

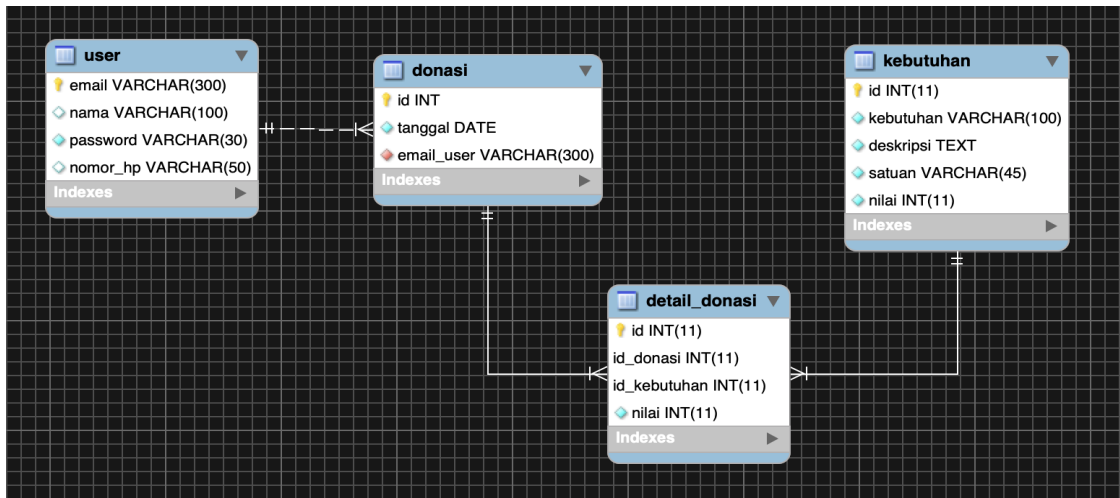
Dokumentasi Pengerjaan Project UAS “Donasiku”

Pemrograman Nirkabel Genap 2019/2020

Dikerjakan Oleh: Varianto (160717023)

Database

Pertama-tama, saya membuat ERD (*Entity Relationship Diagram*) dari tampilan tabel-tabel database yang terdapat pada soal project. Pembuatan ERD saya lakukan dengan software MySQL Workbench. Di sini saya akan membuat seluruh tabel beserta dengan relasi yang dimiliki, sesuai dengan ketentuan project. Hasil akhir pembuatan ERD pada Workbench adalah sebagai berikut.



Setelah selesai membuat ERD, saya kemudian menghubungkan Model ERD tersebut ke database MySQL yang saya beri nama “donasiku”. Setelah koneksi telah sukses terbentuk, selanjutnya saya mengintegrasikan Model ERD dengan cara memilih “Database > Synchronize Model”. Sebelumnya, saya melakukan percobaan pada server lokal di komputer saya, kemudian struktur data tersebut saya ekspor lalu import ke database “s160717023b” pada url server “<http://ubaya.prototype.net/phpmyadmin>”. Sehingga tampilan struktur database project adalah sebagai berikut.

←

127.0.0.1 » s160717023b

Structure

SQL

Search

Query

Export

























Import

Operations

Routines

Events

Triggers

	Table	Action					Rows	Type	Collation	Size	Overhead	
<input type="checkbox"/>	detail_donasi							~14	InnoDB	utf8_general_ci	48 KiB	-
<input type="checkbox"/>	donasi							~10	InnoDB	utf8_general_ci	32 KiB	-
<input type="checkbox"/>	kebutuhan							~4	InnoDB	utf8_general_ci	16 KiB	-
<input type="checkbox"/>	user							~2	InnoDB	utf8_general_ci	16 KiB	-
4 tables		Sum					30	InnoDB	utf8mb4_general_ci	112 KiB	0 B	

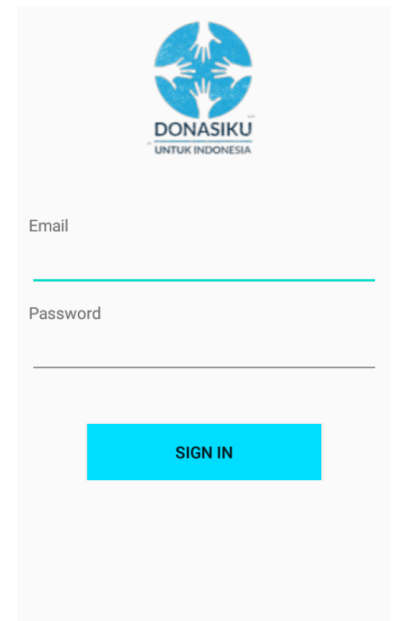
Desain

Pada tahapan ini, akan saya awali dengan membuat project baru di Android Studio yang saya beri nama “160717023_donasiku”. Saya memilih “Navigation Drawer Activity” sebagai Activity awal. Kemudian proses desain User Interface (UI) pun saya mulai dengan memposisikan komponen (margin, padding, width, height dan letak) pada setiap Activity disesuaikan agar tampilan UI seperti yang tertera pada project.

