

LAPORAN PRAKTIKUM
PEMROGRAMAN PERANGKAT BERGERAK



DISUSUN OLEH :

NAMA : Agustiawan

NIM : 21120116120008

KELOMPOK : 37

DEPARTEMEN TEKNIK KOMPUTER
FAKULTAS TEKNIK
UNIVERSITAS DIPONEGORO
SEMARANG

2018

LAPORAN PRAKTIKUM
PEMROGRAMAN PERANGKAT BERGERAK



DISUSUN OLEH :

NAMA : Agustiawan

NIM : 21120116120008

KELOMPOK : 37

DEPARTEMEN TEKNIK KOMPUTER
FAKULTAS TEKNIK
UNIVERSITAS DIPONEGORO
SEMARANG

2018

Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik
Universitas Diponegoro Semarang



LEMBAR PENGESAHAN

PRAKTIKUM PEMROGRAMAN PERANGKAT BERGERAK 2018

Nama : Agustiawan

NIM : 21120116120008

Raisya Rahmah Noor
21120115140065

Najakhul Fahmi
21120115120035

Muhammad Adinugroho
21120115130063

Bima Fajar Setiawan
21120115140088

Pramudya Erviansyah
21120115130055

Tito Anugerah Maharizky
21120115130049

Lian Cahyo Wijaya
21120115120001

Pradipita Sekar Ayu Putri W
21120115120030

Muchammad Dwi Cahyo N.
21120115120039

Pandu Kent Ellian
21120115120009

Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik
Universitas Diponegoro Semarang



Dhesti Rosytawati Rahayu
21120115120033

Fakhry Fauzan
21120115140062

Mengetahui
Koordinator Praktikum

Pramudya Erviansyah
21120115130055

KATA PENGANTAR

Assalamualaikum Wr. Wb.

Puji syukur ke hadirat Tuhan Yang Maha Esa karena atas berkah, rahmat, dan hidayah-Nya sehingga Laporan Praktikum Pemrograman Perangkat Bergerak 2018 ini dapat diselesaikan dan disusun dengan baik.

Laporan ini berisi mengenai pembelajaran mengenai pengembangan aplikasi mobile pada berbagai platform. Diharapkan laporan ini dapat membantu kita dalam mempelajari dan memahami segala sesuatu yang berkaitan dengan pengembangan aplikasi mobile. Sehingga kita dapat mengimplementasikan ilmu yang kita peroleh di kemudian hari.

Ucapan terima kasih diberikan kepada :

1. Allah SWT, yang telah melimpahkan rahmat-Nya,
2. Kedua orang tua, yang selalu mendukung dan mendoakan saya,
3. Bapak Ir. Kodrat Imam Satoto, MT, selaku dosen pengampu mata kuliah Pemrograman Perangkat Bergerak,
4. Seluruh asisten praktikum, yang telah membimbing saya dalam praktikum,
5. Seluruh teman-teman Teknik Komputer angkatan 2016 dan semua pihak yang terlibat lainnya.

Semoga laporan ini dapat bermanfaat bagi yang membacanya. Kritik dan saran sangat diterima demi lebih baiknya laporan ini.

Wassalamualaikum Wr. Wb.

Semarang, 22 November 2018
Penulis,



Agustiawan
21120116120008

DAFTAR ISI

Halaman Cover.....	i
Halaman Pengesahan	ii
Kata Pengantar	iv
Daftar Isi.....	v
Daftar Gambar.....	viii
Abstrak	xviii
BAB I PENDAHULUAN	
1.1.Latar Belakang Praktikum	1
1.2.Tujuan Praktikum.....	2
1.3.Metode Penulisan Laporan.....	3
BAB II Xamarin	
2.1. Tujuan	5
2.2. Alat dan Bahan.....	6
2.3. Dasar Teori.....	8
2.4. Langkah Kerja.....	16
2.5. Analisa Hasil Percobaan	41
2.6. Tugas	60
2.7. Kesimpulan	69
BAB III Android Studio 1	
3.1. Tujuan	70
3.2. Alat dan Bahan.....	71
3.3. Dasar Teori.....	73
3.4. Langkah Kerja.....	82
3.5. Analisa Hasil Percobaan	120
3.6. Tugas	153
3.7. Kesimpulan	168
BAB IV Android Studio 2	
4.1. Tujuan	169
4.2. Alat dan Bahan.....	170

4.3. Dasar Teori.....	172
4.4. Langkah Kerja.....	174
4.5. Analisa Hasil Percobaan	201
4.6. Tugas	221
4.7. Kesimpulan	228
BAB V Progressive Web Application 1	
5.1. Tujuan	229
5.2. Alat dan Bahan.....	230
5.3. Dasar Teori.....	234
5.4. Langkah Percobaan	240
5.5. Hasil Percobaan dan Analisa.....	257
5.6. Tugas	262
5.7. Kesimpulan	275
BAB VI Progressive Web Application 2	
6.1. Tujuan	276
6.2. Alat dan Bahan.....	277
6.3. Dasar Teori.....	279
6.4. Langkah Percobaan	282
6.5. Analisa Percobaan.....	295
6.6. Tugas	300
6.7. Kesimpulan	316
BAB VII iOS	
7.1. Tujuan	317
7.2. Dasar Teori.....	318
7.3. Langkah Kerja.....	321
7.4. Hasil Percobaan dan Analisa.....	338
7.5. Kesimpulan	342
BAB VIII PENUTUP	
8.1. Kesimpulan	343
8.2. Saran.....	344
Daftar Pustaka	

LAMPIRAN I (Lembar Asistensi)

LAMPIRAN II (Tugas Pendahuluan)

DAFTAR GAMBAR

Gambar 2.1 PC/Notebook	6
Gambar 2.2 Visual Studio Community 2017	6
Gambar 2.3 Device Android dan Kabel Data USB.....	7
Gambar 2.4 Android SDK.....	7
Gambar 2.5 Arsitektur dari platform Xamarin.....	9
Gambar 2.6 Solution explorer pada Visual Studio.....	10
Gambar 2.7 Xamarin Form UI	11
Gambar 2.8 SDK pada Xamarin	13
Gambar 2.9 Sistem Operasi iOS	14
Gambar 2.10 Membuat Project baru	16
Gambar 2.11 Membuat folder view dan model	16
Gambar 2.12 Cara menambahkan class	17
Gambar 2.13 Membuat class User.cs.....	17
Gambar 2.14 Membuat class UserLogin.cs	17
Gambar 2.15 Cara menambahkan New Item	18
Gambar 2.16 Menambahkan Content Page LoginPage.xaml	19
Gambar 2.17 Menambahkan Content Page SignUpPage.xaml.....	19
Gambar 2.18 Membuat Content Page (C#) LoginPageCS.cs	23
Gambar 2.19 Membuat Content Page (C#) MainPageCS.cs	24
Gambar 2.20 Membuat Content Page (C#) SignUpPageCS.cs.....	24
Gambar 2.21 Membuat Content Page App.xaml	29
Gambar 2.22 Manage NuGet Packages	30
Gambar 2.23 Instalasi NuGet Packages	31
Gambar 2.24 Menambahkan Class baru pada folder Model.....	31
Gambar 2.25 Membuat class DataMahasiswa.cs	32
Gambar 2.26 Membuat Content Page (C#) pada folder View dengan nama HalamanUtama.cs	33
Gambar 2.27 Content Page (C#) pada folder View dengan nama HalamanTambahData.cs	33

Gambar 2.28 Membuat Content Page (C#) pada folder View dengan nama HalamanLihatData.cs	34
Gambar 2.29 Memilih Opsi Pengembang pada Android Phone	37
Gambar 2.30 Mengaktifkan Debugging USB	38
Gambar 2.31 Mengaktifkan Debugging USB	39
Gambar 2.32 Clean Solution	39
Gambar 2.33 Tampilan Pada Visual Studio 2017 ketika smartphone telah terhubung	40
Gambar 2.34 Tampilan awal pada smartphone ketika program di deploy	40
Gambar 2.35 Halaman Login	51
Gambar 2.36 Halaman SignUp	52
Gambar 2.37 Halaman Main Page	52
Gambar 2.38 Halaman Utama	57
Gambar 2.39 Menambah data Agustiawan	57
Gambar 2.40 Tambah data Agustiawan berhasil	58
Gambar 2.41 Menambah data Saskia.....	58
Gambar 2.42 Tambah data saskia berhasil.....	59
Gambar 2.43 Halaman Lihat Data	59
Gambar 2.44 Tampilan Halaman Utama	65
Gambar 2.45 Tampilan Awal Lihat Data	66
Gambar 2.46 Mengubah Data Agustiawan	66
Gambar 2.47 Data Agustiawan berubah menjadi Muhammad Agustiawan	67
Gambar 2.48 Menghapus Data Agustiawan	67
Gambar 2.49 Data Agustiawan telah terhapus	68
Gambar 3.1 Notebook	71
Gambar 3.2 Android Studio	71
Gambar 3.3 Perangkat Android dan Kabel USB	72
Gambar 3.4 Android Studio	74
Gambar 3.5 Layout Editor Mode Design pada Android Studio.....	75
Gambar 3.6 Layout Editor XML Mode Text pada Android Studio.....	76
Gambar 3.7 Struktur View	76

Gambar 3.8 Contoh Constraint Layout	77
Gambar 3.9 Linear Layout	78
Gambar 3.10 Relative Layout	78
Gambar 3.11 Grid Layout	79
Gambar 3.12 Frame Layout	80
Gambar 3.13 New Project.....	82
Gambar 3.14 Tab Create New Project	82
Gambar 3.15 Target Android Devices	83
Gambar 3.16 Pilihan Activity	83
Gambar 3.17 Isi nama activity	84
Gambar 3.18 Proses gradle selesai.....	84
Gambar 3.19 Tampilan membuat drawable resource	86
Gambar 3.20 Membuat file roundedbutton.xml.....	87
Gambar 3.21 Gambar-gambar pada folder res→drawable	88
Gambar 3.22 Layout activity_login.xml	91
Gambar 3.23 Membuat class baru User	94
Gambar 3.24 Membuat class baru DatabaseHandler	95
Gambar 3.25 Membuat Empty Activity.....	99
Gambar 3.26 Membuat Register Activity	99
Gambar 3.27 Membuat MainActivity	103
Gambar 3.28 File gambar splashscreen di res-drawable.....	108
Gambar 3.29 Membuat Activity Splashscreen.....	108
Gambar 3.30 Tampilan Splashscreen.....	109
Gambar 3.31 Membuat EmailActivity	112
Gambar 3.32 Membuat AboutActivity	116
Gambar 3.33 Tampilan LoginActivity	120
Gambar 3.34 Tampilan RegisterActivity	131
Gambar 3.35 Tampilan MainActivity	136
Gambar 3.36 Tampilan Splashscreen.....	141
Gambar 3.37 Tampilan EmailActivity	142
Gambar 3.38 Tampilan activity_about.....	148

Gambar 3.39 Tampilan sebelum dan sesudah logo diganti pada halaman activity_login	153
Gambar 3.40 Tampilan sebelum dan sesudah logo diganti pada halaman activity_register.....	154
Gambar 3.41 Tampilan sebelum dan sesudah logo diganti pada halaman activity_main.....	154
Gambar 3.42 Membuat construct layout (1)	155
Gambar 3.43 Tampilan setelah di convert menjadi construct layout pada activity_login	155
Gambar 3.44 Membuat construct layout (2)	156
Gambar 3.45 Tampilan setelah di convert menjadi construct layout pada activity_register.....	156
Gambar 3.46 Tampilan activity_splashscreen.xml	157
Gambar 3.47 Tampilan activity_login.xml	158
Gambar 3.48 Tampilan activity_register.xml	158
Gambar 3.49 Tampilan activity_main.xml	159
Gambar 3.50 Tampilan activity_email.xml	159
Gambar 3.51 Tampilan activity_about.xml	160
Gambar 3.52 Tampilan activity_main.xml dengan button edit	161
Gambar 3.53 Tampilan activity_main.xml	166
Gambar 3.54 Tampilan activity_edit.xml	166
Gambar 3.55 Tampilan Halaman Registrasi	167
Gambar 4.1. PC / Notebook	170
Gambar 4.2. Android Studio	170
Gambar 4.3. Logo Java	171
Gambar 4.4. Android SDK.....	171
Gambar 4.5. Device Android dan kabel USB	171
Gambar 4.6. API	172
Gambar 4.7. Retrofit	173
Gambar 4.8. MVP	173
Gambar 4.9. Android Studio	174

Gambar 4.10. File – Open	174
Gambar 4.11. Pilih Mode Android.....	174
Gambar 4.12 New Package	175
Gambar 4.13. Tambah Beberapa Package	175
Gambar 4.14. Double click build.grade	176
Gambar 4.15. Klik Sync Now	176
Gambar 4.16. Membuat Java Class.....	177
Gambar 4.17. Membuat New Class DataCar	177
Gambar 4.18. Membuat Java Class.....	180
Gambar 4.19. Membuat New Class Interface Api	180
Gambar 4.20. Membuat New Class RetrofitClient	181
Gambar 4.21. Membuat New Class Constant	182
Gambar 4.22. Membuat Class HomePresenter	183
Gambar 4.23. Membuat Interface HomeView	185
Gambar 4.24. Membuat activity baru	185
Gambar 4.25. Membuat HomeActivity.....	185
Gambar 4.26. Memilih activity_home.xml	186
Gambar 4.27. Membuat new Layout resorce file.....	187
Gambar 4.28. Membuat layout resource file item_car.....	187
Gambar 4.29. Menambahkan kode pada HomeActivity	189
Gambar 4.30. Membuat class CarAdapter	191
Gambar 4.31. Membuat class CarListerner.....	192
Gambar 4.32. Membuat class CarViewHolder	193
Gambar 4.33. Membuat Interface AddView.....	195
Gambar 4.34. Membuat activity AddActivity.....	197
Gambar 4.35. Mengubah MainActivity menjadi HomeActivity	199
Gambar 4.36. Mengganti pada splashscreen.....	199
Gambar 4.37. Import class	200
Gambar 4.38. Tampilan splashscreen	218
Gambar 4.39. Tampilan login	219
Gambar 4.40. Tampilan Halaman utama	219

Gambar 4.41. Tampilan Tambah data.....	220
Gambar 4.42. Tampilan tambah data sukses.....	220
Gambar 4.43. Tampilan Utama.....	227
Gambar 4.44. Tampilan Detail Car	227
Gambar 5.1 Laptop.....	230
Gambar 5.2 Smartphone.....	230
Gambar 5.3 Google Chrome Pada Laptop	231
Gambar 5.4 Google Chrome Pada Handphone	231
Gambar 5.5 Visual Studio Code.....	232
Gambar 5.6 Node.js.....	232
Gambar 5.7 Akun google aktif.....	233
Gambar 5.8 Folder pwa-1-kel37	240
Gambar 5.9 Membuka folder dengan VS Code	240
Gambar 5.10 Folder dan File yang dibuat pada Pwa-1-kel37.....	241
Gambar 5.11 Install NPM menggunakan terminal VSCode	242
Gambar 5.12 node_modules dan package-lock.json.....	243
Gambar 5.13 Mengetikkan perintah npm start.....	246
Gambar 5.14 Tampilan yang muncul di browser.....	247
Gambar 5.15 Halaman inspect elements	247
Gambar 5.16 Halaman web saat offline	248
Gambar 5.17 Halaman web setelah ditambahkan service worker	250
Gambar 5.18 Struktur folder yang baru	251
Gambar 5.19 Menambahkan proyek baru	251
Gambar 5.20 Membuat proyek baru bernama pwa-1-kel37	251
Gambar 5.21 Menginstall firebase	252
Gambar 5.22 Berhasil login ke firebase dengan akun Gmail.....	252
Gambar 5.23 Menginisialisasi proyek yang akan dideploy ke firebase.....	252
Gambar 5.24 Menyetujui proses inisialisasi	252
Gambar 5.25 Menentukan fitur yang akan digunakan	253
Gambar 5.26 Menentukan proyek firebase yang akan digunakan	253
Gambar 5.27 Menjawab pertanyaan yang muncul.....	253

Gambar 5.28 Deploy proyek ke firebase.....	254
Gambar 5.29 Tampilan halaman web yang telah berhasil dideploy ke firebase	254
Gambar 5.30 Tampilan halaman web hosting pada smartphone	255
Gambar 5.31 Add Webpage to Home Screen	255
Gambar 5.32 Pop Up yang muncul untuk menambahkan ke layar utama	256
Gambar 5.33 Tampilan shortcut webpage pada layar utama	256
Gambar 5.34 Tampilan project pada web	257
Gambar 5.35 Tampilan project pada web saat kondisi Offline.....	257
Gambar 5.36 Tampilan project setelah tombol Tampilkan ditekan.....	258
Gambar 5.37 Tampilan project saat melakukan pencarian data	258
Gambar 5.38 Tampilan splash screen project pada mobile.....	259
Gambar 5.39 Tampilan project pada mobile.....	259
Gambar 5.40 Tampilan project pada saat melakukan pencarian pada mobile..	260
Gambar 5.41 Tampilan project tugas pada web.....	262
Gambar 5.42 Tampilan project tugas pada web daftar kode pos berjumlah 15	262
Gambar 5.43 Tampilan project tugas pada web saat Offline ..	263
Gambar 5.44 Tampilan project tugas halaman About pada web	263
Gambar 5.45 Tampilan project tugas halaman About pada web saat Offline ..	264
Gambar 5.46 Tampilan project saat melakukan pencarian pada web	264
Gambar 5.47 Tampilan splash screen project tugas pada mobile	265
Gambar 5.48 Tampilan project tugas pada mobile	265
Gambar 5.49 Tampilan project tugas pada mobile	266
Gambar 5.50 Tampilan project tugas saat melakukan pencarian pada mobile .	267
Gambar 5.51 Tampilan project tugas halaman About pada mobile.....	267
Gambar 6.1 Laptop.....	277
Gambar 6.2 Google Chrome Pada Laptop	277
Gambar 6.3 Web Server for Chrome	278
Gambar 6.4 Visual Studio Code.....	278
Gambar 6.5 Memilih folder Latihan pada Web Server for Chrome	282
Gambar 6.6 Tampilan link dari web server.....	282

Gambar 6.7 Hasil tampilan link dari web server setelah diedit dan dimuat ulang	283
Gambar 6.8 Penambahan kode di bawah // TODO add saveSelectedCities function here	283
Gambar 6.9 Penambahan kode di bawah // TODO add startup code here.....	284
Gambar 6.10 Mengubah kode untuk penanganan tombol add city.....	285
Gambar 6.11 Penambahan kode di bawah // TODO add service worker code here	286
Gambar 6.12 Membuat file service-worker.js.....	286
Gambar 6.13 Tampilan Dev Tools pada Google Chrome setelah ditambah kode pada program.....	287
Gambar 6.14 Hasil pada Cache Storage setelah dimuat ulang.....	288
Gambar 6.15 Mengaktifkan mode offline pada Service Worker	289
Gambar 6.16 Hasil yang ditampilkan pada mode offline	289
Gambar 6.17 Menghapus semua data yang tersimpan.....	290
Gambar 6.18 Menambahkan kota baru saat mode Online	292
Gambar 6.19 Tampilan saat mode Offline	292
Gambar 6.20 Membuat shortcut dari PWA cuaca	293
Gambar 6.21 Shortcut pada homescreen.....	294
Gambar 6.22 Tampilan aplikasi ketika dibuka	294
Gambar 6.23 Penampilan fake data pada saat offline	295
Gambar 6.24 Penampilan data terakhir pada saat offline	295
Gambar 6.25 Hasil deploy aplikasi pada icon homescreen.....	298
Gambar 6.26 Tampilan splash screen aplikasi pada mobile	298
Gambar 6.27 Tampilan index aplikasi pada mobile.....	299
Gambar 6.28 Tampilan splash screen aplikasi pada mobile	313
Gambar 6.29 Tampilan data weather real time beserta footer	313
Gambar 6.30 Tampilan pilihan kota.....	314
Gambar 6.31 Tampilan data weather real time pada browser.....	314
Gambar 6.32 Tampilan pilihan kota.....	315
Gambar 7.1 iOS 10.....	319

Gambar 7.2 Logo Swift.....	319
Gambar 7.3 Logo Xcode	320
Gambar 7.4 Tampilan Awal Xcode	321
Gambar 7.5 Memilih Single View App	322
Gambar 7.6 Memberi Nama Project	322
Gambar 7.7 Memilih Main Storyboard.....	323
Gambar 7.8 Membuat Tampilan Aplikasi.....	323
Gambar 7.9 Menekan ViewController pada View.....	324
Gambar 7.10 Melakukan Constraint Otomatis.....	324
Gambar 7.11 Menambahkan Segue	325
Gambar 7.12 Membuat Segue.....	325
Gambar 7.13 Show Segue.....	326
Gambar 7.14 Memberi nama view pertama.....	326
Gambar 7.15 Memberi nama view kedua	326
Gambar 7.16 Menekan Show the Assistant Editor	327
Gambar 7.17 Menghubungkan Widget dengan halaman viewcontroller.swift.	328
Gambar 7.18 Memberi nama TextField1	328
Gambar 7.19 Memberi nama TextField2	328
Gambar 7.20 Memberi nama Button.....	328
Gambar 7.21 Memberi nama Action.....	329
Gambar 7.22 Secure Text Entry.....	330
Gambar 7.23 New File	330
Gambar 7.24 Memilih Cocoa Touch Class	330
Gambar 7.25 Memberi Nama HalamanAplikasi.....	331
Gambar 7.26 Memilih HalamanAplikasi (1)	331
Gambar 7.27 Memilih HalamanAplikasi (2)	331
Gambar 7.28 Memilih HamalanAplikasi.swift	331
Gambar 7.29 Memasukkan Fungsi Tombol Logout ke Code	332
Gambar 7.30 Membuat button logout	332
Gambar 7.31 Membuat LogoutAction	332
Gambar 7.32 Membuat Tampilan Halaman Generate Number	333

Gambar 7.33 Memasukkan komponen ke dalam source code.....	333
Gambar 7.34 memberi nama variabel cekGG.....	334
Gambar 7.35 memberi nama variabel hasil.....	334
Gambar 7.36 memberi nama variabel nilaiMin	334
Gambar 7.37 memberi nama variabel nilaiMax.....	334
Gambar 7.38 memberi nama variabel generate.....	335
Gambar 7.39 memberi nama action generateTombol.....	335
Gambar 7.40 Tampilan Login	336
Gambar 7.41 Tampilan Number Generator	337
Gambar 7.42 Tampilan Program pada Halaman Login	340
Gambar 7.43 Tampilan Program pada HalamanAplikasi saat genap	340
Gambar 7.44 Tampilan Program pada HalamanAplikasi saat ganjil.....	341

ABSTRAK

Perkembangan teknologi informasi saat ini sangat cepat. Kebutuhan akan sumber informasi seiring naik dengan perkembangan teknologi tersebut. Karena itu manusia perlu menciptakan suatu terobosan baru untuk mempermudah dalam melakukan akses informasi yang diinginkannya. Dengan informasi tersebut, manusia dapat mempermudah mencapai keinginannya. Bukan hanya dalam perangkat keras saja tetapi juga perangkat lunaknya. Manusia akan sangat perlu suatu perintah-perintah yang dapat menghubungkan keinginannya dengan komputer dan dunia internet. Untuk menyediakan keinginan manusia dalam bidang informasi dalam bentuk genggaman tangan diperlukan pemrograman pada mobile device untuk sistem operasinya.

Dalam laporan ini, dibahas mengenai pembuatan program aplikasi menggunakan Xamarin, Android Studio, Visual Code, serta Xcode. Aplikasi tersebut digunakan untuk mengembangkan berbagai platform, seperti: *cross platform*, Android, *Progressive Web Application*, dan juga untuk iOS. Setiap jenis *platform* akan dibahas masing-masing dalam bab yang berbeda.

Semua percobaan tersebut akan dibahas dalam laporan praktikum Pemrograman Perangkat Bergerak kali ini, dimana pemrograman perangkat bergerak merupakan pengembangan aplikasi yang ditujukan untuk *mobile device*. Praktikum ini dilakukan sebagai syarat menempuh SKS dan menambah pemahaman seputar pemrograman perangkat bergerak terutama ilmu-ilmu dasarnya.

Kata kunci : *mobile device, pemrograman, android, cross, platform, pwa, ios, xcode, swift*

BAB I

PENDAHULUAN

1.1. LATAR BELAKANG PRAKTIKUM

Praktikum Pemrograman Perangkat Bergerak adalah praktikum wajib yang harus diambil oleh mahasiswa di semester 5 Departemen Teknik Komputer, Fakultas Teknik, Universitas Diponegoro. Praktikum ini merupakan praktikum dari mata kuliah Pemrograman Perangkat Bergerak yang sudah dipelajari pada semester sebelumnya. Praktikum ini dilaksanakan agar mahasiswa dapat mengetahui tentang pemrograman untuk membuat aplikasi pada *mobile device* dengan baik dan jelas.

1.2. TUJUAN PRAKTIKUM

1. Praktikan mampu mengetahui ilmu Pemrograman Perangkat Bergerak dengan baik.
2. Praktikan mampu membuat program sederhana menggunakan bahasa pemrograman yang ada pada modul praktikum.
3. Menambah wawasan mahasiswa mengenai *Mobile Device*.
4. Mengetahui bagaimana menggunakan perangkat lunak untuk pemrograman dan penerapan aplikasinya.
5. Melatih mahasiswa untuk melakukan praktikum selanjutnya di Departemen Teknik Komputer.

1.3. SISTEMATIKA PENULISAN

Laporan ini dibuat dengan sistematika sebagai berikut :

BAB I PENDAHULUAN

- 1.1. Latar Belakang Praktikum
- 1.2. Tujuan Praktikum
- 1.3. Metode Penulisan Laporan

BAB II Xamarin

- 2.1. Tujuan
- 2.2. Dasar Teori
- 2.3. Hasil Percobaan dan Analisa
- 2.4. Tugas Praktikum
- 2.5. Kesimpulan

BAB III Android Studio 1

- 3.1. Tujuan
- 3.2. Dasar Teori
- 3.3. Langkah Kerja
- 3.4. Hasil Percobaan
- 3.5. Tugas Praktikum
- 3.6. Kesimpulan

BAB IV Android Studio 2

- 4.1. Tujuan
- 4.2. Dasar Teori
- 4.3. Langkah Kerja
- 4.4. Hasil Percobaan dan Analisa
- 4.5. Tugas Praktikum
- 4.6. Kesimpulan

BAB V Progressive Web Application 1

- 5.1. Tujuan
- 5.2. Alat dan Bahan
- 5.3. Dasar Teori
- 5.4. Langkah Kerja

5.5. Hasil Percobaan dan Analisa

5.6. Tugas Praktikum

5.7. Kesimpulan

BAB VI Progressive Web Application 2

6.1. Tujuan

6.2. Alat dan Bahan

6.3. Dasar Teori

6.4. Langkah Kerja

6.5. Hasil Percobaan dan Analisa

6.6. Tugas

6.7. Kesimpulan

BAB VII iOS

7.1. Tujuan

7.2. Alat dan Bahan

7.3. Dasar Teori

7.4. Hasil Percobaan dan Analisa

7.5. Kesimpulan

BAB VIII PENUTUP

8.1. Kesimpulan

8.2. Saran

Daftar Pustaka

LAMPIRAN I (Lembar Asistensi)

LAMPIRAN II (Tugas Pendahuluan)

BAB II

Xamarin

2.1 Tujuan

1. Praktikan mampu mengetahui lingkungan pengembangan Xamarin
2. Praktikan memahami struktur *file* Xamarin
3. Praktikan memahami penggunaan Xamarin Form
4. Praktikan dapat memahami bagaimana cara untuk menyimpan data secara lokal menggunakan *database* SQLite
5. Praktikan mampu menerapkan penggunaan Xamarin dan *database* SQLite dalam pembuatan program sederhana.

2.2 Alat dan Bahan

2.2.1 PC/Notebook

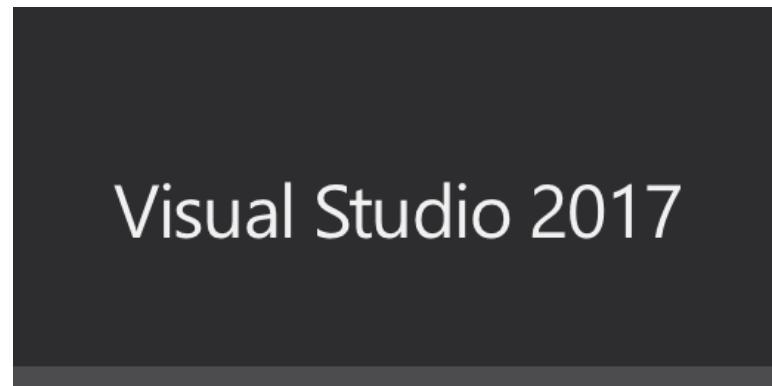
PC/Notebook digunakan untuk menginstall aplikasi Visual Studio Community 2017 dengan spesifikasi minimal yaitu memiliki RAM 4 GB. Hal ini bertujuan agar praktikum dapat berjalan dengan lancar.



Gambar 2.1 PC/Notebook

2.2.2 Visual Studio Community 2017

Visual Studio program untuk membuat aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi console, aplikasi Windows, ataupun aplikasi Web. Visual Studio mencakup kompiler, SDK, Integrated Development Environment (IDE), dan dokumentasi.



Gambar 2.2 Visual Studio Community 2017

2.2.3 Device Android dan Kabel Data

Device yang digunakan pada praktikum BAB II ini adalah perangkat yang berjalan pada sistem operasi Android dengan didukung kabel data USB.



Gambar 2.3 Device Android dan Kabel Data USB

2.2.4 Android SDK

Android SDK (Software Development Kit) adalah satu set alat pengembangan yang digunakan untuk mengembangkan aplikasi untuk platform Android.



Gambar 2.4 Android SDK

2.3 Dasar Teori

2.3.1 Visual Studio

Microsoft Visual Studio merupakan sebuah perangkat lunak lengkap (suite) yang dapat digunakan untuk melakukan pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi console, aplikasi Windows, ataupun aplikasi Web. Visual Studio mencakup kompiler, SDK, Integrated Development Environment (IDE), dan dokumentasi (umumnya berupa MSDN Library). Kompiler yang dimasukkan ke dalam paket Visual Studio antara lain Visual C++, Visual C#, Visual Basic, Visual Basic .NET, Visual InterDev, Visual J++, Visual J#, Visual FoxPro, dan Visual SourceSafe.

Microsoft Visual Studio dapat digunakan untuk mengembangkan aplikasi dalam native code (dalam bentuk bahasa mesin yang berjalan di atas Windows) ataupun managed code(dalam bentuk Microsoft Intermediate Language di atas .NET Framework). Selain itu, Visual Studio juga dapat digunakan untuk mengembangkan aplikasi Silverlight, aplikasi Windows Mobile (yang berjalan di atas .NET Compact Framework).

(Sumber : Modul 1 BAB II Xamarin – Praktikum MDP 2018)

2.3.2 Xamarin

Xamarin adalah platform pengembangan aplikasi mobile dari Microsoft. Dengan Xamarin anda dapat membuat aplikasi mobile cross platform pada perangkat Android, IOS, Windows 10, Windows 8.1, dan Windows Phone. Saat ini terdapat 3 produk Xamarin yang dapat anda pilih yaitu:

- Xamarin for Android
- Xamarin for IOS
- Xamarin Forms

Seperti namanya Xamarin for Android digunakan jika ingin membuat aplikasi native Android. Sedangkan Xamarin for IOS digunakan untuk membuat aplikasi native IOS. Jika menggunakan salah satu atau kedua teknologi tersebut maka dapat membagi komponen yang sudah dibuat pada satu project untuk digunakan di project yang lain. Namun yang dapat dibagi hanya komponen selain UI (User

Interface). Xamarin for Android dan Xamarin for IOS masih menggunakan komponen UI yang spesifik dengan platform.

Xamarin Forms adalah teknologi yang lebih baru, kelebihan dari Xamarin Forms adalah dapat menggunakan satu UI yang dapat di gunakan pada platform yang berbeda.

Arsitektur dari platform Xamarin dapat dilihat pada gambar dibawah ini:



Gambar 2.5 Arsitektur dari platform Xamarin

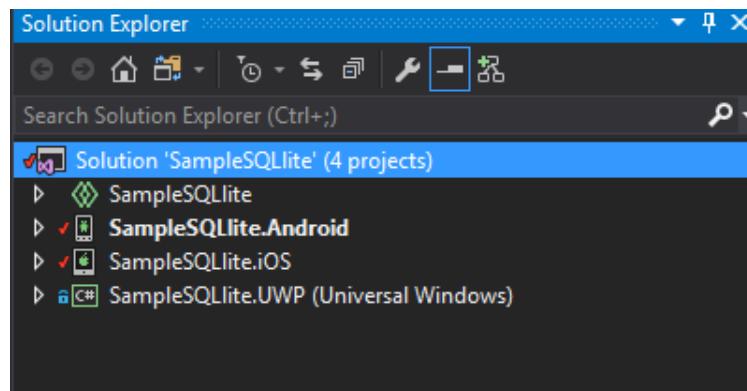
(Sumber : Modul 1 BAB II Xamarin – Praktikum MDP 2018)

2.3.3 Xamarin Form

Xamarin Forms adalah teknologi yang lebih baru, kelebihan dari Xamarin Forms adalah dapat menggunakan satu UI yang dapat di gunakan pada platform yang berbeda. Dengan Xamarin Forms anda cukup mengembangkan satu aplikasi dengan target platform yang berbeda. Xamarin Forms menggunakan format XAML (eXtensible Application Markup Language) untuk membuat tampilan UI. Tampilan solution explorer pada Visual Studio yang berisi 5 project yaitu:

- **Portable:** berisi project utama yang akan dishare secara otomatis ke project yang mempunyai platform spesifik.
- **Droid:** project untuk platform Android.
- **IOS:** project untuk platform IOS.

- **Windows 8.1:** project untuk platform Windows 8.1.
- **Windows Phone 8.1:** project untuk platform Windows Phone 8.1.
- **UWP (Windows 10) –** jika anda menggunakan sistem operasi Windows 10 dan menginstal Windows 10 SDK.



Gambar 2.6 Solution explorer pada Visual Studio

1. Application Lifecycle pada Xamarin

Xamarin Forms mempunyai application lifecycle yang sederhana. Ada tiga method utama yang digunakan untuk menangani method lifecycle, ketiga proses tersebut adalah:

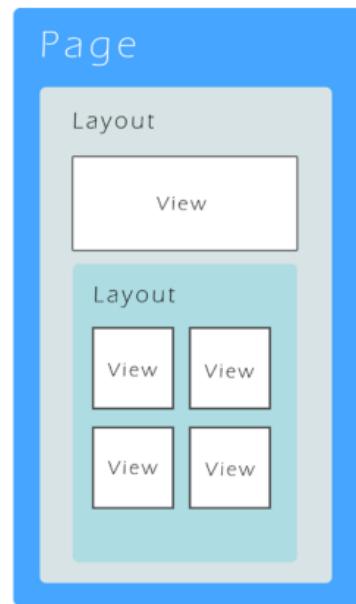
- **OnStart:** method ini akan dipanggil ketika aplikasi pertama kali dijalankan.
- **OnSleep:** method ini akan dipanggil setiap kali app berpindah ke mode background
- **OnResume:** method ini akan dipanggil ketika app masuk ke mode resume setelah sebelumnya masuk ke mode background.

Ketika pengguna memilih tombol Back atau Home pada ponsel, maka aplikasi yang sedang aktif pada saat itu akan masuk ke mode background. Ketika pengguna memilih kembali aplikasi tersebut, maka app tersebut di resume dan app tersebut dibawa kembali ke mode foreground. Saat ketika aplikasi dijalankan pertama kali, saat app masuk ke mode background dan masuk ke mode foreground lagi, sampai app tersebut ditutup (terminate).

2. Xamarin Form UI (User Interface)

Ada tiga komponen utama yang membentuk Xamarin Forms UI yaitu:

- **Page:** Halaman yang digunakan untuk menampilkan view yang disusun menggunakan layout.
- **Views:** komponen kontrol yang ada pada Xamarin Forms seperti *Entry*, *Button*, dll.
- **Layout:** komponen yang digunakan untuk mengatur posisi views pada halaman.



Gambar 2.7 Xamarin Form UI

3. Page

Class Page adalah kontainer utama untuk setiap tampilan pada halaman aplikasi di Xamarin Forms. Class ini diturunkan dari class **Xamarin.Forms.VisualElement**. Class Page adalah baseclass untuk membuat dari semua class UI pada Xamarin Forms. Berikut adalah beberapa contoh Page yang umum digunakan:

- ContentPage
- MasterDetailPage
- NavigationPage
- TabbedPage

- CarouselPage

4. View

View adalah kontrol interaktif yang ada pada Page. Berikut ini adalah jenis-jenis view yang ada pada Xamarin Forms.

- **Basic – fundamental views**
 - Label
 - Image
 - *Button*
 - BoxView
- **List – scrollabe dan selectable list**
 - ListView
- **Text Entry – entry input pengguna menggunakan keyboard**
 - Entry
 - Editor
- **Selection – pilihan pengguna yang lebih dari satu.**
 - Picker
 - DatePicker
 - TimePicker
 - Stepper
 - Slider
 - Switch
- **User Feedback – notifikasi pengguna**
 - Activity

5. Layout

Layout adalah wadah yang digunakan untuk pengaturan posisi kontrol (view, atau layout lain). Ada beberapa macam layout yang didukung oleh Xamarin Form yaitu:

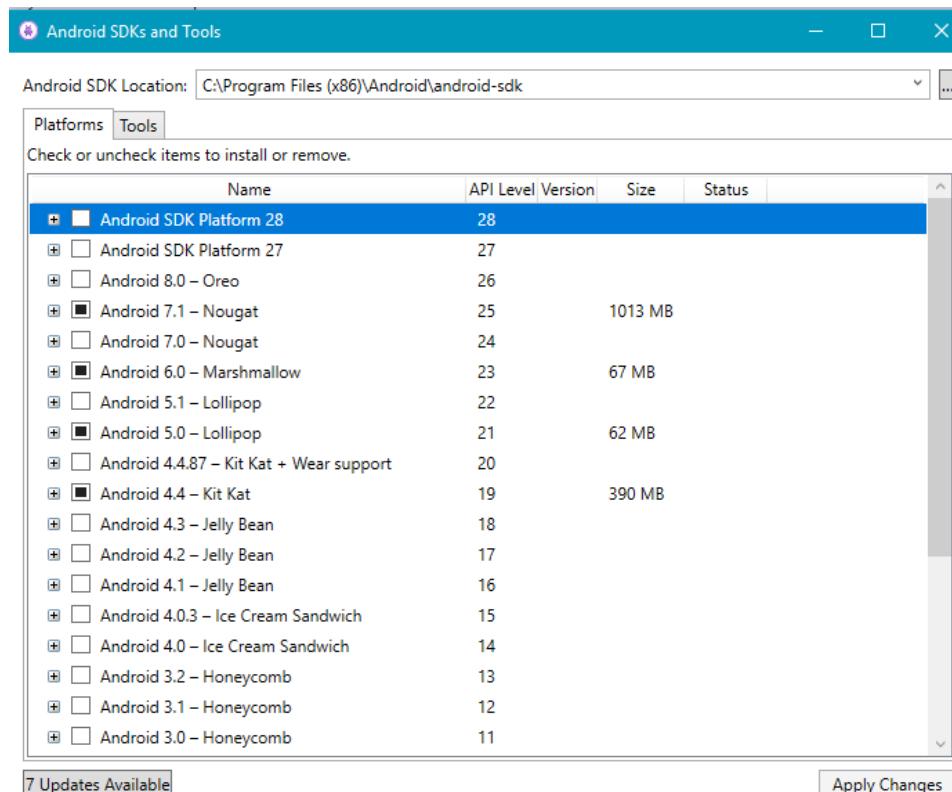
- **StackLayout**: mengatur kontrol secara horisontal atau vertikal.
- **AbsoluteLayout**: pengaturan posisi berdasarkan letak yang pasti.

- **RelativeLayout:** pengaturan posisi kontrol berdasarkan kontrol yang lain.
- **Grid:** membuat layout yang terdiri dari kolom dan baris seperti tabel.

(Sumber : Modul 1 BAB II Xamarin – Praktikum MDP 2018)

2.3.4 SDK Pada Xamarin

Android menggunakan beberapa pengaturan level API Android untuk menentukan kompatibilitas aplikasi yang dibuild di berbagai versi Android. Tergantung pada level Android API yang ditargetkan, sehingga diperlukan mengunduh dan memasang komponen Android SDK tambahan. Selain itu, mungkin perlu menginstal tool opsional dan gambar emulator yang disediakan di Android SDK. Untuk melakukannya, dapat dengan menggunakan Android SDK Manager. Pada Visual Studio 2017 Android SDK Manager dapat disetting dengan mengklik Tools> Android> Android SDK Manager.



Gambar 2.8 SDK pada Xamarin

(Sumber : Modul 1 BAB II Xamarin – Praktikum MDP 2018)

2.3.5 iOS

iOS (sebelumnya iPhone OS) adalah sistem operasi perangkat bergerak yang dikembangkan dan didistribusikan oleh Apple Inc. Sistem operasi ini pertama diluncurkan tahun 2007 untuk iPhone dan iPod Touch, dan telah dikembangkan untuk mendukung perangkat Apple lainnya seperti iPad dan Apple TV.

Tidak seperti IDE lain pada Xamarin.Forms aplikasi yang dibuat dapat kompatibel pada 3 platform sekaligus termasuk salah satunya iOS.



Gambar 2.9 Sistem Operasi iOS

(Sumber : Modul 1 BAB II Xamarin – Praktikum MDP 2018)

2.3.6 SQLite

SQLite merupakan sebuah sistem manajemen basisdata relasional yang bersifat ACID-compliant dan memiliki ukuran pustaka kode yang relatif kecil, ditulis dalam bahasa C. SQLite merupakan proyek yang bersifat public domain yang dikerjakan oleh D. Richard Hipp.

Tidak seperti pada paradigma client-server umumnya, Inti SQLite bukanlah sebuah sistem yang mandiri yang berkomunikasi dengan sebuah program, melainkan sebagai bagian integral dari sebuah program secara keseluruhan. Sehingga protokol komunikasi utama yang digunakan adalah melalui pemanggilan API secara langsung melalui bahasa pemrograman. Mekanisme seperti ini tentunya membawa keuntungan karena dapat mereduksi *overhead, latency times*, dan secara keseluruhan lebih sederhana.

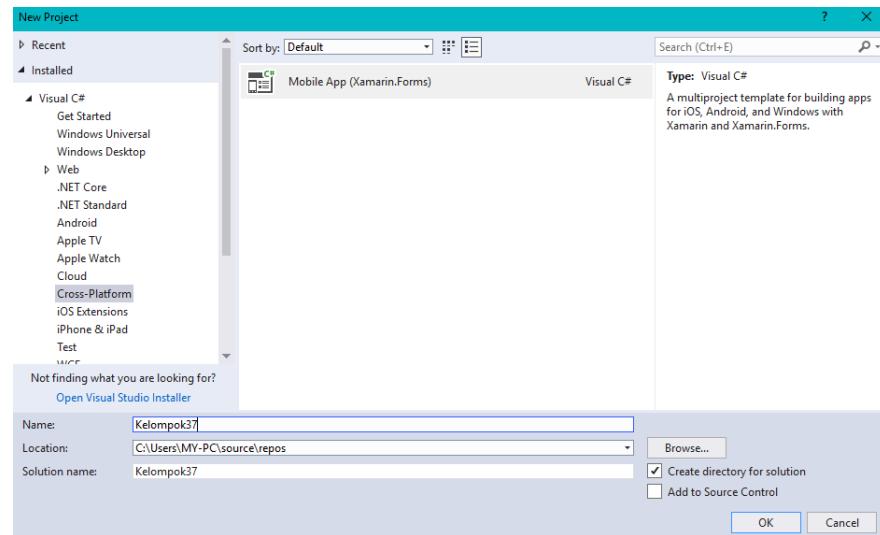
Seluruh elemen basisdata (definisi data, tabel, indeks, dan data) disimpan sebagai sebuah *file*. Kesederhanaan dari sisi disain tersebut bisa diraih dengan cara mengunci keseluruhan *file* basis data pada saat sebuah transaksi dimulai.

