Rancang Bangun Aplikasi "W-Mass (*Weight Monitor Assistant*)" Berbasis Android Studio Dengan Bahasa Native Java

I Gede Susrama[1], Ariyono Setiawan[2], Moch. Kholis[3]

[1,2,3)]Politeknik Penerbangan Surabaya
Jl. Jemur Andayani I, No. 73 Surabaya 60236
Email : rmaryo4u@gmail.com

ABSTRAK

Memiliki tubuh ideal merupakan impian semua orang. Namun, keinginan untuk memiliki tubuh ideal punya konsekuensi beragam, termasuk dieat dan olahraga teratur, sehingga untuk mengetehaui penurunan berat badan akan dicatat secara konvensional yaitu dicatat pada secarik kertas, sehingga kemungkinan catatan tersebut akan bisa hilang. Dilain pihak perkembangan teknologi mobile (smartphone) sangat mempengaruhi kehidupan manusia saat ini, sampai di kehidupan sehari-hari, sehingga akan menjadi sangat terikat dengan smartphone, sehingga muncul banyak aplikasi yang memudahkan pada kehidupan manusia. Untuk itulah, pada penelitian ini dibuat suatu aplikasi yang digunakan untuk menyimpan pencatatan dan monitoring berat badan secara berkala pada user. Dan dengan adanya aplikasi W-MAss (weight Monitor Assistant) berbasis android ini diharapkan dapat menjadi motivasi bagi pengguna untuk lebih hidup sehat dengan menjaga berat badan. Serta Aplikasi ini dibuat dengan Android Studio dan Java sebagai bahasa pemrogramannya..

Kata Kunci: Rancang Bangun, Aplikasi, W-MAss, Android, Native Java

PENDAHULUAN

Obesitas atau kelebihan berat badan diketahui merupakan faktor dari berbagai masalah kesehatan seperti gangguan pernapasan, infertilitas bahkan kanker. Efek buruk dari kegemukan mempengaruhi kesehatan seseorang mulai dari ujung kepala hingga ujung kaki [1]. Berikut dampak buruk obesitas terhadap kesehatan seseorang, seperti dilansir *emaxhealth***,** Selasa antara lain: Kepikunan, Depresi, Maslah kesehatan mata, Masalah kesehatan gigi dan mulut, Infeksi telinga kronis, Sleep apnea, Berbagai jenis kanke, dan lain-lainya**.**

Sebanarnya tidak ada tanda gejala pasti dari obesitas [2]. Orang yang obesitas cenderung terlihat lebih gemuk dan besar. Namun perlu dipahami bahwa orang yang gemuk belum tentu mengalami obesitas, sementara mereka yang obesitas sudah pasti gemuk [2].

Untuk menentukan apakah seseorang termasuk dalam obesitas atau tidak, terdapat beberapa cara menentukannya yakni dengan mengukur:

1. *Body Mass Index* (BMI)
2. Lingkar pinggang
3. Rasio lingkar pinggang dan panggul (RLPP)
4. Tebal lipatan kulit menggunakan alat ukur yang bernama *skinfold*
5. Kadar lemak tubuh menggunakan sebuah alat *bioelectrical impedance analysis* (BIA)

Dari berbagai cara tersebut, mengukur BMI adalah cara yang paling sering digunakan karena cukup mudah untuk dilakukan. Perhitungan BMI ini menggunakan berat badan dan tinggi badan. Rumus dari perhitungan BMI adalah:

$$\text{BMI} = \text{berat badan (kg)} / (\text{tinggi (m)} \times \text{tinggi (m)})$$

Orang-orang dengan BMI lebih besar dari 25 dapat dikategorikan sebagai *overweight*, pada 30 atau lebih termasuk ke dalam obesitas, dan pada 40 ke atas merupakan tingkat obesitas yang serius. Dengan demikian diperlukan suatu aplikasi untuk menghitung dan memonitor perkembangan berat badan secara berkala.

Penelitian terdahulu yang terkait dengan monitoring berat badan ini, salah satunya adalah Afdali, dkk. (2017), yaitu tentang alat pengukur tinggi badan dan penimbang berat badan yang sekaligus memberikan informasi berat badan ideal yang terukur [3]. Alat ukur ini menggunakan *Arduino Uno* sebagai otaknya, sensor ultrasonik untuk mengukur tinggi badan, dan sensor *strain gauge* untuk mengukur berat badan, akan tetapi pada penelitian ini tidak dapat mencatat dan menyimpan data hasil penimbangan berat badan.

Penelitian lainnya yang dilakukan oleh Fadlur, dkk. (2017), bertujuan untuk menghitung berat badan dengan menggunakan pengolahan citra. Pendekatan matematis didasarkan pada perhitungan *Body Surface Area* dan volume *elips* tubuh manusia. Pengolahan citra yang berupa foto digital diolah untuk menghasilkan informasi mengenai berat badan seseorang pada foto tersebut. Selanjutnya dilakukan investigasi kemungkinan, analisis perhitungan, dan peningkatan akurasi pada sistem.

Lain halnya pada penelitian yang dilakukan oleh Vicky, dkk. (2013) yang menitik beratkan pada pembangunan aplikasi pencatatan diet berbasis perangkat bergerak. Rancangan desain penelitian dibuat dengan menggunakan UML, sedangkan aplikasi menggunakan bahasa pemrograman J2ME, serta file RMS untuk menyimpan datanya [5]. Penggunaan aplikasi pencatatan, catatan aktivitas diet yang dilakukan masih bersifat off-line (hanya pada alat pencatat mandiri). Ariyono, dkk (2018), melakukan penelitian tentang pencatat kebugaran tubuh untuk mempermudah pengguna untuk mengetahui nilai dari indeks massa tubuh, tingkat metabolisme, kadar air, berat badan ideal, dan kategori berat badan berdasarkan indeks massa tubuh [6], maka dibuatlah perhitungan untuk mendapatkan sebuah indeks massa tubuh yang diiniginkan dengan menggunakan perhitungan Body Mass Index (BMI) dan Basal Metabolisme Rate (BMR). Dengan adanya perhitungan tersebut dapat dihasilkan tingkat kebugaran tubuh berdasarkan dari nilai BMI, apakah masuk kedalam kategori kurus, normal, gemuk, ataupun obesitas. Dari hasil peneliitian tersebut dapat dianalisa dan dikembangkan menjadi beberapa fungsionalitas meliputi aplikasi mampu menghitung nilai dari indeks massa tubuh, berat badan ideal, angka metabolisme, dan kadar air yang terdapat pada tubuh. Kemudian pada menu histori, user menaampilkan sebuah list view yang berfungsi untuk melihat berat badan sesuai dengan tanggal yang telah diinputkan, akan tetapi penelitian ini masih bersifat off-line.