Login Page

Saya membuat activity baru, berjenis Empty Activity yang saya namai “LoginActivity”, kemudian pada “activity_login.xml” saya mulai mendesain. Sebagai basis layout, saya menggunakan Kontainer ScrollView dengan LinearLayout (Vertical). Kemudian, secara berurutan saya menambahkan komponen:

- imageView, sebagai tampilan logo aplikasi, yang sudah saya taruh di folder drawable.
- textView, dengan tulisan “Email”.
- editText, sebagai kolom untuk memasukkan email.
- textView, dengan tulisan “Password”.
- editText, sebagai kolom untuk memasukkan password.
- button, sebagai tombol untuk melakukan Sign In → Atribut layout_gravity = center. Dan, background bernilai “@android:color/holo_blue_bright”



Kemudian, tampilan Login Page akan dibuat Full Screen dan Hide Action Bar dengan cara menambahkan codingan berikut:

```
requestWindowFeature(Window.FEATURE_NO_TITLE)
window.addFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN)
supportActionBar?.hide()
```

Drawer Layout

Karena di awal pembuatan project saya memilih “Navigation Drawer Activity”, maka Drawer Layout ini secara default sudah memiliki Toolbar, Drawer, dan Floating Action Button. Drawer Layout ini bisa digunakan untuk Main, Donation & Tambah Kebutuhan Page. Konfigurasi yang dilakukan:

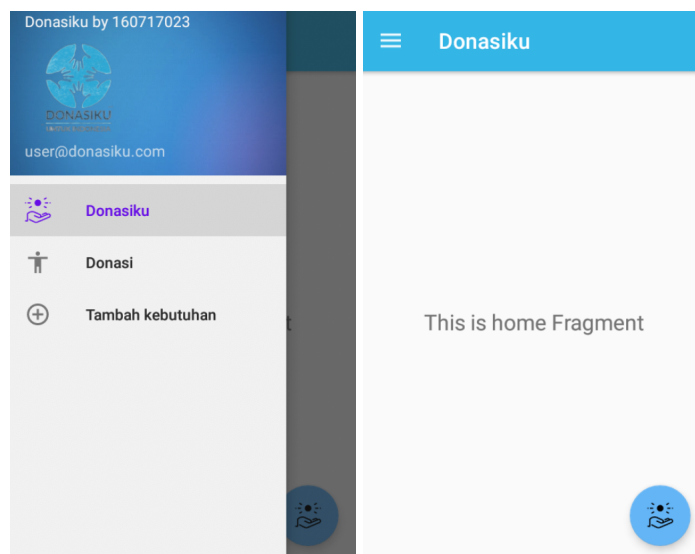
- Pada “folder res > layout > app_bar_main.xml”:
 - ❖ Hapus Floating Action Button, yang telah disediakan sebelumnya. Karena FAB tersebut akan diletakkan pada fragment-fragment yang membutuhkan.
 - ❖ Ganti tampilan Toolbar:
 - ❑ Atribut background diset dengan color “Holo Blue Light”.
 - ❑ Menghilangkan opsi setting (dengan simbol 3 titik vertikal) → Pada MainActivity, hilangkan atau diberi comment pada perintah “menuInflater.inflate(R.menu.main, menu)” pada function onCreateOptionsMenu.

- Mengganti nama halaman yang akan tampil pada toolbar dan navigation drawer, dengan mengubah nilai pada “folder res > values > strings.xml” di line 11-13 menjadi:

```
<string name="menu_main">Donasiku</string>
<string name="menu_donation">Donasi</string>
<string name="menu_add_needs">Tambah kebutuhan</string>
```

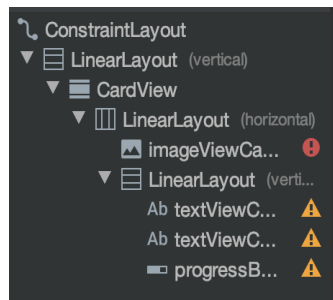
- Menyesuaikan nilai string dengan name “menu_main”, “menu_donation”, dan “menu_add_needs” agar dapat digunakan pada komponen lainnya.
 - ❖ Pada “folder res > navigation > mobile_navigation.xml”, saya mengubah id dari setiap nav yang telah terbentuk secara default. Sehingga nav-nav tersebut adalah: nav_main, nav_donation, nav_add_needs. Masing-masing nav memiliki label yang mengacu pada menu masing-masing, yang terletak pada strings.xml.
 - ❖ Pada “folder res > menu > activity_main_drawer.xml”, Masing-masing nav nilai title diisi dengan nilai menu masing-masing, yang terletak pada strings.xml, serta icon juga dipilih dengan icon yang telah saya tambah di folder drawable.
- Mengganti tampilan pada “folder res > menu > nav_header_main.xml” agar Navigation Drawer secara berurutan terdapat nama dan logo aplikasi, kemudian diikuti oleh email user yang sedang login. Background drawer pun disesuaikan agar berwarna biru.

Hasil dari Desain Drawer Layout yang telah saya lakukan adalah sebagai berikut.



Main Page

Pertama-tama, saya membuat card view sebagai struktur dasar untuk menampilkan kebutuhan medis pada Main Page. Caranya, saya membuat “listview_card_needs” pada “folder res > layout > layout resource file”. Hasil dari struktur dan tampilan card view untuk kebutuhan kesehatan adalah sebagai berikut.