(Sumber : Modul 1 BAB II Xamarin – Praktikum MDP 2018)

2.4 Langkah Kerja

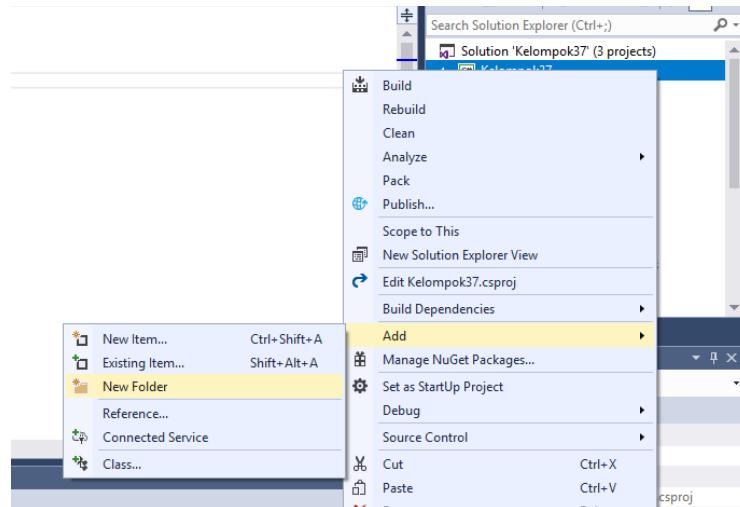
2.4.1 Membuat Login

1. Membuat project baru dengan nama Kelompok37. Pilih Cross-Platform App (Xamarin.Forms).



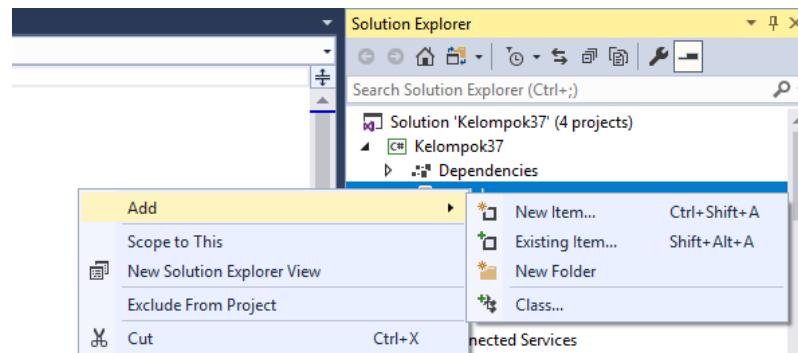
Gambar 2.10 Membuat Project baru

2. Membuat folder View dan Model pada project Xamarin.Forms Kelompok37. Klik kanan pada project Kelompok37 > Add > New Folder

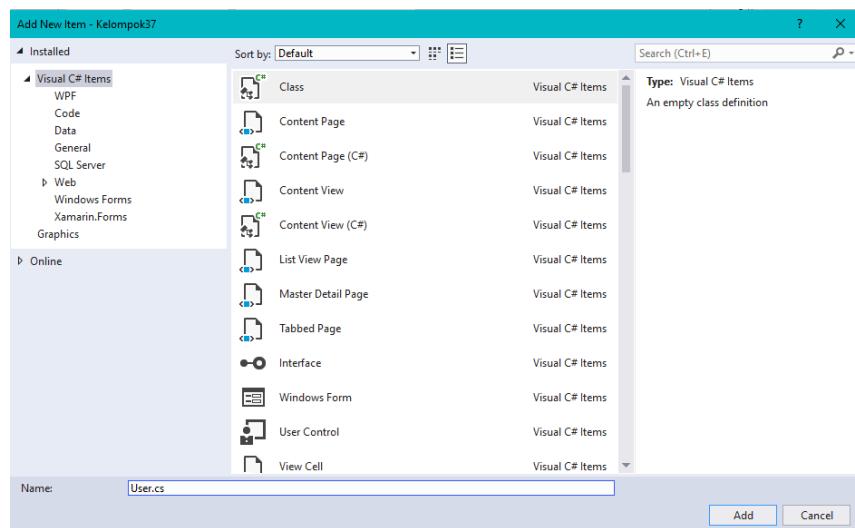


Gambar 2.11 Membuat folder view dan model

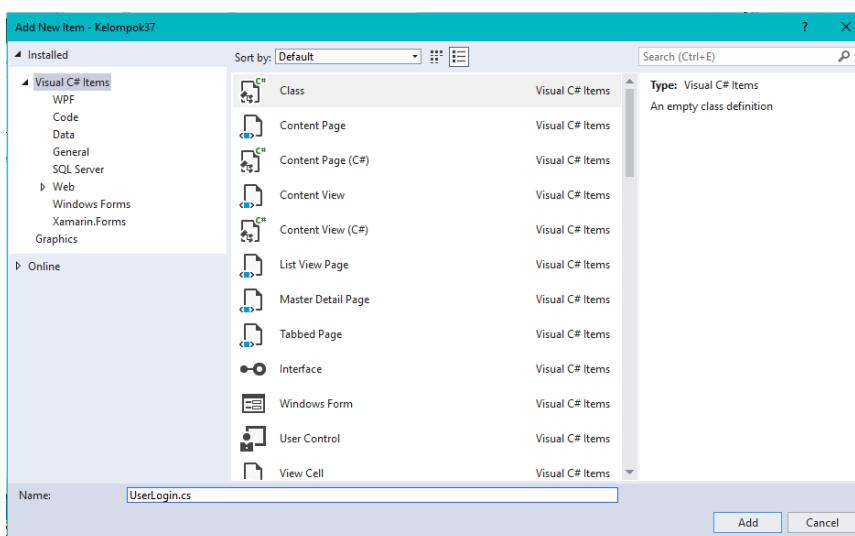
3. Membuat Class User.cs dan UserLogin.cs pada direktori atau folder model yang sudah dibuat sebelumnya. Klik kanan pada folder Model > Add > Class



Gambar 2.12 Cara menambahkan class



Gambar 2.13 Membuat class User.cs



Gambar 2.14 Membuat class UserLogin.cs

4. Masukkan *source code* di bawah ini ke dalam file User.cs. Pada file ini terdapat beberapa parameter yang digunakan untuk melakukan SignUp.

```

using System;
using System.Collections.Generic;
using System.Text;

namespace Kelompok37.Model
{
    public class User
    {
        public string Username { get; set; }

        public string Password { get; set; }

        public string Email { get; set; }
    }
}

```

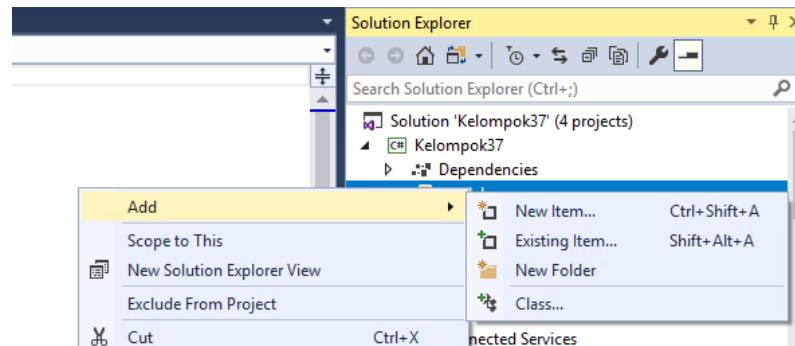
- Masukkan source code berikut ini ke dalam *file* UserLogin.cs. Pada *file* ini terdapat dua variabel yang digunakan sebagai user dan password ketika akan melakukan login ke dalam aplikasi.

```

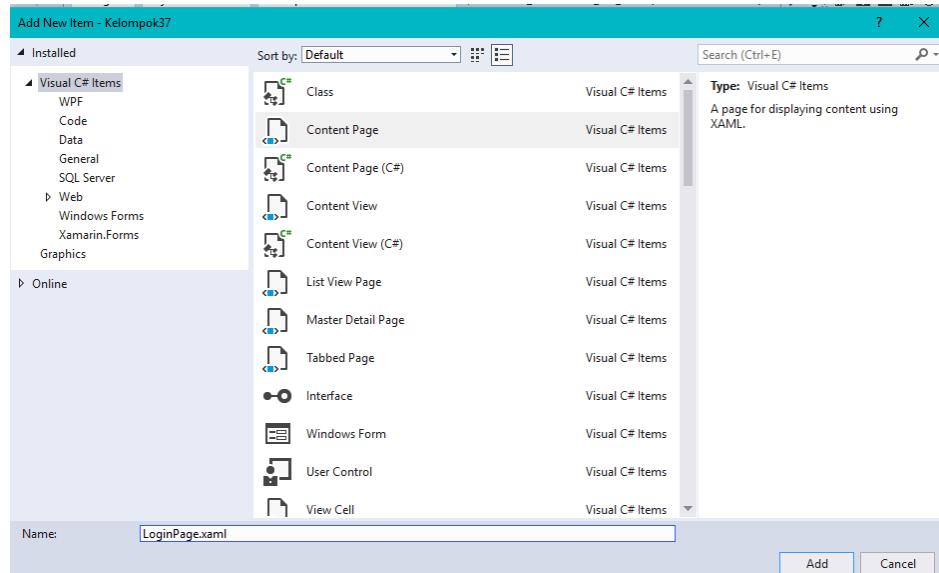
namespace Kelompok37
{
    public static class UserLogin
    {
        public static string Username = "Kelompok37";
        public static string Password = "password";
    }
}

```

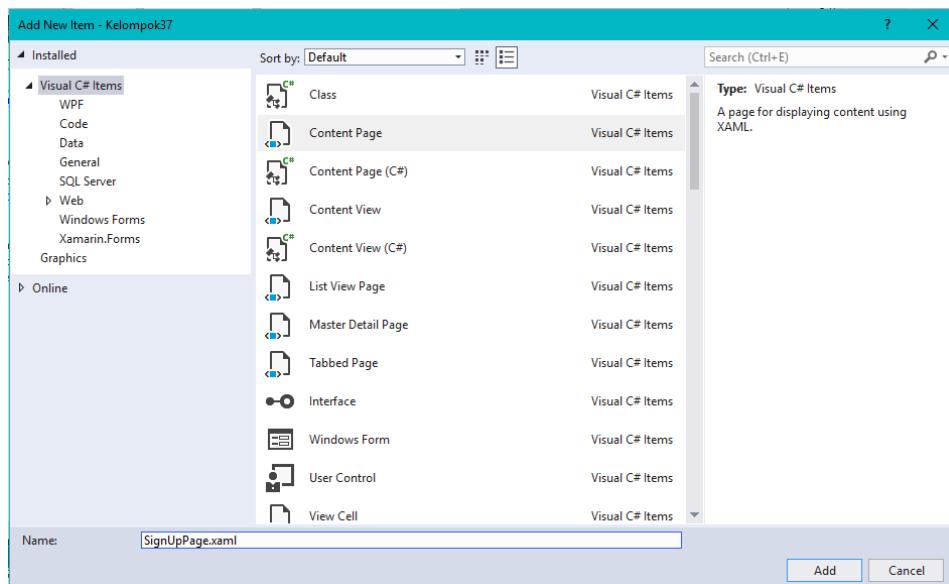
- Membuat Content Page, masing – masing *file* diberi nama LoginPage.xaml, SignUpPage.xaml. Pada *file* Content Page terdapat dua jenis *file* sekaligus yaitu *file* dengan format .xaml dan .cs. Klik kanan pada project Kelompok37 > Add > New item > Content Page



Gambar 2.15 Cara menambahkan New Item



Gambar 2.16 Menambahkan Content Page LoginPage.xaml



Gambar 2.17 Menambahkan Content Page SignUpPage.xaml

7. Masukkan *source code* dibawah ini ke dalam file LoginPage.xaml

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
x:Class="Kelompok37.LoginPage" Title="Login">
    <ContentPage.ToolbarItems>
        <ToolbarItem Text="Sign Up"
Clicked="OnSignUpButtonClicked" />
    </ContentPage.ToolbarItems>
    <ContentPage.Content>
        <StackLayout VerticalOptions="StartAndExpand">
            <Label Text="Username" />
            <Entry x:Name="usernameEntry" />
        </StackLayout>
    </ContentPage.Content>
</ContentPage>
```

```

        Placeholder="Masukkan Username" />
        <Label Text="Password" />
        <Entry x:Name="passwordEntry" IsPassword="true"
Placeholder="Masukkan Password"/>
        <Button Text="Login"
Clicked="OnLoginButtonClicked" />
        <Label x:Name="messageLabel" />
    </StackLayout>
</ContentPage.Content>
</ContentPage>
```

8. Masukkan *source code* dibawah ini ke dalam file LoginPage.xaml.cs

```

using Kelompok37.Model;
using Kelompok37.View;
using System;
using Xamarin.Forms;

namespace Kelompok37
{
    public partial class LoginPage : ContentPage
    {
        public LoginPage()
        {
            InitializeComponent();
        }

        async void OnSignUpButtonClicked(object sender,
EventArgs e)
        {
            await Navigation.PushAsync(new SignUpPage());
        }

        async void OnLoginButtonClicked(object sender,
EventArgs e)
        {
            var user = new User
            {
                Username = usernameEntry.Text,
                Password = passwordEntry.Text
            };

            var isValid = AreCredentialsCorrect(user);
            if (isValid)
            {
                App.IsUserLoggedIn = true;
                Navigation.InsertPageBefore(new
HalamanUtama(), this);
                await Navigation.PopAsync();
            }
            else
            {
                messageLabel.Text = "Login failed";
                passwordEntry.Text = string.Empty;
            }
        }
    }
}
```

```

        bool AreCredentialsCorrect(User user)
        {
            return user.Username == UserLogin.Username &&
user.Password == UserLogin.Password;
        }
    }
}

```

9. Masukkan *source code* berikut kedalam file SignUpPage.xaml

```

<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="Kelompok37.SignUpPage"
    Title="Sign Up">
    <ContentPage.Content>
        <StackLayout VerticalOptions="StartAndExpand">
            <Label Text="Username" />
            <Entry x:Name="usernameEntry"
Placeholder="username" />
            <Label Text="Password" />
            <Entry x:Name="passwordEntry" IsPassword="true"
/>
            <Label Text="Email address" />
            <Entry x:Name="emailEntry" />
            <Button Text="Sign Up"
Clicked="OnSignUpButtonClicked" />
            <Label x:Name="messageLabel" />
        </StackLayout>
    </ContentPage.Content>
</ContentPage>

```

10. Masukkan *source code* berikut ke dalam file SignUpPage.xaml.cs

```

using Kelompok37.Model;
using System;
using System.Linq;
using Xamarin.Forms;

namespace Kelompok37
{
    public partial class SignUpPage : ContentPage
    {
        public SignUpPage()
        {
            InitializeComponent();
        }

        async void OnSignUpButtonClicked(object sender,
EventArgs e)
        {
            var user = new User()
            {
                Username = usernameEntry.Text,
                Password = passwordEntry.Text,
                Email = emailEntry.Text
            };
        }
    }
}

```

```

        // Sign up logic goes here

        var signUpSucceeded = AreDetailsValid(user);
        if (signUpSucceeded)
        {
            var rootPage =
Navigation.NavigationStack.FirstOrDefault();
            if (rootPage != null)
            {
                App.IsUserLoggedIn = true;
                Navigation.InsertPageBefore(new
MainPage(), Navigation.NavigationStack.First());
                await Navigation.PopToRootAsync();
            }
        }
        else
        {
            messageLabel.Text = "Sign up failed";
        }
    }

    bool AreDetailsValid(User user)
    {
        return
(!string.IsNullOrWhiteSpace(user.Username) &&
!string.IsNullOrWhiteSpace(user.Password) &&
!string.IsNullOrWhiteSpace(user.Email) &&
user.Email.Contains("@"));
    }
}
}

```

11. Masukkan *source code* berikut ke dalam file MainPage.xaml

```

<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
x:Class="Kelompok37.MainPage" Title="Main Page">
    <ContentPage.ToolbarItems>
        <ToolbarItem Text="Logout"
Clicked="OnLogoutButtonClicked" />
    </ContentPage.ToolbarItems>
    <ContentPage.Content>
        <StackLayout>
            <Label Text="Main app content goes here"
HorizontalOptions="Center" VerticalOptions="CenterAndExpand" />
        </StackLayout>
    </ContentPage.Content>
</ContentPage>

```

12. Masukkan *source code* berikut ini ke dalam file MainPage.xaml.cs

```

using Kelompok37;
using System;
using Xamarin.Forms;

```

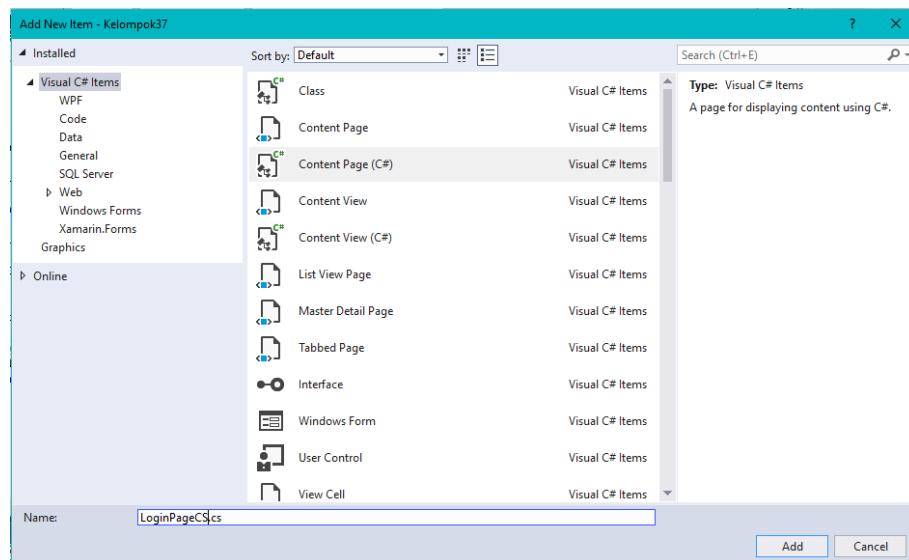
```

namespace Kelompok37
{
    public partial class MainPage : ContentPage
    {
        public MainPage()
        {
            InitializeComponent();
        }

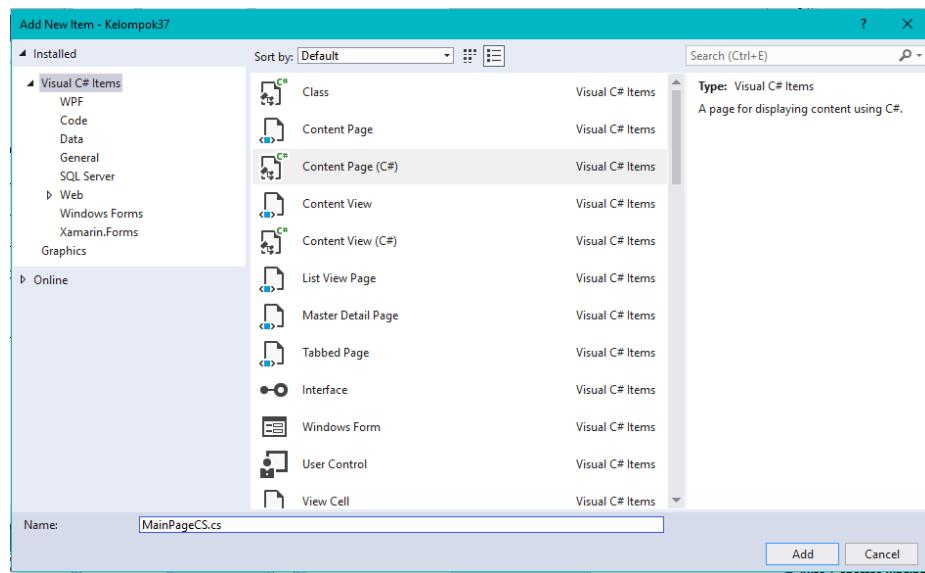
        async void OnLogoutButtonClicked(object sender,
EventArgs e)
        {
            App.IsUserLoggedIn = false;
            Navigation.InsertPageBefore(new LoginPage(),
this);
            await Navigation.PopAsync();
        }
    }
}

```

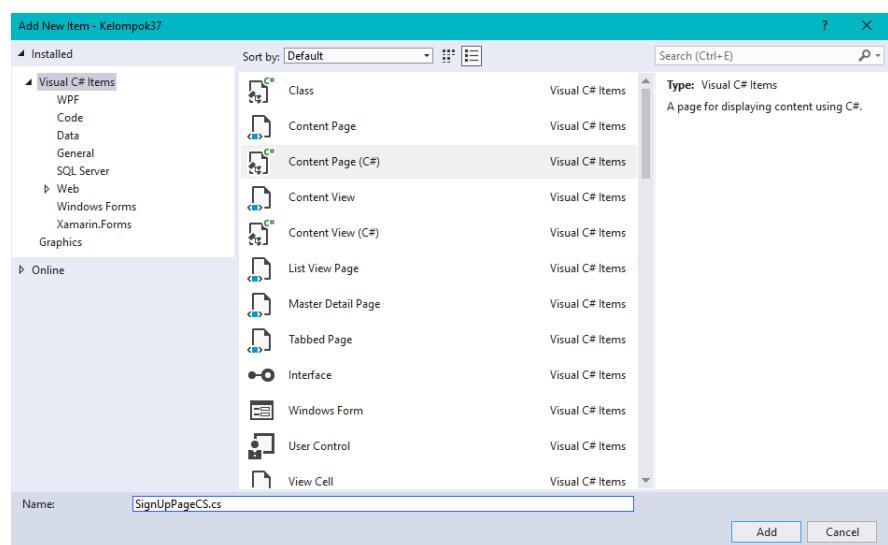
13. Kemudian langkah selanjutnya yaitu membuat Content Page (C#), masing-masing *file* dibuat dan diberi nama dengan LoginPageCS.cs, MainPageCS.cs dan SignUpPageCS.cs. Klik kanan pada project Kelompok37 > Add > New item > Content Page (C#)



Gambar 2.18 Membuat Content Page (C#) LoginPageCS.cs



Gambar 2.19 Membuat Content Page (C#) MainPageCS.cs



Gambar 2.20 Membuat Content Page (C#) SignUpPageCS.cs

14. Masukkan *source code* dibawah ini kedalam file LoginPageCS.cs

```
using Kelompok37.Model;
using System;
using Xamarin.Forms;

namespace Kelompok37
{
    public class LoginPageCS : ContentPage
    {
        Entry usernameEntry, passwordEntry;
        Label messageLabel;

        public LoginPageCS()
        {
```

```
var toolbarItem = new ToolbarItem
{
    Text = "Sign Up"
};
toolbarItem.Clicked += OnSignUpButtonClicked;
ToolbarItems.Add(toolbarItem);

messageLabel = new Label();
usernameEntry = new Entry
{
    Placeholder = "username"
};
passwordEntry = new Entry
{
    IsPassword = true
};
var loginButton = new Button
{
    Text = "Login"
};
loginButton.Clicked += OnLoginButtonClicked;

Title = "Login";
Content = new StackLayout
{
    VerticalOptions =
LayoutOptions.StartAndExpand,
    Children =
    {
        new Label { Text = "Username" },
        usernameEntry,
        new Label { Text = "Password" },
        passwordEntry,
        loginButton,
        messageLabel
    }
};
}

async void OnSignUpButtonClicked(object sender,
EventArgs e)
{
    await Navigation.PushAsync(new SignUpPageCS());
}

async void OnLoginButtonClicked(object sender,
EventArgs e)
{
    var user = new User
    {
        Username = usernameEntry.Text,
        Password = passwordEntry.Text
    };

    var isValid = AreCredentialsCorrect(user);
    if (isValid)
    {
```

```
        App.IsUserLoggedIn = true;
        Navigation.InsertPageBefore(new
MainPageCS(), this);
            await Navigation.PopAsync();
        }
    else
    {
        messageLabel.Text = "Login failed";
        passwordEntry.Text = string.Empty;
    }
}

bool AreCredentialsCorrect(User user)
{
    return user.Username == UserLogin.Username &&
user.Password == UserLogin.Password;
}
}
```

15. Masukkan *source code* dibawah ini kedalam file MainPageCS.cs

```
using System;
using Xamarin.Forms;

namespace Kelompok37
{
    public class MainPageCS : ContentPage
    {
        public MainPageCS()
        {
            var toolbarItem = new ToolbarItem
            {
                Text = "Logout"
            };
            toolbarItem.Clicked += OnLogoutButtonClicked;
            ToolbarItems.Add(toolbarItem);

            Title = "Main Page";
            Content = new StackLayout
            {
                Children = {
                    new Label {
                        Text = "Main app content goes here",
                        HorizontalOptions =
LayoutOptions.Center,
                        VerticalOptions =
LayoutOptions.CenterAndExpand
                    }
                }
            };
        }

        async void OnLogoutButtonClicked(object sender,
EventArgs e)
{
    App.IsUserLoggedIn = false;
}
```

```
        Navigation.InsertPageBefore(new LoginPageCS(),  
this);  
        await Navigation.PopAsync();  
    }  
}  
}
```

16. Masukkan *source code* berikut kedalam *file* SignUpPageCS.cs

```
using Kelompok37;
using Kelompok37.Model;
using System;
using System.Linq;
using Xamarin.Forms;

namespace Kelompok37
{
    public class SignUpPageCS : ContentPage
    {
        Entry usernameEntry, passwordEntry, emailEntry;
        Label messageLabel;

        public SignUpPageCS()
        {
            messageLabel = new Label();
            usernameEntry = new Entry
            {
                Placeholder = "username"
            };
            passwordEntry = new Entry
            {
                IsPassword = true
            };
            emailEntry = new Entry();
            var signUpButton = new Button
            {
                Text = "Sign Up"
            };
            signUpButton.Clicked += OnSignUpButtonClicked;

            Title = "Sign Up";
            Content = new StackLayout
            {
                VerticalOptions =
LayoutOptions.StartAndExpand,
                Children = {
                    new Label { Text = "Username" },
                    usernameEntry,
                    new Label { Text = "Password" },
                    passwordEntry,
                    new Label { Text = "Email address" },
                    emailEntry,
                    signUpButton,
                    messageLabel
                }
            };
        }
    }
}
```

```

        async void OnSignUpButtonClicked(object sender,
EventArgs e)
{
    var user = new User()
    {
        Username = usernameEntry.Text,
        Password = passwordEntry.Text,
        Email = emailEntry.Text
    };

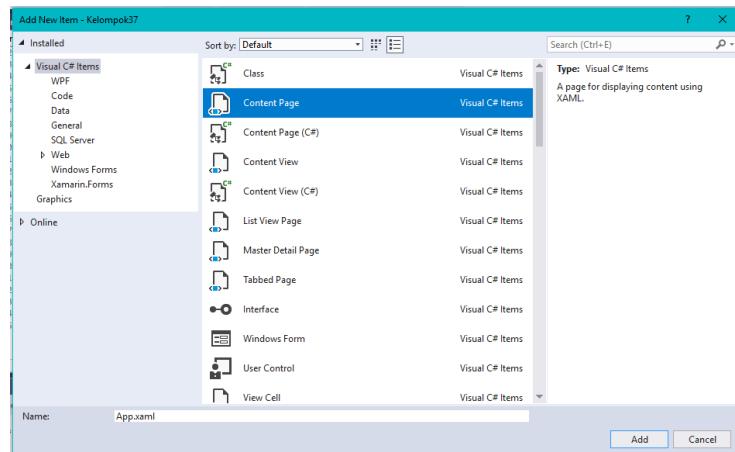
    // Sign up logic goes here

    var signUpSucceeded = AreDetailsValid(user);
    if (signUpSucceeded)
    {
        var rootPage =
Navigation.NavigationStack.FirstOrDefault();
        if (rootPage != null)
        {
            App.IsUserLoggedIn = true;
            Navigation.InsertPageBefore(new
 MainPageCS(), Navigation.NavigationStack.First());
            await Navigation.PopToRootAsync();
        }
    }
    else
    {
        messageLabel.Text = "Sign up failed";
    }
}

bool AreDetailsValid(User user)
{
    return
(!string.IsNullOrWhiteSpace(user.Username) &&
!string.IsNullOrWhiteSpace(user.Password) &&
!string.IsNullOrWhiteSpace(user.Email) &&
user.Email.Contains("@"));
}
}

```

17. Membuat content page dengan nama App.xaml dengan cara klik kanan project Kelompok37 > Add > New Item > Content Page



Gambar 2.21 Membuat Content Page App.xaml

18. Isikan source code berikut pada App.xaml.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Kelompok37.View;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

[assembly: XamlCompilation(XamlCompilationOptions.Compile)]
namespace Kelompok37
{
    public partial class App : Application
    {
        public static bool IsUserLoggedIn { get; internal
set; }
        public App()
        {
            if (!IsUserLoggedIn)
            {
                MainPage = new NavigationPage(new
LoginPage());
            }
            else
            {
                MainPage = new NavigationPage(new
HalamanUtama());
            }
        }

        protected override void OnStart()
        {
            // Handle when your app starts
        }

        protected override void OnSleep()
    }
}

```

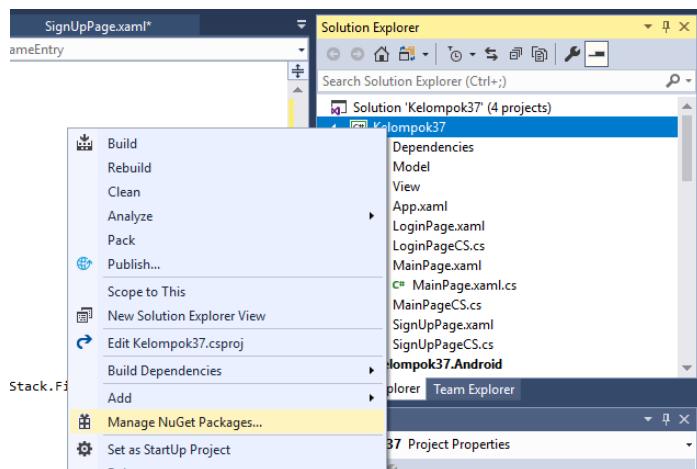
```
        {
            // Handle when your app sleeps
        }

protected override void OnResume()
{
    // Handle when your app resumes
}

}
```

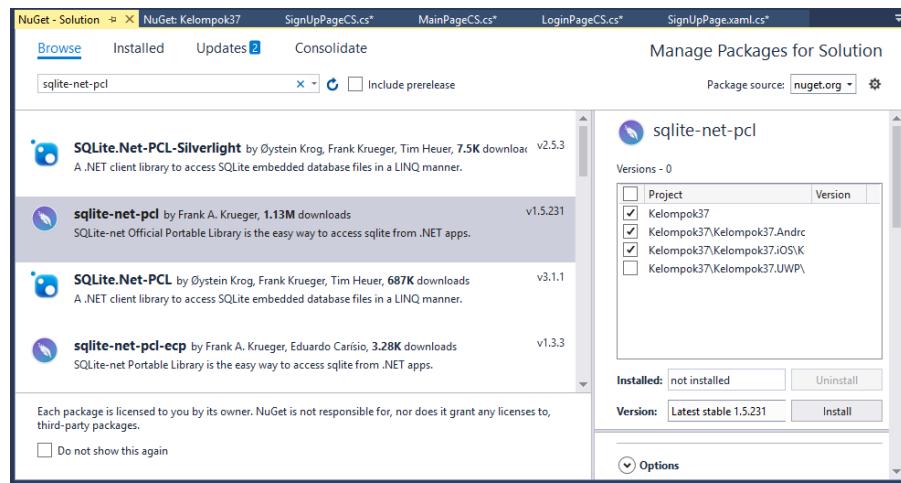
2.4.2 Percobaan SQLite

1. Klik kanan pada project Solution Kelompok37 kemudian pilih menu Manage NuGet Packages for Solution.



Gambar 2.22 Manage NuGet Packages

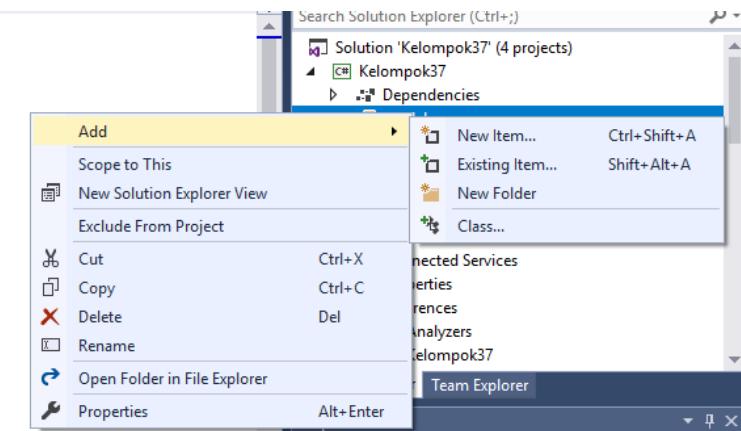
2. Pada tab Browse terdapat kolom pencarian NuGet Packages yang akan dicari. Kemudian ketik “sqlite-net-pcl” setelah itu terdapat checkbox pada project yang akan *diinstall* NuGet Packages yang sudah dipilih. Apabila NuGet Packages sudah *terinstall* maka *button* yang akan muncul yaitu *button Uninstall*.



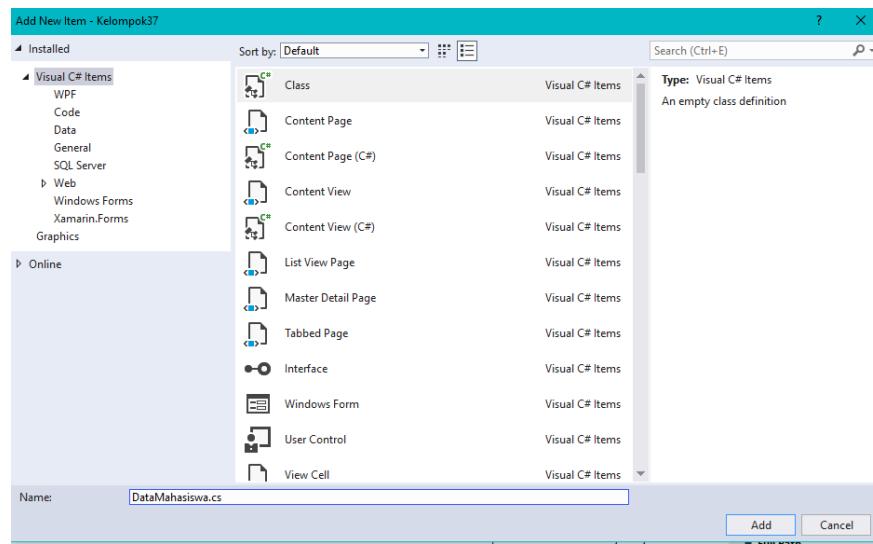
Gambar 2.23 Instalasi NuGet Packages

- Membuat Class baru pada folder Model yang diberi nama DataMahasiswa.cs.

Pada *File* ini terdapat parameter data apa saja yang akan kita gunakan pada aplikasi. Klik kanan pada folder Model > Add > Class



Gambar 2.24 Menambahkan Class baru pada folder Model



Gambar 2.25 Membuat class DataMahasiswa.cs

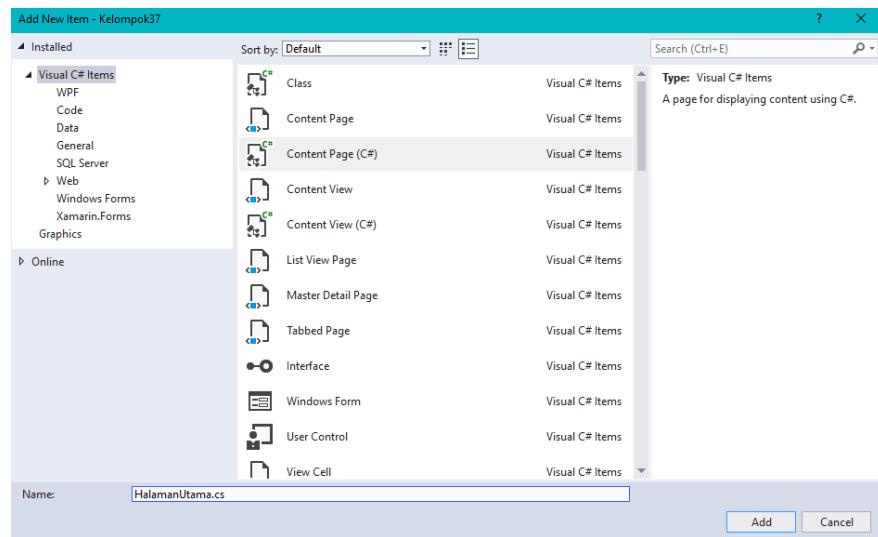
4. Ketik *source code* berikut ke dalam DataMahasiswa.cs

```
using SQLite;
using System;
using System.Collections.Generic;
using System.Text;

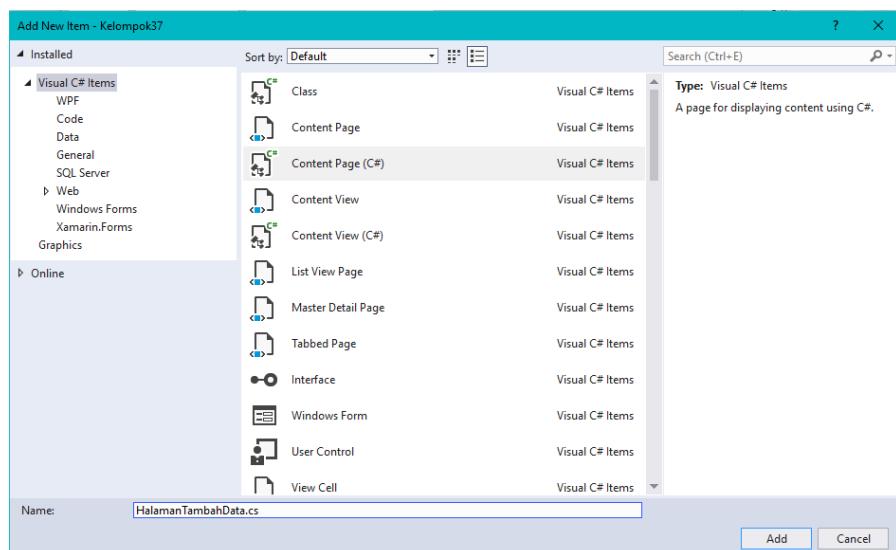
namespace Kelompok37.Model
{
    public class DataMahasiswa
    {
        [PrimaryKey]
        public int Id { get; set; }
        public string Nama { get; set; }
        public string Jurusan { get; set; }

        public override string ToString()
        {
            return "Nama : " + this.Nama + " ==> Jurusan : "
+ this.Jurusan;
        }
    }
}
```

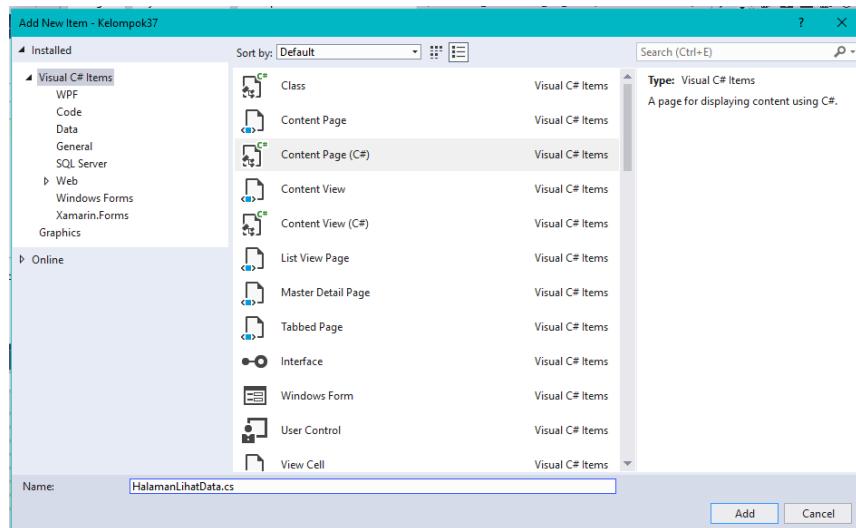
5. Membuat *file Content Page (C#)* pada folder View, masing – masing *file* diberi nama HalamanUtama.cs, HalamanTambahData.cs dan HalamanLihatData.cs. Klik kanan pada folder View> Add > New item > Content Page (C#)



Gambar 2.26 Membuat Content Page (C#) pada folder View dengan nama HalamanUtama.cs



Gambar 2.27 Content Page (C#) pada folder View dengan nama HalamanTambahData.cs



Gambar 2.28 Membuat Content Page (C#) pada folder View dengan nama HalamanLihatData.cs

6. Ketik *source code* berikut ke dalam HalamanUtama.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

using Xamarin.Forms;

namespace Kelompok37.View
{
    public class HalamanUtama : ContentPage
    {
        public HalamanUtama()
        {
            this.Title = "Data Mahasiswa";

            StackLayout stacklayout = new StackLayout();
            Button button = new Button();
            button.Text = "Tambah Data";
            button.Clicked += Button_Tambah_Clicked;
            stacklayout.Children.Add(button);

            button = new Button();
            button.Text = "Lihat Data";
            button.Clicked += Button_Lihat_Clicked;
            stacklayout.Children.Add(button);

            Content = stacklayout;
        }

        private async void Button_Tambah_Clicked(object
sender, EventArgs e)
        {
    
```

```

        await Navigation.PushAsync(new
HalamanTambahData());
    }
    private async void Button_Lihat_Clicked(object sender,
EventArgs e)
{
    await Navigation.PushAsync(new
HalamanLihatData());
}

}
}

```

7. Ketik *source code* berikut ke dalam HalamanTambahData.cs

```

using Kelompok37.Model;
using SQLite;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;

using Xamarin.Forms;

namespace Kelompok37.View
{
    public class HalamanTambahData : ContentPage
    {
        private Entry _nama;
        private Entry _jurusan;
        private Button _simpan;

        string _dbPath =
Path.Combine(System.Environment.GetFolderPath(System.Environment.SpecialFolder.Personal), "myDB.db4");

        public HalamanTambahData()
        {
            this.Title = "Tambah Data Mahasiswa";

            StackLayout stackLayout = new StackLayout();

            _nama = new Entry();
            _nama.Keyboard = Keyboard.Text;
            _nama.Placeholder = "Nama Mahasiswa";
            stackLayout.Children.Add(_nama);

            _jurusan = new Entry();
            _jurusan.Keyboard = Keyboard.Text;
            _jurusan.Placeholder = "Jurusan";
            stackLayout.Children.Add(_jurusan);

            _simpan = new Button();
            _simpan.Text = "Tambah";
            _simpan.Clicked += _simpan_Clicked;
            stackLayout.Children.Add(_simpan);
        }

        void _simpan_Clicked(object sender, EventArgs e)
        {
            Model.Mahasiswa mahasiswa = new Model.Mahasiswa();
            mahasiswa.Nama = _nama.Text;
            mahasiswa.Jurusan = _jurusan.Text;
            Model.SQLite1.SQLite1.TambahData(mahasiswa);
            await Navigation.PushAsync(new HalamanLihatData());
        }
    }
}

```

```
        Content = stackLayout;

    }

    private async void _simpan_Clicked(object sender,
EventArgs e)
{
    var db = new SQLiteConnection(_dbPath);
    db.CreateTable<DataMahasiswa>();

    var maxPk =
db.Table<DataMahasiswa>().OrderByDescending(c =>
c.Id).FirstOrDefault();

    DataMahasiswa dbm = new DataMahasiswa()
{
    Id = (maxPk == null ? 1 : maxPk.Id + 1),
    Nama = _nama.Text,
    Jurusan = _jurusan.Text
};

    db.Insert(dbm);
    await DisplayAlert(null, "Data " + dbm.Nama + " Berhasil Disimpan", "Ok");
    await Navigation.PopAsync();
}
}
```

8. Ketik *source code* berikut ke HalamanLihatData.cs

```
using Kelompok37.Model;
using SQLite;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;

using Xamarin.Forms;

namespace Kelompok37.View
{
    public class HalamanLihatData : ContentPage
    {
        private ListView _listView;
        string _dbPath =
Path.Combine(System.Environment.GetFolderPath(System.Environment.SpecialFolder.Personal), "myDB.db4");

        public HalamanLihatData()
        {
            this.Title = "Data Mahasiswa";

            var db = new SQLiteConnection( dbPath);
```

```

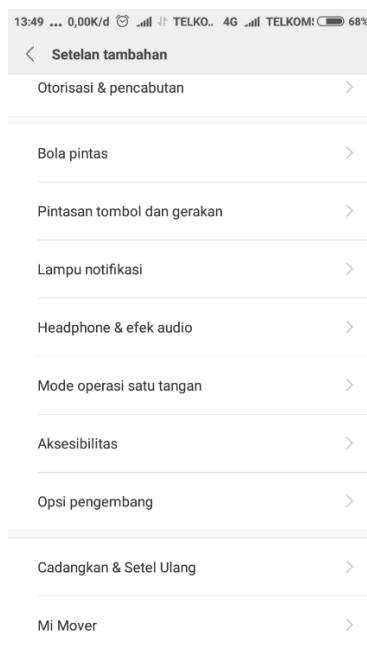
        StackLayout stackLayout = new StackLayout();

        _listView = new ListView();
        _listView.ItemsSource =
db.Table<DataMahasiswa>().OrderBy(x => x.Nama).ToList();
        stackLayout.Children.Add(_listView);

        Content = stackLayout;
    }
}
}

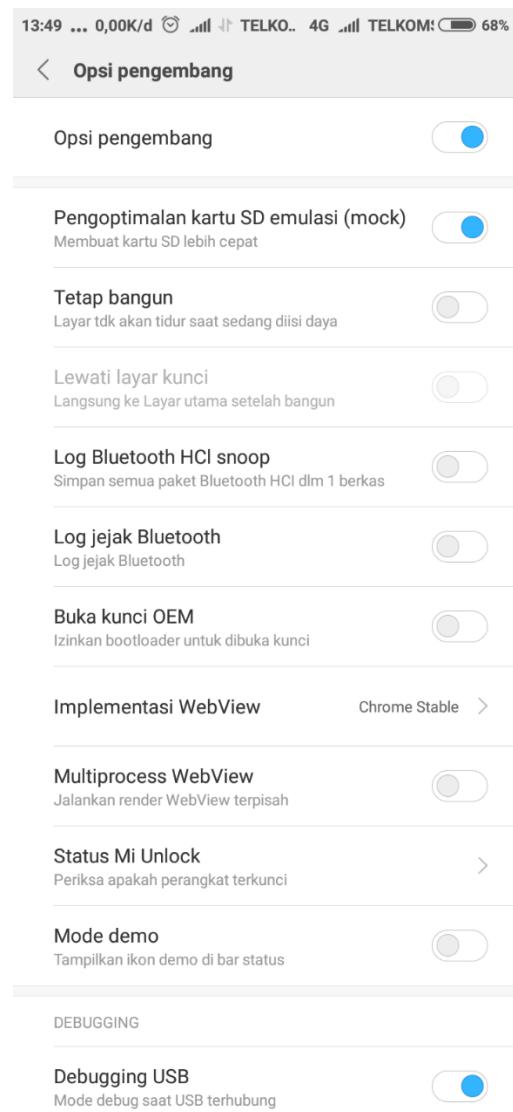
```

9. Penjelasan __: Pada file HalamanUtama.cs, HalamanTambahData.cs dan HalamanLihatData.cs diatas terdapat variabel `_dbpath` yang di dalamnya merupakan *database* lokal dengan nama **myDB.db4**. File database tersebut akan disimpan kedalam folder pada device android anda yaitu pada folder **/data/data/[your.package.name]/files**.
10. Sebelum melakukan Build Project dan Deploy Project pastikan *smartphone* sudah terdapat menu Developer options.



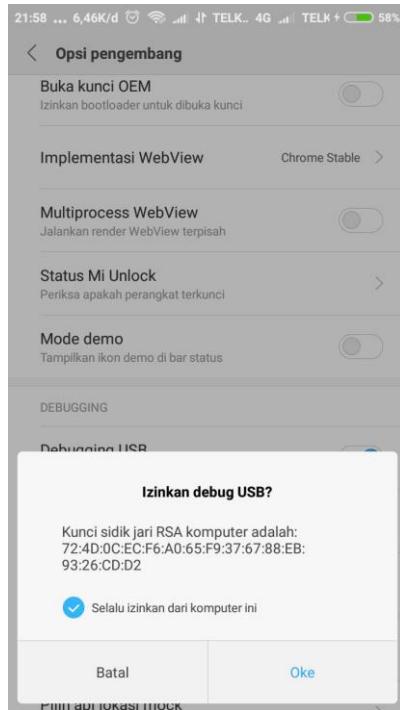
Gambar 2.29 Memilih Opsi Pengembang pada Android Phone

11. Pada menu Developer options checklist pada checkbox USB debugging



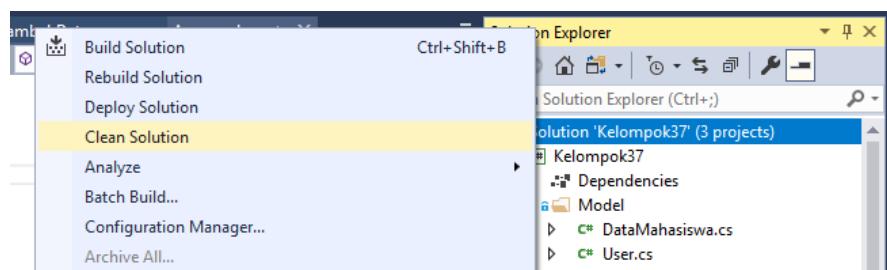
Gambar 2.30 Mengaktifkan Debugging USB

12. Ketika *smartphone* sudah terhubung melalui kabel USB dengan PC atau Laptop maka akan muncul dialog seperti berikut. Pilih OK



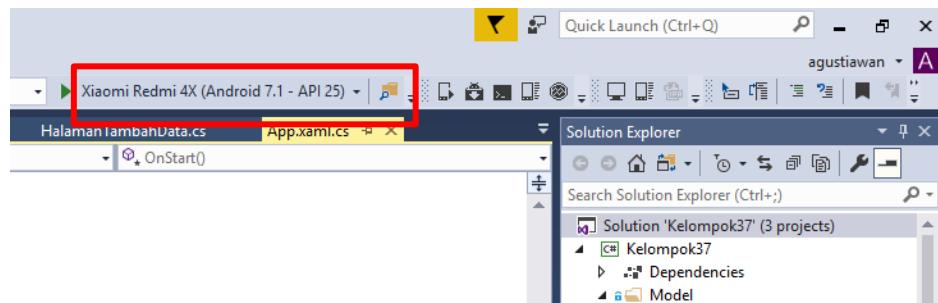
Gambar 2.31 Mengaktifkan Debugging USB

13. Kemudian sebelum membuild project maka lakukan Clean Solution terlebih dahulu pada Project Solution Kelompok37.



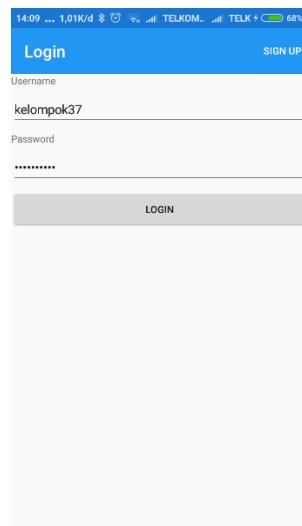
Gambar 2.32 Clean Solution

14. Apabila *smartphone* sudah terhubung dengan PC atau Laptop maka akan terdeteksi seperti pada gambar dibawah ini. Untuk melakukan build project sekaligus dploy project dapat dilakukan dengan langsung menekan tombol play.



Gambar 2.33 Tampilan Pada Visual Studio 2017 ketika *smartphone* telah terhubung

15. Apabila aplikasi sudah sukses dibuild dan *dideploy* maka akan muncul tampilan seperti pada gambar dibawah ini.



Gambar 2.34 Tampilan awal pada *smartphone* ketika program di *deploy*

2.5. Analisa Hasil Percobaan

2.5.1. Percobaan 1

Pada percobaan 1, praktikan membuat login pada aplikasi Xamarin. Sebelum program di *deploy* ke *smartphone*, *smartphone* sudah harus dalam keadaan USB Debugging agar program dapat *diinstall* menggunakan kabel USB .

Sebelum melakukan *deploy* program, dilakukan *Clean Project* terlebih dahulu. Setelah program di *deploy* pada *smartphone*, maka aplikasi akan otomatis terbuka dan menampilkan halaman *login*. Pada halaman ini user diminta untuk memasukkan data username dan password. Namun jika belum mempunyai akun maka user dapat membuat akun terlebih dahulu pada *button Sign Up* di samping kanan atas layar *smartphone*. Di halaman *Sign Up*, *user* diminta memasukkan *username*, *password*, dan alamat *email*.

Setelah *login* berhasil maka akan ditampilkan halaman utama yang berisi *button Tambah Data* dan *Lihat Data*. *Button Tambah Data* digunakan untuk menambahkan data mahasiswa berupa nama dan jurusan. Sedangkan *button Lihat Data* digunakan untuk melihat data yang sudah dimasukkan.

Berikut ini adalah *sourcecode* percobaan *login* :

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Kelompok37.Model
{
    public class User
    {
        public string Username { get; set; }

        public string Password { get; set; }

        public string Email { get; set; }
    }
}
```

Dapat dilihat pada *sourcecode* diatas bahwa *user* memiliki beberapa parameter atau data yaitu *Username*, *Password*, dan *Email*.

```
namespace Kelompok37
{
    public static class UserLogin
    {
        public static string Username = "Kelompok37";
        public static string Password = "password";
```

```
}
```

Sourcecode diatas digunakan untuk *login*. Untuk masuk ke halaman utama, maka user dapat memasukkan username berupa “Kelompok37” dan mengisi password yaitu “password”.