Dari beberapa penelitian terdahulu, maka pada penelitian ini, membuat suatu aplikasi sistem pencatat dan monitoring berat badan yang dibuat dengan Android Studio dan Java sebagai bahasa pemrogramannya, secara online. Sehingga diharapkan sistem ini bisa dipakai oleh berbagai kalangan.

ANALISIS SISTEM

Dalam proses ini dilakukan analisis terhadap kebutuhan untuk memecahkan masalah yang ada, juga membuat sebuah *Proof of Concept* (PoC) yang bertujuan sebagai sebuah pembuktian bahwa hasil analisis yang telah dirancang merupakan suatu hal yang logis dan dapat dibangun dalam ruang lingkup pengembangan yang telah dilakukan. Selain itu PoC juga digunakan untuk meminimalisir terjadinya perubahan besar atau menyeluruh saat proses pengembangan telah berjalan lama.

Melalui *requirement* yang telah diberikan oleh pihak *client*, maka dapat dirumuskan kebutuhan awal system yang akan dibuat sebagai berikut :

1. Aplikasi ditujukan untuk *platform* Android.
2. *Development and Working Environment*, selama masa pengembangan.
3. Spesifikasi kebutuhan minimum dari *device* yang dituju.

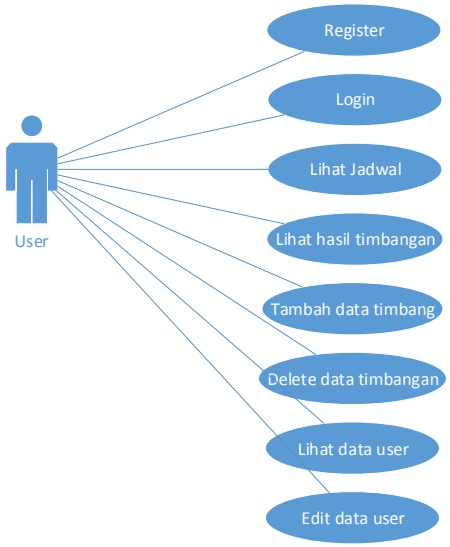
Rancangan awal cara kerja sistem yang akan dibangun, terdiri dari *Use Case Diagram*, *Activity Diagram* dan *Sequence Diagram* secara garis besar.

Development Environment

Berdasarkan kebutuhan yang telah dirumuskan beserta analisis terhadap perbandingan pada *framework-framework* pengembangan aplikasi Android yang ada, Android Studio dengan bahasa *native Java* dipilih sebagai *tools* dan bahasa utama pada pengembangan aplikasi ini, dan berdasarkan versi Android Studio dan ketergantungan antara *library third-party* yang digunakan, maka spesifikasi ditentukan: *Android* dengan minimum API 19 atau *Kitkat* 4.4 beserta target dengan API rilis terakhir, dalam hal ini *Oreo* 8.0 atau API 26, Windows untuk pengembangan dengan RAM minimal 4GB

Pembuatan *Use Case* Diagram

Aktor yang terlibat pada *use case* diagram aplikasi android ini hanya satu yaitu *user*. *Use case* pada aplikasi timbangan berbasis android ini yaitu register, login, lihat jadwal, lihat hasil timbangan, tambah data timbang, delete data timbang, lihat data user, dan edit data user.



Gambar 1. *Use Case* Diagram W-MAss (*Weight Monitor Assistant*)

Pada *use case register*, bukan hanya calon *user* yang dapat mendaftar tetapi calon admin NC (*Nutrition Club*) pun bisa, dan calon admin NC hanya bisa mendaftar saja. Pada *use case* lihat hasil timbang, data yang ditampilkan berupa grafik dan juga list data. *User* dapat melihat data perminggu, perbulan, dan pertahun. Berikut ini adalah penjelasan setiap *use case* Tabel 1. Untuk proses register, Tabel 2. Untuk proses login, Tabel 3. Proses Lihat Jadwal, Tabel 4. Proses Lihat Hasil Timbangan, Tabel 5. Proses Tambah Data Timbangan, Tabel 6. Proses Delete Data Timbangan, Tabel 7. Proses Lihat Data User dan Tabel 8. Proses Edit Data User

Tabel 1. Proses Register

Nama Use Case	Register
Deskripsi	*Use case* ini digunakan untuk menambah data *User* baru. User baru dapat berarti calon ser dan calon admin NC
Aktor	*User*
Kondisi Awal	Aktor belum pernah terdaftar sebelumnya dan mengakses aplikasi timbangan berbasis android
Kondisi Akhir	Terdaftar sebagai *User*
Alur	1. Aktor membuka aplikasi 2. Aktor memilih register 3. Aktor memilih mendaftar sebagai apa 4. Aktor mengisi form 5. Data tersimpan di basis data

Tabel 2. Proses Login

Nama Use Case	*Login*
Deskripsi	*Use case* ini digunakan untuk masuk kedalam aplikasi agar dapat mengakses aplikasi
Aktor	*User*
Kondisi Awal	Aktor belum masuk kedalam aplikasi
Kondisi	Aktor dapat mengakses aplikasi

Akhir	
Alur	1. Aktor membuka aplikasi 2. Aktor mengisi *Username* dan *password* pada form login 3. Aktor mengakses aplikasi

Tabel 3. Proses Lihat Jadwal

Nama Use Case	Lihat jadwal
Deskripsi	*Use case* ini digunakan untuk melihat jadwal user untuk men-imbang
Aktor	*User*
Kondisi Awal	Aktor belum login
Kondisi Akhir	Aktor melihat jadwal
Alur	1. Aktor membuka aplikasi 2. Aktor mengisi *form login* 3. Aktor memilih menu jadwal 4. Aktor melihat jadwal