Name

Description

Kebutuhan: N satuan

n/N

Kemudian, Pada “folder res > layout > fragment_main.xml”, ditambahkan sebuah listview dengan id “@android:id/list”. Dan Floating Action Button dengan id “ic_donate”, dengan penyesuaian sebagai berikut:

- Atribut srcCompact diset dengan icon “ic_donate”, yang telah saya tambahkan sendiri di folder drawable.
- Atribut backgroundTint diset dengan nilai “#64b5f6” agar berwarna biru muda (blue-lighten-2).

Selanjutnya saya akan mencoba untuk menampilkan kebutuhan dengan data dummy, yaitu:

- Pertama-tama, menyiapkan Class Kebutuhan yang akan menampung data member yang diperlukan.
- Membuat Custom Adapter dengan nama “CustomKebutuhanAdapter” yang nantinya meng-extend class ArrayAdapter dengan tipe data “Kebutuhan”. Selanjutnya, akan dilakukan override getView, inflate listview_layout, dan update gambar serta tulisan dari context. Lalu, Custom Adapter ini akan dipasang ke listView dengan id “list”.

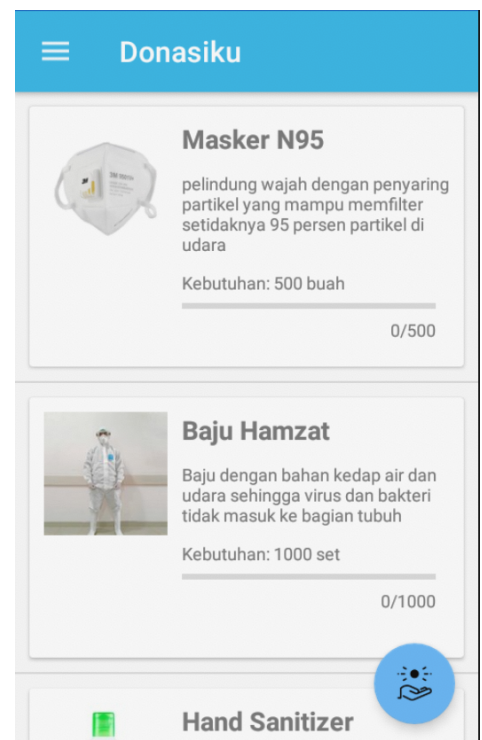
Pada “folder java > com.ubaya.s160717023 > ui > main > MainFragment” class MainFragment akan di-extend menjadi ListFragment sehingga menjadi demikian:

```
class MainFragment : ListFragment() { ... }
```

Kemudian, saya menyiapkan array yang berisi object bertipe “Kebutuhan” dengan data dummy. Kemudian menambahkan codingan berikut:

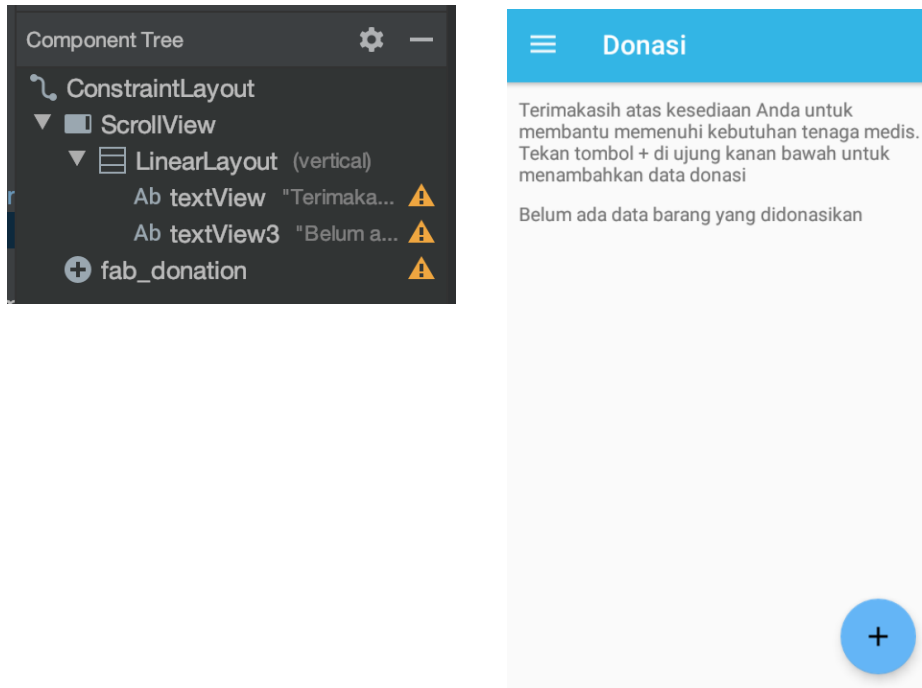
```
override fun onActivityCreated(savedInstanceState: Bundle?) {
    super.onActivityCreated(savedInstanceState)
    val arrayAdapter = CustomKebutuhanAdapter(
        this.getActivity()!!.getApplicationContext(),
        kebutuhan.toCollection(ArrayList()))
    list?.adapter = arrayAdapter
}
```

Hasil akhir tampilan Main Page dengan data dummy adalah sebagai berikut.