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
x:Class="Kelompok37.LoginPage" Title="Login">
    <ContentPage.ToolbarItems>
        <ToolbarItem Text="Sign Up"
Clicked="OnSignUpButtonClicked" />
    </ContentPage.ToolbarItems>
    <ContentPage.Content>
        <StackLayout VerticalOptions="StartAndExpand">
            <Label Text="Username" />
            <Entry x:Name="usernameEntry" Placeholder="Masukkan Username" />
            <Label Text="Password" />
            <Entry x:Name="passwordEntry" IsPassword="true"
Placeholder="Masukkan Password"/>
            <Button Text="Login" Clicked="OnLoginButtonClicked" />
            <Label x:Name="messageLabel" />
        </StackLayout>
    </ContentPage.Content>
</ContentPage>
```

Sourcecode diatas merupakan isi dari *file* LoginPage.xaml. XAML diatas digunakan untuk menentukan konten visual dari halaman *Login*. XAML sendiri menentukan konten visual suatu halaman dan bekerja sama dengan *file* C# di belakang kode.

Ketika user melakukan klik pada *button* Login maka perintah tersebut akan diproses pada LoginPage.xaml.cs. Berikut adalah *sourcecode* LoginPage.xaml.cs :

```
using Kelompok37.Model;
using Kelompok37.View;
using System;
using Xamarin.Forms;

namespace Kelompok37
{
    public partial class LoginPage : ContentPage
    {
        public LoginPage()
        {
            InitializeComponent();
        }
    }
}
```

```

        async void OnSignUpButtonClicked(object sender, EventArgs e)
    {
        await Navigation.PushAsync(new SignUpPage());
    }

        async void OnLoginButtonClicked(object sender, EventArgs e)
    {
        var user = new User
        {
            Username = usernameEntry.Text,
            Password = passwordEntry.Text
        };

        var isValid = AreCredentialsCorrect(user);
        if (isValid)
        {
            App.IsUserLoggedIn = true;
            Navigation.InsertPageBefore(new HalamanUtama(), this);
            await Navigation.PopAsync();
        }
        else
        {
            messageLabel.Text = "Login failed";
            passwordEntry.Text = string.Empty;
        }
    }

    bool AreCredentialsCorrect(User user)
    {
        return user.Username == UserLogin.Username &&
user.Password == UserLogin.Password;
    }
}
}

```

Sourcecode diatas digunakan untuk menentukan apakah username dan password yang dimasukkan user benar atau salah. Jika username dan atau password yang dimasukkan user salah maka akan muncul *messageLabel* “Login Failed” dibawah *button* Login. Namun apabila *username* dan *password* yang dimasukkan user benar, maka akan ditampilkan Halamna Utama.

Sebelumnya, jika user ingin membuat akun baru dapat membuat akun melalui *Button Sign Up* yang terdapat pada pojok kanan atas layar *smartphone*. Berikut adalah *sourcecode* *SignUp.xaml* yang digunakan untuk menentukan konten visual pada halaman Sign Up.

```

<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"

```

```

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="Kelompok37.SignUpPage"
    Title="Sign Up">
<ContentPage.Content>
    <StackLayout VerticalOptions="StartAndExpand">
        <Label Text="Username" />
        <Entry x:Name="usernameEntry" Placeholder="username" />
        <Label Text="Password" />
        <Entry x:Name="passwordEntry" IsPassword="true" />
        <Label Text="Email address" />
        <Entry x:Name="emailEntry" />
        <Button Text="Sign Up" Clicked="OnSignUpButtonClicked" />
        <Label x:Name="messageLabel" />
    </StackLayout>
</ContentPage.Content>
</ContentPage>

```

Pada halaman Sign Up, user dapat memasukkan data *username*, *password*, dan alamat email. Pada bagian bawah juga terdapat *button* Sign Up untuk menuju ke proses selanjutnya. Proses back-end halaman Sign Up adalah sebagai berikut sebagaimana *sourcecode* pada SignUpPage.xaml.cs :

```

using Kelompok37.Model;
using System;
using System.Linq;
using Xamarin.Forms;

namespace Kelompok37

{
    public partial class SignUpPage : ContentPage
    {
        public SignUpPage()
        {
            InitializeComponent();
        }

        async void OnSignUpButtonClicked(object sender, EventArgs e)
        {
            var user = new User()
            {
                Username = usernameEntry.Text,
                Password = passwordEntry.Text,
                Email = emailEntry.Text
            };

            // Sign up logic goes here

            var signUpSucceeded = AreDetailsValid(user);
            if (signUpSucceeded)
            {

```

```

        var             rootPage
Navigation.NavigationStack.FirstOrDefault();
        if (rootPage != null)
        {
            App.IsUserLoggedIn = true;
            Navigation.InsertPageBefore(new MainPage(),
Navigation.NavigationStack.First());
            await Navigation.PopToRootAsync();
        }
    }
    else
    {
        messageLabel.Text = "Sign up failed";
    }
}

bool AreDetailsValid(User user)
{
    return (!string.IsNullOrWhiteSpace(user.Username) &&
!string.IsNullOrWhiteSpace(user.Password) &&
!string.IsNullOrWhiteSpace(user.Email) &&
user.Email.Contains("@"));
}
}
}

```

Setelah memasukkan *username*, password, dan alamat email dan user melakukan klik *button Sign Up* maka akan dilakukan pengecekan menggunakan pengkondisian, jika terdapat salah satu parameter yang belum diisi maka akan muncul message label “Sign Up Failed” namun tetap berada pada halaman Sign Up. Namun jika data sudah dimasukkan semua maka proses akan menuju halaman berikutnya yaitu Main page.

```

<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
x:Class="Kelompok37.MainPage" Title="Main Page">
    <ContentPage.ToolbarItems>
        <ToolbarItem Text="Logout" Clicked="OnLogoutButtonClicked"/>
    </ContentPage.ToolbarItems>
    <ContentPage.Content>
        <StackLayout>
            <Label Text="Main app content goes here"
HorizontalOptions="Center" VerticalOptions="CenterAndExpand" />
        </StackLayout>
    </ContentPage.Content>
</ContentPage>

```

Sourcecode diatas adalah *sourcecode* MainPage.xaml yang digunakan untuk mengatur tampilan ketika user telah berhasil melakukan sign up. *File xaml* ini diikuti oleh MainPage.xaml.cs sebagai berikut :

```

using Kelompok37;
using System;
using Xamarin.Forms;

namespace Kelompok37
{
    public partial class MainPage : ContentPage
    {
        public MainPage()
        {
            InitializeComponent();
        }

        async void OnLogoutButtonClicked(object sender, EventArgs e)
        {
            App.IsUserLoggedIn = false;
            Navigation.InsertPageBefore(new LoginPage(), this);
            await Navigation.PopAsync();
        }
    }
}

```

File diatas digunakan untuk melakukan kondisi pada halaman main page. Halaman main page sendiri akan muncul ketika user telah melakukan sign up. Pada halaman main page, user hanya bisa melakukan log out dan ketika user telah melakukan log out maka user harus melakukan proses login lagi.

Karena melakukan proses login lagi, maka masuk ke halaman LoginPageCS.cs. Berikut adalah *listing code* LoginPageCS.cs :

```

using Kelompok37.Model;
using System;
using Xamarin.Forms;

namespace Kelompok37
{
    public class LoginPageCS : ContentPage
    {
        Entry usernameEntry, passwordEntry;
        Label messageLabel;

        public LoginPageCS()
        {
            var toolbarItem = new ToolbarItem
            {
                Text = "Sign Up"
            };
            toolbarItem.Clicked += OnSignUpButtonClicked;
            ToolbarItems.Add(toolbarItem);

            messageLabel = new Label();
            usernameEntry = new Entry
            {

```

```

        Placeholder = "username"
    };
    passwordEntry = new Entry
    {
        IsPassword = true
    };
    var loginButton = new Button
    {
        Text = "Login"
    };
    loginButton.Clicked += OnLoginButtonClicked;

    Title = "Login";
    Content = new StackLayout
    {
        VerticalOptions = LayoutOptions.StartAndExpand,
        Children = {
            new Label { Text = "Username" },
            usernameEntry,
            new Label { Text = "Password" },
            passwordEntry,
            loginButton,
            messageLabel
        }
    };
}

async void OnSignUpButtonClicked(object sender, EventArgs e)
{
    await Navigation.PushAsync(new SignUpPageCS());
}

async void OnLoginButtonClicked(object sender, EventArgs e)
{
    var user = new User
    {
        Username = usernameEntry.Text,
        Password = passwordEntry.Text
    };

    var isValid = AreCredentialsCorrect(user);
    if (isValid)
    {
        App.IsUserLoggedIn = true;
        Navigation.InsertPageBefore(new MainPageCS(),
this);
        await Navigation.PopAsync();
    }
    else
    {
        messageLabel.Text = "Login failed";
        passwordEntry.Text = string.Empty;
    }
}
}

```

```

        bool AreCredentialsCorrect(User user)
    {
        return user.Username == UserLogin.Username &&
user.Password == UserLogin.Password;
    }
}
}

```

Sourcecode diatas berfungsi untuk mengatur fungsi-fungsi dan tampilan pada halaman *login*. Berdasarkan source diatas, yang dilakukan pertama kali yaitu mengatur bentuk *layout* halaman *login*. Halaman ini diberi judul Login, terdapat label untuk memasukkan *username* dan password, serta juga terdapat *button* dengan tulisan Login. Kemudian atur fungsi-fungsi yang dibutuhkan untuk menjalaankan halaman *login*. User diminta untuk memasukkan username dan password yang sudah dibuat sebelumnya. Jika *username* dan password yang dimasukkan benar, maka user akan diarahkan ke *main page*. Tetapi apabila data yang dimasukkan salah, maka akan muncul pesan “Login Failed” dan diminta untuk memasukkan username dan password lagi.

```

using Kelompok37;
using Kelompok37.Model;
using System;
using System.Linq;
using Xamarin.Forms;

namespace Kelompok37
{
    public class SignUpPageCS : ContentPage
    {
        Entry usernameEntry, passwordEntry, emailEntry;
        Label messageLabel;

        public SignUpPageCS()
        {
            messageLabel = new Label();
            usernameEntry = new Entry
            {
                Placeholder = "username"
            };
            passwordEntry = new Entry
            {
                IsPassword = true
            };
            emailEntry = new Entry();
            var signUpButton = new Button
            {
                Text = "Sign Up"
            };
            signUpButton.Clicked += OnSignUpButtonClicked;
        }
    }
}

```

```

        Title = "Sign Up";
        Content = new StackLayout
        {
            VerticalOptions = LayoutOptions.StartAndExpand,
            Children = {
                new Label { Text = "Username" },
                usernameEntry,
                new Label { Text = "Password" },
                passwordEntry,
                new Label { Text = "Email address" },
                emailEntry,
                signUpButton,
                messageLabel
            }
        };
    }

    async void OnSignUpButtonClicked(object sender, EventArgs
e)
{
    var user = new User()
    {
        Username = usernameEntry.Text,
        Password = passwordEntry.Text,
        Email = emailEntry.Text
    };

    // Sign up logic goes here

    var signUpSucceeded = AreDetailsValid(user);
    if (signUpSucceeded)
    {
        var rootPage = Navigation.NavigationStack.FirstOrDefault();
        if (rootPage != null)
        {
            App.IsUserLoggedIn = true;
            Navigation.InsertPageBefore(new MainPageCS(),
Navigation.NavigationStack.First());
            await Navigation.PopToRootAsync();
        }
    }
    else
    {
        messageLabel.Text = "Sign up failed";
    }
}

bool AreDetailsValid(User user)
{
    return (!string.IsNullOrWhiteSpace(user.Username) &&
!string.IsNullOrWhiteSpace(user.Password) &&
!string.IsNullOrWhiteSpace(user.Email) &&
user.Email.Contains("@"));
}

```

```

    }
}
```

Source code tersebut merupakan *sourcecode* pada SignUpCS.cs yang berfungsi untuk mengatur tampilan halaman sign up dan juga mengatur fungsi-fungsi yang diinginkan. Halaman ini berjudul Sign Up, terdapat bagian dimana user dapat memasukkan nama, password, dan email. Dibawah bagian untuk menulis data tersebut terdapat *button* Sign Up. Jika data yang diisi sesuai maka akan beralih ke halaman main page. Namun jika gagal maka akan muncul label “Sign Up failed”.

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Kelompok37.View;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

[assembly: XamlCompilation(XamlCompilationOptions.Compile)]
namespace Kelompok37
{
    public partial class App : Application
    {
        public static bool IsUserLoggedIn { get; internal set; }
        public App()
        {
            if (!IsUserLoggedIn)
            {
                MainPage = new NavigationPage(new LoginPage());
            }
            else
            {
                MainPage = new NavigationPage(new HalamanUtama());
            }
        }

        protected override void OnStart()
        {
            // Handle when your app starts
        }

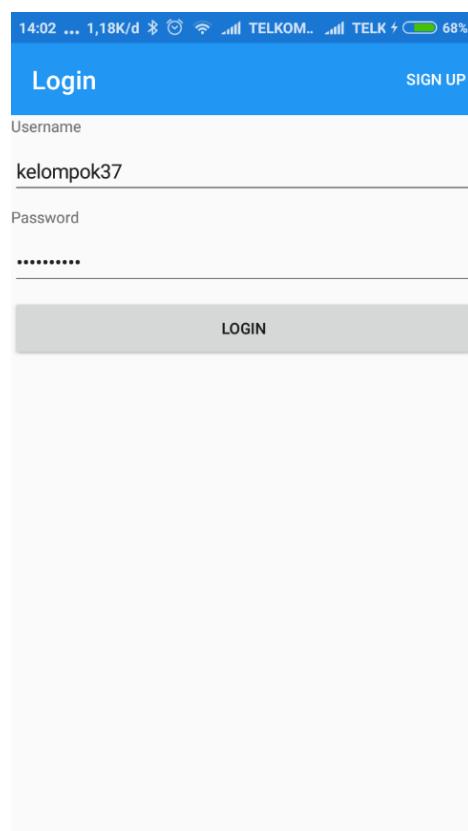
        protected override void OnSleep()
        {
            // Handle when your app sleeps
        }

        protected override void OnResume()
        {
            // Handle when your app resumes
        }
    }
}
```

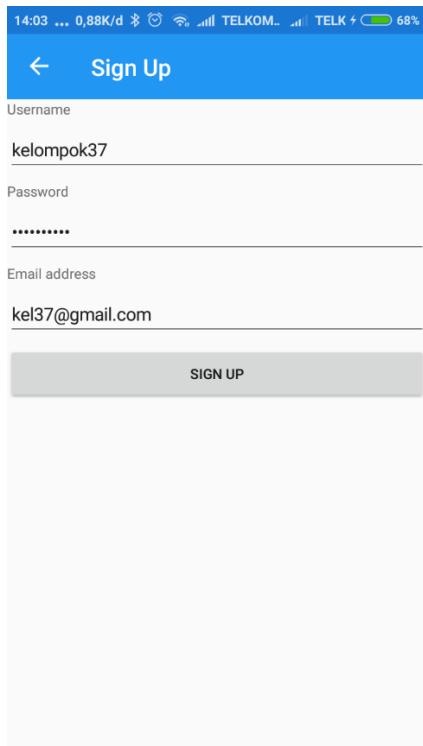
```
        }  
    }
```

Sourcecode diatas merupakan *sourcecode* App.xaml.cs yang digunakan untuk mengatur class yang akan dijalankan pertama kali saat aplikasi dijalankan.

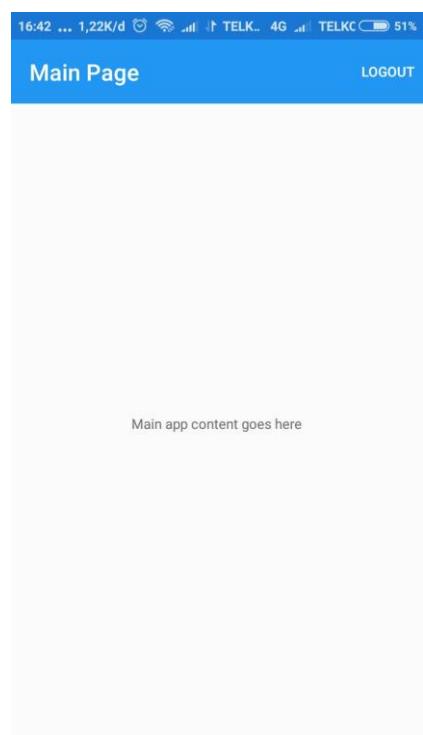
Berikut adalah screenchot percobaan 1 :



Gambar 2.35 Halaman Login



Gambar 2.36 Halaman SignUp



Gambar 2.37 Halaman Main Page

2.5.2 Percobaan 2

Pada percobaan 2, dilakukan pembuatan aplikasi Xamarin dengan memanfaatkan *database* SQLite. SQLite didownload dan diinstall terlebih dahulu sebelumnya.

Berikut adalah *sourcecode* class DataMahasiswa yang dibuat pada folder model. Pada class ini terdapat beberapa parameter pada class ini antara lain ID, Nama, dan Jurusan.

```
using SQLite;
using System;
using System.Collections.Generic;
using System.Text;

namespace Kelompok37.Model
{
    public class DataMahasiswa
    {
        [PrimaryKey]
        public int Id { get; set; }
        public string Nama { get; set; }
        public string Jurusan { get; set; }

        public override string ToString()
        {
            return "Nama : " + this.Nama + " ==> Jurusan : " +
this.Jurusan;
        }
    }
}
```

Kemudian terdapat class HalamanUtama.cs yang digunakan untuk membuat halaman utama dengan judul “Data Mahasiswa”. Pada class ini terdapat pengaturan untuk mengatur layout tampilan dan juga terdapat *button* Tambah Data dan Lihat Data. Apabila user menekan *button* Tambah Data maka user akan diarahkan ke halaman tambah data dan menambahkan data yang diinginkan. Apabila menekan tombol Lihat Data maka user akan diarahkan ke halaman lihat data dan dapat melihat data apa saja yang ada pada halaman lihat data. Berikut adalah *sourcecode* HalamanUtama.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

using Xamarin.Forms;
```

```

namespace Kelompok37.View
{
    public class HalamanUtama : ContentPage
    {
        public HalamanUtama()
        {
            this.Title = "Data Mahasiswa";

            StackLayout stacklayout = new StackLayout();
            Button button = new Button();
            button.Text = "Tambah Data";
            button.Clicked += Button_Tambah_Clicked;
            stacklayout.Children.Add(button);

            button = new Button();
            button.Text = "Lihat Data";
            button.Clicked += Button_Lihat_Clicked;
            stacklayout.Children.Add(button);

            Content = stacklayout;
        }

        private async void Button_Tambah_Clicked(object sender, EventArgs e)
        {
            await Navigation.PushAsync(new HalamanTambahData());
        }
        private async void Button_Lihat_Clicked(object sender, EventArgs e)
        {
            await Navigation.PushAsync(new HalamanLihatData());
        }
    }
}

```

Ketika user melakukan klik pada *button Tambah Data*, maka muncul tampilan untuk menambahkan data. Pengaturan tampilan dan fungsi pada Halaman ini diatur oleh HalamanTambahData.cs. Halaman ini berjudul Tambah Data Mahasiswa. Terdapat bagian untuk memasukkan nama dan jurusan serta terdapat *button tambah data* utnuk menyimpan data yang telah dimasukkan tadi. Data yang dsimpan akan disimpan di *database SQLite* yang telah diinstall sebelumnya. Apabila user melakukan klik Tambah Data maka akan muncul message atau pesan “Data xxx berhasil disimpan” dimana xxx adalah nama yang tadi dimasukkan. Setelah itu akan kembali ke Halaman Utama. Berikut adalah *sourcecode* HalamanTambahData.cs

```

using Kelompok37.Model;
using SQLite;

```

```

using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;

using Xamarin.Forms;

namespace Kelompok37.View
{
    public class HalamanTambahData : ContentPage
    {
        private Entry _nama;
        private Entry _jurusan;
        private Button _simpan;

        string dbPath = Path.Combine(System.Environment.GetFolderPath(System.Environment.SpecialFolder.Personal), "myDB.db4");

        public HalamanTambahData()
        {
            this.Title = "Tambah Data Mahasiswa";

            StackLayout stackLayout = new StackLayout();

            _nama = new Entry();
            _nama.Keyboard = Keyboard.Text;
            _nama.Placeholder = "Nama Mahasiswa";
            stackLayout.Children.Add(_nama);

            _jurusan = new Entry();
            _jurusan.Keyboard = Keyboard.Text;
            _jurusan.Placeholder = "Jurusan";
            stackLayout.Children.Add(_jurusan);

            _simpan = new Button();
            _simpan.Text = "Tambah";
            _simpan.Clicked += _simpan_Clicked;
            stackLayout.Children.Add(_simpan);

            Content = stackLayout;
        }

        private async void _simpan_Clicked(object sender, EventArgs e)
        {
            var db = new SQLiteConnection(_dbPath);
            db.CreateTable<DataMahasiswa>();

            var maxPk = db.Table<DataMahasiswa>().OrderByDescending(c => c.Id).FirstOrDefault();

```

```
        DataMahasiswa dbm = new DataMahasiswa()
    {
        Id = (maxPk == null ? 1 : maxPk.Id + 1),
        Nama = _nama.Text,
        Jurusan = _jurusan.Text
    };

        db.Insert(dbm);
        await DisplayAlert(null, "Data " + dbm.Nama + " Berhasil Disimpan", "Ok");
        await Navigation.PopAsync();
    }
}
```

Setelah kembali ke halaman utama, maka untuk melihat data yang telah dimasukkan yaitu dengan menekan *button* Lihat Data pada halaman utama. Untuk melihat data menggunakan HalamanLihatData.cs. Halaman inilah yang mengatur layout dan fungsi-fungsi yang ada. Halaman ini berjudul Data Mahasiswa dan menampilkan data-data mahasiswa yang telah disimpan pada SQLite. Berikut adalah *sourcecode* HalamanLihatData.cs,

```
using Kelompok37.Model;
using SQLite;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;

using Xamarin.Forms;

namespace Kelompok37.View
{
    public class HalamanLihatData : ContentPage
    {
        private ListView _listView;
        string _dbPath = Path.Combine(System.Environment.GetFolderPath(System.Environment.SpecialFolder.Personal), "myDB.db4");

        public HalamanLihatData()
        {
            this.Title = "Data Mahasiswa";

            var db = new SQLiteConnection(_dbPath);

            StackLayout stackLayout = new StackLayout();

            _listView = new ListView();
            _listView.ItemsSource =
db.Table<DataMahasiswa>().OrderBy(x => x.Nama).ToList();
```

```
        stackLayout.Children.Add(_listView);

        Content = stackLayout;
    }
}
```

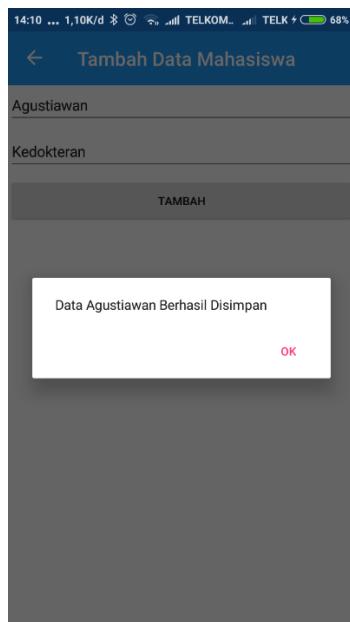
Pada HalamanUtama.cs, HalamanTambahdata.cs, dan HalamanLihatData.cs terdapat variabel `_dbpath` yang mana merupakan *database* lokal dengan nama `myDB.db4`. Berikut adalah *screenshot* hasil percobaan 2,



Gambar 2.38 HalamanUtama



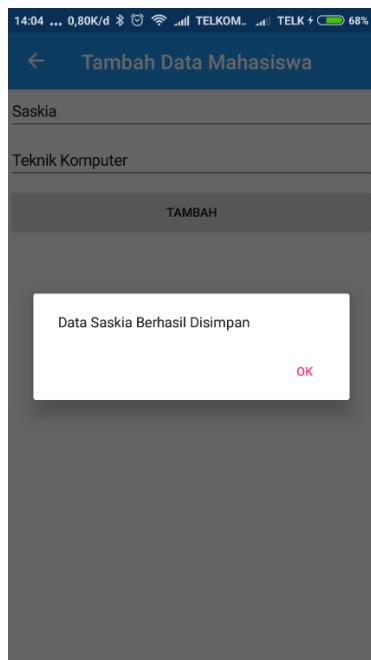
Gambar 2.39 Menambah data Agustiawan



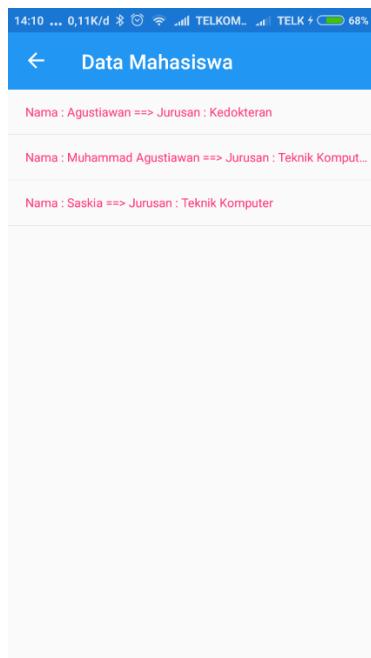
Gambar 2.40 Tambah data Agustiawan berhasil



Gambar 2.41 Menambah data Saskia



Gambar 2.42 Tambah data saskia berhasil



Gambar 2.43 Halaman Lihat Data

2.6. Tugas

Tugas pada bab ini, yaitu praktikan diminta untuk menambahkan menu edit data dan menu hapus data. Untuk itu dibuatlah beberapa class baru pada folder view, yaitu HalamanEditData.cs dan HalamanHapusData.cs. Selain itu ada penambahan code pada Halaman Utama.

Pada HalamanUtama.cs ditambahkan beberapa kodingan untuk membuat *button* Edit Data dan Hapus Data, *sourcecode* HalamanUtama.cs yang baru adalah sebagai berikut,

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

using Xamarin.Forms;

namespace Kelompok37.View
{
    public class HalamanUtama : ContentPage
    {
        public HalamanUtama()
        {
            this.Title = "Data Mahasiswa";

            StackLayout stacklayout = new StackLayout();
            Button button = new Button();
            button.Text = "Tambah Data";
            button.Clicked += Button_Tambah_Clicked;
            stacklayout.Children.Add(button);

            button = new Button();
            button.Text = "Lihat Data";
            button.Clicked += Button_Lihat_Clicked;
            stacklayout.Children.Add(button);

            button = new Button();
            button.Text = "Edit Data";
            button.Clicked += Button_Edit_Clicked;
            stacklayout.Children.Add(button);

            button = new Button();
            button.Text = "Hapus Data";
            button.Clicked += Button_Hapus_Clicked;
            stacklayout.Children.Add(button);

            Content = stacklayout;
        }
    }
}
```

```
private async void Button_Hapus_Clicked(object sender,  
EventArgs e)  
{  
    await Navigation.PushAsync(new HalamanHapusData());  
}  
  
private async void Button_Edit_Clicked(object sender,  
EventArgs e)  
{  
    await Navigation.PushAsync(new HalamanEditData());  
}  
  
private async void Button_Tambah_Clicked(object sender,  
EventArgs e)  
{  
    await Navigation.PushAsync(new HalamanTambahData());  
}  
private async void Button_Lihat_Clicked(object sender,  
EventArgs e)  
{  
    await Navigation.PushAsync(new HalamanLihatData());  
}  
}
```

Dalam menambah *button* edit, yang dilakukan yaitu menambahkan *sourcecode* button = new button (); kemudian *button* tersebut diberi tulisan yang akan ditampilkan di button.Text = "Edit Data" untuk *button* edit. Kemudian ditambahkan button.Clicked += Button_Edit_Clicked; dimana ketika *button* diklik nanti akan mengarahkan ke content di folder view yang bernama HalamanEditData.cs. Untuk *button* hapus, yang dilakukan yaitu menambahkan *sourcecode* button = new button (); kemudian *button* tersebut diberi tulisan yang akan ditampilkan di button.Text = "Hapus Data" untuk *button* edit. Kemudian ditambahkan button.Clicked += Button_Hapus_Clicked; dimana ketika *button* diklik nanti akan mengarahkan ke content di folder view yang bernama HalamanHapusData.cs

Kemudian membuat HalamanEditData.cs di folder view, berikut adalah sourcecode HalamanEditData.cs

```
using Kelompok37.Model;  
using Kelompok37.View;  
using SQLite;
```

```

using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;

using Xamarin.Forms;

namespace Kelompok37.View
{

    public class HalamanEditData : ContentPage
    {
        private ListView _listView;
        private Entry _id;
        private Entry _nama;
        private Entry _jurusan;
        private Button _ubah;

        DataMahasiswa dataMahasiswa = new DataMahasiswa();

        string _dbPath =
Path.Combine(System.Environment.GetFolderPath(System.Environment.SpecialFolder.Personal), "myDB.db4");

        public object Popasync { get; private set; }

        public HalamanEditData()
        {
            this.Title = "Edit Data";

            var db = new SQLiteConnection(_dbPath);

            StackLayout stackLayout = new StackLayout();

            _listView = new ListView();
            _listView.ItemsSource =
db.Table<DataMahasiswa>().OrderBy(x => x.Nama).ToList();
            _listView.ItemSelected += _listView_ItemSelected;
            stackLayout.Children.Add(_listView);

            _id = new Entry();
            _id.Placeholder = "Id";
            _id.IsVisible = false;
            stackLayout.Children.Add(_id);

            _nama = new Entry();
            _nama.Keyboard = Keyboard.Text;
            _nama.Placeholder = "Nama";
            stackLayout.Children.Add(_nama);

            _jurusan = new Entry();
            _jurusan.Keyboard = Keyboard.Text;
            _jurusan.Placeholder = "Jurusan";
            stackLayout.Children.Add(_jurusan);
    }
}

```

```

        _ubah = new Button();
        _ubah.Text = "Ubah";
        _ubah.Clicked += _ubah_Clicked;
        stackLayout.Children.Add(_ubah);

        Content = stackLayout;

    }

    private async void _ubah_Clicked(object sender, EventArgs e)
    {
        var db = new SQLiteConnection(_dbPath);
        DataMahasiswa dataMahasiswa = new DataMahasiswa()
        {
            Id = Convert.ToInt32(_id.Text),
            Nama = _nama.Text,
            Jurusan = _jurusan.Text
        };
        db.Update(dataMahasiswa);
        await Navigation.PopAsync();
    }

    private void _listView_ItemSelected(object sender, SelectedItemChangedEventArgs e)
    {
        dataMahasiswa = (DataMahasiswa)e.SelectedItem;
        _id.Text = dataMahasiswa.Id.ToString();
        _nama.Text = dataMahasiswa>Nama;
        _jurusan.Text = dataMahasiswa.Jurusan;

    }
}
}

```

Pada HalamanEditData.cs dilakukan inisialisasi Listview dengan variabel `_id`, `_nama`, `_jurusan`, `_ubah`, yang nantinya digunakan untuk menyimpan data yang diubah pada halama edit ini. Selanjutnya yaitu membuat kelas baru bernama mahasiswa. Disini kita juga tambahkan *databasenya* sama seperti di halaman utama. Lalu kita buat judul pada halaman edit ini dengan nama “Edit Data”. Lalu kita tambahkan Entry `_id`, `_nama`, dan jurusan untuk ketika kita membuka halaman tersebut, lalu kita mengeklik data yang ada akan muncul entry untuk mengganti nama dan jurusan. Lalu kita buat button `_ubah` yang nantinya ketika kita sudah selesai mengedit kita tekan simpan data akan langsung di ganti di *database*.

Selanjutnya dibuat HalamanHapusData.cs yang digunakan untuk menghapus data mahasiswa. Berikut adalah *sourcecode* HalamanHapusData.cs.

```

using Kelompok37.Model;
using SQLite;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;

using Xamarin.Forms;

namespace Kelompok37.View
{
    public class HalamanHapusData : ContentPage
    {
        private ListView _listView;
        private Button _hapus;

        DataMahasiswa dataMahasiswa = new DataMahasiswa();

        string _dbPath =
Path.Combine(System.Environment.GetFolderPath(System.Environment.SpecialFolder.Personal), "myDB.db4");

        public HalamanHapusData()
        {
            this.Title = "Hapus Data";

            var db = new SQLiteConnection(_dbPath);

            StackLayout stackLayout = new StackLayout();

            _listView = new ListView();
            _listView.ItemsSource =
db.Table<DataMahasiswa>().OrderBy(x => x.Nama).ToList();
            _listView.ItemSelected += _listView_ItemSelected;
            stackLayout.Children.Add(_listView);

            _hapus = new Button();
            _hapus.Text = "Hapus";
            _hapus.Clicked += _hapus_Clicked;
            stackLayout.Children.Add(_hapus);

            Content = stackLayout;
        }

        private async void _hapus_Clicked(object sender, EventArgs e)
        {
            var db = new SQLiteConnection(_dbPath);
            db.Table<DataMahasiswa>().Delete(x => x.Nama ==
dataMahasiswa.Nama);
            await Navigation.PopAsync();
        }
    }
}

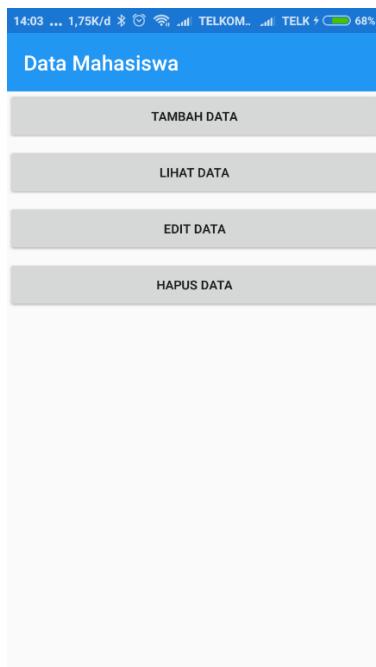
```

```
        }

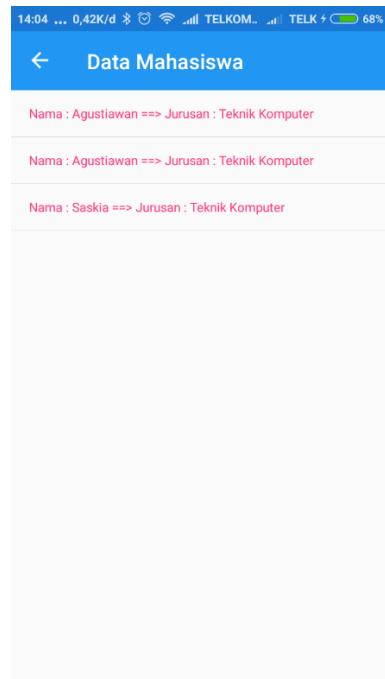
        private void _listView_SelectedIndexChanged(object sender,
SelectedEventArgs e)
{
    dataMahasiswa = (DataMahasiswa)e.SelectedItem;
}
}
```

Pada HalamanHapusData.cs ini pertama-tama yaitu menginisialisasi ListView dan juga *button _hapus*. Listview digunakan untuk menampilkan data yang ada di datamahasiswa. Ketika masuk ke halaman hapus data, maka akan ditampilkan data mahasiswa yang ada. Kemudian untuk menghapus data mahasiswa dengan cara meng-klik data yang ada pada daftar dan kemudian klik tombol hapus.

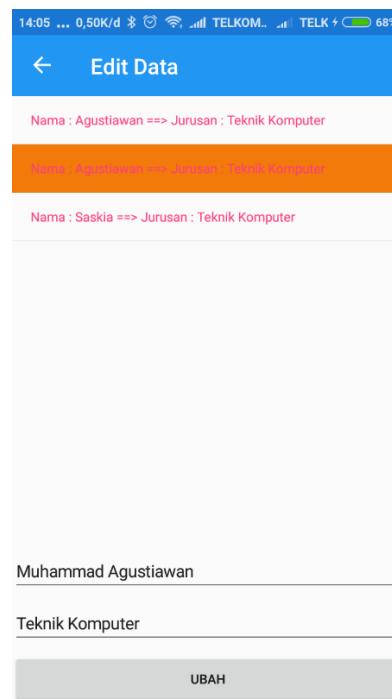
Berikut adalah hasil *screenshot* tugas ini,



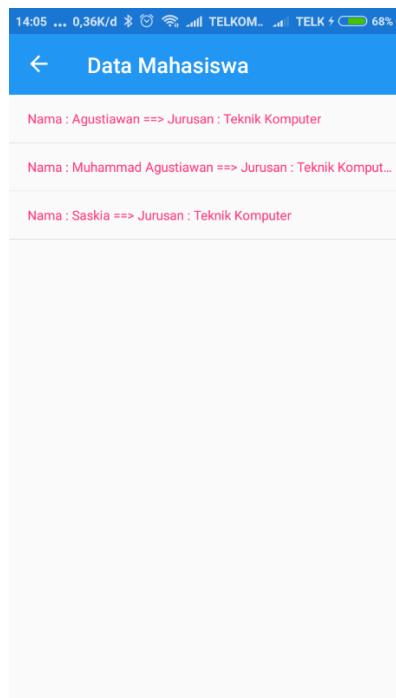
Gambar 2.44 Tampilan Halaman Utama



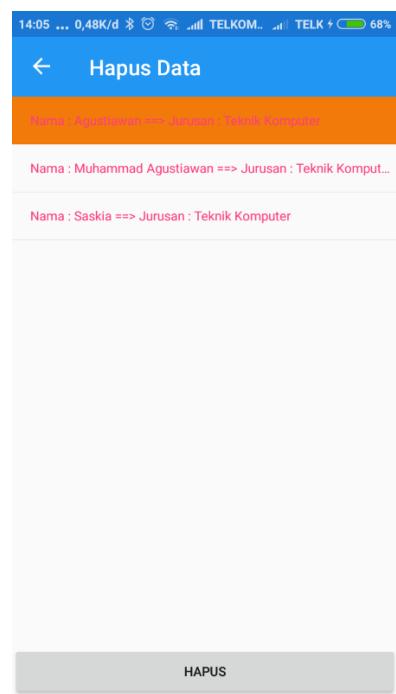
Gambar 2.45 Tampilan Awal Lihat Data



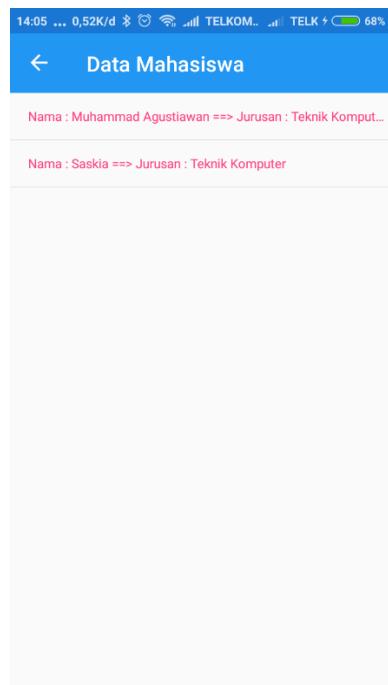
Gambar 2.46 Mengubah Data Agustiawan



Gambar 2.47 Data Agustiawan berubah menjadi Muhammad Agustiawan



Gambar 2.48 Menghapus Data Agustiawan



Gambar 2.49 Data Agustiawan telah terhapus

(Link github : github.com/agustiawan0798/appmodul1kelompok37)

2.7. Kesimpulan

1. *Database* yang digunakan pada praktikum ini yaitu SQLite
2. Sebelum melakukan praktikum, harus terinstall terlebih dahulu SDK yang mana sesuai dengan perangkat android yang akan dipakai.
3. Untuk menggunakan *database* SQLite maka harus diinstall terlebih dahulu SQLite-net-pcl nya terlebih dahulu melalui menu Manage NuGet Package.
4. Untuk mengatur layout atau tampilan suatu halaman dapat menggunakan *file.xaml*
5. Variabel _dbpath digunakan untuk membuat suatu halaman dapat terhubung dengan *database* lokal dalam hal ini yaitu *database* dengan nam myDB.db4
6. Fungsi dari *file.cs* yaitu untuk mengatur fungsi-fungsi dan kondisi apa saja yang terdapat pada halaman tersebut.
7. Xamarin merupakan *platform* untuk pengembangan aplikasi *cross platform* di mana pengembangannya menggunakan bahasa C# dan .NET framework.

