makanan/minuman pada sebelah kanan berada pada orientasi *landscape*.

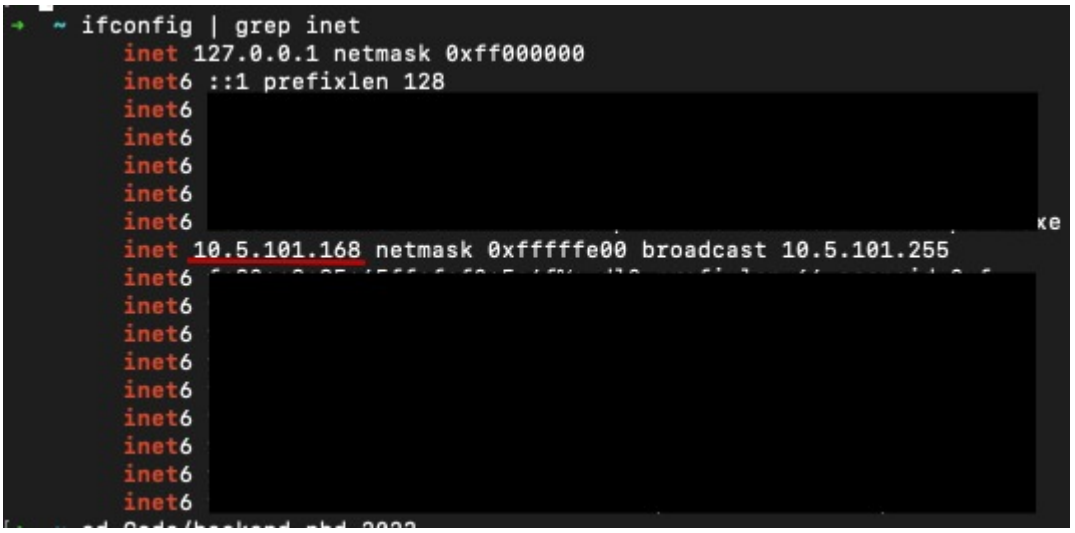
4. **Fragment**
   Fragment digunakan untuk Header dan Navbar. Komponen untuk setiap daftar makanan/minuman juga dapat diimplementasikan menggunakan Fragment.
5. **User Interface**
   Diperbolehkan menggunakan *wireframe* bebas, asalkan seluruh spesifikasi tetap dipenuhi. Disediakan *mockup* pada bagian Daftar Halaman sebagai acuan pembuatan UI aplikasi, tidak diwajibkan untuk mengikuti persis. Tampilan yang indah (enak dilihat), efisien, dan menarik akan mendapatkan **nilai tambahan**.
6. **Intent**
   Intent digunakan untuk mengkomunikasikan aksi dalam dan antar aplikasi. Dalam aplikasi ini, Intent digunakan untuk membuka **Google Maps** saat menampilkan lokasi cabang restoran lainnya dan **Gmail** untuk mengirim *feedback* ke restoran. Penggunaan Intent pada tempat lain dibebaskan sesuai pengerjaan kelompok.

Backend

1. *Executable Backend* dapat diunduh melalui link berikut. Silakan diunduh sesuai dengan spesifikasi dan sistem operasi masing-masing.
2. Cara menjalankan executable bukan di-*double click*. Melainkan di *execute* melalui terminal. Contoh untuk Linux dan MacOS: `./nama_executable_file`
3. Untuk menjalankan:
   a. `./nama_executable_file -a true`
      Untuk melihat List Endpoint
   b. `./nama_executable_file -p 8000`
      Untuk menjalankan *backend* di port 8000 pada *host device* (komputer anda) dengan menggunakan random data seed 0.
   c. `./nama_executable_file -p 8000 -s 123`
      Untuk menjalankan *backend* di port 8000 pada *host device* (komputer anda) dengan menggunakan random data seed 123.
4. **Backend mengirim data yang selalu sama untuk sebuah *seed*.**
   Jangan melakukan hardcode data di aplikasi.
   Demo Tugas Besar akan menggunakan *backend* dengan *seed* yang ditentukan asisten.
5. *Backend* yang dijalankan pada suatu *host device* (komputer anda) akan tidak bisa diakses melalui *android* baik *device* eksternal maupun *virtual device* melalui *localhost*. Jadi bagaimana cara mengaksesnya?
   a. Untuk Eksternal *Device* (*Device* fisik melalui USB): Cari tahu *ip address host device* kalian.
      i. Untuk Windows: `ipconfig /all`
      ii. Untuk Linux & MacOS: `ifconfig | grep inet`
      Perintah diatas pastinya akan memunculkan banyak angka, silakan aplikasikan ilmu *jaringan komputer* semester lalu.