Donasi Page

Pada constraint layout, saya menambahkan scroll view agar bisa mengakomodir banyak barang donasi. Setelah itu, pada LinearLayout saya menambahkan 2 buah text view yang berisi ucapan terima kasih dan tulisan belum ada barang karena awal aplikasi digunakan list barang masih kosong. Yang terakhir, saya menambah Floating Action Button dengan id "fab_donation", yang telah diberikan nilai src: "@drawable/ic_add" dan backgroundTint: "#64b5f6". Berikut adalah gambar component tree dan tampilan aplikasi.



Kemudian saya akan membuat tampilan donasi yang telah ditambahkan dari halaman tambah kebutuhan donasi. Pertama-tama, saya membuat *resource file* dengan nama "listview_donations.xml" terlebih dahulu, seperti gambar di bawah ini.

Nama Kebutuhan	Jumlah Donasi
	Satuan

Selanjutnya, saya membuat class "Donasi" yang memiliki atribut berupa nama, jumlah, dan satuan. Lalu, saya akan menyiapkan sebuah adapter bernama "CustomDonasiAdapter", yang akan menampilkan data donasi ke layout "listview_donations.xml" sesuai dengan data ArrayList yang memiliki objek bertipe "Donasi". ListView akan ditampilkan secara dinamis dari DonationFragment, sehingga pada layout "fragment_donation.xml", saya menambahkan widget ListView bernama "listDonation" yang akan menampung list data donasi.

Pada file "DonationFragment.kt", saya akan menyiapkan beberapa data dummy donasi untuk sementara dengan cara berikut:

```
var donations = arrayOf(  
    Donasi("Masker N95", 500, "buah"),  
    Donasi("Baju Hamzat", 1000, "set"),  
    Donasi("Hand Sanitizer", 8000, "liter"),  
    Donasi("Nasal Swab", 750, "pcs")  
)
```

Dan selanjutnya, array donasi tersebut akan di-passingkan ke CustomDonasiAdapter, yang akan diset sebagai adapter pada listDonation, dengan cara berikut.

```
val arrayAdapter = CustomDonasiAdapter(this.getActivity()!!.getApplicationContext(),
                                     donations.toCollection(ArrayList()))
var listDonation:ListView = view.findViewById(R.id.listDonation)
listDonation.adapter = arrayAdapter
```

Hasil akhirnya adalah sebagai berikut.

Masker N95	500 buah
Baju Hamzat	1000 set
Hand Sanitizer	8000 liter
Nasal Swab	750 DCS

Tambah Kebutuhan Page (Detail Donasi Page)

Pertama-tama saya menambahkan LinearLayout, kemudian secara berurutan saya menambahkan komponen-komponen sebagai berikut:

- spinner, dengan id “spinnerNeeds” sebagai pilihan kebutuhan donasi.
- textView, dengan tulisan “Jumlah Kebutuhan”.
- editText, dengan id “editTextNumberOfNeeds” sebagai kolom untuk memasukkan jumlah kebutuhan.
- textView, dengan tulisan “Masukkan Jumlah Donasi”.
- editText, dengan id “editTextNumberOfDonations” sebagai kolom untuk memasukkan jumlah donasi.
- button, dengan id “buttonSubmit” sebagai tombol untuk menambahkan kebutuhan → Atribut layout_gravity = center. Dan, background bernilai “@android:color/holo_blue_bright”

Kedua, pada “folder res > values > strings.xml” saya menambahkan string array resource untuk digunakan pada spinner, sebagai berikut.

```
<string-array name="needs">
    <item>Masker N95</item>
```

```
<item>Pakaian Hamzat</item>
```

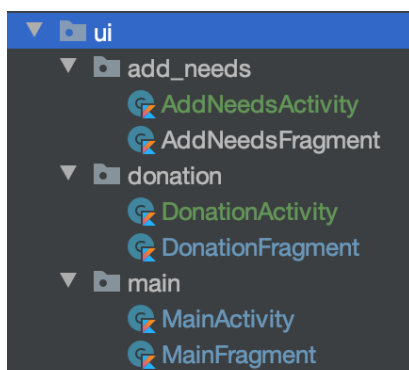
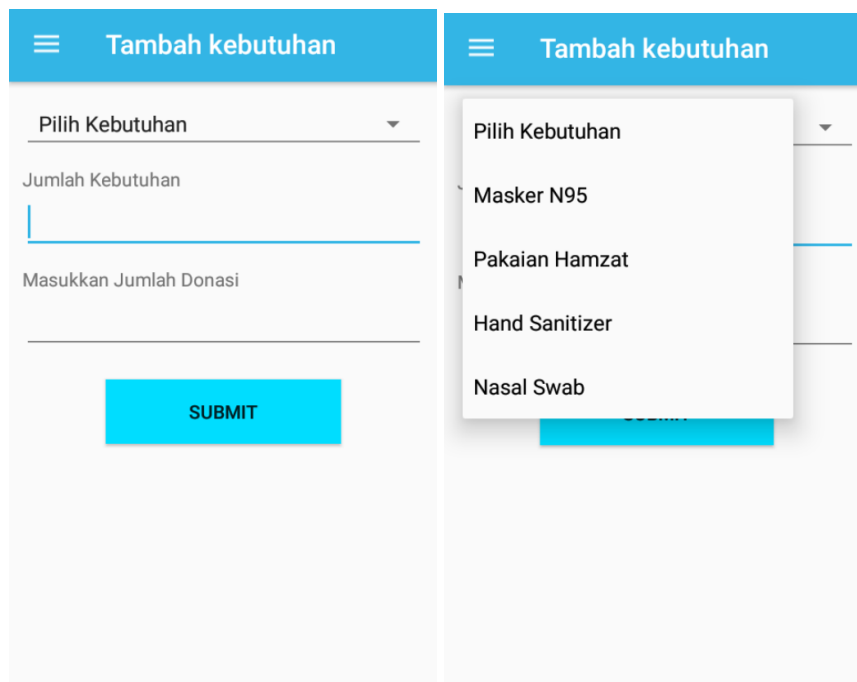
```
<item>Hand Sanitizer</item>
```

```
<item>Nasal Swab</item>
```

```
</string-array>
```