BAB III

Android Studio #1

3.1 Tujuan

1. Praktikan mampu mengetahui lingkungan pengembangan Android
2. Praktikan memahami siklus activity pada aplikasi Android
3. Praktikan memahami penggunaan layout-layout pada Android Studio
4. Praktikan dapat memahami penggunaan intent
5. Praktikan dapat mengetahui fitur-fitur pada Android Studio
6. Praktikan dapat belajar bagaimana membuat project dengan Android Studio

3.2 Alat dan Bahan

1. PC / Notebook (minimal RAM 4GB)



Gambar 3.1 Notebook

Notebook berguna sebagai perangkat yang akan diinstallasi program dan sebagai perangkat yang digunakan untuk melakukan kegiatan praktikum.

2. Android Studio 3.2.1, JDK, Andoid SDK



Gambar 3.2 Android Studio

Android Studio merupakan sebuah IDE untuk Android Development yang dikenalkan pihak google pada acara Google I/O di tahun 2013. Android Studio merupakan IDE resmi untuk pengembangan aplikasi. Android Studio digunakan sebagai IDE yang akan digunakan dalam praktikum.

3. Device Android dan kabel USB



Gambar 3.3 Perangkat Android dan Kabel USB

Perangkat Android digunakan sebagai objek pengujian dalam praktikum. Kabel USB digunakan untuk menyambungkan laptop/PC dengan perangkat Android yang akan dideploykan projectnya nanti ke dalamnya.

3.3 Dasar Teori

3.3.1 Android

Android adalah sistem operasi untuk telepon seluler yang berbasiskan Linux. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri sehingga dapat digunakan oleh bermacam peranti penggerak.

Android tersedia secara open source bagi manufaktur perangkat keras untuk memodifikasi sesuai kebutuhan. Meskipun konfigurasi perangkat Android tidak sama antara satu dengan yang lain, namun Android sendiri mendukung beberapa fitur berikut ini:

- a. Penyimpanan (Storage) - menggunakan SQLite yang merupakan basis data relasional yang ringan untuk menyimpan data.
- b. Koneksi(Connectivity) - Mendukung GSM/EDGE, IDEN, CDMA, EVDO, UMTS, Bluetooth, WiFi, LTE, dan WiMAX.
- c. Pesan (Messaging) Mendukung SMS dan MMS
- d. Web Browser - Menggunakan open source WebKit termasuk di dalamnya engine Chrome V8 JavaScript.
- e. Media - media di dukung antara lain : H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, MP3, MIDI, Ogg Vorbis, JPEG, PNG, GIF, dan BMP.
- f. Perangkat keras - terdapat Accelerometer Sensor, kamera, kompas digital, sensor proximity, dan GPS.
- g. Multi-touch - mendukung layar multi-touch
- h. Multi-tasking - mendukung aplikasi multi-tasking.
- i. Dukungan Flash - Android 2.3 mendukung Flash 10.1

(Sumber : Modul 2 Android Studio RC2.I.1 Praktikum MDP 2018)

3.3.2 Android Studio



Gambar 3.4 Android Studio

Android Studio adalah sebuah **IDE** untuk Android Development yang dikenalkan pihak google pada acara Google I/O di tahun 2013. Android Studio merupakan suatu pengembangan dari Eclipse IDE, dan dibuat berdasarkan IDE Java populer, yaitu **IntelliJ IDEA**. Android Studio merupakan IDE resmi untuk pengembangan aplikasi Android. Berbeda dengan Eclipse yang menggunakan Ant, Android Studio menggunakan Gradle sebagai **build environment**.

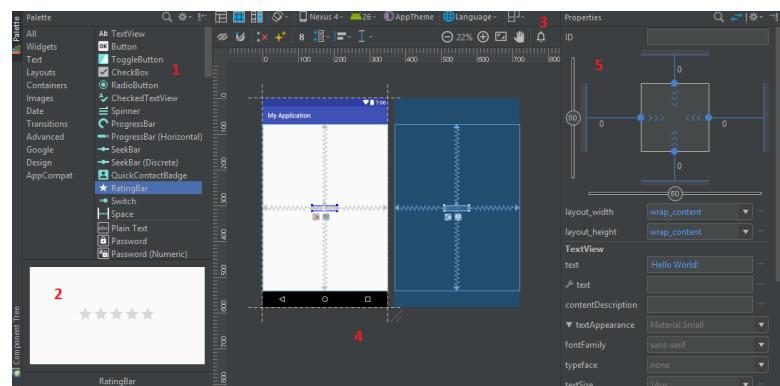
Fitur-fitur lainnya yang sudah disematkan pada Android Studio adalah sebagai berikut :

1. Menggunakan Gradle-based build system yang fleksibel.
2. Bisa mem-build multiple APK .
3. Template support untuk Google Services dan berbagai macam tipe perangkat.
4. Layout editor yang lebih bagus.
5. Built-in support untuk Google Cloud Platform, sehingga mudah untuk integrasi dengan Google Cloud Messaging dan App Engine.
6. Import library langsung dari Maven repository dan sebagainya.

(Sumber : Pengenalan Android Studio,
<http://www.jadibaru.com/android/pengenalan-android-studio-2/>)

3.3.3 Layout

Pada Android Studio layout di definisikan menggunakan bahasa XML, layout di bedakan menjadi dua yaitu design dan text. Pada design, sama halnya Frame GUI di Java, menampilkan layout untuk mengatur komponen UI pada aplikasi yang kita buat. Kita dapat men drag and drop komponen yang tersedia pada tab Palette dan mengatur ukuran komponen di properties.



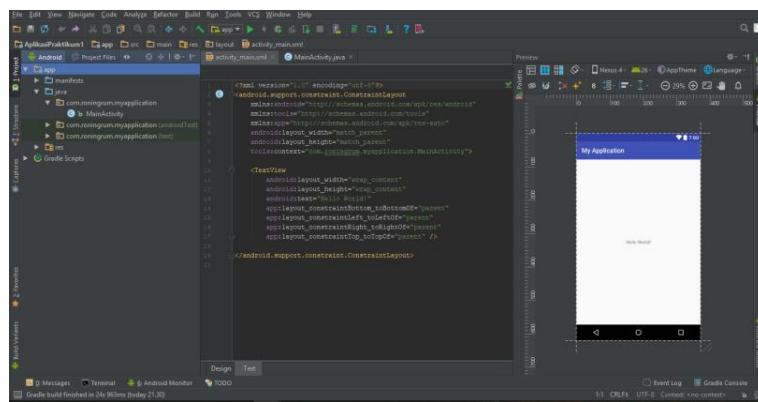
Gambar 3.5 Layout Editor Mode Design pada Android Studio

Region atau daerah pada yang ada pada layout editor design dibagi menjadi 5 area yaitu:

- Palette: Menyediakan daftar widget dan layout yang bisa Anda seret ke dalam layout di editor.
- Component Tree: Menampilkan hierarki tampilan untuk layout Anda. Klik sebuah item di sini untuk melihatnya dipilih dalam editor.
- Toolbar: Menyediakan tombol untuk mengonfigurasi penampilan layout di editor dan untuk mengedit properti layout.
- Design Editor: Menampilkan layout Anda bersama tampilan Design dan Blueprint.
- Properties: Menyediakan kontrol properti untuk tampilan yang dipilih saat ini.

Sementara pada mode text, akan menampilkan barisan kode atau barisan susunan kode dari text XML nya. dalam Android studio sudah memiliki fitur **Live**

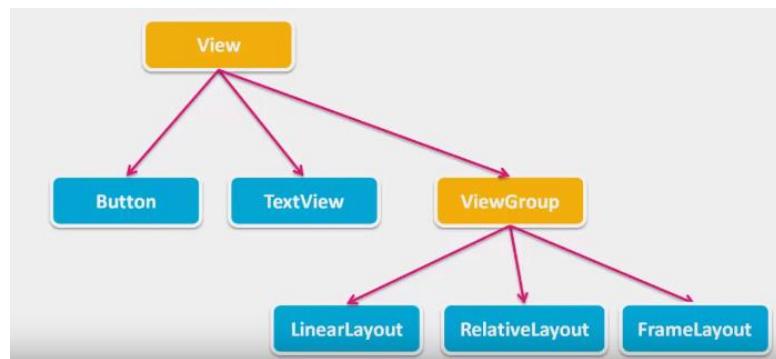
Preview, jadi dengan adanya fitur ini, kalian tidak perlu bolak-balik ke mode design hanya untuk melihat hasil kode XML yang sudah dibuat atau edit.



Gambar 3.6 Layout Editor XML Mode Text pada Android Studio

Pada Layout di android studio setiap komponen GUInya disebut View, View adalah: basic class atau kelas dasar dari setiap komponen User Interface pada Android termasuk tata letak (layout).

Terdapat sebuah subclass dalam view yaitu ViewGroup, ViewGroup merupakan class yang diperuntukan untuk layout saja.



Gambar 3.7 Struktur View

Layout adalah struktur visual untuk antarmuka pengguna, seperti User Interface sebuah aktivitas atau widget aplikasi. Mendeklarasikan layout dengan dua cara:

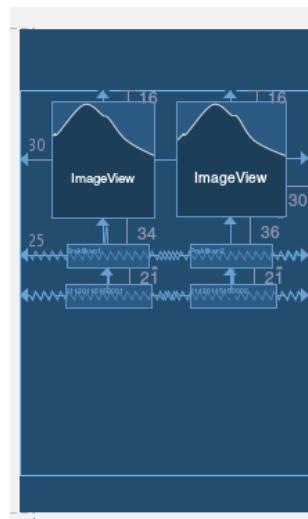
- **Deklarasikan elemen UI dalam XML.** Android menyediakan sebuah kosakata XML sederhana yang sesuai dengan kelas dan subkelas View, seperti halnya untuk widget dan layout.

- **Buat instance elemen layout saat waktu proses.** Aplikasi Anda bisa membuat objek View dan ViewGroup (dan memanipulasi propertinya) lewat program.

Pada Layout terdapat kelas – kelas yang sering digunakan yaitu

1. Constraint Layout

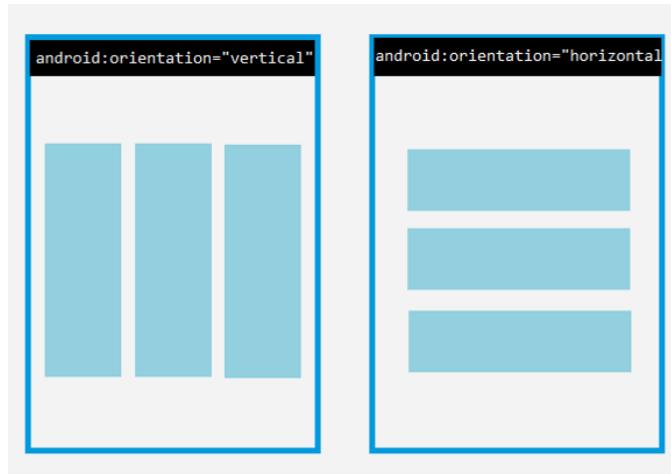
Constraint Layout adalah tampilan grup yang tersedia di pustaka Constraint Layout, yang telah disertakan bersama Android Studio 2.2 dan yang lebih tinggi. Layout ini dibangun dari nol bersama Layout Editor, jadi apa saja dapat diakses dari Design editor dan tidak perlu mengedit XML secara manual. Yang paling penting, sistem layout berbasis batasan ini memungkinkan untuk membuat hampir semua layout tanpa menambahkan tampilan grup apa pun.



Gambar 3.8 Contoh Constraint Layout

2. Linear Layout

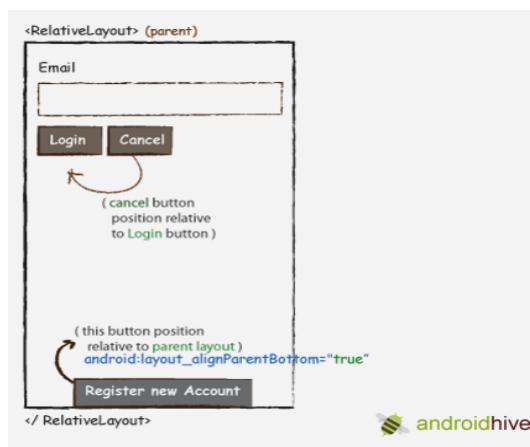
LinearLayout akan menempatkan komponen-komponen didalamnya secara horizontal atau vertical (menyamping atau menurun). LinearLayout memiliki atribut weight untuk masing-masing child view yang terdapat didalam LinearLayout yang berguna untuk mengontrol porsi ukuran view secara Relatif dalam sebuah ruang (space) yang tersedia.



Gambar 3.9 Linear Layout

3. Relative Layout

RelativeLayout adalah layout yang menampilkan elemen-elemen view dalam posisi yang relatif. Posisi dari sebuah view yang dapat diletakkan relatif terhadap posisi elemen view di sekitarnya atau relatif terhadap area layout utama. Sebuah desain tampilan aplikasi bisa dibuat lebih mudah dan sederhana menggunakan RelativeLayout dibandingkan dengan LinearLayout.

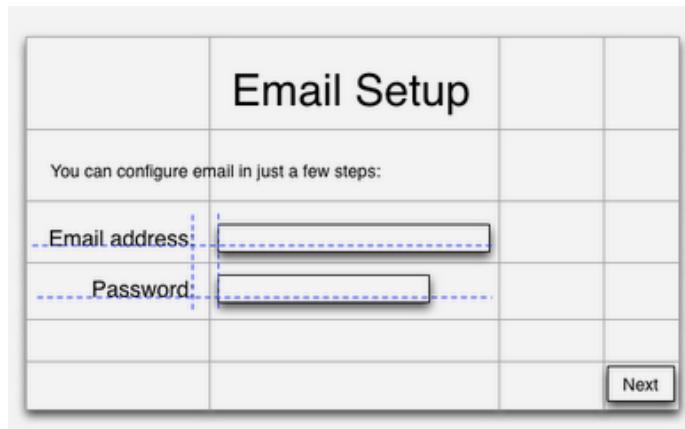


Gambar 3.10 Relative Layout

4. Grid Layout

Diperkenalkan pada API level 14 (Android 4.0 / Ice Cream Sandwich), layout ini akan memberikan kemudahan dengan mengakomodir komponen didalamnya ke

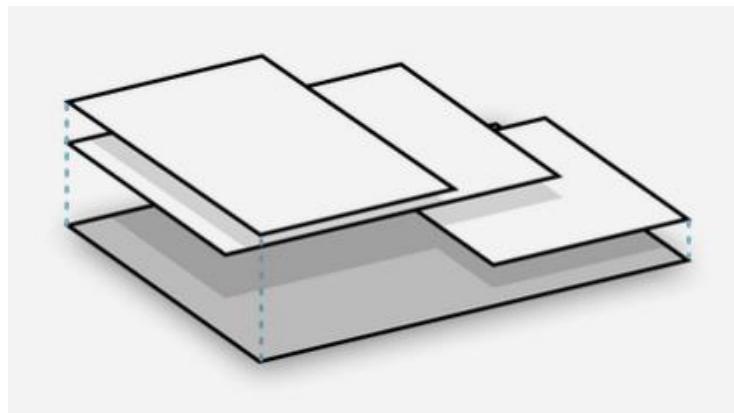
dalam bentuk Grid (Kolom dan Baris). Dalam sebuah referensi, GridLayout merupakan komponen layout yang sangat flexibel dan dapat dimanfaatkan untuk menyederhanakan pembuatan Layout UI yang bersifat kompleks dan bersarang yang terdapat di komponen Layout lainnya.



Gambar 3.11 Grid Layout

5. Frame Layout

Layout ini adalah layout yang paling sederhana. Layout ini akan membuat komponen yang ada didalamnya menjadi menumpuk atau saling menutupi satu dengan yang lainnya (*layering*). Komponen yang paling pertama pada layout ini akan berada dibawah komponen-komponen diatasnya. Pada materi penggunaan fragment di materi sebelumnya, FrameLayout memiliki kemampuan untuk menjadi container untuk fragment-fragment didalam sebuah Activity. Berikut ilustrasi dari penggunaan FrameLayout terhadap child view yang dimiliki didalamnya.



Gambar 3.12 Frame Layout

(Sumber : <https://blog.dicoding.com/android-module-5-android-layout/>)

3.3.4 Shared Preferences

Sebuah penyimpanan data yang ada di dalam Android yang bersifat data primitif pada file internal aplikasi dengan pasangan key-value.

(Sumber : Modul 2 Android Studio RC2.1.1 Praktikum MDP 2018)

3.3.5 SQLite

SQLite merupakan sistem manajemen database relasional yang dibangun dalam sebuah library bahasa pemrograman C. Berbeda dengan kebanyakan sistem manajemen database lainnya, SQLite bukanlah mesin database client-server melainkan berupa file tunggal atau beberapa file saja.

(Sumber : Modul 2 Android Studio RC2.1.1 Praktikum MDP 2018)

3.3.6 Intent

Intent adalah sebuah kelas dalam programming Android yang berfungsi untuk perpindahan halaman. Intent juga merupakan suatu objek yang terdapat dalam suatu activity dimana objek tersebut dapat komunikasi dengan activity yang lain, baik activity pada fungsi internal android misal seperti memanggil activity dalam satu package atau beda package yang masih berada dalam satu project. Intent merupakan objek tipe android.content.Intent. Melalui metode `startActivity()` yang digunakan untuk memulai sebuah activity lain.

Intent dibagi menjadi 2, yaitu :

1. Explicit Intent berfungsi untuk mengaktifkan komponen-komponen dalam satu aplikasi yang sama. Misalnya seperti : Berpindah Activity.
2. Implicit Intent berfungsi untuk memanggil fungsi activity yang sudah ada di fungsi internal android seperti Dial Number, Open Browser dan lainnya.

(Sumber : Modul 2 Android Studio RC2.I.1 Praktikum MDP 2018)

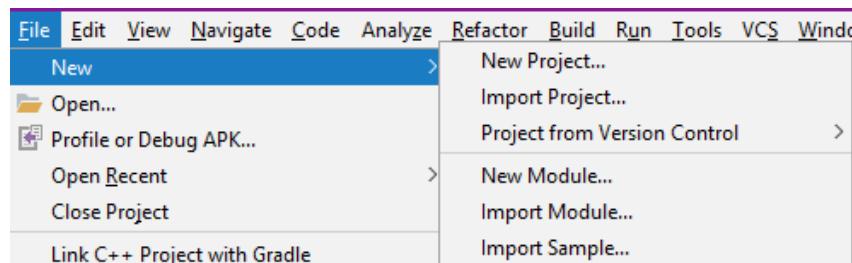
3.4 Langkah Kerja

3.4.1 Mengatur Device Android

- Pengaturan → Opsi Pengembang → Ubah menjadi “aktif” → aktifkan debugging USB

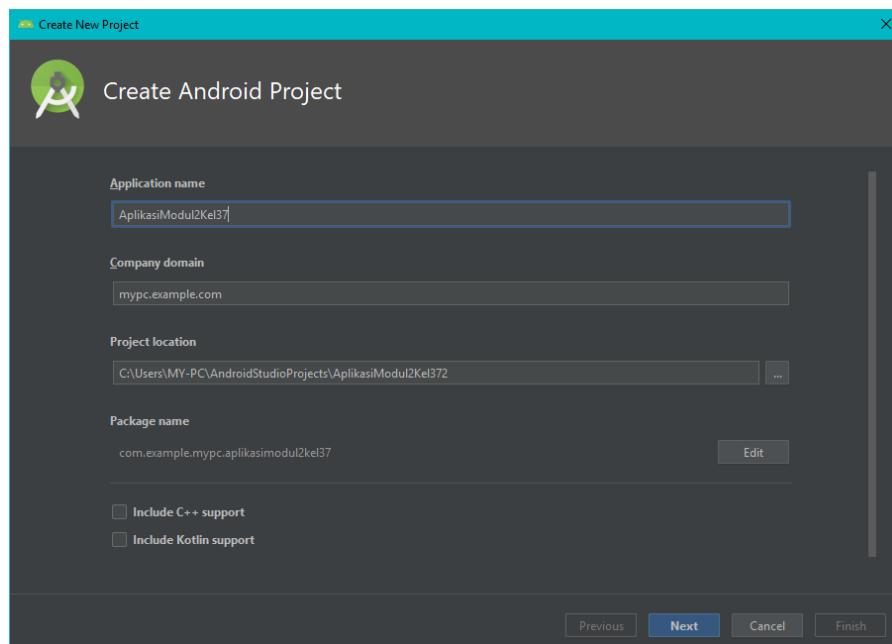
3.4.2 Membuat Project Aplikasi Android

- Buka Android Studio
- Klik menu File → New → New Project



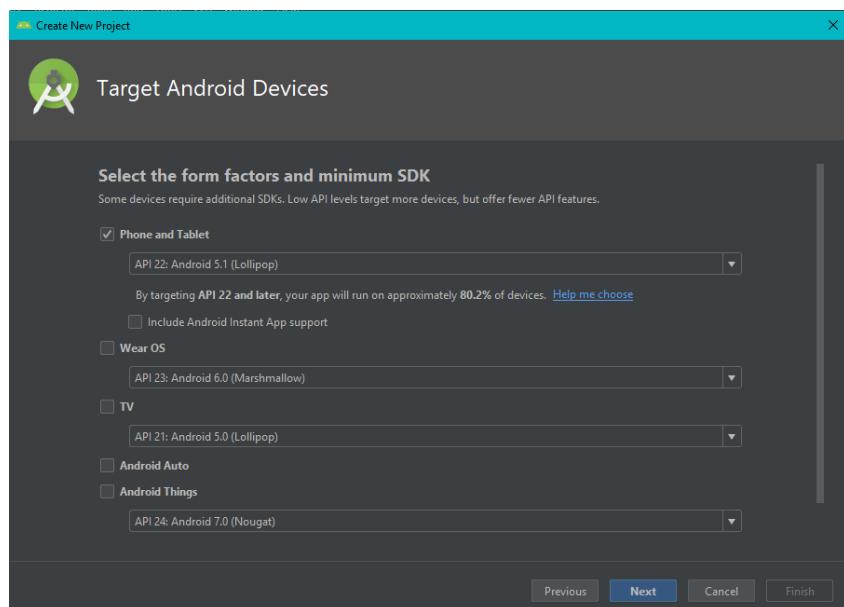
Gambar 3.13 New Project

- Buat project baru dengan nama AplikasiModul2Kel37



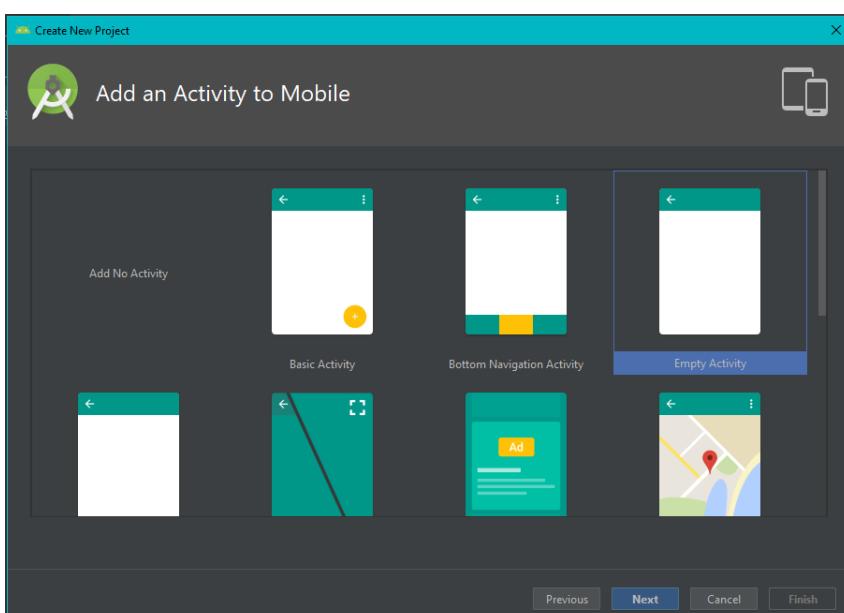
Gambar 3.14 Tab Create New Project

- Kemudian pilih min.SDK sesuai dengan kebutuhan pengguna. Namun, pada praktikum menggunakan min SDK 22 yaitu Android Lollipop dikarenakan kebanyakan pengguna Android sekarang menggunakan versi Lollipop.



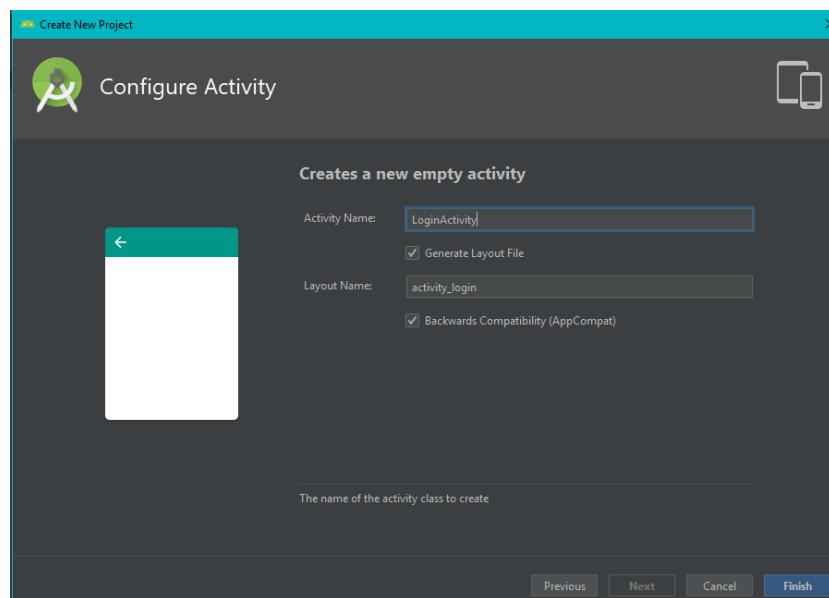
Gambar 3.15 Target Android Devices

- Kemudian pilih jenis activity yang akan ditampilkan yaitu empty activity.

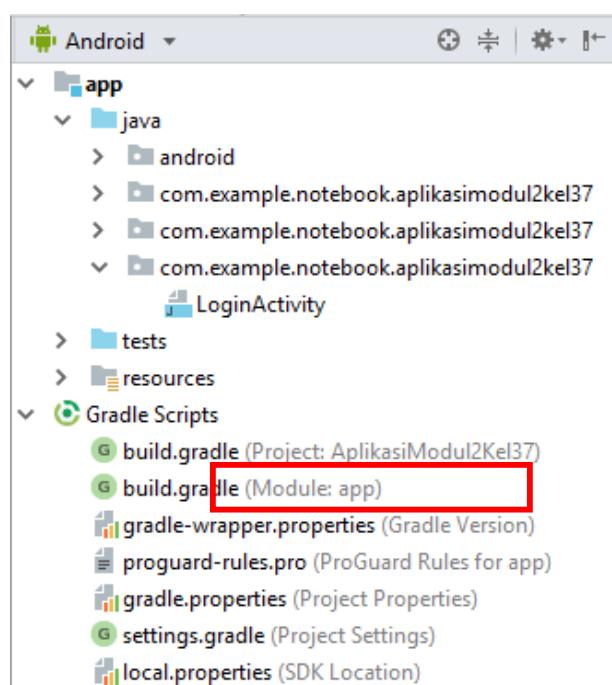


Gambar 3.16 Pilihan Activity

6. Kemudian, beri nama activity dengan nama ‘Login Activity’ lalu klik finish. Kemudian, tunggu sampai proses gradle selesai. Proses gradle memerlukan beberapa waktu untuk selesai. Saat selesai, akan ada build.gradle(Module: app).



Gambar 3.17 Isi nama activity



Gambar 3.18 Proses gradle selesai

7. Setelah itu kita cari build.gradle(Module:app) untuk men-copy dependency material, berikut *sourcecode* nya:

```
dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:appcompat-v7:27.1.1'
    implementation 'com.android.support:design:28.0.0'
    implementation 'com.android.support:cardview-v7:28.0.0'
    implementation 'com.android.support.constraint:constraint-
layout:1.1.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation
    'com.android.support.test:runner:1.0.2'
    androidTestImplementation
    'com.android.support.test.espresso:espresso-core:3.0.2'
}
```

8. Tambahkan beberapa code di build.gradle yang telah kita *click* sebelumnya. Kurang lebih seperti pada di atas.
9. Pertama, kita akan desain tampilan colors, strings, dan styles nya untuk color string dan style xml terletak di folder res – value di panel tree pada android studio. Berikut *sourcecode* dari colors.xml :

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#82aaff</color>
    <color name="colorPrimaryDark">#4d7bcb</color>
    <color name="colorAccent">#bec4ea</color>
    <color name="white">#ffffff</color>
    <color name="button">#b6dbff</color>
</resources>
```

10. Setelah itu kita masuk ke strings.xml. Sesuai Namanya file xml ini berguna untuk menyimpan nilai string, berikut *sourcecode* nya:

```
<resources>
    <string name="app_name">Android Modul</string>
    <string name="tittle">PRAKTIKUM MDP MODUL ANDROID</string>
    <string name="regtittle">REGISTRASI DATA</string>
    <string name="user">username</string>
    <string name="password">password</string>
    <string name="login">masuk</string>
    <string name="register">register</string>
    <string name="logout">Logout</string>
    <string name="Header">Selamat datang Di Praktikum MDP
2018</string>
```

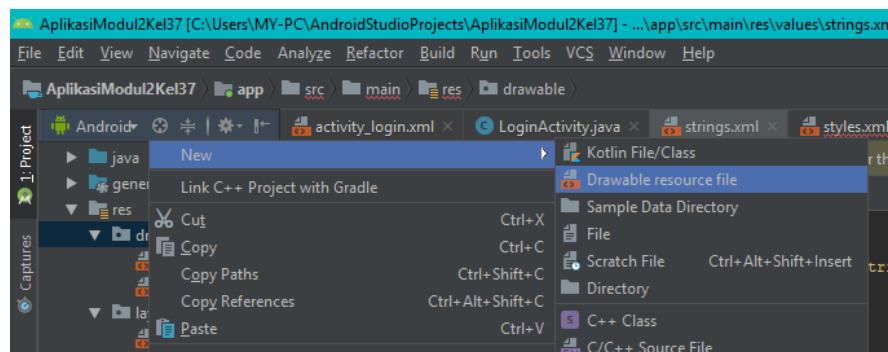
```
<string name="name">Nama Kamu</string>
<string name="description">Selanjutnya akan di jelaskan di
modul android berikutnya Mengenai API dan MVP</string>
</resources>
```

11. Kemudian di dalam styles.xml, di dalam *values* ini mengatur memprsentasikan *style* atau ketika dalam bahas pemrograman web, seperti file css (*cascading style sheet*) yang mengatur style tampilan aplikasi android studio, berikut *sourcecode* nya :

```
<resources>

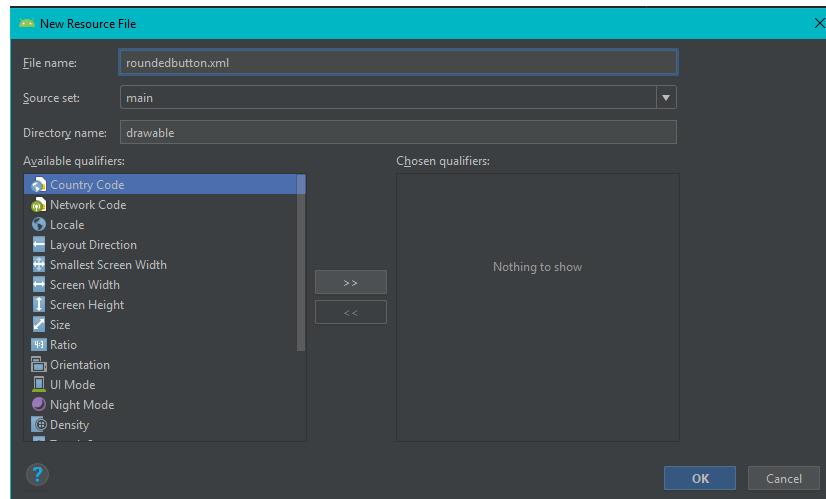
    <!-- Base application theme. -->
    <style name="AppTheme"
parent="Theme.AppCompat.Light.DarkActionBar">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item
name="colorPrimaryDark">@color/colorPrimaryDark</item>
            <item name="colorAccent">@color/colorAccent</item>
    </style>
</resources>
```

12. Kemudian buat drawable resource file untuk membuat desain tombol di aplikasi melengkung, dengan mengklik kanan folder drawable dan pilih new kemudian klik menu drawable resource file.



Gambar 3.19 Tampilan membuat drawable resource

13. Buat file dengan nama roundedbutton.xml

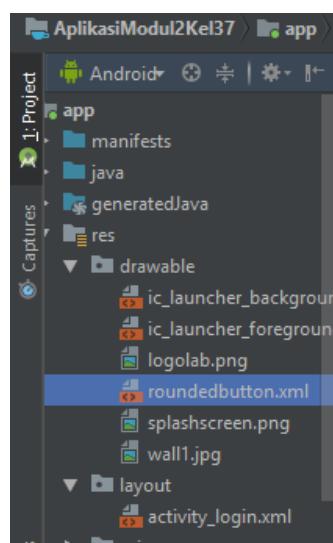


Gambar 3.20 Membuat file roundedbutton.xml

14. Buat file tersebut dan copy kan source codenya

```
<?xml version="1.0" encoding="utf-8"?>
<selector
    xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_pressed="false">
        <shape android:shape="rectangle">
            <corners android:radius="5dp" />
            <solid android:color="@color/button" />
            <stroke
                android:width="1dp"
                android:color="@color/button" />
            <padding
                android:bottom="4dp"
                android:left="4dp"
                android:right="4dp"
                android:top="4dp" />
        </shape>    </item>
    <item android:state_pressed="true">
        <shape android:shape="rectangle">
            <corners android:radius="5dp" />
            <solid android:color="@color/button" />
            <stroke
                android:width="1dp"
                android:color="@color/button" />
            <padding
                android:bottom="4dp"
                android:left="4dp"
                android:right="4dp"
                android:top="4dp" />
        </shape>
    </item>
</selector>
```

15. Setelah mengatur di folder values dan drawable, kemudian kita akan memasukkan sebuah gambar gambar yang akan kita gunakan nantinya untuk tampilan aplikasinya. Kita masukkan logonya di folder res→drawable (drag and drop di folder ini).



Gambar 3.21 Gambar-gambar pada folder res→drawable

16. Masuk ke bagian design activity yang akan kita atur melalui folder res ->layout, yaitu activiy_login.xml. Berikut *sourcecode* nya :

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/white"
    tools:context=".LoginActivity">

    <RelativeLayout
        android:id="@+id/rllLogin"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="60dp">

        <ImageView
            android:id="@+id/ivLogin"
            android:layout_width="100dp"
            android:layout_height="100dp"
```

```
    android:layout_gravity="center_horizontal"
    android:layout_marginLeft="130dp"
    android:src="@drawable/logolab"
    android:contentDescription="LOGO" />

<android.support.v7.widget.AppCompatTextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="@string/tittle"
    android:textAlignment="center"
    android:textSize="20sp"
    android:textStyle="bold"
    android:layout_marginTop="120dp" />

<LinearLayout
    android:id="@+id/l1Login"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_centerInParent="true"
    android:layout_marginTop="320dp"
    android:layout_marginEnd="50dp"
    android:orientation="vertical"
    android:padding="18dp">

    <EditText
        android:id="@+id/etUsernameLogin"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/user"
        android:inputType="textEmailAddress"
        android:maxLines="1"
        android:singleLine="true" />

    <EditText
        android:id="@+id/etPasswdLogin"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="30dp"
        android:hint="@string/password"
        android:imeActionId="6"
        android:imeActionLabel="Sign In"
        android:imeOptions="actionUnspecified"
        android:inputType="textPassword"
        android:maxLines="1"
        android:singleLine="true" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <Button
```

```
        android:id="@+id/btnSignUpLogin"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"

    android:background="@drawable/roundedbutton"
        android:text="@string/register"
        android:textColor="@color/white"/>

    <LinearLayout
        android:layout_width="2dp"
        android:layout_height="2dp"
        android:layout_weight="0"

    android:orientation="vertical"></LinearLayout>

    <Button
        android:id="@+id/btnSignInLogin"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"

    android:background="@drawable/roundedbutton"
        android:text="@string/login"
        android:textColor="@color/white" />
    </LinearLayout>
</LinearLayout>
</RelativeLayout>
</ScrollView>
```

17. Maka tampilannya akan seperti gambar dibawah ini



Gambar 3.22 Layout activity_login.xml

18. Setelah itu kita masukkan *sourcecode* LoginActivity.java yang digunakan untuk masuk ke dalam aplikasi nya. Kali ini kita menggunakan SharedPreferences untuk menyimpan database username dan password yang kita masukkan. Sifat SharedPreferences hamper sama dengan kebanyakan database layaknya SQLite tapi hanya bisa menyimpan bentuk data yang sederhana dan lebih kecil.

```

import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.Log;
import android.util.Printer;
import android.view.KeyEvent;
import android.view.View;
import android.view.inputmethod.EditorInfo;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import java.security.PrivateKey;

public class LoginActivity extends AppCompatActivity {

    private EditText etUsernameLogin, etPasswdLogin;

```

```

private Button btnSignUpLogin, btnSignInLogin;
String username, password;
private User user;
private DatabaseHandler presenter;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_login);

    initView();
    initData();
    login();
    register();
}

private void initView() {
    etUsernameLogin = findViewById(R.id.etUsernameLogin);
    etPasswdLogin = findViewById(R.id.etPasswdLogin);
    btnSignInLogin = findViewById(R.id.btnSignInLogin);
    btnSignUpLogin = findViewById(R.id.btnSignUpLogin);
}

private void register() {
    btnSignUpLogin.setOnClickListener(V -> {
        Intent reg = new Intent(this,
RegisterActivity.class);
        startActivity(reg);
        finish();
    });
}

private void initData() {
    user = new User();
    presenter = new DatabaseHandler(this);
}

private Boolean validation() {
    username = etUsernameLogin.getText().toString();
    password = etPasswdLogin.getText().toString();

    if(username.isEmpty()){
        Toast.makeText(this, "Isikan username",
Toast.LENGTH_SHORT).show();
        Log.e("Validation", "false");
        return false;
    }

    if(password.isEmpty()){
        Toast.makeText(this, "Isikan Password",
Toast.LENGTH_SHORT).show();
    }
}

```

```

        Log.e("Validation", "false");
        return false;
    }

    Log.e("Validation", "true");
    return true;
}

private void login() {
    btnSignInLogin.setOnClickListener(v -> actLogin());
}

private void actLogin(){
    if(validation()){
        if(loginData()){
            Log.e("actLogin", "true");
            initPreference();
            Intent main = new Intent(this,
MainActivity.class);
            startActivity(main);
            finish();
        } else {
            Log.e("actLogin", "false");
            Toast.makeText(this, "Login gagal",
Toast.LENGTH_LONG).show();
        }
    }
}

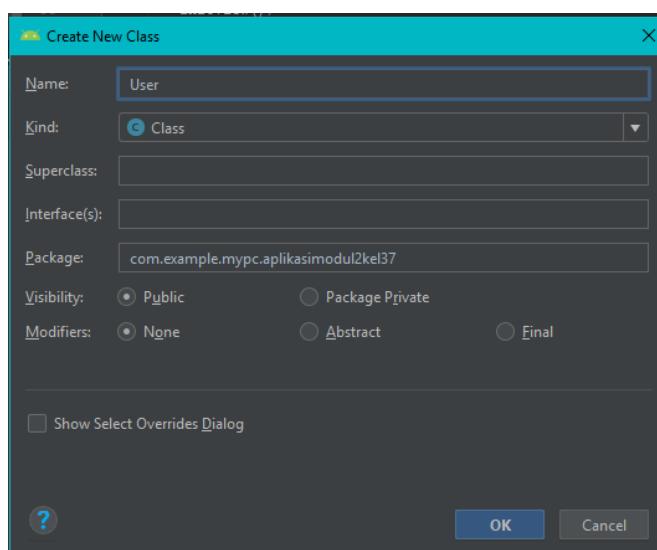
private Boolean loginData(){
    user = presenter.getLogin(username);

    if (password.equals(user.getPassword())){
        Log.e("loginData", "true");
        return true;
    } else {
        Log.e("loginData", "false");
        return false;
    }
}

private void initPreference() {
    SharedPreferences preferences =
getSharedPreferences("LoginPreference", Context.MODE_PRIVATE);
    SharedPreferences.Editor editor = preferences.edit();
    editor.putString("password", password);
    editor.putString("username", username);
    editor.commit();
    editor.apply();
}
}
}

```

19. Kemudian kita akan membuat sebuah class baru yang berfungsi untuk menampung segala kerperluan seorang user yang bersikan username dan password.
20. Kita namani class nya “User” (Di pemograman Bahasa java penulisan Casesensitive).



Gambar 3.23 Membuat class baru User

21. Setelah membuat class nya kita masukkan *sourcecode* nya

```
package com.example.mypc.aplikasimodul2kel37;

public class User {
    int id;
    static String username;
    static String password;

    public User() {
    }

    public User(int id, String username, String password) {
        this.id = id;
        this.username = username;
        this.password = password;
    }

    public int getId() {
        return id;
    }
}
```

```

public void setId(int id) {
    this.id = id;
}

public String getUsername() {
    return username;
}

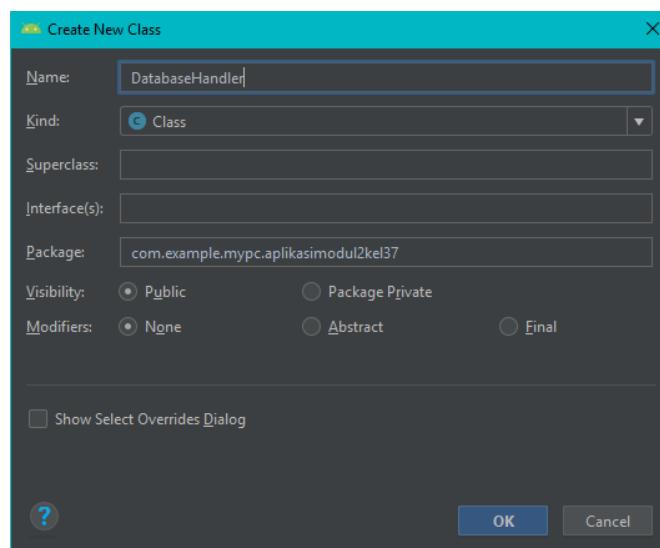
public void setUsername(String username) {
    this.username = username;
}

public String getPassword() {
    return password;
}

public void setPassword(String password) {
    this.password = password;
}
}

```

22. Kita membuat class baru lagi yang bernama “DatabaseHandler” yang berguna untuk operasi database create, update, update, dan delete.