Tabel 4. Proses Lihat Hasil Timbangan

Nama Use Case	Melihat hasil timbangan
Deskripsi	*Use case* ini digunakan untuk melihat hasil timbangan
Aktor	*User*
Kondisi Awal	Aktor belum login
Kondisi Akhir	Aktor melihat hasil timbangan
Alur	1. Aktor membuka aplikasi 2. Aktor mengisi form login 3. Aktor memilih menu grafik 4. Aktor melihat hasil timbangan

Tabel 5. Proses Tambah Data Timbangan

Nama Use Case	Tambah data timbangan

Deskripsi	Use case ini digunakan untuk menambah data timbangan
Aktor	User
Kondisi Awal	Aktor belum login
Kondisi Akhir	Aktor menambah data timbangan
Alur	1. Aktor membuka aplikasi 2. Aktor mengisi form login 3. Aktor memilih symbol “+” 4. Aktor mengisi form tambah data 5. Data akan tersimpan di data-base

Tabel 6. Proses Delete Data Timbangan

Nama Use Case	*Delete* data timbangan
Deskripsi	*Use case* ini digunakan untuk menghapus hasil timbangan
Aktor	*User*
Kondisi Awal	Aktor belum login
Kondisi Akhir	Aktor Menghapus data timbangan
Alur	1. Aktor membuka aplikasi 2. Aktor mengisi form login 3. Aktor memilih menu grafik 4. Aktor memilih data yang akan dihapus 5. Database akan memperbarui data

Tabel 7. Proses Lihat Data User

Nama Use Case	Lihat data user
Deskripsi	*Use case* ini digunakan untuk melihat data *user*
Aktor	*User*
Kondisi Awal	Aktor belum *login*

Kondisi Akhir	Aktor Menghapus data timbangan
Alur	1. Aktor membuka aplikasi 2. Aktor mengisi *form login* 3. Aktor memilih menu profile

Tabel 8. Proses Edit Data User

Nama Use Case	Edit data user
Deskripsi	Use case ini digunakan untuk mengedit data *user*
Aktor	*User*
Kondisi Awal	Aktor belum login
Kondisi Akhir	Aktor Menghapus data timbangan
Alur	1. Aktor membuka aplikasi 2. Aktor mengisi form login 3. Aktor memilih menu profile 4. Aktor mengganti data yang ingin diganti pada form data user 5. Data pada database akan diperbarui

PEMBUATAN PROGRAM

Aplikasi *Tracking Berat Badan* berbasis Android ini dibuat menggunakan Android Studio dengan Bahasa pemrograman Java. Bahasa pemrograman Java bersifat *Object Oriented*, sehingga dalam pembuatan aplikasi terdapat beberapa *class* yang memiliki jenis dan fungsi masing masing. Selain itu digunakan juga beberapa *library* dari pihak ketiga untuk mempermudah proses pengerjaan aplikasi. Dalam pembuatan aplikasi dapat dibagi menjadi dua bagian besar pengerjaan, yaitu pengerjaan untuk logika aplikasi yakni berkaitan dengan kode program dan *class-class* yang dibuat, dan perancangan desain untuk tampilan aplikasi yaitu berupa *blueprint* desain dan kode XML (*eXtensible Markup Language*).

Penggunaan *Library* Pihak ke Tiga

Dalam proses pembuatan aplikasi, penulis menggunakan beberapa library untuk mempermudah dan mempercepat proses pembuatan aplikasi. Pada versi android studio terbaru penggunaan *library* sudah tidaklagi memasukkan file ".jar" dan mengatur dependensi file *library* yang akan digunakan, kini android studio sudah menggunakan package manager yang bernama *gradle.* Penggunaan *gradle* hanya membutuhkan penulisan *source* dimana *library* itu disimpan oleh pemiliknya, dan pengguna hanya akan memasukkan link alamat *library* itu.

Gradle akan secara otomatis menyusun sesuai dengan dependensi yang seharusnya. Berikut ia contoh cara memasukkan lsebuah library kedalam *fle gradle*;*"implementation 'com.github.PhilJay:MPAndroidChart:v3.0.3'"*

Perancangan dan Pembuatan Tampilan Aplikasi

Setelah melakuakan proses analisis system maka hal berikutnya yang dikerjakan adalah perancangan dan pembuatan desain tampilan aplikasi. Proses ini akan menghasilkan dua file, yaitu blue print rancangan dan XML dari tampilan yang sudah dibuat. Ada beberapa halaman yang harus dibuat dalam perancangan dan pembuatan aplikasi ini, antara lain;

1. *Splash Screen*

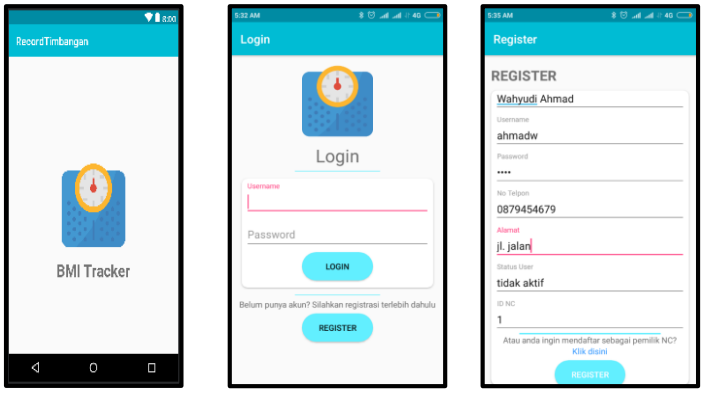
Halaman ini adalah halaman pembuka. Halaman paling awal yang akan dibuka ketika user pertama kali mengetuk/membuka aplikasi. Halaman splash screen ini memberikan jeda satu setengah detik untuk menyiapkan aplikasi saat aplikasi akan dibuka, selain berfungsi untuk memberikan jeda persiapan aplikasi untuk dibuka, *splash screen* ini juga memiliki fungsi sebagai pemanis pengalaman dari user saat menggunakan aplikasi. Pada halaman *splash screen* ini hanya berisi logo dari aplikasi dan warna latar aplikasi, seperti pada Gambar 3(a)

2. Halaman *Login*

Halaman login akan terbuka jika user belum pernah *login* sebelumnya, atau *user* telah melakukan proses *logout* pada sesi terakhir user menggunakan aplikasi ini. Jika *user* sudah pernah masuk dan tidak pernah melakukan proses logout maka user akan langsung dialihkan setelah halaman *splash screen* ke halaman timbangan. Di halaman login ada dua *text field* yang bisa diisi oleh *username* dan *password user*, kemudian ada dua button yaitu *button* untuk melanjutkan proses *login* dan button untuk registrasi (Gambar 3.b).