Ketiga, pada spinnerNeeds saya menset entries dengan nilai “@array/needs” dan menset nilai style dengan “@style/Widget.AppCompat.Spinner.Underlined”.

Keempat, pada “folder com.ubaya.s160717023_donasiku > ui > add_needs > AddNeedsFragment”, saya membuat sebuah array adapter berdasarkan array resource yang telah dibuat, lalu menambahkan adapter ke spinner. Sehingga hasil akhir tampilan adalah sebagai berikut.



Untuk melengkapi Activity yang dibutuhkan, saya membuat 2 (dua) Activity lainnya, yaitu DonationActivity dan AddNeedsActivity. Isi dari kedua Activity tersebut sama persis dengan MainActivity. Hanya saja, sebelum fungsi onCreate pada DonationActivity berakhir, terdapat perintah `navController.navigate(R.id.nav_donation)` dan sebelum fungsi onCreate pada AddNeedsActivity berakhir, terdapat perintah `navController.navigate(R.id.nav_add_needs)`. Kedua perintah tersebut berguna untuk men-load fragment yang sesuai ketika berpindah Activity. Kemudian, struktur folder untuk Activity dan

Fragment saya atur sebagai berikut:

- > Folder add_needs akan mengurus segala hal yang berkaitan dengan penambahan kebutuhan dan detail Activity donasi.
- > Folder donation akan mengurus segala hal yang berkaitan dengan list barang donasi.
- > Folder main akan mengurus segala hal yang berkaitan dengan data kebutuhan donasi.

Pada MainActivity, DonationActivity, dan AddNeedsActivity saya menambahkan potongan program berikut agar saat item menu pada drawer ditekan, maka aplikasi akan melewati Activity terlebih dahulu untuk setiap destinasi yang dituju.

```
navController.addOnDestinationChangeListener { _, destination, _ ->
    if(destination.id == R.id.nav_main) {
        // startActivity<MainActivity>()
    }
    else if(destination.id == R.id.nav_donation) {
        startActivity<DonationActivity>()
    }
    else if(destination.id == R.id.nav_add_needs) {
        startActivity<AddNeedsActivity>()
    }
}
```


Implementasi

Pertama-tama saya menambahkan library volley agar dapat berkomunikasi dengan database melalui protokol HTTP pada module gradle dan internet permission access pada AndroidManifest.xml

Login Page

Saya mengisikan beberapa data user sebelumnya pada tabel user, seperti berikut.

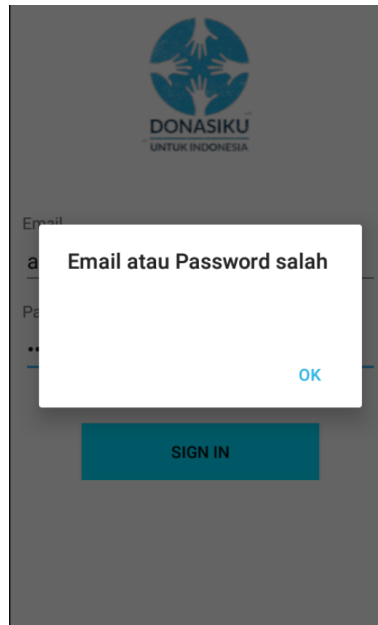
	email	nama	password	nomor_hp
<input type="checkbox"/> Edit Copy Delete	admin@donasiku.com	Admin	donasiku	082272397456
<input type="checkbox"/> Edit Copy Delete	budi@user.com	Budi Agung	rahasia	082272397456

Lalu, saya membuat file sebuah file php bernama “login.php” yang akan mengurus proses login dari user. File tersebut akan mengecek apakah data user yang diisikan pada form telah tercatat pada tabel user. Output dari file ini adalah json dengan properti “status” dan “message”. “Status” akan berisi “success” atau “error”. Sedangkan, “message” akan berisi pesan error dari server.