Setelah dapat, anda bisa mengakses langsung dari Android kalian melalui ip lokal tersebut. Contoh: didapat `192.168.1.100` maka aksesnya HTTP Request ke `192.168.1.100:8000/v1/menu`. Hint: biasanya 192.x.x.x atau 10.x.x.x dan gunakan yang ipv4 saja. Android dan *host device* harus terhubung ke jaringan yang sama agar dapat mengakses IP lokal tersebut.

Contoh:



IP lokal *device*: 10.5.101.168

b. Untuk *Virtual Device* (Melalui ADB): perlu dilakukan *port reversing*.
Perintah: `adb reverse tcp:80 tcp:3000`
Artinya: Port 80 di Android ditujukan ke port 3000 *host device*.
Perlu diperhatikan **tcp:3000** harus disesuaikan dengan port yang kalian letakkan pada argumen langkah nomor 3. Jika anda tulis 8000, maka ganti jadi **tcp:8000**. Setelah berhasil, Android pada *Virtual Device* dapat mengakses `localhost/v1/menu`.

6. **Backend wajib digunakan. *Dilarang hardcode data!* Akan ada penalti besar apabila data di hardcode.**

Pengerjaan

Tugas besar 1 - Android **dikerjakan secara berkelompok**.

Sheets Kelompok: IF3210 PBD - Kelompok

- Kelompok dibebaskan kepada mahasiswa dan harus terdiri dari maksimal **3** orang dan boleh lintas kelas, harap isi segera Sheets Daftar Kelompok Android pada Teams (Deadline Pengisian Sheets: **5 Februari 2023 23:59 WIB**).
- Tidak mengisi Sheets akan dianggap sebagai mengerjakan tugas besar **sendiri**.
- Tugas besar 1 wajib dikerjakan menggunakan ***Kotlin-Android Native***. *Bukan Java ya teman-teman* 🙂.
- Wajib **Android Native**, tidak diperkenankan memakai *framework*/*library* lain seperti React Native, Flutter, dsb.
- Deadline pengerjaan tugas besar adalah **Kamis, 24 Februari 2023 16:00 WIB**.
- Setiap aplikasi dibuat pada *repository* Gitlab Informatika dengan format nama sebagai berikut
  `IF3210-2023-Android-XXX` dengan `XXX` = penomoran kelompok (3 huruf kapital).
- Pastikan aplikasi bisa berjalan. Jika tidak berjalan sama sekali, asisten tidak bisa menilai apapun.
- Disediakan *backend* dari Asisten dan wajib menggunakan *backend* yang telah disediakan.

Pada setiap *repository* ditambahkan sebuah README yang setidaknya berisi:

1. Deskripsi aplikasi.
2. *Library* yang digunakan.
3. *Screenshot* aplikasi (dimasukkan dalam folder *screenshot*).
4. Pembagian kerja anggota kelompok.

Kriteria Penilaian

Berikut merupakan kriteria penilaian dari tugas besar ini:

1. Memilih teknik dan teknologi yang tepat untuk memenuhi spesifikasi.
2. Pemahaman terhadap platform Android.
3. Kreativitas & Keindahan aplikasi.
4. *User Experience* Asisten pada aplikasi android yang dibuat saat dilakukan *testing* pada saat Demo.
   Contoh yang **mengurangi** *User Experience*: Random Crash, Halaman yang sulit untuk digapai (harus ubah-ubah kode), Random behavior (data kadang muncul, kadang tidak), dsb.