Gambar 3.24 Membuat class baru DatabaseHandler

23. Kemudian, masukkan *sourcecode* DatabaseHandler nya

```

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;

```

```

import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

import java.util.ArrayList;
import java.util.List;

public class DatabaseHandler extends SQLiteOpenHelper {

    // static variable
    private static final int DATABASE_VERSION = 1;

    // Database name
    private static final String DATABASE_NAME = "modulandroid";

    // table name
    private static final String TABLE_MAHASISWA = "mahasiswa";

    // column tables
    private static final String KEY_ID = "id";
    private static final String KEY_NAME = "username";
    private static final String KEY_TALL = "password";

    public DatabaseHandler(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    //Create table
    @Override
    public void onCreate(SQLiteDatabase db) {
        String CREATE_MAHASISWA_TABLE = "CREATE TABLE " +
        TABLE_MAHASISWA + "("
            + KEY_ID + " INTEGER PRIMARY KEY," + KEY_NAME +
        " TEXT,"
            + KEY_TALL + " TEXT" + ")";
        db.execSQL(CREATE_MAHASISWA_TABLE);
    }

    // on Upgrade database
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion,
    int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_MAHASISWA);
        onCreate(db);
    }

    public void addUser(User usermodel){
        SQLiteDatabase db = getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put(KEY_NAME, usermodel.getUsername());
        values.put(KEY_TALL, usermodel.getPassword());
        db.insert(TABLE_MAHASISWA, null, values);
        db.close();
    }
}

```

```

}

public User getMahasiswa(int id) {
    SQLiteDatabase db = this.getReadableDatabase();

    Cursor cursor = db.query(TABLE_MAHASISWA, new String[]
{ KEY_ID,
        KEY_NAME, KEY_TALL }, KEY_ID + "=?",
        new String[] { String.valueOf(id) }, null,
        null, null);
    if (cursor != null)
        cursor.moveToFirst();

    User mahasiswa = new
User(Integer.parseInt(cursor.getString(0)),
        cursor.getString(1), cursor.getString(2));
    // return mahasiswa
    return mahasiswa;
}

public User getLogin(String username) {
    User mahasiswa;
    SQLiteDatabase db = this.getReadableDatabase();

    Cursor cursor = db.query(TABLE_MAHASISWA, new String[]
{ KEY_ID,
        KEY_NAME, KEY_TALL }, KEY_NAME + "=?",
        new String[] { username }, null, null, null,
        null);

    if (cursor != null) {
        cursor.moveToFirst();
    }

    int nomor = cursor.getCount();

    if(nomor>0){
        mahasiswa = new
User(Integer.parseInt(cursor.getString(0)),
        cursor.getString(1), cursor.getString(2));
    } else {
        mahasiswa = new User(0,
                "failed", "failed");
    }

    // return mahasiswa
    return mahasiswa;
}

// get All Record
public List<User> getAllRecord() {
    List<User> mahasiswaList = new ArrayList<User>();
}

```

```

// Select All Query
String selectQuery = "SELECT * FROM " +
TABLE_MAHASISWA;

SQLiteDatabase db = this.getWritableDatabase();
Cursor cursor = db.rawQuery(selectQuery, null);

if (cursor.moveToFirst()) {
    do {
        User userModels = new User();

        userModels.setId(Integer.parseInt(cursor.getString(0)));
        userModels.setUsername(cursor.getString(1));
        userModels.setPassword(cursor.getString(2));

        mahasiswaList.add(userModels);
    } while (cursor.moveToNext());
}

// return mahasiswa list
return mahasiswaList;
}

public int updateMahasiswa(User mahasiswa) {
    SQLiteDatabase db = this.getWritableDatabase();

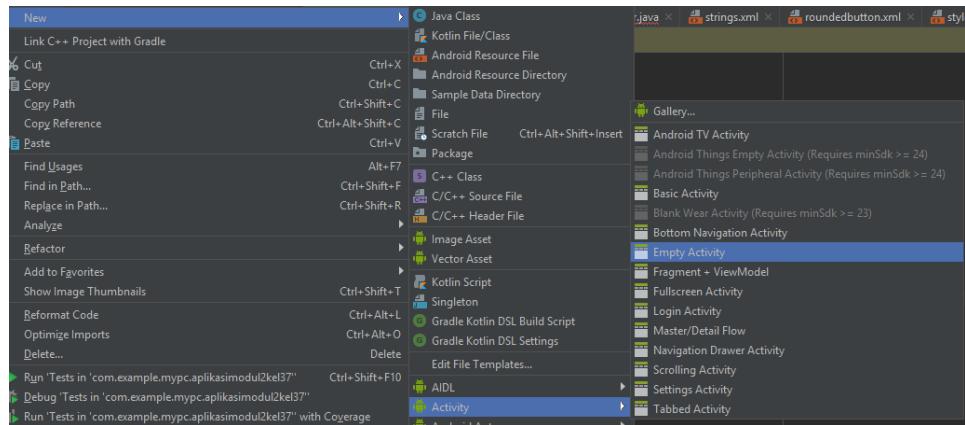
    ContentValues values = new ContentValues();
    values.put(KEY_NAME, mahasiswa.getUsername());
    values.put(KEY_TALL, mahasiswa.getPassword());

    // updating row
    return db.update(TABLE_MAHASISWA, values, KEY_ID + " = ?",
    new String[] {
String.valueOf(mahasiswa.getId()) });
}

public void deleteModel(User mahasiswa) {
    SQLiteDatabase db = this.getWritableDatabase();
    db.delete(TABLE_MAHASISWA, KEY_ID + " = ?",
    new String[] {
String.valueOf(mahasiswa.getId()) });
    db.close();
}
}

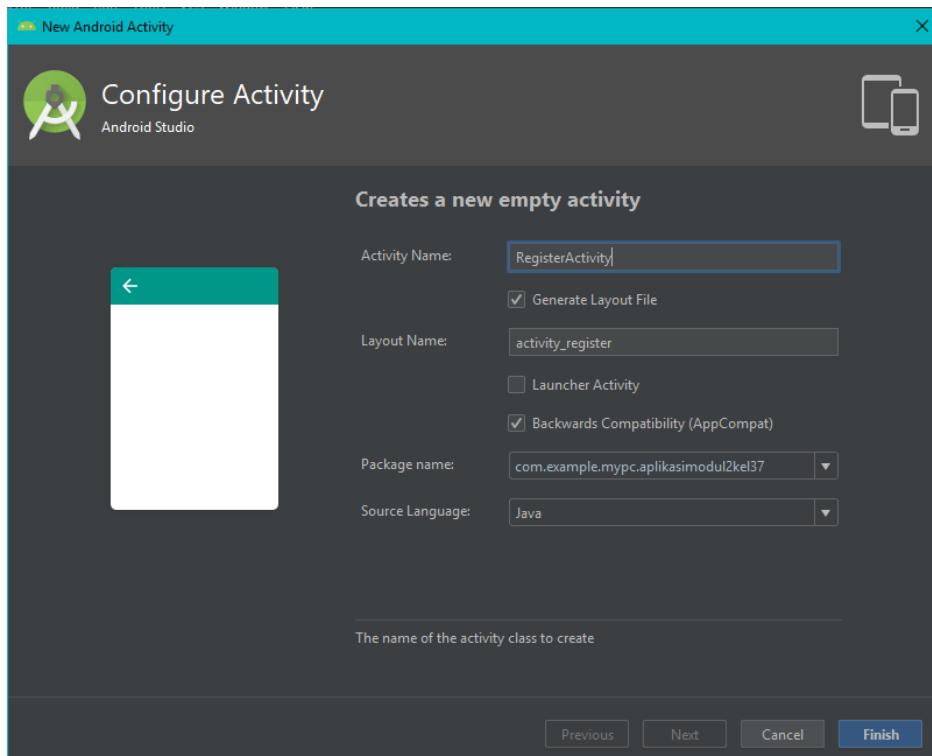
```

24. Untuk menambahkan seorang user dalam aplikasi, maka dibutuhkan suatu Register, kita akan membuat activity baru yang bernama Register Activity



Gambar 3.25 Membuat Empty Activity

25. Activity Name nya kita beri nama “RegisterActivity” dan Layout Name nya kita beri nama “activity_register” . Kemudian kalau sudah benar konfigurasinya maka klik tombol finish untuk menyelesaiakannya



Gambar 3.26 Membuat Register Activity

26. Setelah Activity nya terbuat, kita bikin design nya terlebih dahulu, melalui folder res → layout, berikut source code nya :

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/white"
    tools:context=".LoginActivity">

    <RelativeLayout
        android:id="@+id/rLLogin"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="60dp">

        <ImageView
            android:id="@+id/ivLogin"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:layout_gravity="center_horizontal"
            android:layout_marginLeft="130dp"
            android:src="@drawable/logolab"
            android:contentDescription="LOGO" />

        <android.support.v7.widget.AppCompatTextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="@string/regtitle"
            android:textAlignment="center"
            android:textSize="20sp"
            android:textStyle="bold"
            android:layout_marginTop="120dp" />

        <LinearLayout
            android:id="@+id/lLLogin"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_centerInParent="true"
            android:layout_marginTop="330dp"
            android:layout_marginEnd="50dp"
            android:orientation="vertical"
            android:padding="18dp">

            <EditText
                android:id="@+id/etUsernameRegister"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
```

```

        android:hint="@string/user"
        android:inputType="textEmailAddress"
        android:maxLines="1"
        android:singleLine="true" />

    <EditText
        android:id="@+id/etPasswordRegister"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="30dp"
        android:hint="@string/password"
        android:imeActionId="6"
        android:imeActionLabel="Sign In"
        android:imeOptions="actionUnspecified"
        android:inputType="textPassword"
        android:maxLines="1"
        android:singleLine="true" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <Button
            android:id="@+id/btRegister"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:background="@drawable/roundedbutton"
            android:text="@string/register"
            android:textColor="@color/white" />

    </LinearLayout>
</LinearLayout>
</RelativeLayout>
</ScrollView>

```

28. Selanjutnya kita masukkan *sourcecode* Regsiter Activity nya

```

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.widget.Button;
import android.widget.EditText;

public class RegisterActivity extends AppCompatActivity {

    private DatabaseHandler databaseHandler;
    private User usermodel;
    private EditText etUsernameRegister;
    private EditText etPasswordRegister;

```

```

private Button btRegister;
private String username,password;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_register);

    initView();
    register();
}

private void register() {
    btRegister.setOnClickListener(V ->
        initDataHandler()
    );
}

private void initView() {
    etUsernameRegister =
    findViewById(R.id.etUsernameRegister);
    etPasswordRegister =
    findViewById(R.id.etPasswordRegister);
    btRegister = findViewById(R.id.btRegister);
}

private void initUser() {
    username = etUsernameRegister.getText().toString();
    password = etPasswordRegister.getText().toString();

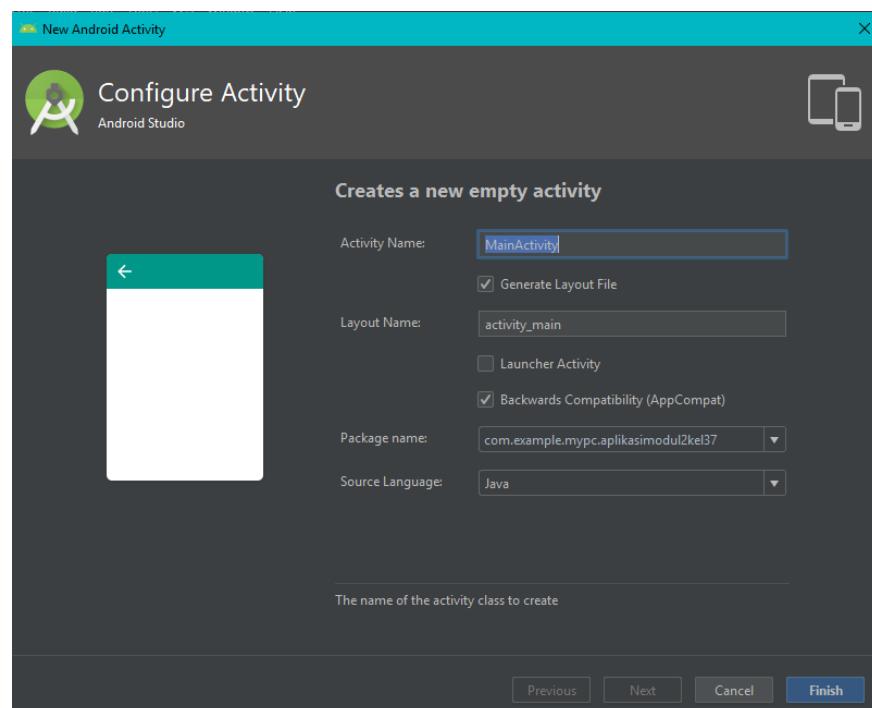
    usermodel = new User();
    usermodel.setUsername(username);
    usermodel.setPassword(password);
}

private void initDataHandler() {
    initUser();

    databaseHandler = new DatabaseHandler(this);
    databaseHandler.addUser(usermodel);
    User model = databaseHandler.getMahasiswa(1);
    Log.e("record", model.getUsername().toString());
    Intent admin = new Intent(RegisterActivity.this,
LoginActivity.class);
    startActivity(admin);
    finish();
}
}

```

29. Activity login dan register sudah kita buat, maka kita akan membuat Activity baru lagi, tahap membuat activity baru seperti langkah sebelumnya. Activity Name nya kita beri nama “MainActivity” dan Layout Name nya kita beri nama “activity_main” harap diperhatikan besar kecilnya penulisan nama.



Gambar 3.27 Membuat MainActivity

30. Masuk ke layout activity_main.xml, kemudian masukkan source code nya :

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <android.support.v7.widget.CardView
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:id="@+id/cardView"
        style="@style/CardView.Light"
        android:layout_width="320dp"
        android:layout_height="400dp"
        android:layout_marginBottom="8dp"
        app:layout_constraintBottom_toBottomOf="parent">
```

```
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.100000024">

    <android.support.constraint.ConstraintLayout
        xmlns:app="http://schemas.android.com/apk/res-auto"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

        <ImageView
            android:id="@+id/media_image"
            android:layout_width="0dp"
            android:layout_height="0dp"
            android:scaleType="fitCenter"
            android:src="@drawable/wall1"

        app:layout_constraintBottom_toTopOf="@+id/primary_text"
            app:layout_constraintDimensionRatio="H,16:9"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"

        app:layout_constraintVertical_chainStyle="packed" />

        <TextView
            android:id="@+id/primary_text"
            android:layout_width="0dp"
            android:layout_height="61dp"
            android:layout_marginStart="16dp"
            android:layout_marginTop="24dp"
            android:layout_marginEnd="16dp"
            android:text="@string/Header"

        android:textAppearance="@style/TextAppearance.AppCompat.Headline"
        android:textColor="#DE000000"

        app:layout_constraintBottom_toTopOf="@+id/tvNameMain"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"

        app:layout_constraintTop_toBottomOf="@+id/media_image" />

        <TextView
            android:id="@+id/tvNameMain"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginStart="16dp"
            android:layout_marginTop="0dp"
            android:layout_marginEnd="16dp"
            android:text="@string/name"
```

```
        android:textAppearance="@style/TextAppearance.AppCompat.Body1"
            android:textColor="#8A000000"
            android:textSize="15sp"
            app:layout_constraintBottom_toTopOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.0"
            app:layout_constraintStart_toStartOf="parent"

        app:layout_constraintTop_toBottomOf="@+id/primary_text" />

        <TextView
            android:id="@+id/tvdescription"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginTop="32dp"
            android:layout_marginBottom="8dp"
            android:text="@string/description"

        android:textAppearance="@style/TextAppearance.AppCompat.Body1"
            android:textColor="#8A000000"
            android:textSize="20sp"
            app:layout_constraintBottom_toBottomOf="parent"

        app:layout_constraintEnd_toEndOf="@+id/tvNameMain"
            app:layout_constraintHorizontal_bias="0.0"

        app:layout_constraintStart_toStartOf="@+id/tvNameMain"

        app:layout_constraintTop_toTopOf="@+id/tvNameMain"
            app:layout_constraintVertical_bias="1.0" />

    </android.support.constraint.ConstraintLayout>

</android.support.v7.widget.CardView>

<Button
    android:id="@+id/btnLogoutMain"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="6dp"
    android:layout_marginBottom="8dp"
    android:background="@drawable/roundedbutton"
    android:minWidth="0dp"
    android:paddingStart="8dp"
    android:paddingEnd="8dp"
    android:text="@string/logout"
    android:textColor="#de101010"
```

```

    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="@+id/cardView"
    app:layout_constraintTop_toBottomOf="@+id/cardView"
    app:layout_constraintVertical_bias="0.753" />

</android.support.constraint.ConstraintLayout>

```

31. Setelah layout sudah kita buat, maka kita masuk ke class MainActivity.java, dan masukkan source code nya :

```

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    private DatabaseHandler databaseHandler;
    private User usermodel;
    private TextView tvNameMain;
    private Button btnLogoutMain;
    private Button exit;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        initView();
        initPreference();
        logout();
    }

    @Override
    public void onBackPressed() {
        showDialog();
    }

    private void initView() {
        tvNameMain = findViewById(R.id.tvNameMain);
        btnLogoutMain = findViewById(R.id.btnLogoutMain);
    }
}

```

```

        exit = findViewById(R.id.btnExitMain);
    }

    private void initPreference() {
        SharedPreferences preferences =
getSharedPreferences("LoginPreference", MODE_PRIVATE);
        String username = preferences.getString("username", "");

        tvNameMain.setText(username);
    }

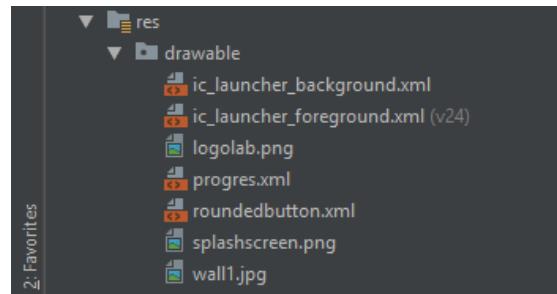
    private void deletePreference(){
        SharedPreferences preferences =
getSharedPreferences("LoginPreference", MODE_PRIVATE);
        preferences.edit().remove("username").commit();
        preferences.edit().remove("password").commit();
    }

    private void logout() {
        exit.setOnClickListener(view -> showAlertDialog());
    }

    public void showAlertDialog() {
        new AlertDialog.Builder(this)
            .setMessage("Apa kalian ingin Logout?")
            .setCancelable(false)
            .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog,
int id) {
                    deletePreference();
                    Intent login = new
Intent(MainActivity.this, LoginActivity.class);
                    startActivity(login);
                    finish();
                }
            })
            .setNegativeButton("No", null)
            .show();
    }
}

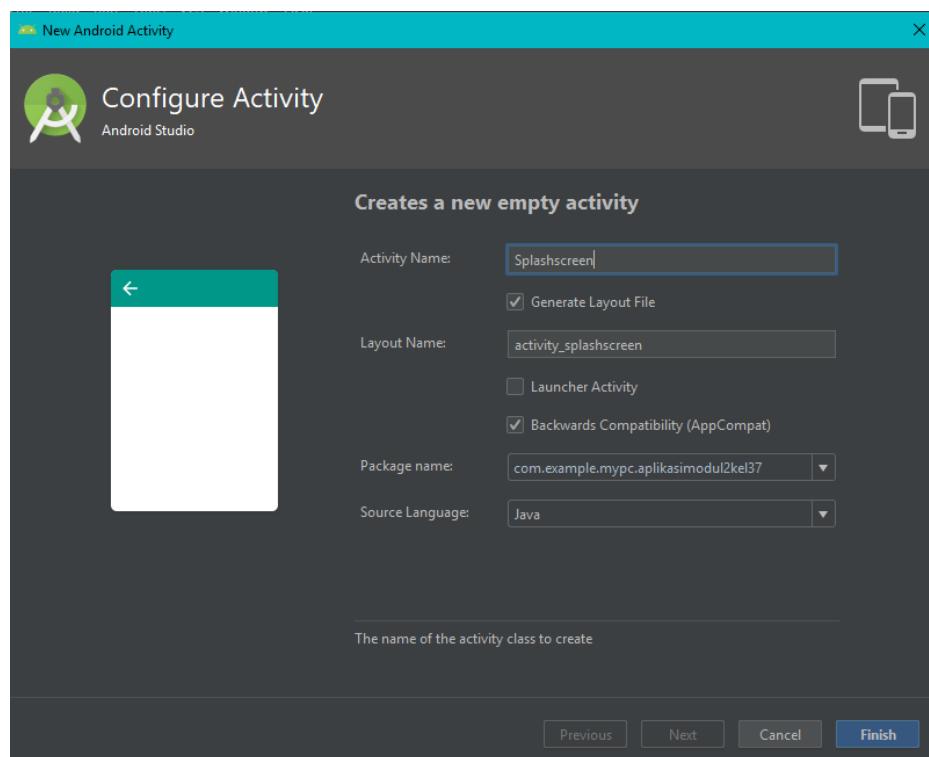
```

32. Kita akan membuat Splashscreen pada tampilan awal aplikasi saat dijalankan, masukkan gambar / icon yang akan digunakan dan menggunakan type .png agar dapat dibaca oleh system. Lakukan drag and drop pada folder berikut res→ drawable.



Gambar 3.28 File gambar splashscreen di res-drawable

33. Langkah selanjutnya, membuat activity baru seperti langkah sebelumnya yang sudah kita buat, dengan nama Activity nya “Splashscreen” dan nama Layout nya “activity_splashscreen”



Gambar 3.29 Membuat Activity Splashscreen

34. Masuk ke bagian layout activiy_splashcreen.xml, dan kemudian masukkan source code nya :

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
```

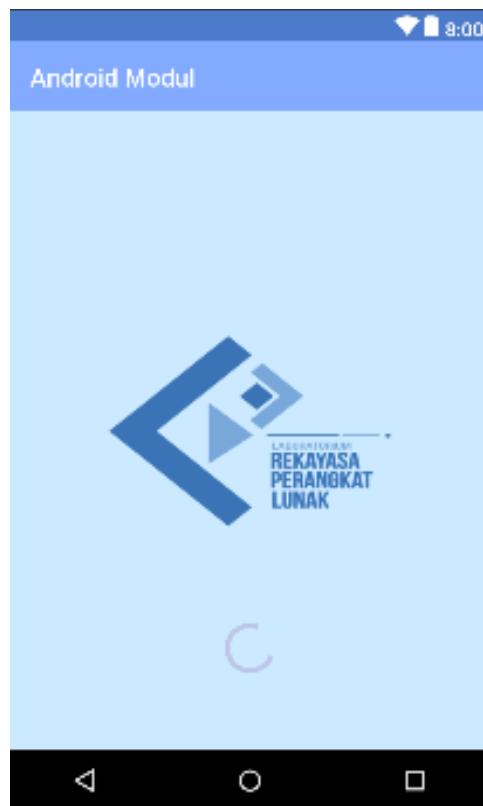
```
        android:layout_width="match_parent"
        android:layout_height="match_parent" >

    <ImageView
        android:src="@drawable/splashscreen"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:scaleType="fitXY"/>

    <ProgressBar
        android:id="@+id/progressBar1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="58dp" />

</RelativeLayout>
```

35. Maka akan muncul tampilannya kurang lebih seperti ini



Gambar 3.30 Tampilan Splashscreen

36. Kemudian masukkan source code SplashScreen.java nya

```

import android.content.SharedPreferences;
import android.support.v7.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;
import android.view.Window;
import android.view.WindowManager;

public class SplashScreen extends AppCompatActivity {

    private static int splashInterval = 2000;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        this.requestWindowFeature(Window.FEATURE_NO_TITLE);

        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
                            WindowManager.LayoutParams.FLAG_FULLSCREEN);

        setContentView(R.layout.activity_splashscreen);

        handler();
    }

    private void handler(){
        new Handler().postDelayed(() -> {
            if (initPreference()){
                Intent main = new Intent(Splashscreen.this,
MainActivity.class);
                startActivity(main);
                finish();
            } else {
                Intent i = new Intent(Splashscreen.this,
LoginActivity.class);
                startActivity(i);
                finish();
            }
        }, splashInterval);
    }

    private Boolean initPreference() {
        SharedPreferences preferences =
getSharedPreferences("LoginPreference", MODE_PRIVATE);
        String username = preferences.getString("username",
"");

        if (username.isEmpty()){
            return false;
        }
    }
}

```

```

        return true;
    }
}
```

37. Setelah itu, kita masuk ke dalam folder Manifest nya, melalui app→Manifest→AndroidManifest.xml. Kemudian masukkan *sourcecode* nya :

```

<?xml version="1.0" encoding="utf-8" ?>
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.mypc.aplikasimodul2kel37">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <!-- Splash screen -->

        <activity
            android:name="com.example.mypc.aplikasimodul2kel37.Splashscreen"
            android:label="@string/app_name"
            android:theme="@style/Splashscreen"
            android:screenOrientation="portrait"
            android:noHistory="true">
            <intent-filter>
                <action
                    android:name="android.intent.action.MAIN" />

                <category
                    android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".RegisterActivity" />
        <activity android:name=".MainActivity" />
        <activity android:name=".LoginActivity"></activity>
    </application>

</manifest>
```

38. Tambahkan beberapa *sourcecode* yang ada di styles.xml untuk mengatur tampilan Splash Screen nya. Berikut di bawah *sourcecode*-nya:

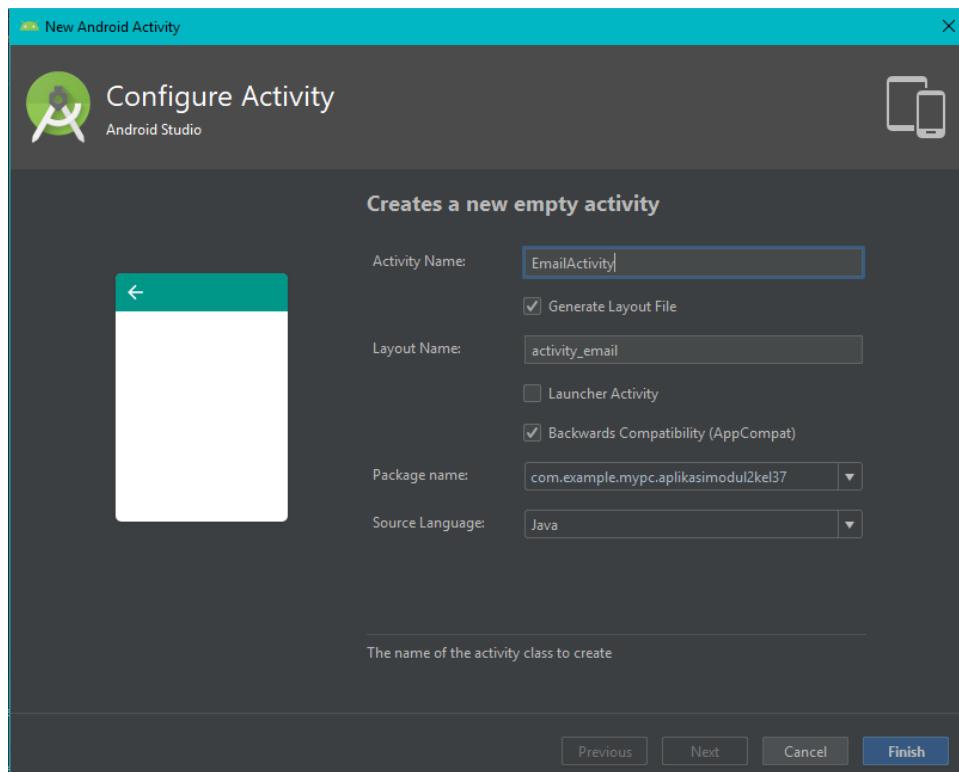
```
<style name="Splashscreen"
```

```

parent="Theme.AppCompat.Light.DarkActionBar">
    <item name="windowActionBar">false</item>
    <item name="windowNoTitle">true</item>
</style>

```

39. Berikutnya kita membuat sebuah *activity* bernama EmailActivity dengan nama layoutnya activity_email



Gambar 3.31 Membuat EmailActivity

40. Kemudian, masukkan *sourcecode* pada EmailActivity.java seperti di bawah:

```

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class EmailActivity extends AppCompatActivity {

    Button buttonSend;
    EditText textTo;

```

```

EditText textSbj;
EditText textMsg;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_email);

    buttonSend = (Button) findViewById(R.id.buttonSend);
    textTo = (EditText) findViewById (R.id.editTextTo);
    textSbj = (EditText) findViewById (R.id.editTextSbj);
    textMsg = (EditText) findViewById (R.id.editTextMsg);

    buttonSend.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View view) {

        String to = textTo.getText().toString();
        String subject = textSbj.getText().toString();
        String message = textMsg.getText().toString();

        Intent email = new Intent(Intent.ACTION_SEND);
        email.putExtra(Intent.EXTRA_EMAIL, new String[] {
to});
        email.putExtra(Intent.EXTRA_SUBJECT, subject);
        email.putExtra(Intent.EXTRA_TEXT, message);

        email.setType("message/rfc822");

        startActivity(Intent.createChooser(email,
"Choose an Email client :"));
    }
});

@Override
public void onBackPressed() {
    back();
}

private void back() {
    Intent reg = new Intent(this, MainActivity.class);
    startActivity(reg);
    finish();
}
}

```

41. Berikutnya masukkan *sourcecode* pada activity_mail.xml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".EmailActivity">

    <EditText
        android:id="@+id/editTextSbj"
        android:layout_width="0dp"
        android:layout_height="44dp"
        android:layout_marginStart="30dp"
        android:ems="10"
        android:inputType="textEmailSubject"
        android:hint="Subject "
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        tools:layout_constraintLeft_creator="1"
        tools:layout_constraintTop_creator="1"
        tools:layout_constraintRight_creator="1"
        android:layout_marginEnd="30dp"
        android:layout_marginTop="25dp"
        app:layout_constraintTop_toBottomOf="@+id/editTextTo"
    />

    <EditText
        android:id="@+id/editTextTo"
        android:layout_width="0dp"
        android:layout_height="44dp"
        android:hint="To "
        android:fontFamily="@string/app_name"
        android:inputType="textEmailAddress"
        tools:layout_constraintTop_creator="1"
        android:layout_marginTop="25dp"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        tools:layout_constraintRight_creator="1"
        android:layout_marginStart="30dp"
        android:layout_marginEnd="30dp"
        tools:layout_constraintLeft_creator="1"
        app:layout_constraintTop_toTopOf="parent" />

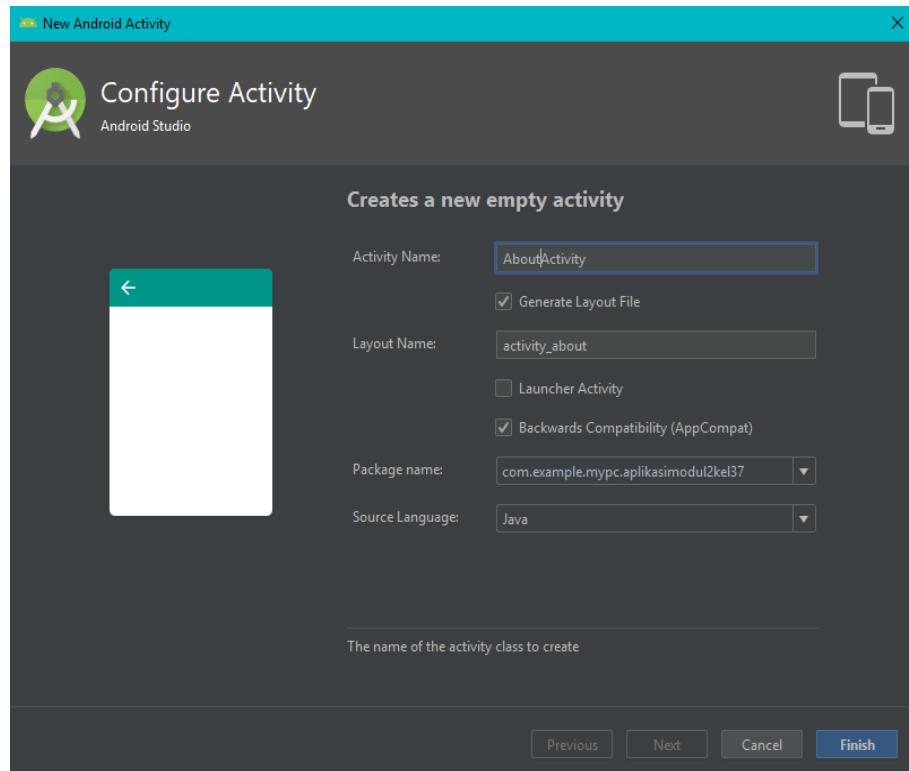
    <EditText
        android:id="@+id/editTextMsg"
        android:layout_width="291dp"
        android:layout_height="176dp"
        android:layout_marginStart="30dp"
        android:ems="10"
        android:inputType="textMultiLine"
```

```
        android:hint="Message "
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        tools:layout_constraintLeft_creator="1"
        tools:layout_constraintTop_creator="1"
        tools:layout_constraintRight_creator="1"
        android:layout_marginEnd="30dp"
        android:layout_marginTop="25dp"
        app:layout_constraintTop_toBottomOf="@+id/editTextSbj"
        app:layout_constraintHorizontal_bias="0.516"
        android:layout_marginLeft="16dp"
        android:layout_marginRight="16dp"
        tools:layout_editor_absoluteX="62dp" />

    <Button
        android:id="@+id/buttonSend"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="30dp"
        android:layout_marginLeft="30dp"
        android:layout_marginRight="30dp"
        android:layout_marginStart="30dp"
        android:background="#26c6da"
        android:text="Send"
        android:textColor="@android:color/background_light"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        tools:layout_constraintBottom_creator="1"
        tools:layout_constraintLeft_creator="1"
        tools:layout_constraintRight_creator="1"
        tools:layout_constraintTop_creator="1"
        android:layout_marginTop="25dp"
        app:layout_constraintTop_toBottomOf="@+id/editTextMsg"
        app:layout_constraintVertical_bias="0.251" />

</android.support.constraint.ConstraintLayout>
```

42. Selanjutnya kita membuat *activity* baru lagi bernama AboutActivity dengan nama layoutnya activity_about



Gambar 3.32 Membuat AboutActivity

43. *Sourcecode* untuk AboutActivity.java

```

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class AboutActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_about);
    }

    @Override
    public void onBackPressed() {
        back();
    }

    private void back() {
        Intent reg = new Intent(this, MainActivity.class);
        startActivity(reg);
        finish();
    }
}

```

44. Selanjutnya, masukkan *sourcecode* pada activity_about.xml seperti di bawah:

```

<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:context=""
com.example.mypc.aplikasimodul2kel37.AboutActivity">

    <ImageView
        android:id="@+id/ivpraktikan1"
        android:layout_width="126dp"
        android:layout_height="145dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@drawable/logolab" />

    <android.support.v7.widget.CardView
        xmlns:android="http://schemas.android.com/apk/res/android"
            android:id="@+id/cardView"
            style="@style/CardView.Light"
            android:layout_width="320dp"
            android:layout_height="80dp"
            android:layout_marginTop="8dp"
            android:layout_marginBottom="8dp"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"

        app:layout_constraintTop_toBottomOf="@+id/ivpraktikan1"
            app:layout_constraintVertical_bias="0.0">

            <android.support.constraint.ConstraintLayout
                xmlns:app="http://schemas.android.com/apk/res-auto"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content">

                <TextView
                    android:id="@+id/twnama1"
                    android:layout_width="97dp"
                    android:layout_height="24dp"
                    android:layout_marginLeft="24dp"
                    android:layout_marginTop="10dp"
                    android:layout_marginEnd="196dp"
                    android:text="Nama kamu"
```

```
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/twnim1"
        android:layout_width="109dp"
        android:layout_height="26dp"
        android:layout_marginTop="12dp"
        android:text="Nim kamu"

    app:layout_constraintStart_toStartOf="@+id/twnama1"
    app:layout_constraintTop_toBottomOf="@+id/twnama1" />

</android.support.constraint.ConstraintLayout>

</android.support.v7.widget.CardView>

<ImageView
    android:id="@+id/ivpraktikan2"
    android:layout_width="126dp"
    android:layout_height="145dp"
    android:layout_marginTop="16dp"
    app:layout_constraintEnd_toEndOf="@+id/ivpraktikan1"
    app:layout_constraintHorizontal_bias="0.0"

    app:layout_constraintStart_toStartOf="@+id/ivpraktikan1"
    app:layout_constraintTop_toBottomOf="@+id/cardView"
    app:srcCompat="@drawable/logolab" />

<android.support.v7.widget.CardView
    android:id="@+id/cardView2"
    style="@style/CardView.Light"
    android:layout_width="320dp"
    android:layout_height="80dp"
    android:layout_marginTop="8dp"
    android:layout_marginBottom="8dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="@+id/cardView"
    app:layout_constraintStart_toStartOf="@+id/cardView"

    app:layout_constraintTop_toBottomOf="@+id/ivpraktikan2">

    <android.support.constraint.ConstraintLayout
    xmlns:app="http://schemas.android.com/apk/res-auto"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <TextView
            android:id="@+id/twnama2"
```

```

        android:layout_width="97dp"
        android:layout_height="24dp"
        android:layout_marginLeft="24dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="196dp"
        android:text="Nama kamu"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/twnim2"
        android:layout_width="115dp"
        android:layout_height="32dp"
        android:layout_marginTop="8dp"
        android:text="Nim kamu"

    app:layout_constraintStart_toStartOf="@+id/twnama2"
    app:layout_constraintTop_toBottomOf="@+id/twnama2" />

</android.support.constraint.ConstraintLayout>

</android.support.v7.widget.CardView>

</android.support.constraint.ConstraintLayout>

```

45. Selanjutnya kita diharuskan menambah method pada MainActivity.java, berikut di bawah ini adaah *sourcecode*-nya:

```

private void email() {
    btnEmail.setOnClickListener(v -> {
        Intent reg = new Intent(this, EmailActivity.class);
        startActivity(reg);
        finish();
    });
}

private void about() {
    btnAboutMain.setOnClickListener(v -> {
        Intent reg = new Intent(this, AboutActivity.class);
        startActivity(reg);
        finish();
    });
}

```

3.5 Analisa Hasil Percobaan

3.5.1 Membuat LoginActivity



Gambar 3.33 Tampilan LoginActivity

Pada percobaan pertama halaman login akan terlihat seperti pada gambar di atas dengan *Source code* activity_login.xml di bawah ini:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/white"
    tools:context=".LoginActivity">

    <RelativeLayout
        android:id="@+id/rlLogin"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="60dp">

        <ImageView
            android:id="@+id/ivLogin"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:layout_gravity="center_horizontal"
            android:layout_marginLeft="130dp"
            android:src="@drawable/logolab"
            android:contentDescription="LOGO" />
    
```

```
<android.support.v7.widget.AppCompatTextView  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="@string/tittle"  
    android:textAlignment="center"  
    android:textSize="20sp"  
    android:textStyle="bold"  
    android:layout_marginTop="120dp" />  
  
<LinearLayout  
    android:id="@+id/l1Login"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:layout_centerInParent="true"  
    android:layout_marginTop="320dp"  
    android:layout_marginEnd="50dp"  
    android:orientation="vertical"  
    android:padding="18dp">  
  
<EditText  
    android:id="@+id/etUsernameLogin"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:hint="@string/user"  
    android:inputType="textEmailAddress"  
    android:maxLines="1"  
    android:singleLine="true" />  
  
<EditText  
    android:id="@+id/etPasswdLogin"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:layout_marginBottom="30dp"  
    android:hint="@string/password"  
    android:imeActionId="6"  
    android:imeActionLabel="Sign In"  
    android:imeOptions="actionUnspecified"  
    android:inputType="textPassword"  
    android:maxLines="1"  
    android:singleLine="true" />  
  
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content">  
  
<Button  
    android:id="@+id/btnSignUpLogin"  
    android:layout_width="0dp"  
    android:layout_height="wrap_content"  
    android:layout_weight="1"
```

```

        android:background="@drawable/roundedbutton"
        android:text="@string/register"
        android:textColor="@color/white"/>

    <LinearLayout
        android:layout_width="2dp"
        android:layout_height="2dp"
        android:layout_weight="0"

    android:orientation="vertical"></LinearLayout>

    <Button
        android:id="@+id/btnSignInLogin"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:background="@drawable/roundedbutton"
        android:text="@string/login"
        android:textColor="@color/white" />
    </LinearLayout>
</LinearLayout>
</RelativeLayout>
</ScrollView>
```

Source code diatas digunakan untuk membuat tampilan pada login dengan Bahasa XML. Tampilannya terdiri dari *image*, *button*, *textfield*. Ada 2 cara yang dapat dilakukan dalam proses merancang tampilan dengan Android Studio yaitu, dengan melakukan *coding* keseluruhan rancangan yang diinginkan atau dengan metode *drag and drop*. Pada *activity* ini, dengan menggunakan kodingan yang berisi kodingan awalnya `<?xml version="1.0" encoding="utf-8" ?>` merupakan inisialisasi awal XML dan versi yang digunakan yaitu 1.0 dan baris ini optional bisa ditulis atau tidak, jika iya maka harus diletakkan diawal dokumen. Kemudian *tag ScrollView* dipakai supaya user bisa melakukan scroll secara vertikal pada program. *Tag* ini merupakan *tag root* dari XML. Kemudian *tag* selanjutnya *RelativeLayout* yang berguna untuk pengaturan layout pada android dimana setiap komponen dapat memposisikan dirinya dengan bergantung pada komponen lain. Selanjutnya *Tag ImageView* ini berfungsi untuk menampilkan gambar, dimana lebar gambar diatur di `android:layout_width="100dp"`, sedangkan untuk tinggi gambar diatur pada `android:layout_height="100dp"`, posisi dari gambar yang bergantung

pada sumbu X dan Y diatur di `android:layout_gravity="center_horizontal"` dan gambar diakses dari `android:src="@drawable/logolab"`. Tag selanjutnya `LinearLayout` yang berfungsi untuk pengaturan layout dimana pengaturannya komponennya dapat rapi secara vertical dan horizontal. Tag `EditText` yang pertama ini digunakan untuk memasukan username kemudian yang kedua untuk mengisi password. Kemudian tag `button` yang pertama bertulisan resigister dengan baris koding `android:text="@string/register"`. Button yang kedua bertuliskan login dengan baris koding untuk mengatur nya `android:text="@string/login"`.

Selanjutnya agar halaman ini dapat bekerja, kita perlu mengatur *back-end language* pada halaman ini yaitu Java pada Android Studio. *Source code* diinput pada file `LoginActivity.java` seperti di bawah:

```
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.Log;
import android.util.Printer;
import android.view.KeyEvent;
import android.view.View;
import android.view.inputmethod.EditorInfo;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import java.security.PrivateKey;

public class LoginActivity extends AppCompatActivity {

    private EditText etUsernameLogin, etPasswdLogin;
    private Button btnSignUpLogin, btnSignInLogin;
    String username, password;
    private User user;
    private DatabaseHandler presenter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
```

```

        initView();
        initData();
        login();
        register();
    }

    private void initView() {
        etUsernameLogin = findViewById(R.id.etUsernameLogin);
        etPasswdLogin = findViewById(R.id.etPasswdLogin);
        btnSignInLogin = findViewById(R.id.btnSignInLogin);
        btnSignUpLogin = findViewById(R.id.btnSignUpLogin);
    }

    private void register() {
        btnSignUpLogin.setOnClickListener(v -> {
            Intent reg = new Intent(this, RegisterActivity.class);
            startActivity(reg);
            finish();
        });
    }

    private void initData() {
        user = new User();
        presenter = new DatabaseHandler(this);
    }

    private Boolean validation() {
        username = etUsernameLogin.getText().toString();
        password = etPasswdLogin.getText().toString();

        if(username.isEmpty()){
            Toast.makeText(this, "Isikan username",
Toast.LENGTH_SHORT).show();
            Log.e("Validation","false");
            return false;
        }

        if(password.isEmpty()){
            Toast.makeText(this, "Isikan Password",
Toast.LENGTH_SHORT).show();
            Log.e("Validation","false");
            return false;
        }

        Log.e("Validation","true");
        return true;
    }

    private void login() {
        btnSignInLogin.setOnClickListener(v -> actLogin());
    }
}

```

```

private void actLogin(){
    if(validation()){
        if(loginData()){
            Log.e("actLogin","true");
            initPreference();
            Intent main = new Intent(this,
MainActivity.class);
            startActivity(main);
            finish();
        } else {
            Log.e("actLogin","false");
            Toast.makeText(this, "Login gagal",
Toast.LENGTH_LONG).show();
        }
    }
}

private Boolean loginData(){
    user = presenter.getLogin(username);

    if (password.equals(user.getPassword())){
        Log.e("loginData", "true");
        return true;
    } else {
        Log.e("loginData", "false");
        return false;
    }
}

private void initPreference() {
    SharedPreferences preferences =
getSharedPreferences("LoginPreference", Context.MODE_PRIVATE);
    SharedPreferences.Editor editor = preferences.edit();
    editor.putString("password", password);
    editor.putString("username", username);
    editor.commit();
    editor.apply();
}
}

```

Bagian awal pada *source code* diatas berisi library-library untuk mendukung fitur-fitur yang akan digunakan pada halaman LoginActivity. Kemudian pendeklarasian variabel yaitu username dan password dengan tipe data String, kemudian etUsernameLogin dan etPasswordLogin yang diambil dari *tag* XML EditText, btnSignUpLogin dan btnSignInLogin yang diambil dari *tag* XML Button, user dengan *modifier private* diambil dari Class User, presenter yang dibuat dari class DatabaseHandler. *Source code* terdiri dari banyak method, method yang

pertama adalah Oncreate dengan modifier protected dan metode akses void (non-return). Pada method ini terletak semua method yang akan diinisialisasi yakni initView(), initData(), login(), dan register(). Method initView merupakan method yang berfungsi untuk menginisialisasi tampilan pada halaman LoginActivity. Method register dengan modifier private dan metode akses void(non-return) berisikan

```

btnSignUpLogin.setOnClickListener(v -> {
    Intent reg = new Intent(this, RegisterActivity.class);
    startActivity(reg); finish(); });

```

yang berguna untuk memproses button register. Ketika button register (btnSignUpLogin) di klik maka activity akan berpindah ke RegisterActivity. Method initData bertugas untuk melakukan inisialisasi data yang yang diambil dari user dan class DatabaseHandler. Kemudian method validation yang betipe Boolean sehingga hanya mempunyai 2 hasil keluaran yaitu true or false. Method ini yang akan mengecek apakah username dan password sudah sesuai apa belum. Nantinya jika username tidak diisi maka akan muncul notifikasi “Isikan username”, jika password belum diisi maka akan muncul notifikasi “Isikan password”, dan jika sudah terisi semua maka akan validation akan bernilai true. Method login berisi

```

btnSignInLogin.setOnClickListener(v -> actLogin());

```

ketika button login di click maka activity baru akan berpindah ke actLogin. Dan diproses pada method actLogin yang berisi validasi dari username dan password jika semua proses sudah sesuai maka akan berpindah ke MainActivity. Kemudian method loginData yang berisi pengecekan password dimana baris kodingannya

```

user = presenter.getLogin(username);
if(password.equals(user.getPassword())) Log.e("loginData" "true");
return true;
} else {
Log.e("loginData", "false");
return false;
}.

```

Method terakhir initPreference yang mengambil hasil username dan password yang tersimpan pada database.

```
package com.example.mypc.aplikasimodul2kel37;
```

```

public class User {
    int id;
    static String username;
    static String password;

    public User() {
    }

    public User(int id, String username, String password) {
        this.id = id;
        this.username = username;
        this.password = password;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }
}

```

Source code diatas diisi pada user.java yang berisi package com.example.mypc.aplikasimodul2kel137; package ini sesuai dengan nama package saat awal projek dibuat. Kodingan ini berisi variable id bertipe integer, username bertipe data string, dan password bertipe data string. Kemudian constructor user dengan parameter int id, string username, string password. Method ini berhubungan dengan segala kebutuhan user dari id, username dan juga password.