3. Halaman Register

Halaman ini berisikan form yang harus diisi oleh user baru ketika ingin membuat akun. Ada dua jenis halaman register, yaitu halaman register untuk user biasa, dan halaman register untuk pemilik NC. Setiap *field* yang harus diisikan data oleh user maupun pemilik NC sudah memiliki metedo input tersendiri, misalkan untuk masukkan data berupa angka maka tampilan keyboard yang muncul hanya angka, tanpa adanya karakter huruf, begitu juga sebaliknya. Gambar 2. (c) adalah tampilan halaman register.



(a) (b) (c)

Gambar 2. (a) Tampilan *Splash Screen*, (b) Tampilan Halaman Login, (c) Tampilan Halaman Register

4. Halaman Timbangan

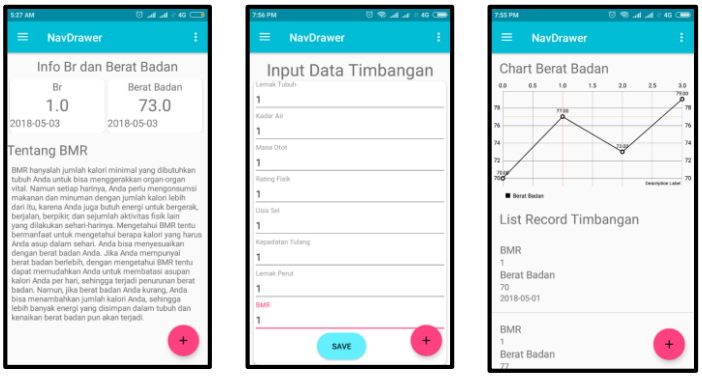
Halaman ini adalah halaman yang dijadikan halaman base pada aplikasi (halaman utama) di halaman ini ditampilkkan berat badan terakhir user saat melakukan timbangan. Di halaman ini juga di tampilkan berat badan ideal untuk user dan sedikit penjelasan tentang apa itu BMR kepada user, dengan adanya penjelasan tersebut dihalaman timbangan diharapkan user dengan mudah untuk mengerti fungsi utama aplikasi *record timbangan.* Seperti pada Gambar 3. (a).

5. Halaman Input Timbangan

Halaman ini merupakan *form* saat *user* melakukan input timbangan, seperti Gambar 3.(b). Pada halaman ini berisikan beberrapa *textfield* yang harus diisi oleh *user* saat ingin menginput data baru timbangan. Pada form telah disesuaikan dimana saja field yang harus diisi dengan angka dan mana saja field yang diisi dengan huruf, hal ini dilakukan dengan menyesuaikan tiap tipe input di tiap field yang ada, seperti contoh input data umur diberikan input type berupa number sehingga user tidak bisa memasukkan selain angka.

6. Halaman Grafik

Halaman ini berisikan grafik dan list data timbangan yang pernah dilakukan oleh user. Pada halaman ini user dapat mengetahui perkembangan berat badan nya melalui grafik yang telah disediakan. Pada tampilan list tiap kelompok item data dapat di click yang nantinya akan menampilkan detail timbangan yang dilakukan pada saat itu. Berikut ini adalah tampilan halaman grafik. Grafik yang digunakan untuk menyajikan data adalah grafik line, dengan axis x berisikan tanggal timbang dan axis y berisikan nilai berat badan pada saat user melakukan timbang, seperti Gambar 3. (c).



(a) (b) (c)

Gambar 3. (a) Tampilan Halaman Timbangan, (b) Halaman Input Data Timbangan, (c) Tampilan Halaman Grafik

7. Halaman Detail Timbangan

Halaman ini berisikan detail dari timbangan yang dilakukan oleh user. Data yang ada pada halaman ini mengacu pada data timbangan yang di pilih user untuk ditampilkan pada saat membuka list di halaman grafik. Berikut adalah tampilan halaman detail timbangan. Yang digunakan dalam halaman detail timbangan bentuk *layout linear* dengan *orientasi vertical* dengan komponen *text view, dan button. Text view* digunakan untuk menampilkan data

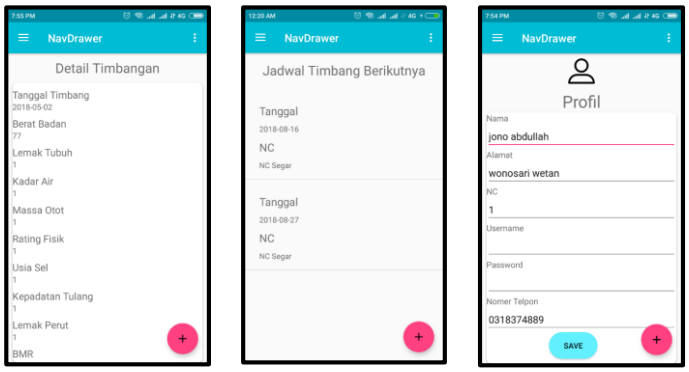
timbangan dan button digunakan untuk melakukan operasi penghapusan data timbangan jika ditemukkan kesalahan pada proses input yang sebelumnya dilakukan (Gambar 4. (a)).

8. Halaman Jadwal

Halaman ini berisikan list jadwal hari apa saja user harus melakukan kegiatan timbang badan ke NC yang telah user tunjuk (Gambar 4.b). Data jadwal yang ada di halaman ini adalah data realtime yang di inputkan langsung oleh admin NC. Pada halaman ini user hanya bisa melihat jadwal, tidak untuk mengubah, menghapus atau menambah jadwal. Berikut adalah tampilan dari halaman jadwal beserta blueprint deasain dari halaman.