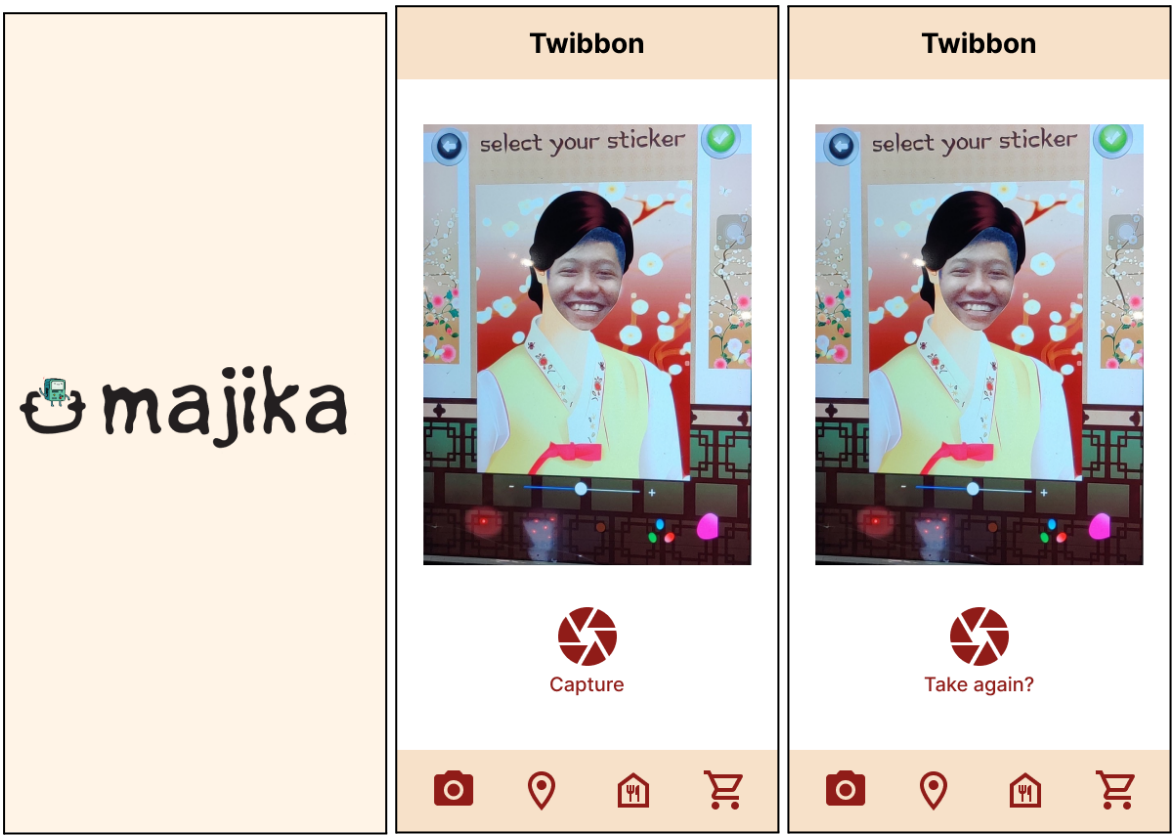
5. Kinerja dan *best practice* dari aplikasi yang dibangun.
6. Kerja sama dan manajemen kelompok (berlaku untuk yang berkelompok).