```

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;

```

```

import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

import java.util.ArrayList;
import java.util.List;

public class DatabaseHandler extends SQLiteOpenHelper {

    // static variable
    private static final int DATABASE_VERSION = 1;

    // Database name
    private static final String DATABASE_NAME = "modulandroid";

    // table name
    private static final String TABLE_MAHASISWA = "mahasiswa";

    // column tables
    private static final String KEY_ID = "id";
    private static final String KEY_NAME = "username";
    private static final String KEY_TALL = "password";

    public DatabaseHandler(Context context){
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    //Create table
    @Override
    public void onCreate(SQLiteDatabase db) {
        String CREATE_MAHASISWA_TABLE = "CREATE TABLE " +
        TABLE_MAHASISWA + "("
            + KEY_ID + " INTEGER PRIMARY KEY," + KEY_NAME + " "
            + TEXT, "
            + KEY_TALL + " TEXT" + ")";
        db.execSQL(CREATE_MAHASISWA_TABLE);
    }

    // on Upgrade database
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_MAHASISWA);
        onCreate(db);
    }

    public void addUser(User usermodel){
        SQLiteDatabase db = getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put(KEY_NAME, usermodel.getUsername());
        values.put(KEY_TALL, usermodel.getPassword());
        db.insert(TABLE_MAHASISWA, null, values);
    }
}

```

```

        db.close();
    }

    public User getMahasiswa(int id) {
        SQLiteDatabase db = this.getReadableDatabase();

        Cursor cursor = db.query(TABLE_MAHASISWA, new String[] {
KEY_ID,
                KEY_NAME, KEY_TALL }, KEY_ID + "=?",
                new String[] { String.valueOf(id) }, null, null,
null, null);
        if (cursor != null)
            cursor.moveToFirst();

        User mahasiswa = new
User(Integer.parseInt(cursor.getString(0)),
                cursor.getString(1), cursor.getString(2));
        // return mahasiswa
        return mahasiswa;
    }

    public User getLogin(String username) {
        User mahasiswa;
        SQLiteDatabase db = this.getReadableDatabase();

        Cursor cursor = db.query(TABLE_MAHASISWA, new String[] {
KEY_ID,
                KEY_NAME, KEY_TALL }, KEY_NAME + "=?",
                new String[] { username }, null, null, null,
null);

        if (cursor != null) {
            cursor.moveToFirst();
        }

        int nomor = cursor.getCount();

        if(nomor>0){
            mahasiswa = new
User(Integer.parseInt(cursor.getString(0)),
                cursor.getString(1), cursor.getString(2));
        } else {
            mahasiswa = new User(0,
                    "failed", "failed");
        }

        // return mahasiswa
        return mahasiswa;
    }

    // get All Record
    public List<User> getAllRecord() {

```

```

List<User> mahasiswaList = new ArrayList<User>();
// Select All Query
String selectQuery = "SELECT * FROM " + TABLE_MAHASISWA;

SQLiteDatabase db = this.getWritableDatabase();
Cursor cursor = db.rawQuery(selectQuery, null);

if (cursor.moveToFirst()) {
    do {
        User userModels = new User();

        userModels.setId(Integer.parseInt(cursor.getString(0)));
        userModels.setUsername(cursor.getString(1));
        userModels.setPassword(cursor.getString(2));

        mahasiswaList.add(userModels);
    } while (cursor.moveToNext());
}

// return mahasiswa list
return mahasiswaList;
}

public int updateMahasiswa(User mahasiswa) {
    SQLiteDatabase db = this.getWritableDatabase();

    ContentValues values = new ContentValues();
    values.put(KEY_NAME, mahasiswa.getUsername());
    values.put(KEY_TALL, mahasiswa.getPassword());

    // updating row
    return db.update(TABLE_MAHASISWA, values, KEY_ID + " = ?",
        new String[] { String.valueOf(mahasiswa.getId()) });
}

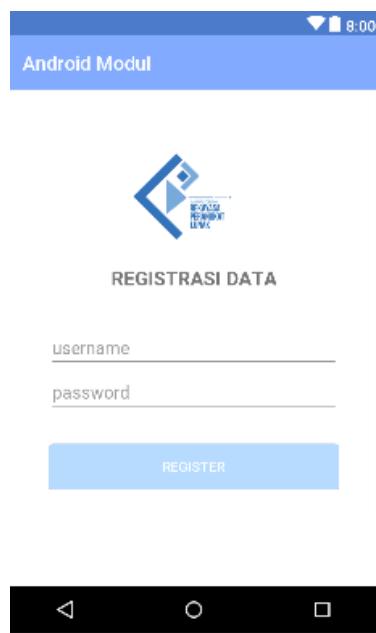
public void deleteModel(User mahasiswa) {
    SQLiteDatabase db = this.getWritableDatabase();
    db.delete(TABLE_MAHASISWA, KEY_ID + " = ?",
        new String[] { String.valueOf(mahasiswa.getId()) });
    db.close();
}
}

```

Pada DatabaseHandler.java yang berisi source code seperti diatas digunakan untuk mengatur segala yang berhubungan dengan database. Database yang dipakai dengan nama modulandroid baris kodingannya private static final String DATABASE_NAME = "modulandroid";. Dan nama tabelnya mahasiswa baris

kodingannya private static final String TABLE_MAHASISWA = "mahasiswa"; . pada *source code* ini berisi tambah user, update mahasiswa, Delete mahasiswa. Pada method addUser akan menambahkan data ke database dengan kodingan SQLiteDatabase db = getWritableDatabase(); ContentValues values = new ContentValues(); values.put(KEY_NAME, usermodel.getUsername()); values.put(KEY_TALL, usermodel.getPassword()); db.insert(TABLE_MAHASISWA, null, values); db.close();. Pada method updateMahasiswa akan mengupdate data dari tabel mahasiswa yang sudah dimasukin sebelumnya. Kemudian method deleteModel untuk menghapus nilai dalam tabel sesuai dengan id yang terpilih.

3.5.2 Membuat RegisterActivity



Gambar 3.34 Tampilan RegisterActivity

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/white"
    tools:context=".LoginActivity">

    <RelativeLayout
```

```
    android:id="@+id/rlLogin"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="60dp">

    <ImageView
        android:id="@+id/ivLogin"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:layout_gravity="center_horizontal"
        android:layout_marginLeft="130dp"
        android:src="@drawable/logolab"
        android:contentDescription="LOGO" />

    <android.support.v7.widget.AppCompatTextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/regtitle"
        android:textAlignment="center"
        android:textSize="20sp"
        android:textStyle="bold"
        android:layout_marginTop="120dp" />

    <LinearLayout
        android:id="@+id/l1Login"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:layout_marginTop="330dp"
        android:layout_marginEnd="50dp"
        android:orientation="vertical"
        android:padding="18dp">

        <EditText
            android:id="@+id/etUsernameRegister"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="@string/user"
            android:inputType="textEmailAddress"
            android:maxLines="1"
            android:singleLine="true" />

        <EditText
            android:id="@+id/etPasswordRegister"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginBottom="30dp"
            android:hint="@string/password"
            android:imeActionId="6"
            android:imeActionLabel="Sign In"
            android:imeOptions="actionUnspecified"
```

```

        android:inputType="textPassword"
        android:maxLines="1"
        android:singleLine="true" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <Button
            android:id="@+id/btRegister"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:background="@drawable/roundedbutton"
            android:text="@string/register"
            android:textColor="@color/white" />

        </LinearLayout>
    </LinearLayout>
</RelativeLayout>
</ScrollView>
```

Source code diatas untuk membuat tampilan pada RegisterActivity. Yang terdiri dari fitur *image*, *textfield*, *button*. Pengaturan gambar dikelola di *Tag ImageView* ini berfungsi untuk menampilkan gambar, dimana lebar dambar diatur di *android:layout_width="100dp"*, sedangkan untuk ukuran tinggi gambar diatur di *android:layout_height="100dp"*, posisi dari gambar yang bergantung pada sumbu X dan Y diatur di *android:layout_gravity="center_horizontal"*. Dan gambar diakses dari *android:src="@drawable/logolab"*. *Tag* selanjutnya *LinearLayout* yang berfungsi untuk pengaturan layout dimana pengaturannya komponennya dapat rapi secara vertical dan horizontal. *Tag EditText* yang pertama ini digunakan untuk memasukan username kemudian yang kedua untuk mengisi *password*. Kemudian *tag button* yang pertama bertulisan resigister dengan baris koding *android:text="@string/register"*.

```

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.widget.Button;
import android.widget.EditText;

public class RegisterActivity extends AppCompatActivity {
```

```

private DatabaseHandler databaseHandler;
private User usermodel;
private EditText etUsernameRegister;
private EditText etPasswordRegister;
private Button btRegister;
private String username,password;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_register);

    initView();
    register();
}

private void register() {
    btRegister.setOnClickListener(V ->
        initDataHandler()
    );
}

private void initView() {
    etUsernameRegister =
        findViewById(R.id.etUsernameRegister);
    etPasswordRegister =
        findViewById(R.id.etPasswordRegister);
    btRegister = findViewById(R.id.btRegister);
}

private void initUser() {
    username = etUsernameRegister.getText().toString();
    password = etPasswordRegister.getText().toString();

    usermodel = new User();
    usermodel.setUsername(username);
    usermodel.setPassword(password);
}

private void initDataHandler() {
    initUser();

    databaseHandler = new DatabaseHandler(this);
    databaseHandler.addUser(usermodel);
    User model = databaseHandler.getMahasiswa(1);
    Log.e("record", model.getUsername().toString());
    Intent admin = new Intent(RegisterActivity.this,
        LoginActivity.class);
    startActivity(admin);
    finish();
}

```

{}

Source code diatas dimasukan pada *file* RegisterActivity.java. dimana pada bagian awal kodingan adalah *import library* untuk pendukung fitur-fitur yang akan dipakai, yakni

```
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.widget.Button;
import android.widget.EditText;
```

didalam *class* RegisterActivity berikan variable-variabel yaitu databaseHandler yang dibuat dari *class* DatabaseHandler, usermodel yang dibuat dari *class* User, etUsernameRegister yang dibuat dari *class* Tag EditText, etPasswordRegister yang dibuat dari *class* Tag EditText, btRegister sebagai nama variable dari Button, username dan password yang bertipe data string dengan modifier private. Pada method Oncreate yang akan menginisialisasi program ini. method ini berisi method initView yang berfungsi untuk inisialisasi *view* dengann barisan kodingan

```
etUsernameRegister = findViewById(R.id.etUsernameRegister);
etPasswordRegister = findViewById(R.id.etPasswordRegister);
btRegister = findViewById(R.id.btRegister);
```

, kemudian register berisi proses ketika *button register* ditekan, setelah ditekan program akan mengalihkan proses ke initDataHandler(). Baris kodingan pada method ini akan mengambil data mahasiswa yang tersimpan pada database. Method terakhir adalah initUser yang berguna untuk inisialisasi user. Dimana hasil ketikan pada text field akan disimpan pada variable username dan password dan disimpan pada database.

3.5.3 Membuat MainActivity



Gambar 3.35 Tampilan MainActivity

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:context=".MainActivity">

    <android.support.v7.widget.CardView
        xmlns:android="http://schemas.android.com/apk/res/android"
            android:id="@+id/cardView"
            style="@style/CardView.Light"
            android:layout_width="320dp"
            android:layout_height="400dp"
            android:layout_marginBottom="8dp"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintVertical_bias="0.100000024">

        <android.support.constraint.ConstraintLayout
            xmlns:app="http://schemas.android.com/apk/res-auto"
                android:layout_width="match_parent"
                android:layout_height="wrap_content">
```

```
<ImageView
    android:id="@+id/media_image"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:scaleType="fitCenter"
    android:src="@drawable/wall1"

    app:layout_constraintBottom_toTopOf="@+id/primary_text"
        app:layout_constraintDimensionRatio="H,16:9"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_chainStyle="packed"
/>

<TextView
    android:id="@+id/primary_text"
    android:layout_width="0dp"
    android:layout_height="61dp"
    android:layout_marginStart="16dp"
    android:layout_marginTop="24dp"
    android:layout_marginEnd="16dp"
    android:text="@string/Header"

    android:textAppearance="@style/TextAppearance.AppCompat.Headline"
        android:textColor="#DE000000"

    app:layout_constraintBottom_toTopOf="@+id/tvNameMain"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"

    app:layout_constraintTop_toBottomOf="@+id/media_image" />

<TextView
    android:id="@+id/tvNameMain"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="0dp"
    android:layout_marginEnd="16dp"
    android:text="@string/name"

    android:textAppearance="@style/TextAppearance.AppCompat.Body1"
        android:textColor="#8A000000"
        android:textSize="15sp"
        app:layout_constraintBottom_toTopOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"

    app:layout_constraintTop_toBottomOf="@+id/primary_text" />
```

```
<TextView  
    android:id="@+id/tvdescription"  
    android:layout_width="0dp"  
    android:layout_height="wrap_content"  
    android:layout_marginTop="32dp"  
    android:layout_marginBottom="8dp"  
    android:text="@string/description"  
  
    android:textAppearance="@style/TextAppearance.AppCompat.Body1"  
    android:textColor="#8A000000"  
    android:textSize="20sp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="@+id/tvNameMain"  
    app:layout_constraintHorizontal_bias="0.0"  
  
    app:layout_constraintStart_toStartOf="@+id/tvNameMain"  
    app:layout_constraintTop_toTopOf="@+id/tvNameMain"  
    app:layout_constraintVertical_bias="1.0" />  
  
</android.support.constraint.ConstraintLayout>  
  
</android.support.v7.widget.CardView>  
  
<Button  
    android:id="@+id/btnLogoutMain"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginStart="8dp"  
    android:layout_marginTop="8dp"  
    android:layout_marginEnd="6dp"  
    android:layout_marginBottom="8dp"  
    android:background="@drawable/roundedbutton"  
    android:minWidth="0dp"  
    android:paddingStart="8dp"  
    android:paddingEnd="8dp"  
    android:text="@string/logout"  
    android:textColor="#de101010"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.0"  
    app:layout_constraintStart_toStartOf="@+id/cardView"  
    app:layout_constraintTop_toBottomOf="@+id/cardView"  
    app:layout_constraintVertical_bias="0.753" />  
  
</android.support.constraint.ConstraintLayout>
```

Pada tampilan MainActivity memiliki beberapa fitur seperti image, button dan paragraph. Paragraph diatur pada tag TextView dengan kodingan seperti diatas.

Dengan id primary text, dan tinggi layout 61dp, lebar 0 dp dll. Kemudian fitur button MainActivity.Java. Berikut adalah listing code mainActivity.java,

```

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    private DatabaseHandler databaseHandler;
    private User usermodel;
    private TextView tvNameMain;
    private Button btnLogoutMain;
    private Button exit;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        initView();
        initPreference();
        logout();
    }

    @Override
    public void onBackPressed() {
        showAlertDialog();
    }

    private void initView() {
        tvNameMain = findViewById(R.id.tvNameMain);
        btnLogoutMain = findViewById(R.id.btnLogoutMain);
        exit = findViewById(R.id.btnExit);
    }

    private void initPreference() {
        SharedPreferences preferences =
        getSharedPreferences("LoginPreference", MODE_PRIVATE);
        String username = preferences.getString("username", "");

        tvNameMain.setText(username);
    }
}

```

```
private void deletePreference(){
    SharedPreferences preferences =
getSharedPreferences("LoginPreference", MODE_PRIVATE);
    preferences.edit().remove("username").commit();
    preferences.edit().remove("password").commit();
}

private void logout() {
    exit.setOnClickListener(view -> showAlertDialog());
}

public void showAlertDialog() {
    new AlertDialog.Builder(this)
        .setMessage("Apa kalian ingin Logout?")
        .setCancelable(false)
        .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog,
int id) {
                deletePreference();
                Intent login = new
Intent(MainActivity.this, LoginActivity.class);
                startActivity(login);
                finish();
            }
        })
        .setNegativeButton("No", null)
        .show();
}

}
```

Pada *source code* diatas untuk *back end program* dari halaman MainActivity, yang berisi awalannya adalah mengimport *library* untuk mensupport fitur-fitur yang dipakai di MainActivity. *Class* ini berisikan variable databaseHandler yang dibuat dari *class* DatabaseHandler, *usermodel* dari *class* User, tvNameMain dari *tag* TextView, btnLogout *variable button*, dan *button exit*. Ketika menekan tombol *exit* maka *method logout* akan berfungi untuk mengeksekusi ke *method* showAlertDialog yang berisi Secara garis besar ketika menekan *button exit (logout)* maka akan muncul dialog alert yang berisikan “apakah kalian ingin logout?” jika iya maka *preference* akan dihapus sehingga perlu login lagi untuk masuk ke program. Jika tidak maka tetap berada pada halaman ini.

3.5.4 Membuat Splashscreen



Gambar 3.36 Tampilan Splashscreen

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <ImageView
        android:src="@drawable/splashscreen"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:scaleType="fitXY"/>

    <ProgressBar
        android:id="@+id/progressBar1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="58dp" />

</RelativeLayout>
```

Pada *source code* ini untuk membuat tampilan seperti gambar di atasnya. Dengan 2 *tag* yang dipakai yaitu *ImageView* untuk menampilkan gambar dan *progressbar* untuk menampilkan *progress* ketika membuka program. Kemudian pada *tag ImageView* berisi *android:src="@drawable/Splashscreen"* mengambil

gambar yang berada pada file *drawable* dengan nama file png/jpg nya *SplashScreen*.

```
    android:layout_width="fill_parent"      sesuai      dengan      parent      nya,  

    android:layout_height="fill_parent"      sesuai      dengan      parent      nya,  

    android:scaleType="fitXY"/>.   Dan    progressbar    berisi    baris    kodingan  

    android:id="@+id/progressBar1"  

    android:layout_width="wrap_content"  

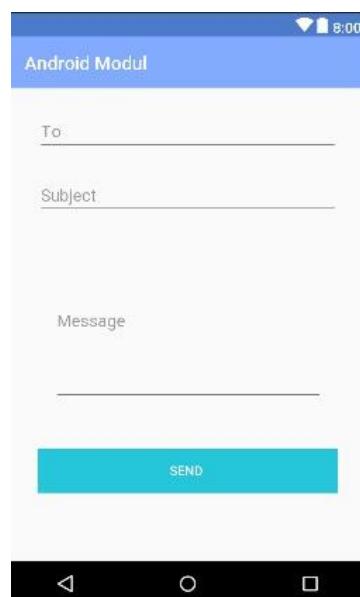
    android:layout_height="wrap_content"  

    android:layout_alignParentBottom="true"  

    android:layout_centerHorizontal="true"  

    android:layout_marginBottom="58dp" />.
```

3.5.5 Membuat EmailActivity



Gambar 3.37 Tampilan EmailActivity

```
import android.content.Intent;  

import android.support.v7.app.AppCompatActivity;  

import android.os.Bundle;  

import android.view.View;  

import android.widget.Button;  

import android.widget.EditText;  
  

public class EmailActivity extends AppCompatActivity {
```

```

Button buttonSend;
EditText textTo;
EditText textSbj;
EditText textMsg;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_email);

    buttonSend = (Button) findViewById(R.id.buttonSend);
    textTo = (EditText) findViewById (R.id.editTextTo);
    textSbj = (EditText) findViewById (R.id.editTextSbj);
    textMsg = (EditText) findViewById (R.id.editTextMsg);

    buttonSend.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {

            String to = textTo.getText().toString();
            String subject = textSbj.getText().toString();
            String message = textMsg.getText().toString();

            Intent email = new Intent(Intent.ACTION_SEND);
            email.putExtra(Intent.EXTRA_EMAIL, new String[] {
                to});
            email.putExtra(Intent.EXTRA_SUBJECT, subject);
            email.putExtra(Intent.EXTRA_TEXT, message);

            email.setType("message/rfc822");

            startActivity(Intent.createChooser(email, "Choose
an Email client :"));
        }
    });
}

@Override
public void onBackPressed() {
    back();
}

private void back() {
    Intent reg = new Intent(this, MainActivity.class);
    startActivity(reg);
    finish();
}
}

```

Source code diatas merupakan tampilan pada EmailActivity. Fitur yang ada pada tampilan EmailActivity ada *Textfield* dan *button*. *Textfield* yang pertama

berikan alamat email yang dituju basis kodingannya

```
android:id="@+id/editTextTo" baris ini sebagai id dari textfield ini dengan nama  
id editTextTo. android:layout_width="0dp", lebar dari text sebesar 0  
dp, android:layout_height="44dp", Panjang dari textfield sebesar  
44dp android:hint="To" ini isi pada saat textfield diawal inisialisasi,  
android:inputType="textEmailAddress" tipe inputan harus ada @ nya karena  
email tipenya. Kemudian pada textfield yang kedua digunakan sebagai subjek saat  
mengirimkan email dengan baris kodingannya android:id="@+id/editTextSbj"  
baris ini menunjukan id dari textfield ini, android:layout_width="0dp"  
lebar dari textfield-nya adalah 0 dp, android:layout_height="44dp"  
panjang dari textfield-nya adalah 44 dp, android:inputType="textEmailSubject"  
tipe inputannya email sehingga sesuai dengan sintak dari email  
android:hint="Subject " ini isi pada saat textfield diawal inisialisasi. Textfield  
yang terakhir untuk menampung pesan yang ingin disampaikan  
android:id="@+id/editTextMsg" nama id dari textfield ini editTextMsg,  
android:layout_width="291dp" lebarnya dari textfielnya adalah 219dp,  
android:layout_height="176dp" panjangnya dari textfield-nya adalah 219dp,  
android:inputType="textMultiLine" tipe inputannya multi line sehingga dapat  
menampung kata-kata/pesan-pesan yang ingin disampaikan,  
android:hint="Message " ini isi pada saat textfield diawal inisialisasi. Terakhir  
tag button yang nantinya akan berguna untuk memproses hasil dari inputan pada  
textfield android:id="@+id/buttonSend" nama id dari button ini buttonSend  
dengan lebar 0 dp, dan tingginya sesuai dengan kontennya  
android:layout_width="0dp" android:layout_height=  
"wrap_content". android:background="#26c6da" baris ini mengatur warna  
dari background button dimana hexa #26c6da bermakna warna dengan persentase  
4.9% red, 77.6% green and 85.5% blue. android:text="Send " ini tulisan yang
```

ada pada *button* android:textColor="@android:color/background_light"
barisan ini mengatur warna dari text nya.

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".EmailActivity">

    <EditText
        android:id="@+id/editTextSbj"
        android:layout_width="0dp"
        android:layout_height="44dp"
        android:layout_marginStart="30dp"
        android:ems="10"
        android:inputType="textEmailSubject"
        android:hint="Subject "
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        tools:layout_constraintLeft_creator="1"
        tools:layout_constraintTop_creator="1"
        tools:layout_constraintRight_creator="1"
        android:layout_marginEnd="30dp"
        android:layout_marginTop="25dp"
        app:layout_constraintTop_toBottomOf="@+id/editTextTo" />

    <EditText
        android:id="@+id/editTextTo"
        android:layout_width="0dp"
        android:layout_height="44dp"
        android:hint="To "
        android:fontFamily="@string/app_name"
        android:inputType="textEmailAddress"
        tools:layout_constraintTop_creator="1"
        android:layout_marginTop="25dp"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        tools:layout_constraintRight_creator="1"
        android:layout_marginStart="30dp"
        android:layout_marginEnd="30dp"
        tools:layout_constraintLeft_creator="1"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/editTextMsg"
        android:layout_width="291dp"
```

```

        android:layout_height="176dp"
        android:layout_marginStart="30dp"
        android:ems="10"
        android:inputType="textMultiLine"
        android:hint="Message "
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        tools:layout_constraintLeft_creator="1"
        tools:layout_constraintTop_creator="1"
        tools:layout_constraintRight_creator="1"
        android:layout_marginEnd="30dp"
        android:layout_marginTop="25dp"
        app:layout_constraintTop_toBottomOf="@+id/editTextSbj"
        app:layout_constraintHorizontal_bias="0.516"
        android:layout_marginLeft="16dp"
        android:layout_marginRight="16dp"
        tools:layout_editor_absoluteX="62dp" />

    <Button
        android:id="@+id/buttonSend"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="30dp"
        android:layout_marginLeft="30dp"
        android:layout_marginRight="30dp"
        android:layout_marginStart="30dp"
        android:background="#26c6da"
        android:text="Send "
        android:textColor="@android:color/background_light"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        tools:layout_constraintBottom_creator="1"
        tools:layout_constraintLeft_creator="1"
        tools:layout_constraintRight_creator="1"
        tools:layout_constraintTop_creator="1"
        android:layout_marginTop="25dp"
        app:layout_constraintTop_toBottomOf="@+id/editTextMsg"
        app:layout_constraintVertical_bias="0.251" />

</android.support.constraint.ConstraintLayout>

```

Kemudian *source code* diatas sebagai *back end* nya, sehingga fitur-fiturnya bisa berjalan sesuai dengan kemauan *developer*. Bagian pertama mengimport *library-library* yang dibutuhkan

```

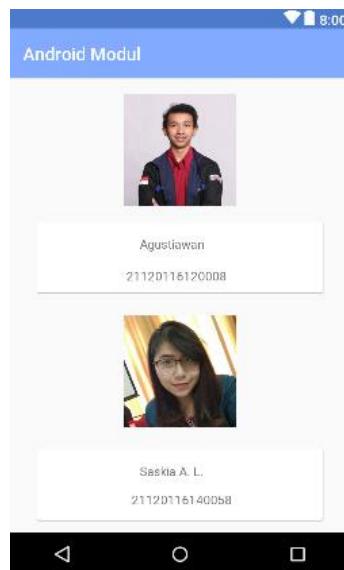
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

```

variable yang dipakai buttonSend variable button,

textTo dari EditText, textSbj dari EditText bagian Subjek, dan textMsg dari EditText bagian Message. Kemudian ketika buttonSend Diclick maka proses eksekusi akan dijalankan dengan baris koding , buttonSend = (Button) findViewById(R.id.buttonSend); textTo = (EditText) findViewById (R.id.editTextTo); textSbj = (EditText) findViewById (R.id.editTextSbj); textMsg = (EditText) findViewById (R.id.editTextMsg); baris ini yang akan mencocokan dengan fitur-fitur yang dibuat pada xaml tadi sesuai dengan idnya. Kemudian String to = textTo.getText().toString(); inputan yang diisi pada textbox dengan id textTo ini akan disimpan pada variable to dengan tipe data String String subject = textSbj.getText().toString(); inputan yang diisi pada textbox dengan id textSbj ini akan disimpan pada variable subject dengan tipe data String. String message = textMsg.getText().toString(); inputan yang diisi pada *textfield* dengan id textMsg ini akan disimpan pada variable message dengan tipe data String. Intent email = new Intent(Intent.ACTION_SEND); objek *email* ini yang akan menyimpan masukan pada *textfieldnya* dan akan dikirimkan sesuai dengan *email tujuan*, email.putExtra(Intent.EXTRA_EMAIL, new String[]{ to}); email.putExtra(Intent.EXTRA_SUBJECT, subject); email.putExtra(Intent.EXTRA_TEXT, message);. Method onBackPressed ketika diklik maka akan memproses *method back* yang berisi Intent reg = new Intent(this, MainActivity.class); akan memindahkan ke MainActivity startActivity(reg); menjalankan objek reg yang menyimpan eksekusi untuk pindah halaman ke MainActivity. finish(); kemudian selesai.

3.5.6 Membuat AboutActivity



Gambar 3.38 Tampilan activity_about

Berikut adalah source code `AboutActivity.java` yang digunakan untuk menampilkan tampilan pada `activity_about.xml`. Terdapat juga method `onBackPressed` yang digunakan agar ketika tombol back ditekan atau sentuh maka kembali ke halaman sebelumnya, yaitu `MainActivity`.

```
package com.example.mypc.aplikasimodul2kel37;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class AboutActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_about);
    }

    @Override
    public void onBackPressed() {
        back();
    }

    private void back() {
```

```

        Intent reg = new Intent(this, MainActivity.class);
        startActivity(reg);
        finish();
    }
}

```

Berikut adalah listing code about_activity.xml

```

<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"

    tools:context="com.example.mypc.aplikasimodul2kel37.AboutActivity">

    <ImageView
        android:id="@+id/ivpraktikan1"
        android:layout_width="126dp"
        android:layout_height="145dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@drawable/agus" />

    <android.support.v7.widget.CardView
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:id="@+id/cardView"
        style="@style/CardView.Light"
        android:layout_width="320dp"
        android:layout_height="80dp"
        android:layout_marginTop="8dp"
        android:layout_marginBottom="8dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/ivpraktikan1"
        app:layout_constraintVertical_bias="0.0" />

    <android.support.constraint.ConstraintLayout
        xmlns:app="http://schemas.android.com/apk/res-auto"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <TextView

```

```

        android:id="@+id/twnama1"
        android:layout_width="97dp"
        android:layout_height="24dp"
        android:layout_marginLeft="24dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="108dp"
        android:text="@string/nameabout1"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintTop_toTopOf="parent"      />

    <TextView
        android:id="@+id/twnim1"
        android:layout_width="127dp"
        android:layout_height="26dp"
        android:layout_marginTop="12dp"
        android:text="@string/nimabout1"
        app:layout_constraintEnd_toEndOf="@+id/twnama1"
        app:layout_constraintStart_toStartOf="@+id/twnama1"
        app:layout_constraintTop_toBottomOf="@+id/twnama1"
    />

</android.support.constraint.ConstraintLayout>

</android.support.v7.widget.CardView>

<ImageView
    android:id="@+id/ivpraktikan2"
    android:layout_width="126dp"
    android:layout_height="145dp"
    android:layout_marginTop="16dp"
    app:layout_constraintEnd_toEndOf="@+id/ivpraktikan1"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="@+id/ivpraktikan1"
    app:layout_constraintTop_toBottomOf="@+id/cardView"
    app:srcCompat="@drawable/saskia"      />

<android.support.v7.widget.CardView
    android:id="@+id/cardView2"
    style="@style/CardView.Light"
    android:layout_width="320dp"
    android:layout_height="80dp"
    android:layout_marginTop="8dp"
    android:layout_marginBottom="8dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="@+id/cardView"
    app:layout_constraintStart_toStartOf="@+id/cardView"
    app:layout_constraintTop_toBottomOf="@+id/ivpraktikan2">

    <android.support.constraint.ConstraintLayout
        xmlns:app="http://schemas.android.com/apk/res-auto"

```

```

        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <TextView
            android:id="@+id/twnama2"
            android:layout_width="97dp"
            android:layout_height="24dp"
            android:layout_marginLeft="24dp"
            android:layout_marginTop="16dp"
            android:layout_marginEnd="108dp"
            android:text="@string/nameabout2"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintLeft_toLeftOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

        <TextView
            android:id="@+id/twnim2"
            android:layout_width="115dp"
            android:layout_height="32dp"
            android:layout_marginTop="8dp"
            android:text="@string/nimabout2"
            app:layout_constraintEnd_toEndOf="@+id/twnama2"
            app:layout_constraintStart_toStartOf="@+id/twnama2"
            app:layout_constraintTop_toBottomOf="@+id/twnama2" />
    />

    </android.support.constraint.ConstraintLayout>
</android.support.v7.widget.CardView>

</android.support.constraint.ConstraintLayout>

```

Sourcecode diatas digunakan untuk membuat tanmpilan activity_about. Pada halaman ini ditampilkan data praktikan 1 dan data praktikan 2. Diatas tiap-tiap data praktikan terdapat foto masing-masing praktikan. Pada sourcecode diatas menggunakan beberapa view seperti imageView dan textView. Pada halaman about ini sudah menggunakan Construct Layout. Setelah halaman ini dibuat kemudian ditambahkan fungsi di class MainActivity utnuk menambahkan fungsi ketika button about ditekan. Berikut adalah source code yang ditambahkan pada MainActivity.java

```

private void about() {
    btnAboutMain.setOnClickListener(v -> {
        Intent reg = new Intent(this, AboutActivity.class);
        startActivity(reg);
        finish();
    });
}

```

```
    );  
}
```

Kemudian dibuat juga tampilan button About pada halam utama. Berikut adalah source code yang ditambahkan pada main_activity.xml

```
<Button  
    android:id="@+id	btnAboutMain"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginStart="36dp"  
    android:layout_marginEnd="8dp"  
    android:layout_marginBottom="8dp"  
    android:background="@drawable/roundedbutton"  
    android:text="About"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="@+id/cardView"  
    app:layout_constraintHorizontal_bias="1.0"  
    app:layout_constraintStart_toEndOf="@+id	btnLogoutMain"  
    app:layout_constraintTop_toTopOf="@+id	btnLogoutMain"  
    app:layout_constraintVertical_bias="0.111" />
```

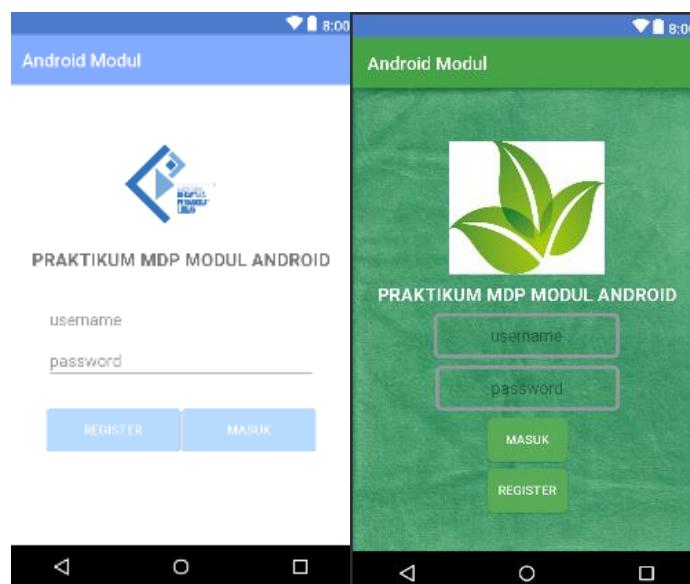
3.6 Tugas

3.6.1 Mengubah Logo

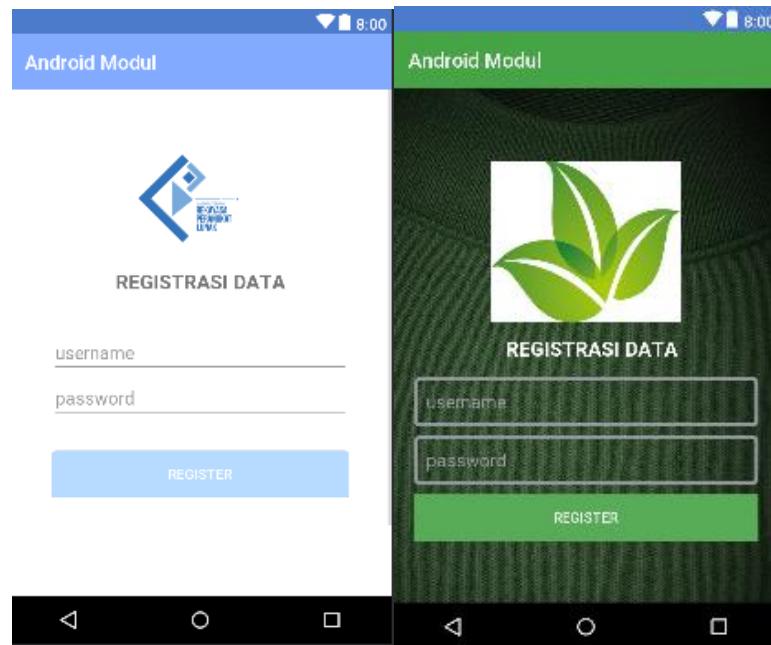
Langkah yang dilakukan untuk mengubah logo sangat sederhana, ada 2 metode dalam mengubah logo sebelumnya yaitu

1. menambahkan logo baru ke dalam package drawable dan mengubah penggunaan logo yang sebelumnya dengan logo baru pada file .xml yang menggunakan logo lama.
2. Menamakan logo baru dengan nama yang sama dengan logo lama, kemudian menghapus logo lama di drawable dilanjutkan dengan melakukan paste logo baru di package drawable.

Dalam hal ini praktikan memilih metode yang kedua karena lebih mudah, berikut adalah tampilan sebelum dan sesudah logo diganti.



Gambar 3.39 Tampilan sebelum dan sesudah logo diganti pada halaman activity_login

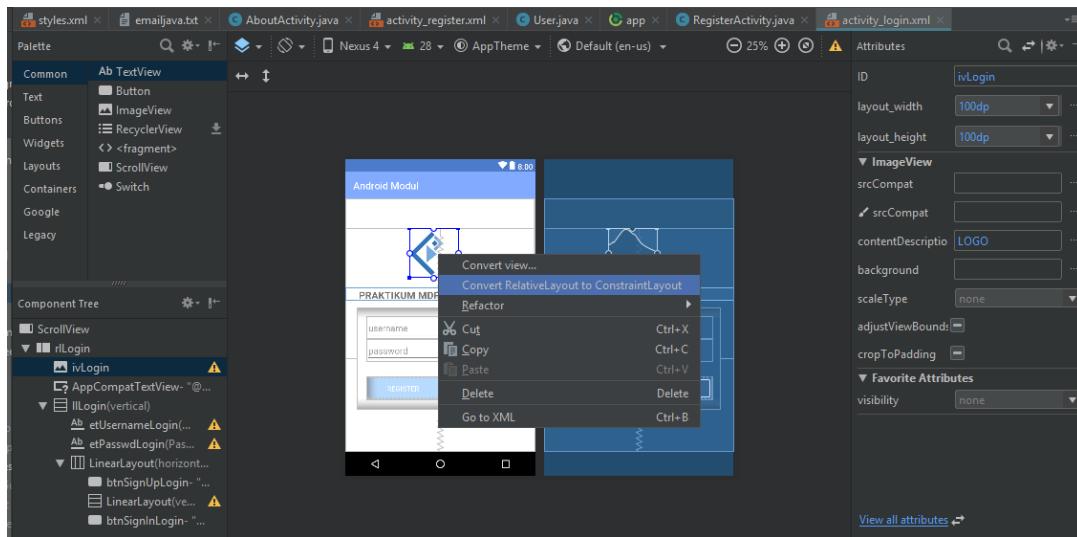


Gambar 3.40 Tampilan sebelum dan sesudah logo diganti pada halaman activity_register



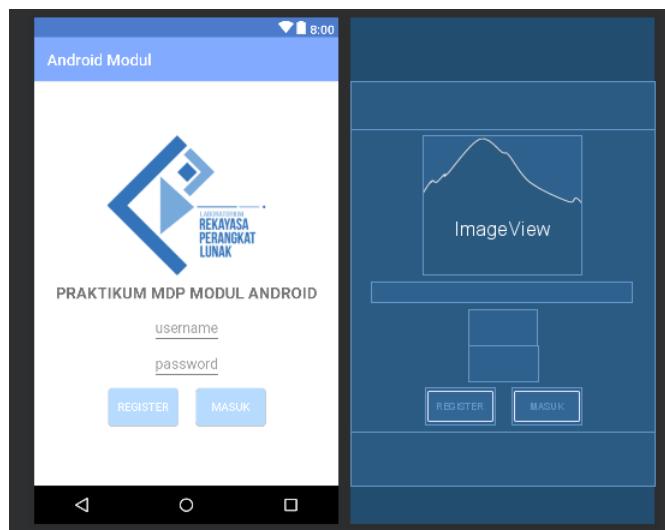
Gambar 3.41 Tampilan sebelum dan sesudah logo diganti pada halaman activity_main

3.6.2 Mengubah Struktur Layout Pada Login dan Registrasi Menjadi Constraint

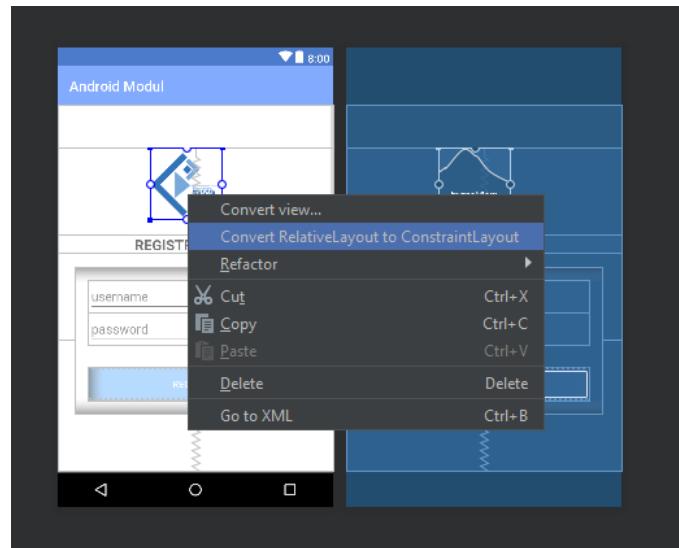


Gambar 3.42 Membuat construct layout (1)

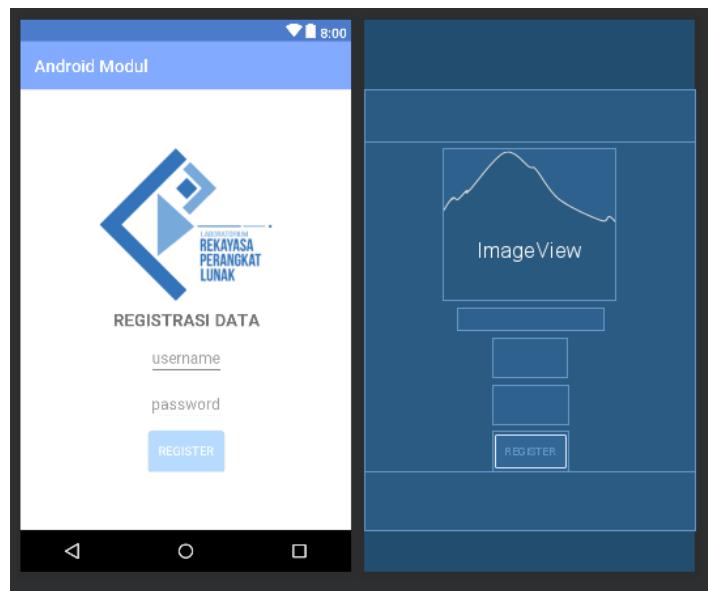
Cara untuk mengubah relative layout menjadi construct layout yaitu dengan cara klik kanan pada object seperti pada gambar diatas dan pilih ConvertRelativeLayout to ConstraintLayout. Setelah itu tampilan akan berubah dan disusun sedemikian rupa sehingga menjadi seperti pada gambar dibawah ini,



Gambar 3.43 Tampilan setelah di convert menjadi construct layout pada activity_login



Gambar 3.44 Membuat construct layout (2)



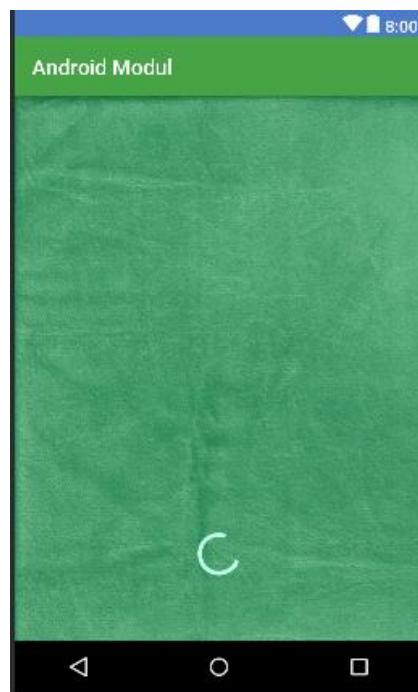
Gambar 3.45 Tampilan setelah di convert menjadi construct layout pada activity_register

3.6.3 Mengubah Tampilan Menjadi Material Design

Perubahan dilakukan pada background yang dipakai, pengaturan tombol atau button, dan juga pada pemilihan warna yang digunakan. Untuk mengubah background yang digunakan dengan cara mengubah pada bagian

android:background="@drawable/baju". "baju" disini merupakan nama file yang terdapat pada folder drawable yang digunakan untuk background activity_register. Hal ini juga dilakukan dalam mengubah background activity yang lainnya.

Untuk mengubah warna tema, button, dan lainnya yang dianggap perlu yaitu dengan mengubah pada folder res/values/colors.xml. Daftar warna yang dibutuhkan di list pada folder tersebut. Dalam hal ini praktikan mengubah warna sehingga menjadi sedemikian rupa. Berikut adalah tampilan-tampilan baru pada aplikasi ini,



Gambar 3.46 Tampilan activity_splashscreen.xml



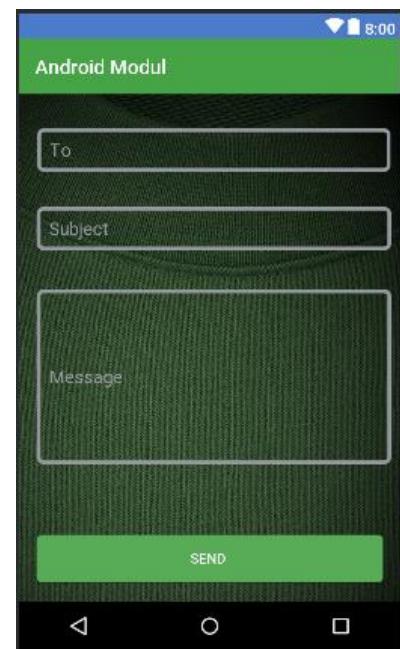
Gambar 3.47 Tampilan activity_login.xml



Gambar 3.48 Tampilan activity_register.xml



Gambar 3.49 Tampilan activity_main.xml



Gambar 3.50 Tampilan activity_email.xml



Gambar 3.51 Tampilan activity_about.xml

3.6.4 Membuat Menu Update dan Delete Untuk User

Membuat button edit pada avtivity_main.xml. Jika button edit ditekan maka nantinya akan muncul halaman edit data user. Berikut adalah listing code untuk menambah button edit pada activity_main.xml