9. Halaman *Profile*

Halaman ini menampilkan detail *profile* dari user. Pada halaman ini user juga dapat melakukan proses mengubah data jika ditemukan data yang salah saat melakukakan proses input data sebelumnya atau ada perubahan data oleh user. Field yang disediakan untuk menampilkan dan mengubah data sudah disesuaikan dengan data tipe apa yang harus dimasukkan, seperti field nomor telpon, jika user memasukkan nomor telpon maka tampilan keyboard hanya berisikan angka tanpa ada karakter huruf. Berikut adalah tampilan halaman *profile* dan *blueprint* desain halaman *profile* (Gambar 4.c).



(a) (b) (c)

Gambar 4. (a) Tampilan Halaman Detail timbangan, (b) Tampilan Halaman Jadwal, (c) Tampilan Halaman *Profile*

Pembuatan Logic Aplikasi

Setelah melakukan proses desain pada tiap halaman yang ada pada aplikasi maka langkah selanjutnya adalah proses pemberian logic atau penulisan program di setiap fungsional halaman yang sudah dibuat sebelumnya. Penulisan program dilakukan dengan mengelompokkan bebeerapa class yang dibuat menjadi beberapa kelompok *package*. Tujuan pengelompokkan class pada suatu *package* ini ia dengan dilakukan pengelompokan didalam satu package pada class yang memiliki fungsi yang sama atau Saling mendukung akan memudahkan pengembang lain untuk mengembangkan aplikasi ini lebih jauh lagi.

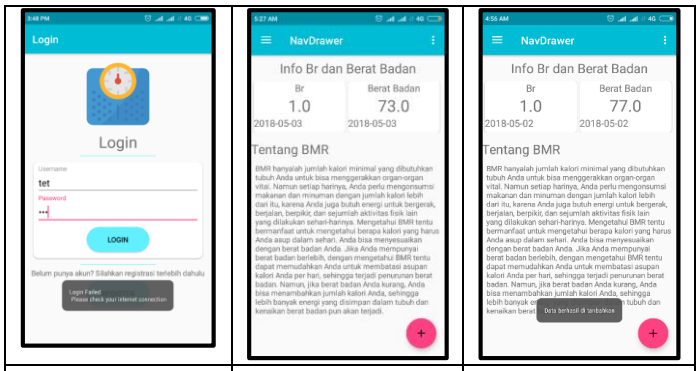
1. *Package view*

Package view berisi class berjenis *activity*, *activity* merupakan sebuah komponen yang berfungsi untuk menampilkan user interface ke layar, antara lain:

a. SplashActivity, merupakan class yang berfungsi sebagai halaman pembuka aplikasi. Selama proses memuat class ini akan menampilkan logo aplikasi Record Timbangan. Setelah selesai memuat activity ini selama satu detik akan ditutup dan berpindah ke activity NavBar.

b. NavDrawer, merupakan activity yang berfungsi sebagai main menu di aplikasi ini dan menjadi penampung dari fragment. Dalam Activity ini terdapat dua komponen utama, yaitu *inavigation bar* yang berisi menu-menu yang ada dalam aplikasi dan container yang merupakan tempat untuk menampilkan fragment yang dibuat, tak hanya itu dalam NavDrawer activity ini juga ada keterangan nama dari user.

c. LoginPanelActivity, merupakan class yang berfungsi untuk menampilkan halaman login bagi user. Pada LoginPanelActivity disediakan dua field edit text yang berfungsi sebagai tempat user memasukkan username dan password yang kemudian nanti dikirim ke server untuk dilakukan pencocokan di database dengan bantuan REST API yang sudah dibuat. Di dalam activity ini juga ada text yang dapat mengantarkan user ke halaman registrasi jika user belum memiliki akun. Secara umum class ini lah yang mengatur proses login, proses login dikarakan berhasil jika user dialihkan secara otomatis ke halaman utama aplikasi yakni halaman timbangan, dan aplikasi akan menyimpan beberapa data yang sering dipakai secara statis seperti id user dan nama user. Proses login dikatakan tidak berhasil jika user menerima notifikasi pendek yang berisikan pessan bahwa proses login tidak berhasil, dan usermasih berada di halaman loogin. Ada beberapa factor yang dapat membuat proses login tidak berhasil yakni tidak adanya koneksi internet saat proses login dilakukan, username dan password salah, dan server database atau RESTAPI mengalami gangguan. Tahap uji coba ketika user mencoba login dengan gangguan yakni tanpa ada koneksi internet, maka aplikasi akan menampilkan notifikasi error seperti yang ditunjukkan di Gambar 5 (a).



(a) (b) (c)

Gambar 5. (a) Notifikasi Proses Login Gagal, (b) Tampilan Halaman Utama Setelah Login, (c) Berhasil Menambah Data Timbangan

Jika proses login berhasil berhail dilakukan maka user akan secara otomatis dialihkan menuju halaman timbangan seperti ditunjukkan pada Gambar 5. (b).

d. RegisterActivity, merupakan class yang berfungsi untuk menampilkan halaman register bagi user. Pada register activity ini, user baru akan diminta untuk memasukkan data diri mereka berupa daftar NC mereka, nama user, username user, password user, no telpon user, alamat user, dan id NC user. Setelah data dimasukkan, kemudian data akan dikirim ke database dengan bantuan REST API. Jika pendaftaran berhasil dilakukan, maka user akan dikembalikan lagi ke halaman login untuk melakukan proses login, jika proses pendaftaran gagal dilakukan maka user akan menerima pemberitahuan bahwa pendaftaran gagal dilakukan dan user masih berada di halaman register. Ada beberapa factor yang dapat membuat proses registrasi tidak berhasil yakni tidak adanya koneksi internet saat proses registrasi dilakukan, data yang dimasukkan belum lengkap, dan server database atau RESTAPI mengalami gangguan. User baru yang telah dibuat masih dalam status tidak aktif, sampai admin NC merubah status user tersebut menjadi aktif barulah user bisa login kedalam aplikasi.