Pada LoginActivity, terdapat pengecekan apakah editText email dan password telah diisi. Jika telah diisi oleh user, maka program akan mengirimkan email dan password tersebut, seperti berikut.

```
override fun getParams(): MutableMap<String, String> {  
    val params = HashMap<String, String>()  
    params.put("email", email)  
    params.put("password", password)  
  
    return params  
}
```

Pada response listener, hasil dari file “login.php” akan diperoleh untuk menentukan apakah akun user telah terdaftar. Jika email atau password salah, maka akan muncul dialog box dengan tulisan “Email atau Password salah”. Jika akun telah terdaftar, maka user akan dipindahkan ke MainActivity dengan cara: `startActivity<MainActivity>(...)`. Dan, email user yang log in tersebut akan dikirimkan melalui intent.



Main Page

Pada MainActivity (di folder java > com.ubaya.s160717023 > ui > main), saya men-disable back button agar user tidak dapat kembali ke halaman login dengan cara berikut:

```
override fun onBackPressed() {
    // super.onBackPressed()
}
```

Lalu, untuk menyimpan email user yang telah login pada aplikasi agar dapat diakses di halaman lainnya, maka saya menyimpan data secara global, dengan cara pertama-tama membuat Class bernama “MyApplication”, yang isinya adalah: `var userEmail: String? = null`. Pada LoginActivity ketika user telah berhasil Log in, maka email akan disimpan dengan cara: `(this.application as MyApplication).userEmail = email`. Pada AndroidManifest.xml, saya menambahkan `android:name="MyApplication"` pada tag application, agar Class dapat diakses pada Activity-Activity lainnya.

Saya menambahkan beberapa data pada tabel kebutuhan di database external untuk keperluan program pada halaman-halaman selanjutnya.

<div><div><div>←</div><div>T</div><div>→</div></div><div></div></div>				id	kebutuhan	deskripsi	satuan	nilai
<div><div><div></div></div><div><div>Edit</div><div>Copy</div><div>Delete</div></div></div>	1	Masker N95	pelindung wajah dengan penyaring partikel yang mam...	buah	500			
<div><div><div></div></div><div><div>Edit</div><div>Copy</div><div>Delete</div></div></div>	2	Baju Hamzat	Baju dengan bahan kedap air dan udara sehingga vir...	set	1000			
<div><div><div></div></div><div><div>Edit</div><div>Copy</div><div>Delete</div></div></div>	3	Hand Sanitizer	cairan atau gel yang umumnya digunakan untuk mengu...	liter	800			
<div><div><div></div></div><div><div>Edit</div><div>Copy</div><div>Delete</div></div></div>	4	Nasal Swab	alat untuk mengumpulkan sampel uji klinis sekresi ...	pcs	750			

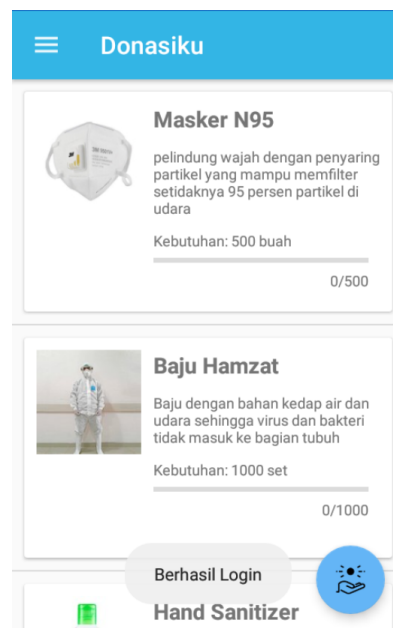
Selanjutnya akan dilakukan pengambilan data pada database dengan memanfaatkan library “volley”. Library tersebut akan menjalankan file “get_all_needs.php”. Isi file php ini adalah mengambil seluruh kebutuhan donasi dari tabel “kebutuhan”. Seluruh data kebutuhan akan di-loop satu per satu, dan setiap data kebutuhan tersebut akan didapatkan jumlah donasi dari user atas kebutuhan tersebut dengan perintah: `“SELECT IFNULL(SUM(nilai), 0) as total_donasi FROM detail_donasi WHERE id_kebutuhan = ?”`

Jika response berhasil, maka setiap elemen dari keseluruhan hasil tersebut akan dijadikan object JSON, yang kemudian akan ditampung dalam `ArrayList<Kebutuhan>`. `ArrayList` tersebut akan di-serialize agar dapat dikirimkan ke `MainFragment` melalui bundle. Namun, sebelumnya class `Kebutuhan` saya extends menjadi `Serializable`, seperti berikut: `class Kebutuhan(...) : Serializable { ... }`

Pada `CustomKebutuhanAdapter`, bagian menset resource logo kebutuhan, diganti agar dapat menampilkan resource secara dinamis sesuai id masing-masing kebutuhan dari server `103.252.100.37/s160717023/` (lokasi server yang telah disediakan) pada android studio dengan menggunakan plugin Picasso, seperti berikut:

```
Picasso.get().load("http://ubaya.prototipe.net/s160717023/donasiku/kebutuhan_images/${kebutuhan[position].id}.jpeg").into(logo).
```

Pada `MainFragment`, di `onActivityCreated`, `ArrayList` kebutuhans didapatkan dari Activity, dengan cara berikut: `var kebutuhans = arguments?.getSerializable("needs") as? ArrayList<Kebutuhan>`. Variabel `kebutuhans` selanjutnya akan dipassingkan ke `CustomKebutuhanAdapter`. Sehingga, sekarang tampilan kebutuhan dapat dihasilkan secara dinamis sesuai data yang ada pada tabel "kebutuhan" di database. Sebagai berikut.