```
<Button
    android:id="@+id/btnEditMain"
    android:layout_width="65dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="36dp"
    android:layout_marginEnd="8dp"
    android:layout_marginBottom="8dp"
    android:background="@drawable/roundedbutton"
    android:text="Edit"
    android:textColor="@color/white"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="@+id/cardView"
    app:layout_constraintHorizontal_bias="0.371"
    app:layout_constraintStart_toEndOf="@+id/btnLogoutMain"
    app:layout_constraintTop_toTopOf="@+id/btnLogoutMain"
    app:layout_constraintVertical_bias="0.111" />
```



Gambar 3.52 Tampilan activity_main.xml dengan button edit

Langkah selanjutnya yaitu membuat ActivityEdit.java dengan listing code seperti berikut,

```
package com.example.mypc.aplikasimodul2kel37;

import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;

public class EditActivity extends AppCompatActivity {

    private User usermodel;
    private EditText etUserAbout, etPassAbout;
    private Button btnUpdate, btnDelete;
    DatabaseHandler cek = LoginActivity.getPresenter();
    User usr = LoginActivity.getUser();
    String name = usr.getUsername();
    String pass = usr.getPassword();
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_edit);
    initView();
    update();
    delete();
    deletePreference();
    etUserAbout.setText(name);
    etPassAbout.setText(pass);
    initUser();
}

private void initView() {
    etUserAbout = findViewById(R.id.etUserAbout);
    etPassAbout = findViewById(R.id.etPassAbout);
    btnUpdate = findViewById(R.id.btnUpdate);
    btnDelete = findViewById(R.id.btnDelete);
}

private void initUser() {
    name = etUserAbout.getText().toString();
    pass = etPassAbout.getText().toString();

    usr.setUsername(name);
    usr.setPassword(pass);
}

private void update() { //untuk update data
    btnUpdate.setOnClickListener(v -> {
        initUser();
        cek.updateMahasiswa(usr);
        Intent reg = new Intent(this, MainActivity.class);
        startActivity(reg);
        finish();
    });
}

private void delete() { //untuk delete data
    btnDelete.setOnClickListener(view ->
showAlertDialog());
}

public void showAlertDialog() {
    new AlertDialog.Builder(this)
        .setMessage("Apa ingin menghapus Akun ini?")
        .setCancelable(false)
        .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
```

```

        public void onClick(DialogInterface dialog,
int id) {
            deletePreference();
            Intent login = new
Intent(EditActivity.this, LoginActivity.class);
            startActivity(login);
            cek.deleteModel(usr);
            finish();
        }
    })
.setNegativeButton("No", null)
.show();
}

private void deletePreference(){
    SharedPreferences preferences =
getSharedPreferences("LoginPreference", MODE_PRIVATE);
    preferences.edit().remove("username").commit();
    preferences.edit().remove("password").commit();
}
@Override
public void onBackPressed() {
    back();
}

private void back() {
    Intent reg = new Intent(this, MainActivity.class);
    startActivity(reg);
    finish();
}
}
}

```

Pada *sourcecode* tersebut, dibuat metode/fungsi untuk button pada halaman menu edit. Pada halaman edit ini, nantinya data user dapat diupdate maupun dihapus. Di mana pada update kita bisa mengubah username dan password sesuai dengan yang kita mau. Selanjunya kita juga bisa menghapus username dan password yang telah terdaftar, dan ketika usename dan password yang sudah di hapus tidak akan bisa kita gunakan untuk login kembali, kecuali kita regis lagi. Berikut adalah listing code activity_edit.xml

```

<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"

```

```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/seagreen"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/btnUpdate"
        android:layout_width="140dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="36dp"
        android:layout_marginTop="8dp"
        android:layout_marginBottom="44dp"
        android:background="@color/colorPrimaryDark"
        android:text="Update"
        android:textColor="@color/white"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/etPassAbout"
        app:layout_constraintVertical_bias="1.0" />

    <Button
        android:id="@+id/btnDelete"
        android:layout_width="140dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="40dp"
        android:layout_marginBottom="44dp"
        android:background="@color/colorPrimaryDark"
        android:text="DELETE"
        android:textColor="@color/white"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/etPassAbout"
        app:layout_constraintVertical_bias="1.0" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="36dp"
        android:layout_marginTop="72dp"
        android:text="Username"
        android:textSize="24sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="34dp"
        android:layout_marginStart="36dp"
```

```
        android:layout_marginTop="96dp"
        android:text="Password"
        android:textSize="24sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView3"    />

    <EditText
        android:id="@+id/etUserAbout"
        android:layout_width="280dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="124dp"
        android:layout_marginEnd="8dp"
        android:ems="10"
        android:inputType="textEmailAddress"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.318"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"                  />

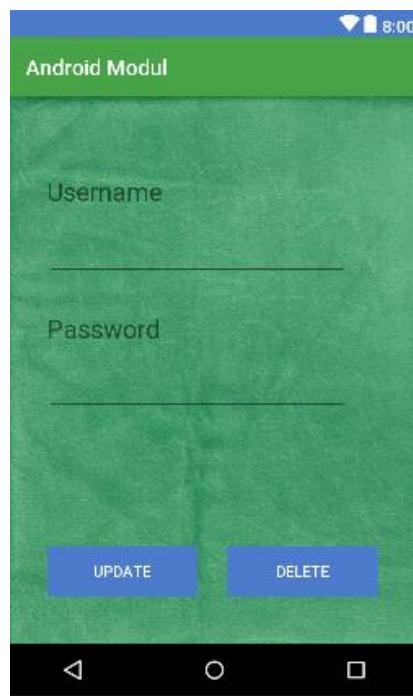
    <EditText
        android:id="@+id/etPassAbout"
        android:layout_width="280dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="36dp"
        android:layout_marginTop="80dp"
        android:ems="10"
        android:inputType="textPassword"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/etUserAbout"
    />

</android.support.constraint.ConstraintLayout>
```

Sourcecode activity_edit.xml digunakan untuk membuat tampilan pada halaman edit. Pada halaman ini terdapat textview utnuk username dan password, textedit yang digunakan untuk mengedit username maupun password, dan juga terdapat button update dan delete.



Gambar 3.53 Tampilan activity_main.xml



Gambar 3.54 Tampilan activity_edit.xml

3.6.5 Membatasi Password Minimal 6 Karakter

```
private void initDataHandler() {
    initUser();
    String pass = etPasswordRegister.getText().toString();
    if (TextUtils.isEmpty(pass) || pass.length() < 6) {
        etPasswordRegister.setError("You must have 6 characters
in your password");
        return;
    } else {
        databaseHandler = new DatabaseHandler(this);
        databaseHandler.addUser(usermodel);
        User model = databaseHandler.getMahasiswa(1);
        Log.e("record", model.getUsername());
        Intent admin = new Intent(RegisterActivity.this,
LoginActivity.class);
        startActivity(admin);
        finish();
    }
}
```

Ditambahkan sourcecode diatas pada ActivityRegister.java. Source code diatas digunakan untuk emngatur agar password yang diisikan oleh user saat register data minimal berjumlah 6 karakter. Jika password yang dimasukkan kurang dari 6 karakter maka akan muncul tulisan “You must have 6 characters in your password” dan akan gagal menambahkan data pada menu register.



Gambar 3.55 Tampilan Halaman Registrasi

(Link Tugas: <https://github.com/agustiawan0798/modul2kelompok37.git>)

3.7 Kesimpulan

1. SharedPreferences digunakan untuk menyimpan database username dan password user.
2. SharedPreferences mirip seperti database yang lainnya namun digunakan untuk menyimpan data dengan skala kecil dan sederhana.
3. Ketika membuat activity pada android, maka akan secara otomatis terbuat activity.xml pada layout.
4. Class Database Handler berfungsi untuk melakukan operasi CRUD (Create, Read, Update, dan Delete).
5. Untuk membuat suatu resources yang nantinya dapat dipakai secara berulang-ulang seperti roundedbutton, dapat dibuat suatu Drawable resources file pada res/drawable dengan cara klik kanan dan pilih new kemudian pilih Drawable resource file.
6. Pada res/values/colors.xml merupakan suatu lokasi yang digunakan untuk database warna yang akan digunakan dan warna tersebut dapat diberi nama agar lebih mudah untuk digunakan pada kelas-kela yang lain.
7. AndroidManifest.xml menyediakan informasi penting tentang aplikasi ke sistem Android yang harus dimiliki sistem agar bisa menjalankan setiap kode aplikasi.
8. Background pada suatu halaman dapat diubah dengan mudah dengan cara menambahkan atau mengubah pada `android:background="@drawable/nama_file_background"` di layout halaman yang ingin kita ubah backgroundnya.

BAB IV

Android Studio #2

4.1 Tujuan

1. Praktikan mampu mengetahui lingkungan pengembangan Sistem Operasi *Mobile* Android.
2. Praktikan mampu memahami model pengembangan MVP (*Model – View – Presenter*)
3. Praktikan memahami konsep penggunaan API (*Application Program Interface*)
4. Praktikan memahami konsep penggunaan retrofit.
5. Praktikan mampu membuat aplikasi Android sederhana menggunakan model pengembangan MVP.
6. Praktikan mampu melakukan deploy program ke device android masing-masing.

4.2 Alat dan Bahan

1. PC / Notebook (minimal RAM 4GB)

PC / Notebook yang digunakan yaitu yang memiliki RAM minimal 4 GB, hal ini dilakukan demi menunjang kelancaran saat praktikum.



Gambar 4.1. PC / Notebook

2. Android Studio 3.2, JDK, Android SDK

Android Studio adalah Integrated Development Environment (IDE) untuk sistem operasi Android, yang dibangun diatas perangkat lunak JetBrains IntelliJ IDEA dan didesain khusus untuk pengembangan Android. IDE ini merupakan pengganti dari Eclipse Android Development Tools (ADT) yang sebelumnya merupakan IDE utama untuk pengembangan aplikasi android.



Gambar 4.2. Android Studio

JDK (Java Development Kit) adalah Perangkat lunak yang digunakan untuk melakukan proses kompilasi dari kode java ke bytecode yang dapat dimengerti dan dapat dijalankan oleh JRE (Java Runtime Environment). JDK

wajib terinstall pada komputer yang akan melakukan proses pembuatan aplikasi berbasis java, namun tidak wajib terinstall di komputer yang akan menjalankan aplikasi yang dibangun dengan java.



Gambar 4.3. Logo Java

Android SDK (Software Development Kit) adalah satu set alat pengembangan yang digunakan untuk mengembangkan aplikasi untuk platform Android.



Android SDK

Gambar 4.4. Android SDK

3. Device Android dan kabel USB

Device yang digunakan di praktikum BAB IV adalah perangkat berjalan yang memiliki operasi sistem Android yang memiliki dukungan kabel data USB.



Gambar 4.5. Device Android dan kabel USB

4.3 Dasar Teori

4.3.1. API

API adalah singkatan dari *Application Program Interface*, dan memungkinkan *developer* untuk mengintegrasikan dua bagian dari aplikasi atau dengan aplikasi yang berbeda secara bersamaan. API terdiri dari berbagai elemen seperti function, protocols, dan tools lainnya yang memungkinkan *developer* untuk membuat aplikasi. Tujuan penggunaan API adalah untuk mempercepat proses *development* dengan menyediakan *function* secara terpisah sehingga developer tidak perlu membuat fitur yang serupa. Penerapan API akan sangat terasa jika fitur yang diinginkan sudah sangat kompleks, tentu membutuhkan waktu untuk membuat yang serupa dengannya. Misalnya: integrasi dengan *payment gateway*. Terdapat berbagai jenis sistem API yang dapat digunakan, termasuk sistem operasi, *library*, dan *web*.

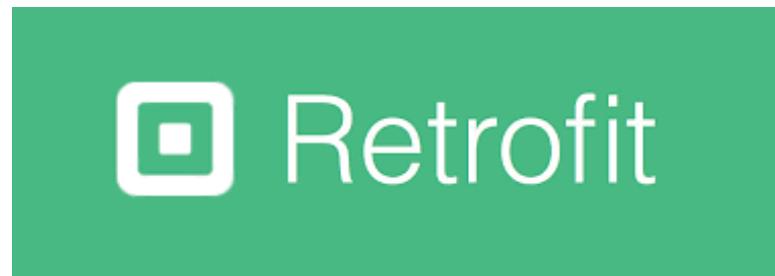


Gambar 4.6. API

(Sumber: codepolitan.com)

4.3.2. Retrofit

Retrofit adalah library Android yang dibuat oleh Squire yang digunakan sebagai REST Client pada Android, yang pasti akan memudahkan kita dalam programing. Karena kita tidak perlu lagi untuk membuat method-method sendiri untuk menggunakan REST Client API dari backend.



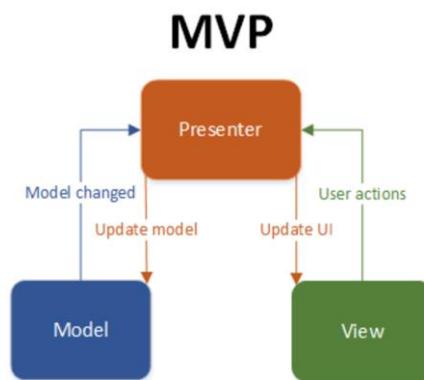
Gambar 4.7. Retrofit

(sumber:medium.com)

4.3.3. MVP

MVP (Model-View-Presenter) adalah pola arsitektur antarmuka pengguna dirancang untuk memfasilitasi unit testing otomatis dan meningkatkan pemisahan keprihatinan di logika presentasi:

- a) Model adalah antarmuka yang mendefinisikan data yang akan ditampilkan atau ditindaklanjuti dalam antarmuka pengguna.
- b) View adalah antarmuka pasif yang menampilkan data (model) dan mengarahkan perintah pengguna (kejadian) ke presenter untuk bertindak atas data tersebut.
- c) Presenter bertindak berdasarkan model dan pandangan. Ini mengambil data dari repositori (model), dan memformatnya untuk ditampilkan dalam tampilan. Biasanya, implementasi tampilan instantiates objek presenter beton, menyediakan referensi untuk dirinya sendiri.



Gambar 4.8. MVP

(sumber:medium.com)

4.4 Langkah Kerja

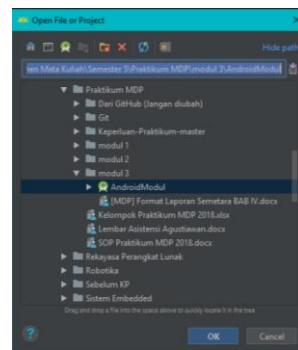
4.4.1 Persiapan Package

1. Buka Android Studio



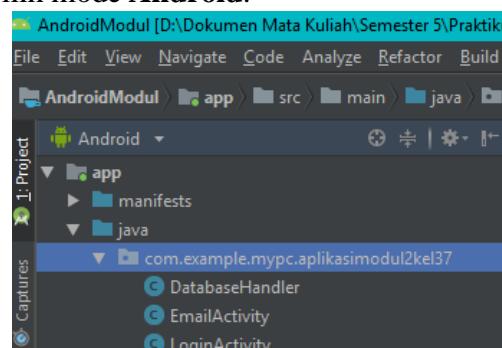
Gambar 4.9. Android Studio

2. Buka backup file project Modul 2, dengan cara klik **File – Open** – Arahkan ke File Project Android Modul 2



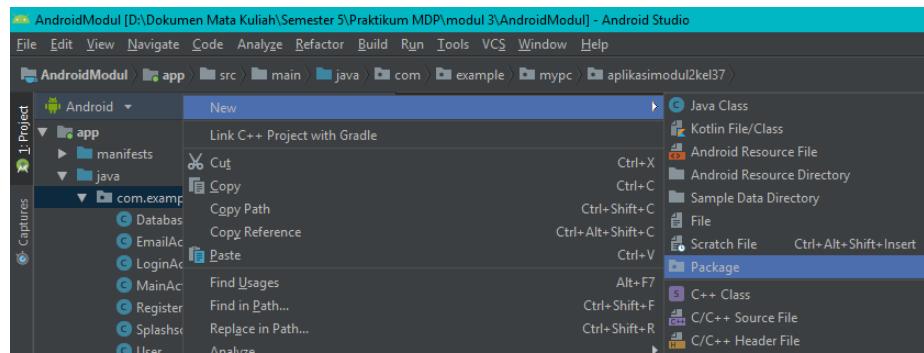
Gambar 4.10. File - Open

3. Tunggu hingga **build.gradle** dan **indexing** selesai seluruhnya
4. Pada tab **project**, pilih mode **Android**.



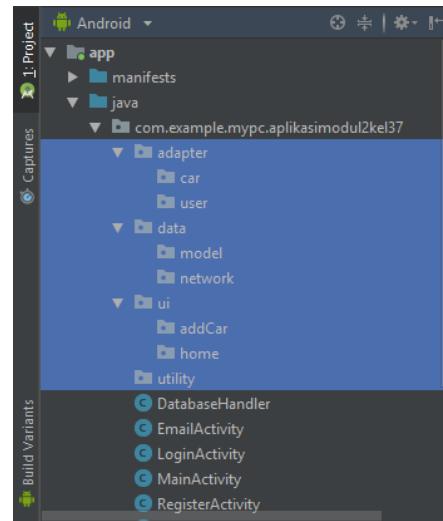
Gambar 4.11. Pilih Mode Android

5. Lalu buat package baru dengan cara klik kanan pada **main package java – new – package**



Gambar 4.12 New Package

6. Tambahkan beberapa package baru sehingga menjadi seperti berikut

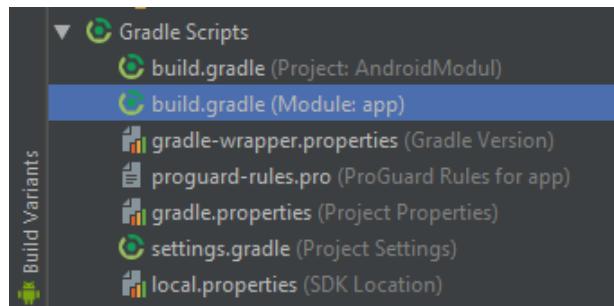


Gambar 4.13. Tambah Beberapa Package

Package – package ini akan digunakan untuk menampung class – class dalam metode MVP sesuai dengan kegunaanya, dengan ini dokumentasi projek akan lebih mudah dan terstruktur.

4.4.2 Persiapan Retrofit

1. Selanjutnya, tambahkan beberapa dependencies ke build.gradle. Dobel klik **build.gradle (Module:app)**.



Gambar 4.14. Double click build.grade

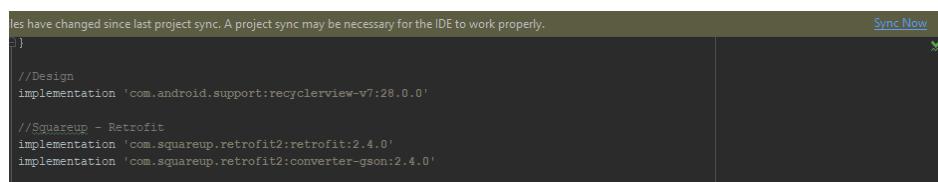
2. Lalu tambahkan, code berikut

```
//Design
implementation 'com.android.support:recyclerview-v7:28.0.0'

//Squareup - Retrofit
implementation 'com.squareup.retrofit2:retrofit:2.4.0'
implementation 'com.squareup.retrofit2:converter-gson:2.4.0'
```

Retrofit merupakan library yang akan digunakan untuk mengubah API yang diperoleh dari Refstull Server (HTTP) ke dalam bentuk data yang dapat diolah oleh Java

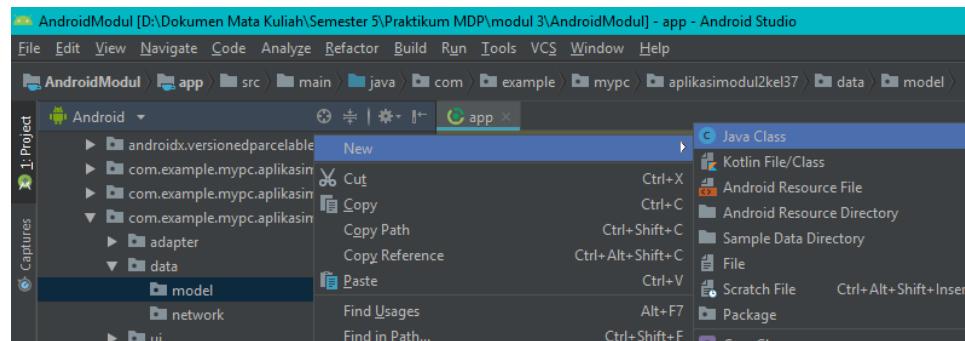
3. Setelahnya, klik tombol **Sync Now** dibagian kanan atas



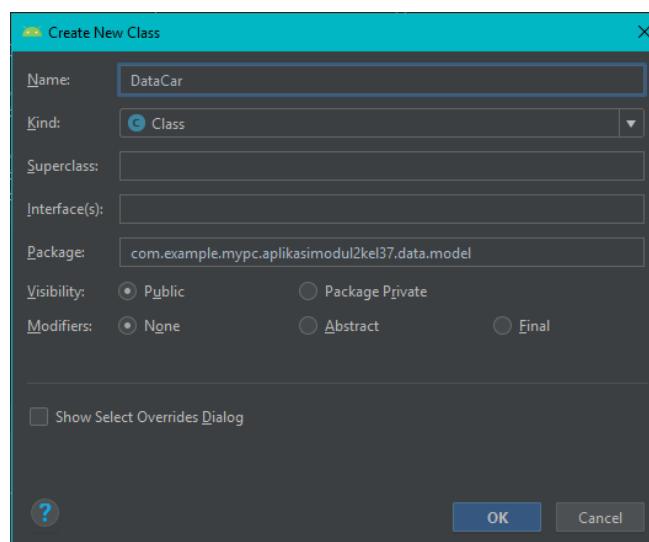
Gambar 4.15. Klik Sync Now

Tunggu hingga **download** dan **build.gradle** selesai

4. Setelah package dan dependencies, selanjutnya mempersiapkan Retrofit client. Buat class model pada package **data - model** dengan nama **DataCar**



Gambar 4.16. Membuat Java Class



Gambar 4.17. Membuat New Class DataCar

Pada model ini, akan menggunakan POJO (Plain Old Java Object) yaitu class yang berdiri sendiri dan tidak ada ketergantungan dengan class lain. Pada class DataCar tambahkan implements Parcelable untuk membungkus data sehingga dapat dikirimkan/dipindahkan antar activity. Kemudian tambahkan, code berikut,

```
import android.os.Parcel;
import android.os.Parcelable;

import com.google.gson.annotations.Expose;
import com.google.gson.annotations.SerializedName;

public class DataCar implements Parcelable {

    @SerializedName("id")
    @Expose
    private Integer id;
    @SerializedName("name")
    @Expose
```

```
private String name;
@SerializedName("merk")
@Expose
private String merk;
@SerializedName("model")
@Expose
private String model;
@SerializedName("year")
@Expose
private String year;

protected DataCar(Parcel in) {
    if (in.readByte() == 0) {
        id = null;
    } else {
        id = in.readInt();
    }
    id = in.readInt();
    name = in.readString();
    merk = in.readString();
    model = in.readString();
    year = in.readString();
}

public static final Creator<DataCar> CREATOR = new
Creator<DataCar>() {
    @Override
    public DataCar createFromParcel(Parcel in) {
        return new DataCar(in);
    }

    @Override
    public DataCar[] newArray(int size) {
        return new DataCar[size];
    }
};

public Integer getId() {
    return id;
}

public void setId(int id) {
    this.id = id;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getMerk() {
    return merk;
}
```

```
public void setMerk(String merk) {
    this.merk = merk;
}

public String getModel() {
    return model;
}

public void setModel(String model) {
    this.model = model;
}

public String getYear() {
    return year;
}

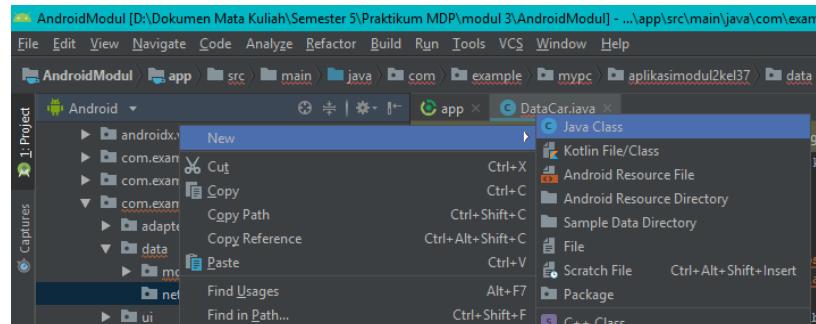
public void setYear(String year) {
    this.year = year;
}

@Override
public String toString() {
    return "DataUser{" +
        "id=" + id +
        ", name='" + name + '\'' +
        ", merk='" + merk + '\'' +
        ", model='" + model + '\'' +
        ", year='" + year + '\'' +
        '}';
}

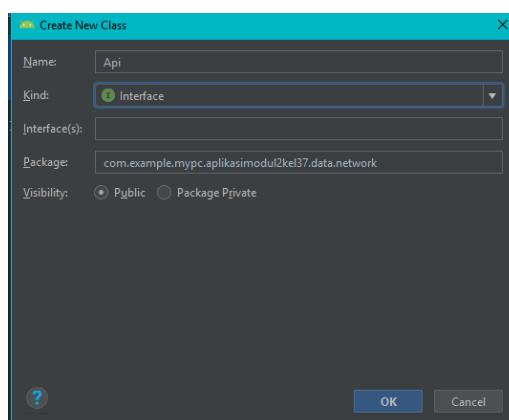
@Override
public int describeContents() {
    return 0;
}

@Override
public void writeToParcel(Parcel dest, int flags) {
    if (id == null) {
        dest.writeByte((byte) 0);
    } else {
        dest.writeByte((byte) 1);
        dest.writeInt(id);
    }
    dest.writeInt(id);
    dest.writeString(name);
    dest.writeString(merk);
    dest.writeString(model);
    dest.writeString(year);
}
}
```

5. Buat **interface** pada package **data - network** dengan nama **Api** sama seperti langkah sebelumnya, namun pada window box, pilih **interface** lalu beri nama **Api**



Gambar 4.18. Membuat Java Class



Gambar 4.19. Membuat New Class Interface Api

Interface ini digunakan untuk menginisialisasi key pada saat POST, GET, DELETE dan lain-lain ke Endpoint Restfull API. Pada praktikum ini akan menggunakan POST dan GET (all). Kemudian tambahkan, code berikut,

```
import com.google.gson.JsonObject;

import retrofit2.Call;
import retrofit2.http.Field;
import retrofit2.http.FormUrlEncoded;
import retrofit2.http.GET;
import retrofit2.http.Headers;
import retrofit2.http.POST;
import retrofit2.http.Query;

public interface Api {

    @Headers("Content-Type: application/json")

    @GET("getAllCar.php")
    Call<JsonObject> getAllCar();

    @GET("getCarById.php")
```

```

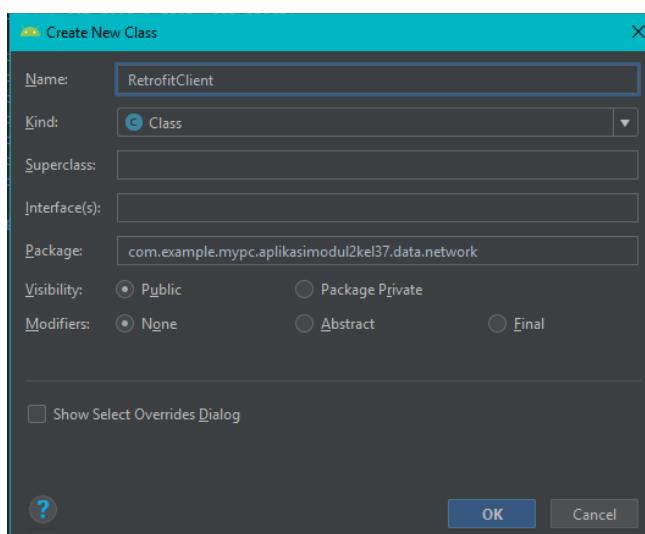
        Call<JsonObject> GetCarById(
            @Query("id") int id
        );

        @FormUrlEncoded
        @POST("addCar.php")
        Call<JsonObject> addCar(
            @Field("name") String name,
            @Field("merk") String merk,
            @Field("model") String model,
            @Field("year") String year
        );

    }
}

```

6. Buat class baru pada package **data – network** dengan nama **RetrofitClient**, yang berfungsi untuk inisialisasi library retrofit.



Gambar 4.20. Membuat New Class RetrofitClient

Kemudian tambahkan, code berikut,

```

import java.util.concurrent.TimeUnit;

import okhttp3.OkHttpClient;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;

public class RetrofitClient {
    private static RetrofitClient retrofitClient;

    private RetrofitClient() {}

    public static RetrofitClient getInstance() {
        if(retrofitClient == null){
            retrofitClient = new RetrofitClient();
        }
    }
}

```

```

        return retrofitClient;
    }

    public Api getApi() {
        return getRetrofit().create(Api.class);
    }

    public Retrofit getRetrofit() {

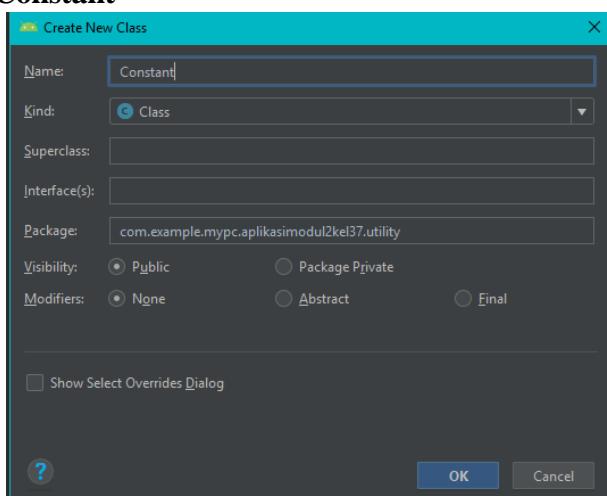
        final OkHttpClient okHttpClient = new
OkHttpClient.Builder()
            .readTimeout(15, TimeUnit.SECONDS)
            .connectTimeout(15, TimeUnit.SECONDS)
            .build();

        return new Retrofit.Builder()
            .baseUrl(Constant.BASE_URL)

.addConverterFactory(GsonConverterFactory.create())
            .client(okHttpClient)
            .build();
    }
}

```

7. Buat class pada package **utility** dengan nama **Constant**, yang akan berisikan variable - variable konstan atau tetap. **Tambahkan** (bukan replace) code berikut kedalam class **Constant**



Gambar 4.21. Membuat New Class Constant

Kemudian tambahkan, code berikut,

```

public static final String BASE_URL =
"https://vizyan.000webhostapp.com/";

public final class Extra {
    public static final String DATA = "EXTRA_DATA";
}

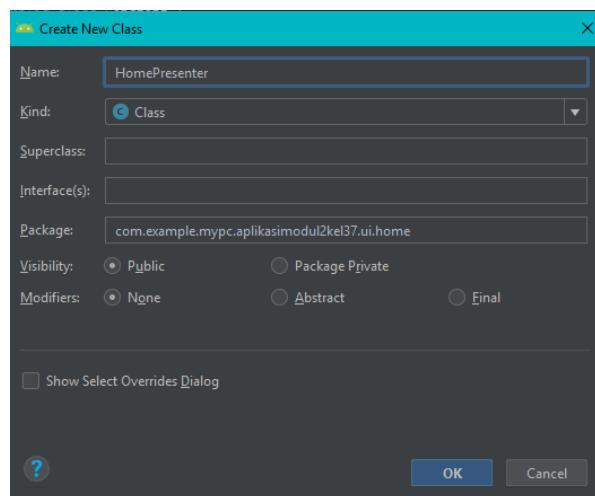
```

6.4.3. Pembuatan Activity

Selanjutnya, membuat activity – activity yang diperlukan pada aplikasi. Dengan konsep MVP, setiap activity nantinya terdiri dari 3 class. 1 class activity, 1 class presenter dan 1 class interface (view).

- Activity berfungsi untuk menghubungkan System Aplikasi dengan User
- Presenter berfungsi untuk menghubungkan Model dengan Retrofit Client
- Interface (View) berfungsi untuk menghubungkan Activity dengan Presenter

1. Membuat home
 - a. Pertama, buatlah presenter pada package **ui - home** dengan cara yang sama seperti membuat class biasa. Beri nama **HomePresenter**.



Gambar 4.22. Membuat Class HomePresenter

```

import com.google.gson.Gson;
import com.google.gson.JsonArray;
import com.google.gson.JsonObject;
import com.google.gson.reflect.TypeToken;

import java.lang.reflect.Type;
import java.util.List;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class HomePresenter {
  
```

```

private HomeView homeView;

public HomePresenter (HomeView homeView) {
    this.homeView = homeView;
}

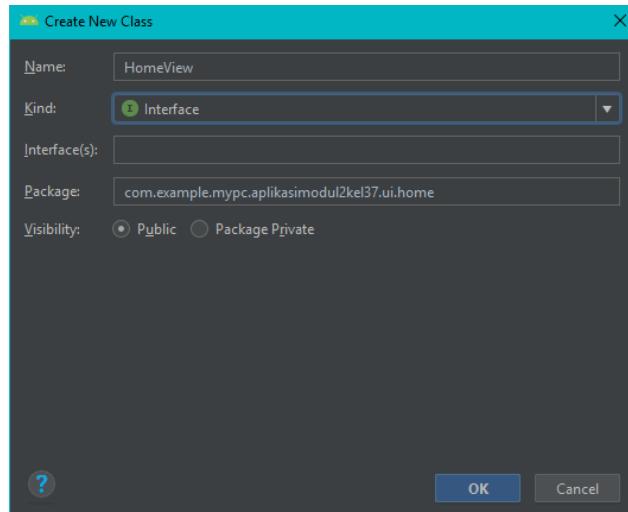
public void getAllCar(){
    final String tag = "Home-getAllCar";
    RetrofitClient.getInstance()
        .getApi()
        .getAllCar()
        .enqueue(new Callback<JsonObject>() {
            @Override
            public void onResponse(Call<JsonObject> call, Response<JsonObject> response) {
                if (response.isSuccessful()) {
                    JsonObject body = response.body();
                    JSONArray array = body.get("result").getAsJSONArray();
                    Type type = new TypeToken<List<DataCar>>().getType();
                    List<DataCar> dataCars = new Gson().fromJson(array, type);

                    homeView.successShowCar(dataCars);
                    Log.e(tag, response.body().toString());
                } else {
                    homeView.failedShowCar("Maaf terjadi kesalahan");
                    Log.e(tag, response.errorBody().toString());
                }
            }

            @Override
            public void onFailure(Call<JsonObject> call, Throwable t) {
                homeView.failedShowCar("Maaf terjadi kesalahan");
                Log.d(tag, t.getMessage().toString());
            }
        });
}
}

```

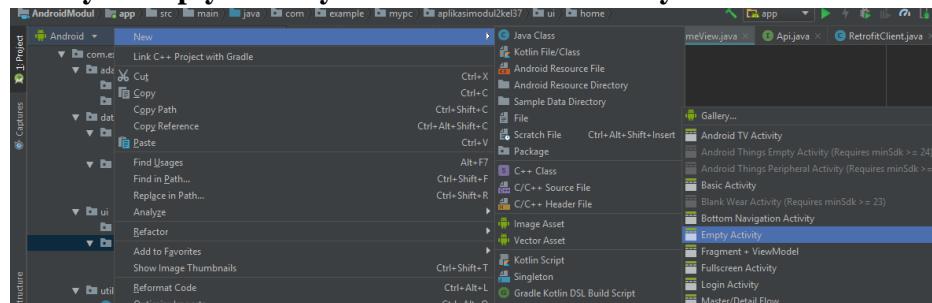
- b. Setelah itu buat **interface** pada package **ui - home** dengan nama **HomeView**. Lalu **tambahkan** (bukan replace) code berikut didalam classnya



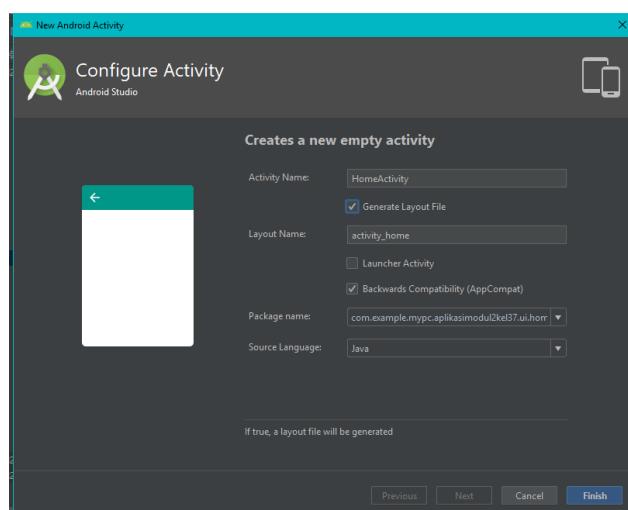
Gambar 4.23. Membuat Interface HomeView

```
void successShowCar(List<DataCar> dataCars);
void failedShowCar(String message);
```

- c. Selanjutnya, buat activity dengan cara klik kanan package **home** – **New** – **Activity** – **Empty Activity**. Beri nama **HomeActivity**. Lalu klik finish

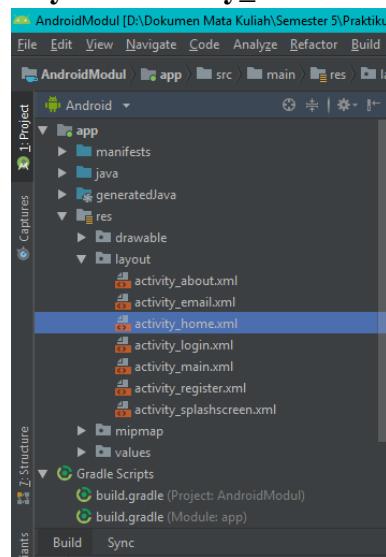


Gambar 4.24. Membuat activity baru



Gambar 4.25. Membuat HomeActivity

- d. Sebelum ke code activity, design terlebih dahulu tampilan pada layout activity nya. Klik package **res – layout – activity_home.xml**



Gambar 4.26. Memilih activity_home.xml

- e. Pilih mode text, lalu tambahkan code berikut kedalamnya

```

<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.widget.SwipeRefreshLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/srlHome"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ui.home.HomeActivity">

    <android.support.constraint.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <android.support.design.widget.FloatingActionButton
            android:id="@+id/fabHome"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginEnd="15dp"
            android:layout_marginBottom="15dp"
            android:clickable="true"

            app:backgroundTint="@color/design_default_color_primary"
            app:fabSize="normal"

            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:srcCompat="@android:drawable/ic_menu_add"
        />
    
```

```

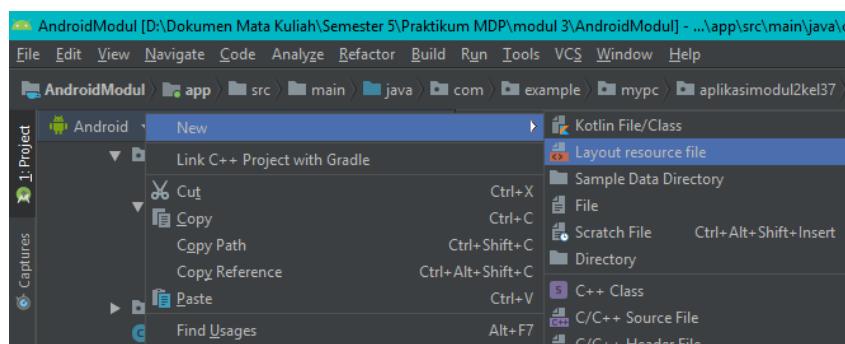
<android.support.v7.widget.RecyclerView
    android:id="@+id/rvHome"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginEnd="5dp"
    android:layout_marginStart="5dp"

    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

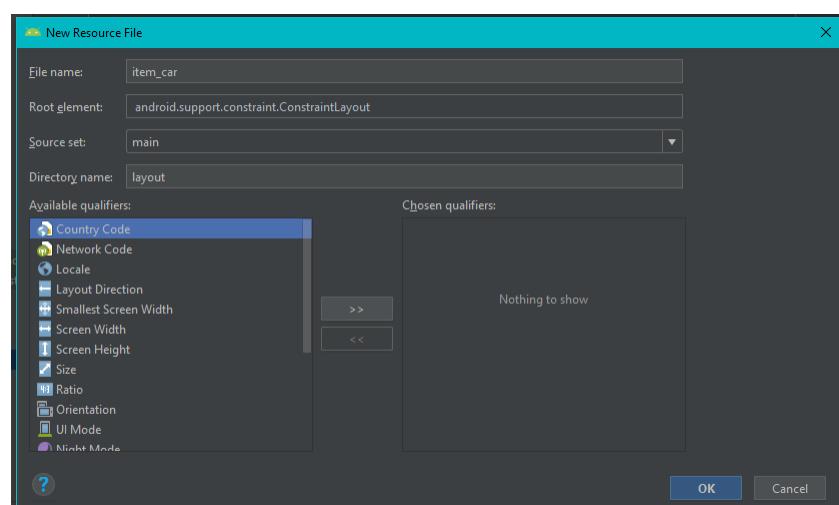
</android.support.constraint.ConstraintLayout>
</android.support.v4.widget.SwipeRefreshLayout>

```

- f. Buat layout item, dengan cara klik kanan pada **res – layout – Layout Resource File**. Beri nama **item_car**



Gambar 4.27. Membuat new Layout resorce file



Gambar 4.28. Membuat layout resource file item_car

```

<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView

```

```
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:minHeight="50dp"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_marginTop="5dp"
    android:layout_marginStart="10dp"
    android:layout_marginEnd="10dp"
    app:cardCornerRadius="15dp"
    android:clickable="true">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:minHeight="50dp"
        android:clickable="true">

        <LinearLayout
            android:id="@+id/l1ItemC"
            android:layout_width="match_parent"
            android:layout_height="60dp"
            android:clickable="true"
            android:orientation="horizontal">

            <ImageView
                android:id="@+id/ivItemC"
                android:layout_width="35dp"
                android:layout_height="35dp"
                android:layout_gravity="center"
                android:layout_marginStart="20dp"
                android:src="@mipmap/ic_launcher" />

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:orientation="vertical"
                android:gravity="center">

                <TextView
                    android:id="@+id/tvItemCName"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_gravity="center"
                    android:layout_marginStart="15dp"
                    android:text="Name"
                    android:textSize="18sp"
                    android:textStyle="bold"
                    tools:text="" />

                <TextView
                    android:id="@+id/tvItemCMerk"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_gravity="center"
```

```

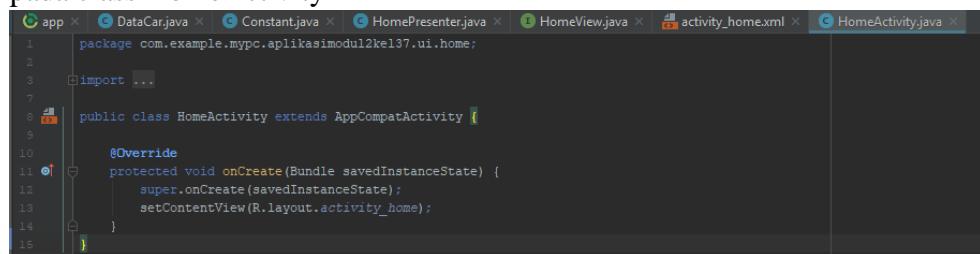
        android:layout_marginStart="15dp"
        android:text="Name"
        android:textSize="18sp"
        tools:text="" />
    </LinearLayout>

    </LinearLayout>
</RelativeLayout>

</android.support.v7.widget.CardView>

```

- g. Selanjutnya tambahkan kode pada HomeActivity. Dengan cara dobel click pada class HomeActivity



Gambar 4.29. Menambahkan kode pada HomeActivity

```

import android.content.Intent;
import android.support.design.widget.FloatingActionButton;
import android.support.v4.widget.SwipeRefreshLayout;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.widget.Toast;

import java.util.List;

public class HomeActivity extends AppCompatActivity
implements HomeView, CarListerner{

    private HomePresenter homePresenter;
    private CarAdapter carAdapter;
    FloatingActionButton fabHome;
    RecyclerView rvHome;
    SwipeRefreshLayout srlHome;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);
        initPresenter();
        initView();
        initDataPresenter();

        addCar();
    }
}

```

```

        refresh();
    }

private void initView() {
    fabHome = findViewById(R.id.fabHome);
    rvHome = findViewById(R.id.rvHome);
    srlHome = findViewById(R.id.srlHome);
}

private void initPresenter() {
    homePresenter = new HomePresenter(this);
}

private void initDataPresenter() {
    homePresenter.getAllCar();
}

private void addCar() {
    fabHome.setOnClickListener(v -> {
        Intent addCar = new Intent(HomeActivity.this,
AddActivity.class);
        startActivityForResult(addCar);
        finish();
    });
}

private void refresh(){
    srlHome.setOnRefreshListener(() ->
initDataPresenter());
}

@Override
public void successShowCar(List<DataCar> dataCars) {
    if (srlHome.isRefreshing()){
        srlHome.setRefreshing(false);
    }
    carAdapter = new CarAdapter(dataCars);
    carAdapter.setAdapterListener(this);
    rvHome.setLayoutManager(new
LinearLayoutManager(this));
    rvHome.setAdapter(carAdapter);
}