Pada proses uji, activity register berjalan dengan baik, ketika error terjadi pun, error handling yang dibuat bekerja dengan semestinya. Tampilan form register ketika user mengisi data dirinya ditunjukkan seperti pada Gambar 2. (b). Field input yang disediakan pada form login dapat diisi dengan character huruf maupun angka, hal ini berlaku pada field username dan password yang ada pada halaman login sehingga user dapat menggunakan kombinasi huruf dan angka pada saat melakukan proses login.

e. RegiserNCActivity, merupakan class yang berfungsi untuk menampilkan halaman register bagi user. Pada register activity ini, user baru akan diminta untuk memasukkan data diri mereka berupa daftar NC mereka, nama NC, username NC, password NC, no telpon NC, dan alamat NC. Setelah data dimasukkan, kemudian data akan dikirim ke database dengan bantuan REST API. Jika pendaftaran berhasil dilakukan, maka user akan dikembalikan lagi ke halaman login untuk melakukan proses login, jika proses pendaftaran gagal dilakukan maka user akan menerima pemberitahuan bahwa pendaftaran gagal dilakukan dan user masih berada di halaman register. Ada beberapa factor yang dapat membuat proses registrasi tidak berhasil yakni tidak adanya koneksi internet saat proses registrasi dilakukan, data yang dimasukkan belum lengkap, dan server database atau RESTAPI mengalami gangguan. User NC baru yang telah dibuat masih dalam status tidak aktif, sampai admin sistem merubah status NC tersebut menjadi aktif barulah user bisa login kedalam aplikasi web service yang telah dibuat.

2. Package Fragment

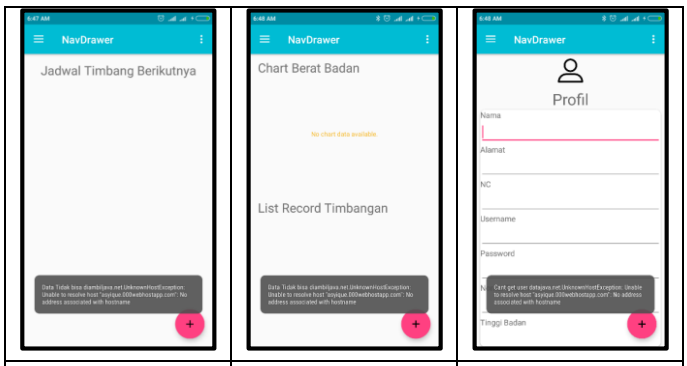
Package ini berisikan class berjenis fragment. Fragment merupakan komponen yang memiliki fungsi untuk menampilkan user interface melalui activity. Dalam project ini yang dimasukkan kedalam kelas ini adalah fungsi-fungsi logika halaman yang berada setalah proses login selesai dilewati. Penjelasan tentang class-class yang ada pada package fragment ini sebagai berikut:

a. AddNewRecord, Fragment ini ditampilkan saat user menekan floating button. Fragment ini berisikan form yang didalamnya user akan diminta untuk mengisikan data timbangan yang sudah dilakukan nya. Data-data yang diminta adalah tanggal timbang, berat

badan, lemak tubuh, kadar air, massa otot, rating fisik, usia sel, kepadatan tulang, lemak perut, dan BMR. Pengiriman data ke server hingga disimpan kedalam database dibantu dengan REST API yang sudah dibuat. Jika data berhasil dimasukkan ke dalam database maka user akan dialihkan ke halaman utama aplikasi timbangan, jika data gagal disimpan didalam database maka user akan menerima pemberitahuan jika data yang dimasukkan gagal disimpan dan user diminta untuk mengecek koneksi internet yang ada pada device mereka, dan user masih tetap berada di halaman penambahan data timbangan. Ada beberapa factor yang dapat membuat proses input data timbangan tidak berhasil yakni tidak adanya koneksi internet saat proses input data dilakukan, data yang dimasukkan belum lengkap, dan server database atau RESTAPI mengalami gangguan.

Pada tahap uji coba class ini berjalan dengan baik, jika user menambahkan data dan berhasil menyimpannya maka user akan otomatis dialihkan menuju halaman timbangan dan diberi notifikasi bahwa proses berhasil dilakukan seperti di tunjukkan pada Gambar 5. (c).

b. JadwalTimbangan, Fragment ini ditampilkan saat user memilih menu jadwal di aplikasi. Jadwal yang ada diurutkan sesuai dari jadwal terakhir yang diinput oleh admin NC. Jadwal timbang ditampilkan dengan bentuk list yang dirutkan dari atas adalah data jadwal terbaru hingga ke bawah adalah data jadwal paling lama. List yang dibuat menggunakan *RecyclerView* membuat data yang banyak pun dapat ditampung dengan efisien karena pembuatan list tidak memakan resource yang cukup banyak. Pada class ini berisi kode untuk menampilkan informasi jadwal dan informasi NC tempat timbang user. Proses pengambilan data jadwal di lakukan dengan bantuan REST API. Proses pengambbilan data berhasil jika data jadwal berhasil ditampilkan dan sebaliknya dikatakan gagal jika user mendapatkan notifikasi bahwa aplikasi tidak dapat mengambil data dari server. Ada beberapa faktor yang membuat proses pengambilan data tidak berhasil, salah satunya adalah tidak adanya koneksi internet pada device user, atau server tempat REST API diletakkan mengalami gangguan, berikut ini adalah tampilan halaman jadwal ditunjukkan pada Gambar 6. (a) ketika dalam tahap pengujian saat mengalami gangguan tidak ada koneksi internet. Pada menu ini user hanya memiliki akses untuk melihat data jadwal saja. User tidak memiliki akses untuk mengubah data jadwal, jika user ingi merubah data jadwal, maka user harus menghubungi admin NC untuk mengubah data jadwal user melalui web service, hal ini dilakukan agar user tidak merubah waktu tanpa adanya koordinasi dengan pemilik NC.