Donasi Page

Pertukaran data donasi dari Donasi Page ke Tambah Kebutuhan Page, dan sebaliknya saya lakukan dengan Intent yang akan mengirimkan `ArrayList` bertipe class `Donasi` yang telah saya buat. Sehingga saya meng-extend class `Donasi` agar menjadi `Serializable`. Pada `DonationActivity`, saya akan mendapatkan data Intent dengan cara berikut:

```
val donasis = this.intent?.getSerializableExtra("donasis") as? ArrayList<Donasi>
```

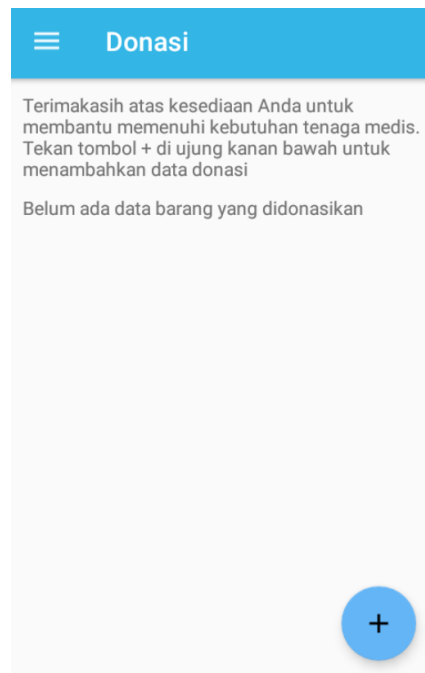
Selanjutnya, data `ArrayList` `Donasi` akan dikirimkan ke `DonationFragment` dengan cara berikut:

```
var bundle = Bundle()
bundle.putSerializable("donasis", donasis?: ArrayList<Donasi>() as Serializable)
navController.navigate(R.id.nav_donation, bundle)
```

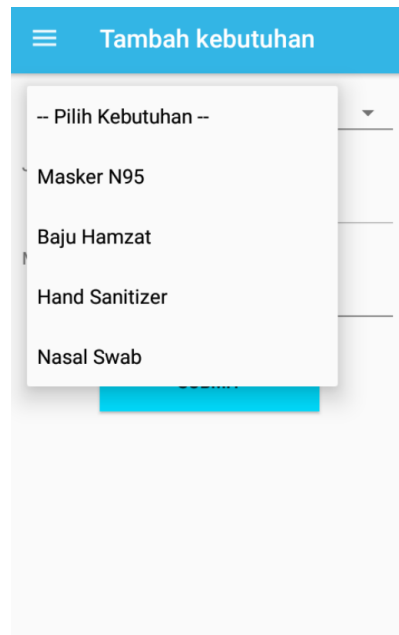
Pada DonationFragment, data ArrayList<Donasi> tersebut, baik yang memiliki isi atau tidak sama sekali akan didapatkan dengan cara berikut:

```
var donasis = arguments?.getSerializable("donasis") as? ArrayList<Donasi>
```

Akan dilakukan pengecekan pada jumlah ArrayList Donasi untuk menentukan apakah halaman baru dibuka sehingga data donasi masih kosong atau sebelumnya *user* telah memilih data dari halaman Tambah Kebutuhan. Jika tidak ada data, maka listview donasi tidak akan ditampilkan, sehingga tampilannya adalah sebagai berikut.