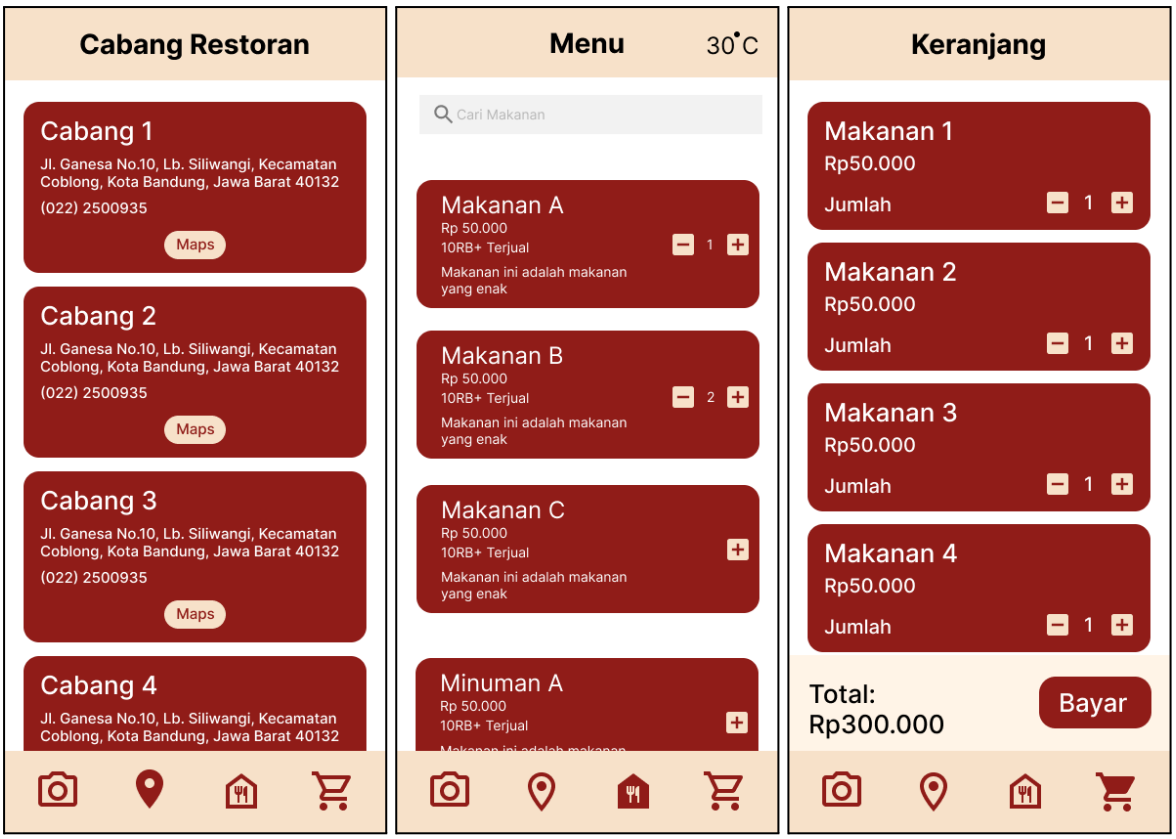
Link Belajar

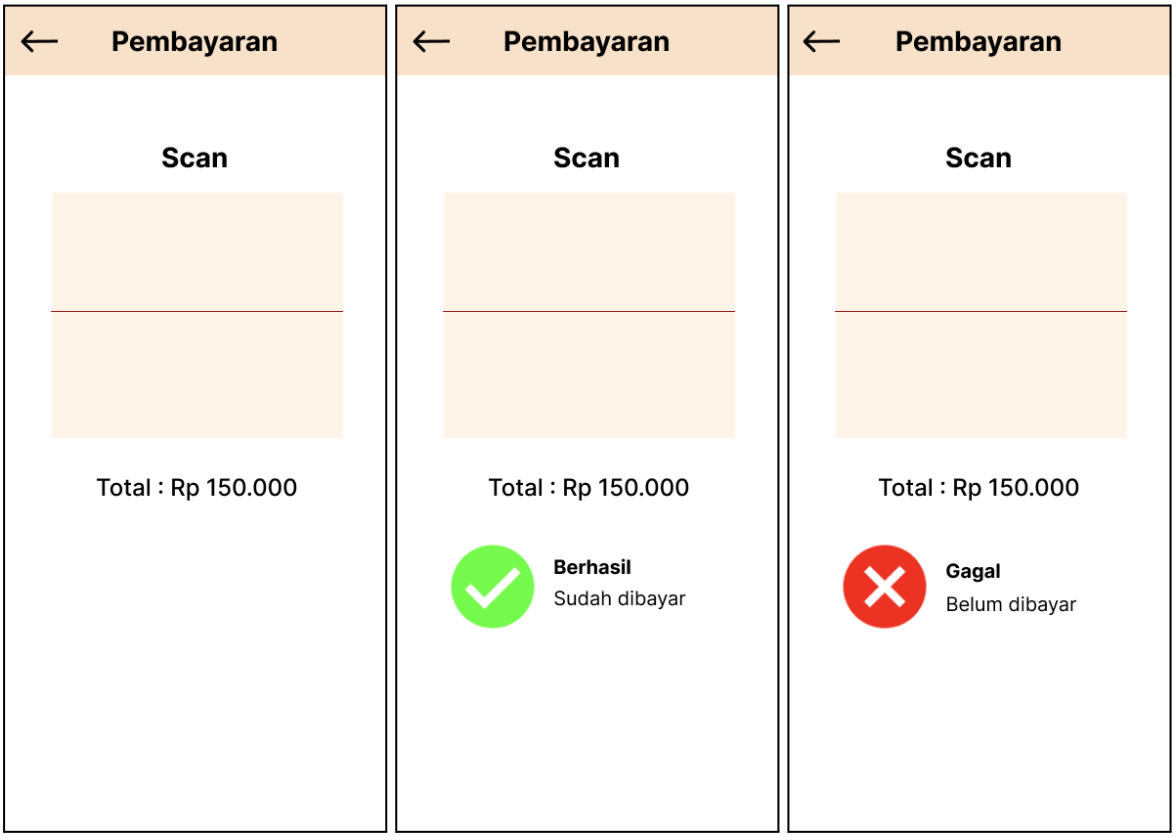
Berikut beberapa *link* yang dapat membantu Anda dalam mengerjakan tubes ini:

1. Belajar Android Kotlin Fundamental (lengkap)
2. Simple App with kotlin
3. FreeCodeCamp 3 jam tutorial
4. Simple GET request using Retrofit in Android | Engineering Education (EngEd) Program | Section
5. Cara Menggunakan Retrofit dalam Mengambil Data dari REST | Rehan Adi Satrya
6. Cara Membuat QR Code Scanner pada Android Studio
7. Google Maps Intents for Android
8. SQLite Database Using Anko Kotlin | by Hanif Abdullah | SANDEC | Medium
9. Implementasi RecyclerView + CardView di Kotlin | by Anto D.
10. BottomNavigationView with Fragments

Lampiran




Cabang Restoran
Cabang 1
Jl. Ganesa No.10, Lb. Siliwangi, Kecamatan Coblong, Kota Bandung, Jawa Barat 40132
(022) 2500935
Maps
Cabang 2