(a) (b) (c)

Gambar 6. (a) Notifikasi Error Halaman Jadwal, (b) Notifikasi Error pada Halaman List Timbangan, (c) Notofikasi Error pada Halaman Profil

c. ListTimbangan, Fragment ini ditampilkan jika user memilih menu grafik. Pada fragment ListTimbangan terdapat logic untuk mengolah data yang ditampilkan oleh grafik timbangan dan list timbangan yang pernah dimasukkan. Grafik yang dibuat berisikan informasi perkembangan berat badan user yang diambil dari database data timbangan user. Grafik yang dibuat tidak mendukung perubahan data secara real-time, sehingga user harus berpindah menu terlebih dahulu jika menambahkan atau terjadi perubahan data baru sehingga tampilan grafik akan berubah. Pada list timbangan yang ada dibuat menggunakan *RecyclerView* membuat data yang banyak pun dapat ditampung dengan efisien karena pembuatan list tidak memakan resource yang cukup banyak. Data list dapat di click di tiap item nya untuk selanjutnya dapat dilihat detail dari item timbangan tersebut, karena pada list timbangan hanya berisikan beberapa data dari timbangan saja. Proses pengambilan data timbangan di lakukan dengan bantuan REST API. Proses pengambilan data berhasil jika data timbangan berhasil ditampilkan di dalam grafik dan juga list dan sebaliknya dikatakan gagal jika user mendapatkan notifikasi bahwa aplikasi tidak dapat mengambil data dari server. Ada beberapa faktor yang membuat proses pengambilan data tidak berhasil, salah satunya adalah tidak adanya koneksi internet pada device user, atau server tempat REST API diletakkan mengalami gangguan, berikut ini adalah contoh halaman list timbangan ketika memunculkan notifikasi error jika dibuka tanpa ada koneksi internet ditunjukkan pada Gambar 6.(b).

d. Profile, Fragment profile berisi logic untuk mengolah data detail profil user. Logic yang ada di fragment ini berfungsi untuk mengedit data profil user jika terjadi kesalahan atau ada perubahan data di kemudian hari. Pengambilan data dan perubahan data yang dilakukan pada profile user dilakukan oleh bantuan REST API, dan langsung berkaitan dengan database yang ada pada server. Proses pengambilan data berhasil jika data profile berhasil ditampilkan dan sebaliknya dikatakan gagal jika user mendapatkan notifikasi bahwa aplikasi tidak dapat mengambil data dari server. Ada beberapa faktor yang membuat proses pengambilan data tidak berhasil, salah satunya adalah tidak adanya koneksi internet pada device user, atau server tempat REST API diletakkan mengalami gangguan, berikut ini adalah gambar halaman profil jika memunculkan error karena dibuka tanpa adanya koneksi internet pada device user, ditunjukkan oleh Gambar 6. (c).

e. TimbanganFragment, TimbanganFragment ia fragment yang dibuat juga sebagai halaman awal ketika aplikasi sudah dibuka (jika user sudah login), dan sebagai halaman pemantauan berat badan paling terkini user. Berbeda dengan class fragment yang lain, class ini tidak berkomunikasi dengan data yang ada pada database di server.

f. ViewRecord, ViewRecord fragment ia fragment yang berisikan logic ketika tampilan detail dari record di *click* oleh user pada salah satu item timbangan yang ada pada halaman list timbangan. Pada fragment ini logic juga berfungsi untuk proses penghapusan data jika ditemukan kesalahan dalam proses memasukkan data timbangan. Semua proses menampilkan hingga menghapus data pada class ini berhubungan langsung dengan dataase yang ada di server sehingga dalam penggunaan nya dibutuhkan koneksi internet dan bantuan REST API. Proses pengambilan data dan hapus berhasil jika data detail timbangan ber-

hasil ditampilkan dan atau berhasil dihapus. sebaliknya dikatakan gagal jika user mendapatkan notifikasi bahwa aplikasi tidak dapat mengambil data dari server. Ada beberapa faktor yang membuat proses pengambilan data tidak berhasil, salah satunya adalah tidak adanya koneksi internet pada device user, atau server tempat REST API diletakkan mengalami gangguan.

3. Package Adapter

Adapter merupakan komponen yang akan mengatur bagaimana menampilkan dataset ke dalam *RecyclerView*. Di sinilah terjadi proses pengisian tampilan (*ViewInflate*) dari file layout xml untuk tiap elemen dari data yang sebelumnya terpasang *(bind)* ke dalam **RecyclerView**. Dalam project ini dibuat beberapa kelas.

Class-class pada package adapter ini berfungsi untuk meng-*handle* tampilan recyclerview yang dibuat. Secara garis besar ada dua jenis class yaitu class adapter dan viewholder. Class adapter berfungsi untuk pengolahan data yang akan ditampilkan ke dalam recyclerview, meliputi jumlah data, object data, dan posisi data, sedangkan class viewholder berfungsi untuk menepatkan data ke tampilan yang sudah dibuat. Berikut adalah contoh code dari salah satu class adapter;

```
public class JadwalAdapter extends RecyclerView.Adapter<JadwalViewHolder> {
   private List<JadwalModel> list = new ArrayList<>();

   public void updateData(List<JadwalModel> items){
      list.addAll(items);
      notifyDataSetChanged();
   }

   @NonNull
   @Override
   public JadwalViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {
      return new JadwalViewHold-
er(LayoutInflater.from(parent.getContext()).inflate(R.layout.jadwal_item, parent,
false));
   }

   @Override
   public void onBindViewHolder(@NonNull JadwalViewHolder holder, int position) {
      holder.bind(list.get(position));
   }

   @Override
   public int getItemCount() {
      return list.size();
   }
```

```
}
```

Dan berikut ini adalah contoh code dari class viewholder yang ada pada package adapter;

```
public class JadwalViewHolder extends RecyclerView.ViewHolder{
    @BindView(R.id.tanggal_jadwal)
    TextView tanggalJadwal;

    public JadwalViewHolder(View itemView) {
        super(itemView);
        ButterKnife.bind(this, itemView);
    }

    public void bind(final JadwalModel item){
        tanggalJadwal.setText(item.getTanggal());

        itemView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                //Do Nothing
            }
        });
    }
}
```

Pemilihan penggunaan recyclerview di setiap list yang dibuat di aplikasi dengan alas an recyclerview merupakan sebuah widget tampilan yang efisien digunakan untuk menampilkan data yang banyak berupa list dibandingkan dengan pendahulunya yaitu listview. Pada recyclerview tidak semua tampilan dari lst akan dibuat, jadi tampilan yang dibuat adalah tampilan yang terlihat di layar, jika data yang akan ditampilkan lebih dari batas layar untuk menampilkan data maka ketika user melakukan *scrolling* terhadap layar maka recyclerview akan membuat item baru dari item yang sudah tidak terpakai atau sudah tidak terlihat di layar, dari kerja itulah kenapa recyclerview menjadi lebih efisien daripada listview untuk digunakan.