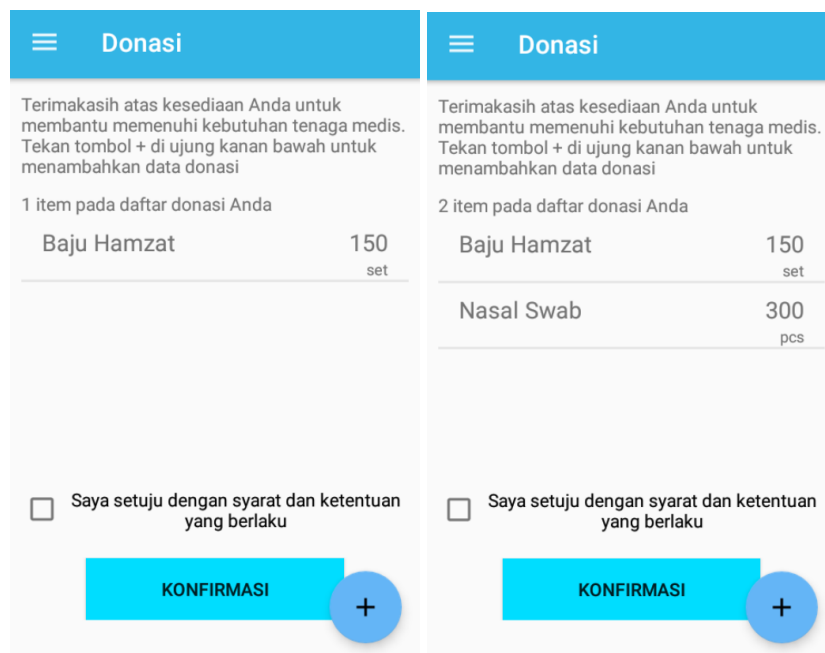


Jika *Floating Action Button* (FAB) diklik, maka halaman akan berpindah ke Tambah Kebutuhan Page dan akan dikirimkan juga data donasi (ArrayList<Donasi>) sebelumnya. AddNeedsActivity akan menerima data intent dengan cara yang sama seperti pada DonationActivity dan data donasi juga akan dikirimkan ke AddNeedsFragment. Untuk memperoleh semua nama dari data kebutuhan yang tersimpan pada *database external*, maka akan dilakukan *request* data ke file "get_all_needs.php". Hasil data dari *server* akan berupa *object* JSON yang merepresentasikan kebutuhan barang. Selanjutnya, data-data *object* JSON tersebut akan di-*looping* satu per satu ke sebuah array bertipe Kebutuhan yang akan di-passingkan ke adapter dari spinner. Sehingga, tampilan spinner dengan data dari *database external* adalah sebagai berikut.



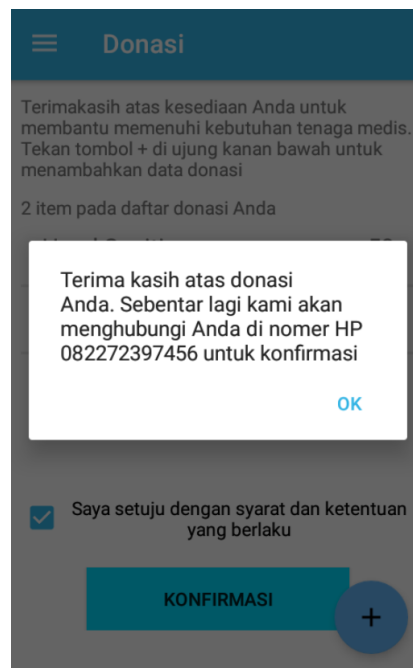
Ketika salah satu data kebutuhan pada spinner di-klik, maka informasi id, nama, dan satuan donasi akan didapatkan melalui dari `ArrayList<Kebutuhan>` sesuai dengan index kebutuhan pada spinner yang dipilih. Saat tombol Submit ditekan, program terlebih dahulu akan melakukan pengecekan untuk menentukan apakah data kebutuhan telah dipilih, jumlah kebutuhan berupa angka saja yang telah diisi, dan bernilai lebih dari 0. Jika data yang diinputkan telah sesuai, maka sistem akan mengirim data donasi yang telah diisikan ke `DonationActivity` menggunakan `Intent`.

Karena sekarang data `Intent` pada `DonationActivity` telah berisi nilai dari `AddNeedsFragment`, maka sistem akan menampilkan data donasi yang telah dipilih dengan contoh sebagai berikut jika data donasi yang diinputkan pertama adalah “150 set Baju Hamzat”, lalu “300 pcs Nasal Swab”.



Pada DonationFragment, email pengguna yang telah tersimpan pada Class MyApplication akan didapatkan dengan cara: `var userEmail: String? = (this.getActivity()!!.application as MyApplication).userEmail ? : ""`. Ketika tombol konfirmasi ditekan, saya melakukan pengecekan terhadap combo box apakah sudah ditekan oleh *user*. Jika *user* telah mencentang combo box syarat & ketentuan, selanjutnya saya akan mendapatkan tanggal saat ini, dengan menambahkan codingan berikut: `val sdf = SimpleDateFormat("yyyy-MM-dd"); val currentDate = sdf.format(Date())`. Untuk mengirimkan data donasi, saya menggunakan *dependency* bernama "Gson". ArrayList donasi akan dikonversi menggunakan Gson dengan cara: `val donasisJson = Gson().toJson(donasis)`.

Kemudian, saya akan melakukan penambahan data donasi ke *file* "add_donations.php", dengan parameter yang di-passingkan adalah email, tanggal, donasi (ArrayList bertipe Donation). Isi dari *file* add_donations.php. Pertama-tama, segala parameter akan disimpan pada variabel-variabel. Lalu, tanggal dan email user akan ditambahkan ke tabel "donasi". Setiap data donasi akan saya loop untuk ditambahkan ke tabel "detail_donasi" dengan menggunakan ID yang didapatkan dari penambahan data donasi *user* tersebut sebelumnya. Yang terakhir, saya akan mendapatkan data nomor HP *user* dari tabel "user". Dan, data nomor HP tersebut yang akan di-output-kan. Hasil perintah penambahan data donasi yang berhasil adalah sebagai berikut.



Setelah tombol OK ditekan, ArrayList donasi akan dikosongkan dan tampilan list donasi akan dikosongkan kembali.