Jl. Ganesa No.10, Lb. Siliwangi, Kecamatan Coblong, Kota Bandung, Jawa Barat 40132
(022) 2500935
Maps
Cabang 3
Jl. Ganesa No.10, Lb. Siliwangi, Kecamatan Coblong, Kota Bandung, Jawa Barat 40132
(022) 2500935
Maps
Cabang 4
Jl. Ganesa No.10, Lb. Siliwangi, Kecamatan Coblong, Kota Bandung, Jawa Barat 40132
Menu
30°C
Cari Makanan
Makanan A
Rp 50.000
10RB+ Terjual
Makanan ini adalah makanan yang enak
1
Makanan B
Rp 50.000
10RB+ Terjual
Makanan ini adalah makanan yang enak
2
Makanan C
Rp 50.000
10RB+ Terjual
Makanan ini adalah makanan yang enak
Minuman A
Rp 50.000
10RB+ Terjual
Keranjang
Makanan 1
Rp50.000
Jumlah
1
Makanan 2
Rp50.000
Jumlah
1
Makanan 3
Rp50.000
Jumlah
1
Makanan 4
Rp50.000
Jumlah
1
Total:
Rp300.000
Bayar


Pembayaran
Scan
Total : Rp 150.000
Pembayaran
Scan
Total : Rp 150.000
Berhasil
Sudah dibayar
Pembayaran
Scan
Total : Rp 150.000
Gagal
Belum dibayar