4. Package Api

Package api dibuat untuk memudahkan pengelompokkan class-class yang berkerja dengan REST API yang dibuat. Pada package api terdapat dua class, yakni class interface APICall, dan class APIClient. Class interface APICall berisikan *endpoint* REST API yang digunakan. Dalam class interface ini digunakan juga library Retrofit, library ini dugunakan untuk memudahkan aplikasi berkomunikasi dengan REST API. Kemudian class APIClient, dalam class ini berisikan logic dengan tujuan utama HttpRequest ke REST API, dengan bantuan library Retrofit untuk memudahkan prosesnya. Berikut ini adalah contoh source code pada class interface APICall di salah satu request ke endpoint REST API;

```
@FormUrlEncoded
@POST("user/login")
Call<LoginModel> loginRequest(@Field("username_user")String username_user,
                              @Field("password_user")String password_user);
```

Berikut ini adalah contoh source code pada class APIClient;

```
public class APIClient {
    private static Retrofit retrofit = null;

    public static Retrofit retrofit(String BASE_URL){
        HttpLoggingInterceptor interceptor = new HttpLoggingInterceptor();
        interceptor.setLevel(HttpLoggingInterceptor.Level.BODY);
        OkHttpClient client = new OkHttpCli-
ent.Builder().addInterceptor(interceptor).build();

        if (retrofit==null){
            retrofit = new Retrofit.Builder()
                    .baseUrl(BASE_URL)
                    .addConverterFactory(GsonConverterFactory.create())
                    .client(client)
                    .build();
        }return retrofit;
    }


    public APICall getService(){
        return retrofit(BuildConfig.BASE_URL).create(APICall.class);
    }
}
```

5. Package Model

Package ini berisikan class POJO (Plain Old Java Object) dalam kegunaan nya POJO merupakan sebuah objek yang berdiri sendiri tidak memiliki ketergantungan dengan class lain atau tidak perlu melakukan extends class lain. POJO sendiri digunakan sebagai model untuk data JSON (JavaScript Object Notation) yang sudah di parsing. Dalam package ini terdapat beberapa class POJO, berikut adalah salah satu contoh source code dari class pada package model;

```
public class LoginModel {
    @SerializedName("message")
    @Expose
```

```
    private String message;
    @SerializedName("status")
    @Expose
    private Boolean status;
    @SerializedName("data")
    @Expose
    private UserModel user;

    public String getMessage() {
        return message;
    }

    public void setMessage(String message) {
        this.message = message;
    }

    public Boolean getStatus() {
        return status;
    }

    public void setStatus(Boolean status) {
        this.status = status;
    }

    public UserModel getUser() {
        return user;
    }

    public void setUser(UserModel user) {
        this.user = user;
    }
}
```

Pada class-class di package model ini hanya berisikan setter dan getter yang nantinya digunakan di class lain ketika ada data yang perlu di tampilkan dari response yang dilakukan ke REST API.

6. Package Util, Package ini berisikan satu class yakni SharedPrefManager. Dalam class ini digunakan sebuah class milik android yakni class SharedPreferenceManager. Dengan class ini penulis dapat melakukan penyimpanan data sederhana didalam aplikasi android. Data yang disimpan ia data yang digunakan berulang-ulang dan tergolong statis, seperti data nama user, id user, dan status login dan logout user di aplikasi.

KESIMPULAN

Untuk membuat sebuah aplikasi android dengan bahasa native kita membutuhkan sebuah tools yang bernama android studio, yang merupakan IDE resmi yang dikeluarkan oleh google bagi pengembang aplikasi android.

Untuk menghubungkan antara aplikasi android dengan web service dibutuhkan API.

Pembuatan aplikasi record timbangan merupakan salah satu solusi untuk membantu memecahkan masalah user yang sering sekali mengalami kesulitan untuk mengetahui dan memantau informasi tentang berat badan dan kesehatannya.

Untuk saran pengembangan penelitian diharapkan dapat menambah fitur pada aplikasi timbangan berbasis android, mengembangkan tampilan yang lebih baik yang memperhatikan sisi user experience serta penyempurnaan di beberapa fungsi yang kini dibuat belum terlalu kompleks dan sesuai dengan kebutuhan di lapangan.

DAFTAR PUSTAKA

Detik Healt, https://health.detik.com/hidup-sehat-detikhealth/d-2136125/ini-akibat-kelebihan-berat-badan-dari-ujung-kepala-hingga-ujung-kaki, Januari 2013

Hello Sehat https://hellosehat.com/penyakit/ obesitas-kegemukan, September 2018

M. Afdali, M. Daud dan R. Putri, "*Perancangan Alat Ukur Digital Untuk Tinggi dan Berat Badan dengan Output suara Berbasis Arduino UNO*", Jurnal ELKOMNIKA, Vol. 5 (1), Juni 2017, pp 106-118

F. Rahman, H. Fauzi dan T.N. Azhar, "*Analisa Metode Pengukuran Berat Badan Manusia Dengan Pengolah Citra*", Jurnal TEKNIK, Vol. 38 (1), 2017, pp 35-39

R.F. Vicky, S.D. Budiwati dan A. Nugraha, "*Aplikasi Pencatatan Aktivitas Diet Pada Penderita Obesitas Berbasis J2ME*", Jurnal Teknologi Informasi, Vol. 1 (5), 2013, pp 171-175

S. Ariyono, I.G.S. Masdiyasa dan M.M. Kholis, "Rancang Bangun "*Pencatat Kebugaran Tubuh Berbasis Indeks Masa Tubuh*", Jurnal Penelitian Poltekbang Surabaya, Vol. 3 (4), 2018, pp 43-51

Kadir, A. *Pengenalan Sistem Informasi.* Yogyakarta: Andi Offest, 2003

Erich Gamma. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994

A. Immamudin dan S. Permana, "*Menjadi Android Developer Expert".* Bandung: PT. Presentologics. , 2018

Priyantini, "*Pemograman Android Untuk Pemula.*" Jakarta, Cerdas Pustaka 2012

P. Sulistyorini, "*Pemodelan Visual Dengan Menggunakan UML dan Rational Rose".* Pekalongan: Informatika, 2016