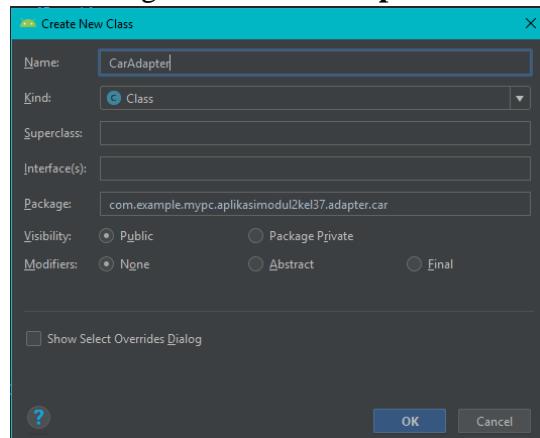
@Override
public void failedShowCar(String message) {
    Toast.makeText(this, "Maaf terjadi kesalahan",
Toast.LENGTH_SHORT).show();
}

@Override
public void onCarClick(DataCar dataCar) {
    //Intent intent = new Intent(this,
DetailActivity.class);
    //intent.putExtra(Constant.Extra.DATA, dataCar);
    //startActivity(intent);
}

```

}

- h. Setelahnya, buat adapter untuk menampung list/daftar data API yang didapatkan, dalam praktikum ini yaitu daftar API Car. Buat class baru pada package **adapter – car** dengan nama **CarAdapter**.



Gambar 4.30. Membuat class CarAdapter

```

import android.support.annotation.NonNull;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import java.util.List;

public class CarAdapter extends RecyclerView.Adapter<CarViewHolder> {

    private List<DataCar> dataCars;
    private CarListerner carListerner;

    public CarAdapter(List<DataCar> dataCars) {
        this.dataCars = dataCars;
    }

    public void setAdapterListener(CarListerner carListerner) {
        this.carListerner = carListerner;
    }

    @NonNull
    @Override
    public CarViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater
            .from(parent.getContext())
            .inflate(R.layout.item_car,
parent,
false);
        return new CarViewHolder(view);
    }
}

```

```

    }

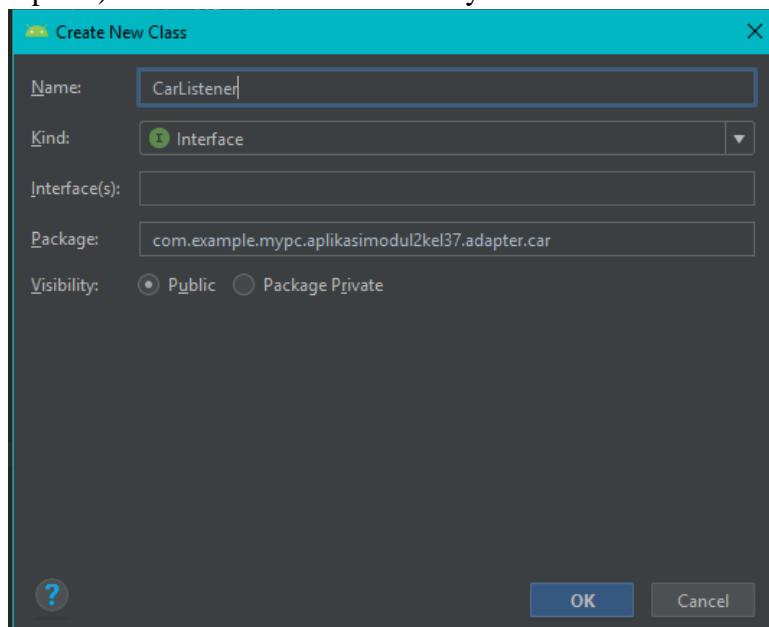
    @Override
    public void onBindViewHolder(@NonNull CarViewHolder
holder, int position) {
        DataCar dataUser = get(position);
        holder.bind(dataUser, carListerner);
    }

    private DataCar get(int position) {
        return dataCars.get(position);
    }

    @Override
    public int getItemCount() {
        if (dataCars == null) return 0;
        return dataCars.size();
    }
}

```

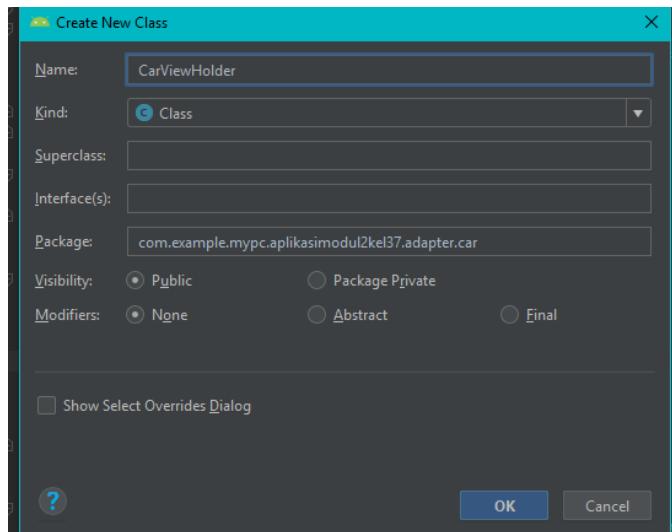
- i. Buat **interface** di package yang sama dengan nama CarListener. **Tambahkan** (bukan replace) code berikut didalam classnya



Gambar 4.31. Membuat class CarListerner

```
void onCarClick(DataCar dataCar);
```

- j. Buat class di package yang sama dengan nama CarViewHolder



Gambar 4.32. Membuat class CarViewHolder

```

import android.support.v7.widget.RecyclerView;
import android.view.View;
import android.widget.LinearLayout;
import android.widget.TextView;

public class CarViewHolder extends RecyclerView.ViewHolder
{
    LinearLayout linearLayout;
    TextView tvHomeName;
    TextView tvHomeMerk;

    public CarViewHolder(View itemView) {
        super(itemView);
        initView(itemView);
    }

    private void initView(View itemView) {
        linearLayout
itemView.findViewById(R.id.llItemC);
        tvHomeName
itemView.findViewById(R.id.tvItemCName);
        tvHomeMerk
itemView.findViewById(R.id.tvItemCMerk);
    }

    public void bind(final DataCar dataCar, final
CarListerner carListerner) {
        tvHomeName.setText(dataCar.getName());
        tvHomeMerk.setText(dataCar.getMerk());

        linearLayout.setOnClickListener(v
carListerner.onCarClick(dataCar));
    }
}

```

2. Membuat addCar

Sama seperti pada home, tambahkan presenter, interface (view) dan activity dengan nama masing – masing. AddPresenter, AddView dan AddActivity. Tambahkan didalam **package addCar** lalu masing – masing replace code yang ada didalamnya dengan code yang ada pada file.txt dibawah

a. AddPresenter

```
import android.util.Log;

import com.google.gson.JsonObject;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class AddPresenter {

    private AddView addView;

    public AddPresenter(AddView addView) {
        this.addView = addView;
    }

    public void addCar() {
        final String tag = "Add-addCar";
        String name = addView.getName();
        String merk = addView.getMerk();
        String model = addView.getModel();
        String year = addView.getYear();

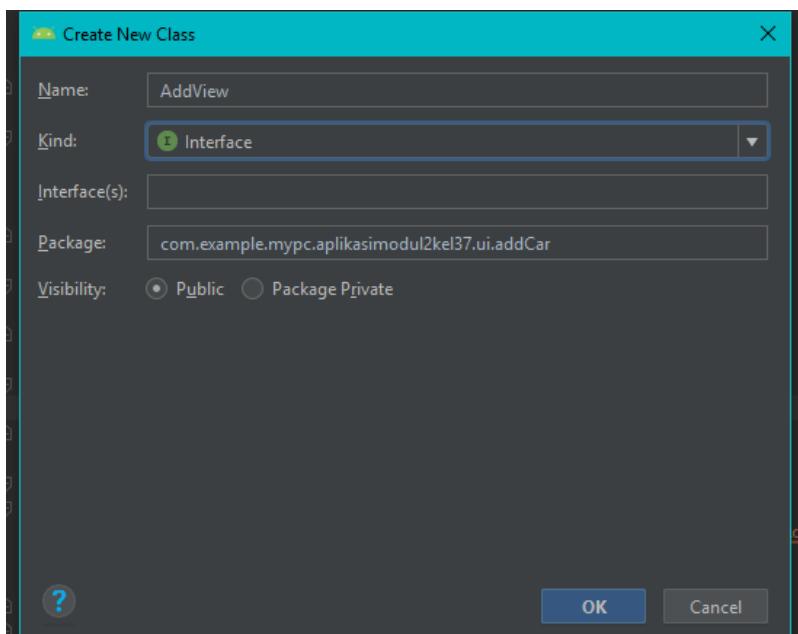
        RetrofitClient.getInstance()
            .getApi()
            .addCar(name, merk, model, year)
            .enqueue(new Callback<JsonObject>() {
                @Override
                public void onResponse(Call<JsonObject> call, Response<JsonObject> response) {
                    if(response.isSuccessful()) {
                        addView.successAddCar();
                        Log.e(tag,
                            response.body().toString());
                    } else {
                        addView.failedAddCar();
                        Log.e(tag,
                            response.body().toString());
                    }
                }

                @Override
                public void onFailure(Call<JsonObject> call, Throwable t) {

```

```
        addView.failedAddCar();
        Log.e(tag,
t.getMessage().toString());
    }
}
}
```

b. AddView (interface)



Gambar 4.33. Membuat Interface AddView

```
public interface AddView {  
  
    String getName();  
  
    String getMerk();  
  
    String getModel();  
  
    String getYear();  
  
    void successAddCar();  
  
    void failedAddCar();  
}
```

c. activity add.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match parent"
```

```
    android:layout_height="match_parent"
    tools:context=".ui.addCar.AddActivity">

    <EditText
        android:id="@+id/etAddName"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginStart="10dp"
        android:layout_marginTop="10dp"
        android:ems="10"
        android:hint="Nama"
        android:inputType="textPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/etAddMerk"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginStart="10dp"
        android:layout_marginTop="5dp"
        android:ems="10"
        android:hint="Merk"
        android:inputType="textPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"

        app:layout_constraintTop_toBottomOf="@+id/etAddName" />

    <EditText
        android:id="@+id/etAddModel"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginStart="10dp"
        android:layout_marginTop="5dp"
        android:ems="10"
        android:hint="Model"
        android:inputType="textPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"

        app:layout_constraintTop_toBottomOf="@+id/etAddMerk" />

    <EditText
        android:id="@+id/etAddYear"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginStart="10dp"
        android:layout_marginTop="5dp"
        android:ems="10"
        android:hint="Tahun"
```

```

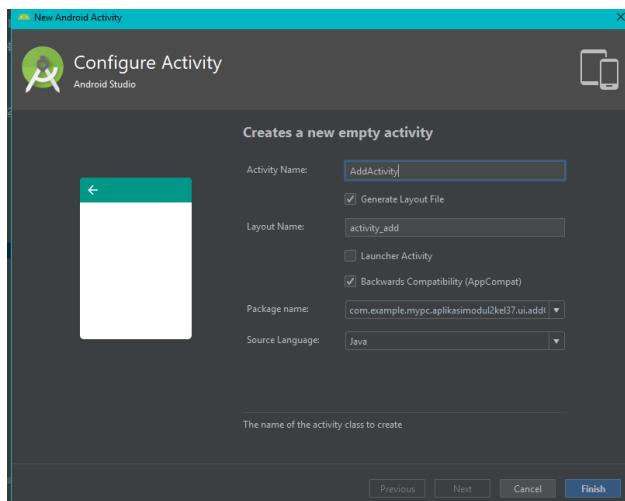
        android:inputType="number"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"

    app:layout_constraintTop_toBottomOf="@+id/etAddModel" />

    <Button
        android:id="@+id	btnAdd"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginTop="10dp"
        android:text="Tambah"
        app:layout_constraintEnd_toEndOf="parent"

    app:layout_constraintTop_toBottomOf="@+id/etAddYear" />
</android.support.constraint.ConstraintLayout>
```

d. AddActivity



Gambar 4.34. Membuat activity AddActivity

```

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast

public class AddActivity extends AppCompatActivity
implements AddView{

    private AddPresenter addPresenter;
    EditText etAddName, etAddMerk, etAddModel, etAddYear;
    Button btnAdd;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add);
        initPresenter();
    }
}
```

```
        initView();

        addCar();
    }

    private void initView() {
        etAddName = findViewById(R.id.etAddName);
        etAddMerk = findViewById(R.id.etAddMerk);
        etAddModel = findViewById(R.id.etAddModel);
        etAddYear = findViewById(R.id.etAddYear);
        btnAdd = findViewById(R.id.btnAdd);
    }

    private void initPresenter() {
        addPresenter = new AddPresenter(this);
    }

    private void addCar() {
        btnAdd.setOnClickListener(v -> {
            addPresenter.addCar();
        });
    }

    @Override
    public String getName() {
        return etAddName.getText().toString();
    }

    @Override
    public String getMerk() {
        return etAddMerk.getText().toString();
    }

    @Override
    public String getModel() {
        return etAddModel.getText().toString();
    }

    @Override
    public String getYear() {
        return etAddYear.getText().toString();
    }

    @Override
    public void successAddCar() {
        Toast.makeText(this, "Berhasil Menambahkan Mobil",
        Toast.LENGTH_SHORT).show();
        Intent home = new Intent(AddActivity.this,
        HomeActivity.class);
        startActivity(home);
        finish();
    }

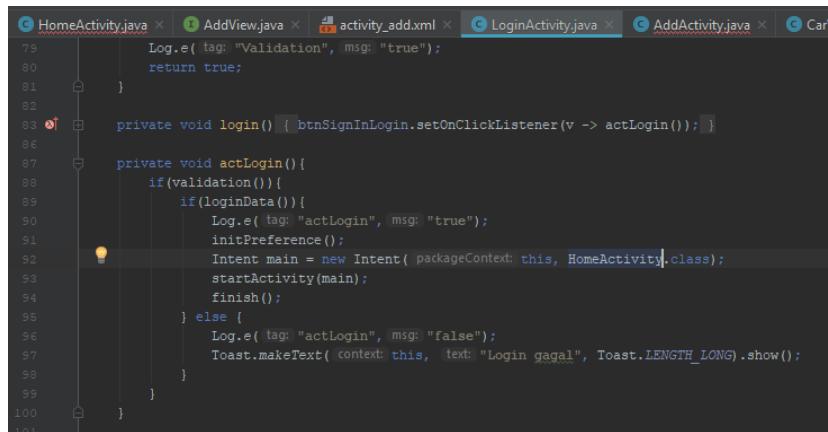
    @Override
    public void failedAddCar() {
```

```

        Toast.makeText(this, "Gagal Menambah Mobil",
Toast.LENGTH_SHORT).show();
    }
}

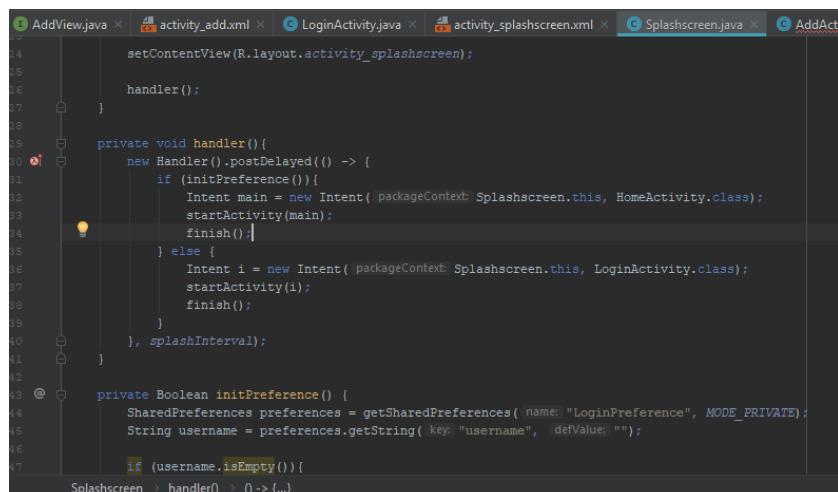
```

3. Pada LoginActivity, ganti bagian intent yang mengarah ke MainActivity menjadi HomeActivity. Seperti pada gambar



Gambar 4.35. Mengubah MainActivity menjadi HomeActivity

4. Ganti pula code pada activity Splashscreen, seperti gambar dibawah



Gambar 4.36. Mengganti pada splashscreen

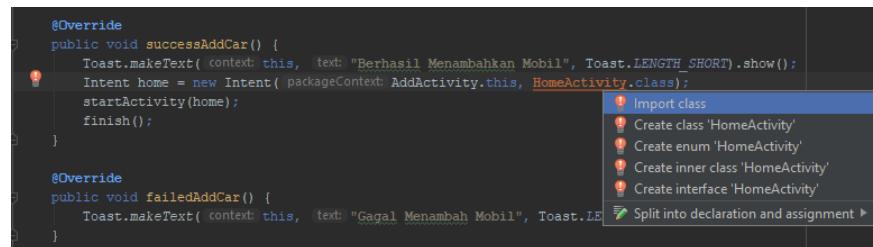
5. Terakhir tambahkan permission pada manifest untuk akses internet dengan cara klik file **Manifest** seperti pada gambar berikut, lalu tambahkan

```

<uses-permission
    android:name="android.permission.INTERNET" />

```

6. Cek semua class yang ada, apabila terdapat code berwarna merah. Arahkan kursor ke code tersebut lalu tekan tombol kombinasi pada keyboard **alt + enter**. Lalu pilih **import class**



Gambar 4.37. Import class

7. Deploy aplikasi ke emulator android atau ke smartphone. Jangan lupa, aktifkan terlebih dahulu **debugging mode** pada smartphone masing – masing melalui developer mode
8. Teliti lagi langkah kerja bila masih gagal, ingat **case-sensitive**

4.5 Analisa Hasil Percobaan

1. DataCar

```

package com.example.mypc.aplikasimodul2kel37.data.model;

import android.os.Parcel;
import android.os.Parcelable;

import com.google.gson.annotations.Expose;
import com.google.gson.annotations.SerializedName;

public class DataCar implements Parcelable { //



    @SerializedName("id")
    @Expose
    private Integer id;
    @SerializedName("name")
    @Expose
    private String name;
    @SerializedName("merk")
    @Expose
    private String merk;
    @SerializedName("model")
    @Expose
    private String model;
    @SerializedName("year")
    @Expose
    private String year;

    protected DataCar(Parcel in) {
        if (in.readByte() == 0) {
            id = null;
        } else {
            id = in.readInt();
        }
        id = in.readInt();
        name = in.readString();
        merk = in.readString();
        model = in.readString();
        year = in.readString();
    }

    public static final Creator<DataCar> CREATOR = new
    Creator<DataCar>() {
        @Override
        public DataCar createFromParcel(Parcel in) {
            return new DataCar(in);
        }

        @Override
        public DataCar[] newArray(int size) {
            return new DataCar[size];
        }
    };

    public Integer getId() {

```

```
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getMerk() {
        return merk;
    }

    public void setMerk(String merk) {
        this.merk = merk;
    }

    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model = model;
    }

    public String getYear() {
        return year;
    }

    public void setYear(String year) {
        this.year = year;
    }

    @Override
    public String toString() {
        return "DataUser{" +
            "id=" + id +
            ", name='" + name + '\'' +
            ", merk='" + merk + '\'' +
            ", model='" + model + '\'' +
            ", year='" + year + '\'' +
            '}';
    }

    @Override
    public int describeContents() {
        return 0;
    }

    @Override
```

```

public void writeToParcel(Parcel dest, int flags) {
    if (id == null) {
        dest.writeByte((byte) 0);
    } else {
        dest.writeByte((byte) 1);
        dest.writeInt(id);
    }
    dest.writeInt(id);
    dest.writeString(name);
    dest.writeString(merk);
    dest.writeString(model);
    dest.writeString(year);
}
}

```

Pada file DataCar ini berisikan objek-objek yang akan digunakan beserta tipe datanya. Objek-objek yang digunakan yaitu id, name, merk, model, dan year yang mana tipe datanya adalah integer untuk id string untuk name, merk, model, dan year. Terdapat juga setter getter yang berguna untuk menambahkan data maupun menampilkan data. `public String toString()` digunakan untuk memasukkan data mobil ke dalam database.

2. Api

```

package com.example.mypc.aplikasimodul2kel37.data.network;

import com.google.gson.JsonObject;

import retrofit2.Call;
import retrofit2.http.Field;
import retrofit2.http.FormUrlEncoded;
import retrofit2.http.GET;
import retrofit2.http.Headers;
import retrofit2.http.POST;
import retrofit2.http.Query;

public interface Api {

    @Headers("Content-Type: application/json")

    @GET("getAllCar.php")
    Call<JsonObject> getAllCar();

    @GET("getCarById.php")
    Call<JsonObject> GetCarById(
        @Query("id") int id
    );

    @FormUrlEncoded
    @POST("addCar.php")
    Call<JsonObject> addCar()
}

```

```

        @Field("name") String name,
        @Field("merk") String merk,
        @Field("model") String model,
        @Field("year") String year
    );
}

}

```

File API berisikan tentang pengambilan data end point yang terdapat pada link yang telah diberikan, seperti getAllCar yang digunakan untuk menampilkan semua data mobil, getCarById yang digunakan untuk menampilkan data mobil sesuai Id nya, dan addCar untuk menambahkan data mobil. Ditentukan juga tipe data yang akan dipakai.

3. RetrofitClient

```

package com.example.mypc.aplikasimodul2kel37.data.network;

import com.example.mypc.aplikasimodul2kel37.utility.Constant;

import java.util.concurrent.TimeUnit;

import okhttp3.OkHttpClient;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;

public class RetrofitClient {
    private static RetrofitClient retrofitClient;

    private RetrofitClient(){}

    public static RetrofitClient getInstance(){
        if(retrofitClient == null){
            retrofitClient = new RetrofitClient();
        }

        return retrofitClient;
    }

    public Api getApi(){
        return getRetrofit().create(Api.class);
    }

    public Retrofit getRetrofit(){

        final OkHttpClient okHttpClient = new
OkHttpClient.Builder() //pakai http client untuk waktu maksimal
koneksi
            .readTimeout(15, TimeUnit.SECONDS)
            .connectTimeout(15, TimeUnit.SECONDS)
            .build();
    }
}

```

```

        return new Retrofit.Builder()
            .baseUrl(Constant.BASE_URL)

        .addConverterFactory(GsonConverterFactory.create())
            .client(okHttpClient)
            .build();
    }
}

```

Retrofit adalah library Android yang dibuat oleh Squire yang digunakan sebagai REST Client pada Android. RetrofitClient berisi tentang retrofit client yang akan digunakan pada aplikasi ini. Pengaturannya yaitu seperti retrofitClient, getApi, dan juga getRetrofit.

4. HomePresenter

```

package com.example.mypc.aplikasimodul2kel137.ui.home;

import android.util.Log;

import com.example.mypc.aplikasimodul2kel137.data.model.DataCar;
import com.example.mypc.aplikasimodul2kel137.data.network.RetrofitClient;
import com.google.gson.Gson;
import com.google.gson.JsonArray;
import com.google.gson.JsonObject;
import com.google.gson.reflect.TypeToken;

import java.lang.reflect.Type;
import java.util.List;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class HomePresenter {

    private HomeView homeView;

    public HomePresenter (HomeView homeView) {
        this.homeView = homeView;
    }

    public void getAllCar(){
        final String tag = "Home-getAllCar";
        RetrofitClient.getInstance()
            .getApi()
            .getAllCar()
            .enqueue(new Callback<JsonObject>() {
                @Override
                public void onResponse(Call<JsonObject> call,
Response<JsonObject> response) { //kalau kita terhubung ke server
                    if (response.isSuccessful()) { //jika data benar

```

```
        JsonObject body = response.body();
        JsonArray array =
body.get("result").getAsJsonArray();
        Type type = new
TypeToken<List<DataCar>>() {}.getType();
        List<DataCar> dataCars = new
Gson().fromJson(array, type);
        homeView.successShowCar(dataCars);
        Log.e(tag,
response.body().toString());
    } else { //jika data salah
        homeView.failedShowCar("Maaf terjadi
kesalahan");
        Log.e(tag,
response.errorBody().toString());
    }
}

@Override
public void onFailure(Call<JsonObject> call,
Throwable t) { //kalau responnya gagal
    homeView.failedShowCar("Maaf terjadi
kesalahan");
    Log.d(tag, t.getMessage().toString());
}
});
```

Home presenter berisikan tentang pengambilan data menggunakan retrofit dari Api pada getAllCar.php. Didalam terdapat sourcecode tentang respon pengambilan data tersebut. Jika berhasil akan menampilkan datanya namun jika gagal akan menampilkan kalimat “Maaf terjadi kesalahan”.

5. HomeView

```
package com.example.mypc.aplikasimodul2kel137.ui.home;

import com.example.mypc.aplikasimodul2kel137.data.model.DataCar;

import java.util.List;

public interface HomeView {
    void successShowCar(List<DataCar> dataCars);
    void failedShowCar(String message);
}
```

HomeView disini berfungsi untuk menghubungkan antara HomeActivity dengan HomePresenter. Di dalamnya berisi void untuk successShowCar dan juga void failedShowCar. Jika successShowCar maka akan ditampilkan list yang berisi

daftar mobil, sedangkan jika failedShowCar maka akan ditampilkan message atau pesannya.

6. HomeActivity

```
package com.example.mypc.aplikasimodul2kel37.ui.home;

import android.content.Intent;
import android.support.design.widget.FloatingActionButton;
import android.support.v4.widget.SwipeRefreshLayout;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.widget.Toast;

import com.example.mypc.aplikasimodul2kel37.R;
import
com.example.mypc.aplikasimodul2kel37.adapter.car.CarAdapter;
import
com.example.mypc.aplikasimodul2kel37.adapter.car.CarListener;
import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;
import com.example.mypc.aplikasimodul2kel37.ui.addCar.AddActivity;
import
com.example.mypc.aplikasimodul2kel37.ui.detail.DetailActivity;
import com.example.mypc.aplikasimodul2kel37.utility.Constant;

import java.util.List;

public class HomeActivity extends AppCompatActivity implements
HomeView, CarListener {

    private HomePresenter homePresenter;
    private CarAdapter carAdapter;
    FloatingActionButton fabHome;
    RecyclerView rvHome;
    SwipeRefreshLayout srlHome;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);
        initPresenter();
        initView();
        initDataPresenter();

        addCar();
        refresh();
    }

    private void initView() {
        fabHome = findViewById(R.id.fabHome);
        rvHome = findViewById(R.id.rvHome);
        srlHome = findViewById(R.id.srlHome);
    }
}
```

```

private void initPresenter() {
    homePresenter = new HomePresenter(this);
}

private void initDataPresenter() {
    homePresenter.getAllCar();
}

private void addCar() {
    fabHome.setOnClickListener(v -> {
        Intent addCar = new Intent(HomeActivity.this,
AddActivity.class);
        startActivity(addCar);
        finish();
    });
}

private void refresh(){
    srlHome.setOnRefreshListener(() -> initDataPresenter());
}

@Override
public void successShowCar(List<DataCar> dataCars) {
    if (srlHome.isRefreshing()){
        srlHome.setRefreshing(false);
    }
    carAdapter = new CarAdapter(dataCars);
    carAdapter.setAdapterListener(this);
    rvHome.setLayoutManager(new LinearLayoutManager(this));
    rvHome.setAdapter(carAdapter);
}

@Override
public void failedShowCar(String message) {
    Toast.makeText(this, "Maaf terjadi kesalahan",
Toast.LENGTH_SHORT).show();
}

@Override
public void onCarClick(DataCar dataCar) {
    Intent intent = new Intent(this, DetailActivity.class);
    intent.putExtra(Constant.Extra.DATA, dataCar);
    startActivity(intent);
}
}

```

Terdapat inisialisasi homePresenter, carAdapter, FloatingActionButton, RecyclerView, dan SwipeRefreshLayout. Kemudian terdapat beberapa method yang akan dipakai seperti initView, initPresenter, initDataPresenter, addCar, dan juga refresh.

7. Activity_home.xml

```

<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.widget.SwipeRefreshLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/srlHome"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ui.home.HomeActivity">

    <android.support.constraint.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <android.support.design.widget.FloatingActionButton
            android:id="@+id/fabHome"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginEnd="15dp"
            android:layout_marginBottom="15dp"
            android:clickable="true"

            app:backgroundTint="@color/design_default_color_primary"
            app:fabSize="normal"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:srcCompat="@android:drawable/ic_menu_add" />

        <android.support.v7.widget.RecyclerView
            android:id="@+id/rvHome"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_marginEnd="5dp"
            android:layout_marginStart="5dp"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

    </android.support.constraint.ConstraintLayout>
</android.support.v4.widget.SwipeRefreshLayout>

```

Activity_home.xml otomatis terbuat saat pembuatan HomeActivity. File ini berfungsi untuk membuat tampilan halaman utama saat program dijalankan setelah login. Dalam pengaturannya dibuat menjadi constraint layout. Terdapat FloatingActionButton yang berfungsi untuk menambah data mobil pada halaman home. Terdapat pula RecyclerView yang berfungsi untuk menampilkan data mobil yang sudah dimasukkan.

8. Item_car.xml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<android.support.v7.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:minHeight="50dp"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_marginTop="5dp"
    android:layout_marginStart="10dp"
    android:layout_marginEnd="10dp"
    app:cardCornerRadius="15dp"
    android:clickable="true">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:minHeight="50dp"
        android:clickable="true">

        <LinearLayout
            android:id="@+id/l1ItemC"
            android:layout_width="match_parent"
            android:layout_height="60dp"
            android:clickable="true"
            android:orientation="horizontal">

            <ImageView
                android:id="@+id/ivItemC"
                android:layout_width="35dp"
                android:layout_height="35dp"
                android:layout_gravity="center"
                android:layout_marginStart="20dp"
                android:src="@mipmap/ic_launcher" />

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:orientation="vertical"
                android:gravity="center">

                <TextView
                    android:id="@+id/tvItemCName"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_gravity="center"
                    android:layout_marginStart="15dp"
                    android:text="Name"
                    android:textSize="18sp"
                    android:textStyle="bold"
                    tools:text="" />

                <TextView
                    android:id="@+id/tvItemCMerk"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_gravity="center"
```

```

        android:layout_marginStart="15dp"
        android:text="Name"
        android:textSize="18sp"
        tools:text="" />
    </LinearLayout>
</LinearLayout>
</RelativeLayout>

</android.support.v7.widget.CardView>

```

Item_car.xml berfungsi untuk mengatur tampilan daftar atau list mobil di halaman home. Pada file ini menggunakan relative layout. Terdapat image view yang digunakan untuk menampilkan logo ic_launcher dan juga terdapat text view yang digunakan untuk menampilkan nama dan merk data mobil.

9. CarAdapter

```

package com.example.mypc.aplikasimodul2kel37.adapter.car;

import android.support.annotation.NonNull;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import com.example.mypc.aplikasimodul2kel37.R;
import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;

import java.util.List;

public class CarAdapter extends
RecyclerView.Adapter<CarViewHolder> {

    private List<DataCar> dataCars;
    private CarListener carListerner;

    public CarAdapter(List<DataCar> dataCars) {
        this.dataCars = dataCars;
    }

    public void setAdapterListener(CarListener carListerner) {
        this.carListerner = carListerner;
    }

    @NonNull
    @Override
    public CarViewHolder onCreateViewHolder(@NonNull ViewGroup
parent, int viewType) {
        View view = LayoutInflater
            .from(parent.getContext())
            .inflate(R.layout.item_car, parent, false);

```

```

        return new CarViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull CarViewHolder holder,
    int position) {
        DataCar dataUser = get(position);
        holder.bind(dataUser, carListerner);
    }

    private DataCar get(int position) {
        return dataCars.get(position);
    }

    @Override
    public int getItemCount() {
        if (dataCars == null) return 0;
        return dataCars.size();
    }
}

```

CarAdapter berfungsi untuk menampung data API yang mana API sendiri berfungsi untuk mengintegrasikan aplikasi yang berbeda secara bersamaan.

10. CarListener

```

package com.example.mypc.aplikasimodul2kel37.adapter.car;

import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;

public interface CarListener {
    void onCarClick(DataCar dataCar);
}

```

CarListener berisi method pengambilan data car dari dataCar.

11. CarViewHolder

```

package com.example.mypc.aplikasimodul2kel37.adapter.car;

import android.support.v7.widget.RecyclerView;
import android.view.View;
import android.widget.LinearLayout;
import android.widget.TextView;

import com.example.mypc.aplikasimodul2kel37.R;
import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;

public class CarViewHolder extends RecyclerView.ViewHolder {

    LinearLayout linearLayout;
    TextView tvHomeName;
    TextView tvHomeMerk;

    public CarViewHolder(View itemView) {
        super(itemView);
    }
}

```

```

        initView(itemView);
    }

    private void initView(View itemView) {
        linearLayout = itemView.findViewById(R.id.llItemC);
        tvHomeName = itemView.findViewById(R.id.tvItemCName);
        tvHomeMerk = itemView.findViewById(R.id.tvItemCMerk);
    }

    public void bind(final DataCar dataCar, final CarListener
carListerner) {
        tvHomeName.setText(dataCar.getName());
        tvHomeMerk.setText(dataCar.getMerk());

        linearLayout.setOnClickListener(v ->
carListerner.onCarClick(dataCar));
    }
}

```

CarViewholder digunakan untuk pengaturan tampilan list pada home activity. Terdapat inisialisasi LinearLayout, tvHomeName, dan juga tvHomeMerk. Lalu pada initView menyambungkan objek dengan tampilan yang berada pada item_car.xml. Untuk pengisian TextViewnya diambil dari DataCar lalu di setText ke tvHomeName dan tvHomeMerk.

12. AddPresenter

```

package com.example.mypc.aplikasimodul2kel37.ui.addCar;

import android.util.Log;

import
com.example.mypc.aplikasimodul2kel37.data.network.RetrofitClient;
import com.google.gson.JsonObject;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class AddPresenter {

    private AddView addView;

    public AddPresenter(AddView addView) {
        this.addView = addView;
    }

    public void addCar() {
        final String tag = "Add-addCar";
        String name = addView.getName();
        String merk = addView.getMerk();
    }
}

```

```

String model = addView.getModel();
String year = addView.getYear();

RetrofitClient.getInstance()
    .getApi()
    .addCar(name, merk, model, year)
    .enqueue(new Callback<JsonObject>() {
        @Override
        public void onResponse(Call<JsonObject> call,
Response<JsonObject> response) {
            if(response.isSuccessful()){
                addView.successAddCar();
                Log.e(tag,
response.body().toString());
            } else {
                addView.failedAddCar();
                Log.e(tag,
response.body().toString());
            }
        }

        @Override
        public void onFailure(Call<JsonObject> call,
Throwable t) {
            addView.failedAddCar();
            Log.e(tag, t.getMessage().toString());
        }
    });
}
}

```

AddPresenter berisi method addcar yng berfungsi untuk menambahkan data mobil ke dalam database API yang digunakan. Terdapat inisialisasi objek dan tipe data yang digunakan yaitu name, merk, model, dan year.

13. AddView

```

package com.example.mypc.aplikasimodul2kel37.ui.addView;

public interface AddView {

    String getName();

    String getMerk();

    String getModel();

    String getYear();

    void successAddCar();

    void failedAddCar();
}

```

File AddView berisi method-method yang mana berfungsi untuk menghubungkan antara AddActivity dan AddPresenter.

14. AddActivity

```
package com.example.mypc.aplikasimodul2kel37.ui.addCar;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.example.mypc.aplikasimodul2kel37.R;
import com.example.mypc.aplikasimodul2kel37.ui.home.HomeActivity;

public class AddActivity extends AppCompatActivity implements
AddView{

    private AddPresenter addPresenter;
    EditText etAddName, etAddMerk, etAddModel, etAddYear;
    Button btnAdd;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add);
        initPresenter();
        initView();

        addCar();
    }

    private void initView() {
        etAddName = findViewById(R.id.etAddName);
        etAddMerk = findViewById(R.id.etAddMerk);
        etAddModel = findViewById(R.id.etAddModel);
        etAddYear = findViewById(R.id.etAddYear);
        btnAdd = findViewById(R.id.btnAdd);
    }

    private void initPresenter() {
        addPresenter = new AddPresenter(this);
    }

    private void addCar() {
        btnAdd.setOnClickListener(v -> {
            addPresenter.addCar();
        });
    }

    @Override
    public String getName() {
        return etAddName.getText().toString();
    }
}
```

```

    }

    @Override
    public String getMerk() {
        return etAddMerk.getText().toString();
    }

    @Override
    public String getModel() {
        return etAddModel.getText().toString();
    }

    @Override
    public String getYear() {
        return etAddYear.getText().toString();
    }

    @Override
    public void successAddCar() {
        Toast.makeText(this, "Berhasil Menambahkan Mobil",
        Toast.LENGTH_SHORT).show();
        Intent home = new Intent(AddActivity.this,
        HomeActivity.class);
        startActivity(home);
        finish();
    }

    @Override
    public void failedAddCar() {
        Toast.makeText(this, "Gagal Menambah Mobil",
        Toast.LENGTH_SHORT).show();
    }
}

```

Pada AddActivity terdapat method utama pada halaman add yang dibuat pada activity_add.xml. Pertama kita inisialisasikan objek-objek yang kita gunakan seperti AddPresenter, EditText, dan button. Lalu inisialisasi method yang akan digunakan pada AddActivity seperti initPresenter, initView, dan addCar. Pada initPresenter kita menginisialisasi presenter yang digunakan yaitu HomePresenter. Lalu pada initView kita menyambungkan objek pada AddActivity dengan yang berada pada activity_add.xml. Lalu pada initPresenter kita menginisialisasi presenter yang kita gunakan yaitu AddPresenter. Pada addCar jika kita tekan tombol add maka akan berpindah ke AddPresenter. Lalu terdapat method getName, getMerk, getModel, dan getYear yang semuanya diambil dari text yang ada pada EditText pada activity_add.xml. Lalu terdapat method jika berhasil menambahkan maka akan menampilkan Toast “Berhasil Menambahkan Mobil” dan terdapat method jika gagal maka akan menampilkan Toast “Gagal Menambah Mobil”.

15. Activity_add.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ui.addCar.AddActivity">

    <EditText
        android:id="@+id/etAddName"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginStart="10dp"
        android:layout_marginTop="10dp"
        android:ems="10"
        android:hint="Nama"
        android:inputType="textPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/etAddMerk"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginStart="10dp"
        android:layout_marginTop="5dp"
        android:ems="10"
        android:hint="Merk"
        android:inputType="textPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/etAddName" />

    <EditText
        android:id="@+id/etAddModel"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginStart="10dp"
        android:layout_marginTop="5dp"
        android:ems="10"
        android:hint="Model"
        android:inputType="textPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/etAddMerk" />

    <EditText
        android:id="@+id/etAddYear"
        android:layout_width="0dp"
```

```
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginStart="10dp"
        android:layout_marginTop="5dp"
        android:ems="10"
        android:hint="Tahun"
        android:inputType="number"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/etAddModel" />

    <Button
        android:id="@+id/btnAdd"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginEnd="10dp"
        android:layout_marginTop="10dp"
        android:text="Tambah"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/etAddYear" />
</android.support.constraint.ConstraintLayout>
```

Activity_add.xml berfungsi untuk mengatur tampilan atau layout pada halaman menambah data mobil. Terdapat 4 editText yaitu untuk memasukkan Nama, Merk, Model, dan Tahun. Terdapat pula satu tombol atau button yang bertuliskan “Tambah” yang berfungsi untuk menambah data mobil yang dituliskan di editText untuk disimpan di database API.

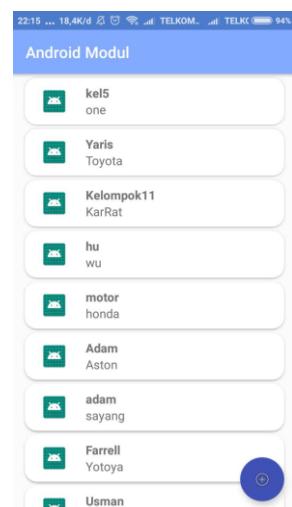
Berikut adalah hasil tangkapan layar tampilan-tampilan pada praktikum ini,



Gambar 4.38. Tampilan splashscreen



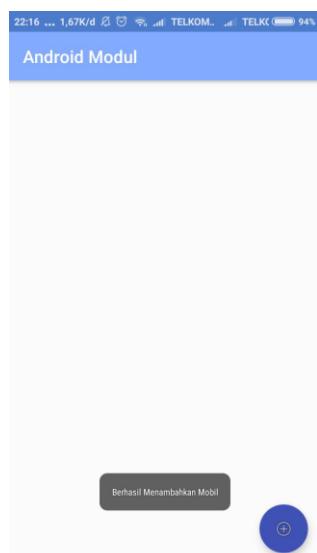
Gambar 4.39. Tampilan login



Gambar 4.40. Tampilan Halaman utama



Gambar 4.41. Tampilan Tambah data



Gambar 4.42. Tampilan tambah data sukses

4.6 Tugas

Pada tugas ini, terdapat dua level dimana untuk level 1 praktikan diminta untuk membuat sebuah activity baru bernama DetailCar yang mana apabila ketika user melakukan klik pada data mobil yang ada pada halaman utama maka akan ditampilkan data detail dari mobil tersebut. Berikut adalah langkah-langkah dalam pembuatan activity baru ini,

1. Detail Activity

```
package com.example.mypc.aplikasimodul2kel37.ui.detail;

import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;
import android.widget.Toast;

import com.example.mypc.aplikasimodul2kel37.R;
import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;
import com.example.mypc.aplikasimodul2kel37.utility.Constant;

import java.util.List;

public class DetailActivity extends AppCompatActivity implements
DetailView {

    private DataCar dataCar;
    private DetailPresenter detailPresenter;
    private TextView tvNama;
    private TextView tvMerk;
    private TextView tvModel;
    private TextView tvTahun;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_detail);
        initView();
        initIntentData();
        initPresenter();
        initData();
    }

    private void initView() {
        tvNama = findViewById(R.id.tvTampil_nama);
        tvMerk = findViewById(R.id.tvTampil_Merek);
        tvModel = findViewById(R.id.tvTampil_Model);
        tvTahun = findViewById(R.id.tvTampil_Tahun);
    }

    private void initData() {
        detailPresenter.getCarById(dataCar);
    }
}
```

```

    }

    private void initPresenter() {
        detailPresenter = new DetailPresenter(this);
    }

    private void initIntentData() {
        dataCar =
getIntent().getParcelableExtra(Constant.Extra.DATA);
        if (dataCar == null) finish();
    }

    @Override
    public void showErrorCarById(String message) {
        Toast.makeText(this, message, Toast.LENGTH_SHORT).show();
    }

    @Override
    public void showSuccessCarById(List<DataCar> car) {
        tvNama.setText(car.get(0).getName());
        tvMerk.setText(car.get(0).getMerk());
        tvModel.setText(car.get(0).getModel());
        tvTahun.setText(car.get(0).getYear());
    }
}
}

```

DetailActivity ini merupakan sebuah activity yang mana akan menampilkan data mobil berupa Nama, Merk, Model, dan Tahun jika showSuccesscarById. Untuk menampilkan data detail dari sebuah mobil menggunakan id. Terdapat inisialisasi objek-objek yang akan digunakan seperti DetailPresenter, DataCar, dan EditText. Kemudian diinisialisasikan method yang akan digunakan pada home activity seperti initPresenter, intiView, initIntentData, dan juga initData. Pada initPresenter diinisialisasi presenter yang digunakan yaitu DetailPresenter. Lalu pada initView kita menyambungkan objek pada DetailActivity dengan yang berada pada activity_Detail.xml. Lalu pada initData kita menginisialisasi data yang kita gunakan. Yaitu getCarById diambil dari DetailPresenter. Lalu terdapat method getName, getMerk, getModel, getYear yang berfungsi untuk mengambil data dari database API lalu dipasangkan pada EditText pada activity_detail.xml.

2. Activity_detail.xml

```

<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"

```

```
tools:context=".ui.detail.DetailActivity">

    <TextView
        android:id="@+id/tvTampil_Tahun"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="28dp"
        android:textSize="24sp"
        app:layout_constraintEnd_toEndOf="@+id/tvTampil_Model"
        app:layout_constraintStart_toStartOf="@+id/tvTampil_Model"
        app:layout_constraintTop_toBottomOf="@+id/tvTampil_Model"
    />

    <TextView
        android:id="@+id/tvTampil_Model"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="28dp"
        android:textSize="24sp"
        app:layout_constraintEnd_toEndOf="@+id/tvTampil_Merek"
        app:layout_constraintStart_toStartOf="@+id/tvTampil_Merek"
        app:layout_constraintTop_toBottomOf="@+id/tvTampil_Merek"
    />

    <TextView
        android:id="@+id/tvTampil_Merek"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="8dp"
        android:textSize="24sp"
        app:layout_constraintEnd_toEndOf="@+id/tvTampil_nama"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="@+id/tvTampil_nama"
        app:layout_constraintTop_toBottomOf="@+id/tvTampil_nama"
    />

    <TextView
        android:id="@+id/tvTampil_nama"
        android:layout_width="247dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="48dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:textSize="24sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"

        app:layout_constraintStart_toEndOf="@+id/textView_tampilNama"
        app:layout_constraintTop_toBottomOf="@+id/textView5_TambilDetail"
        app:layout_constraintVertical_bias="0.0" />
```

```
<TextView
    android:id="@+id/textView_tampilNama"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:text="@string/namacar"
    android:textSize="24sp"

    app:layout_constraintBottom_toBottomOf="@+id/tvTampil_nama"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="@+id/tvTampil_nama"
        app:layout_constraintVertical_bias="1.0" />

<TextView
    android:id="@+id/textView2_TampilMerek"
    android:layout_width="75dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:layout_marginEnd="8dp"
    android:text="@string/merkcar"
    android:textSize="24sp"

    app:layout_constraintBottom_toBottomOf="@+id/tvTampil_Merek"
        app:layout_constraintEnd_toStartOf="@+id/tvTampil_Merek"
        app:layout_constraintHorizontal_bias="1.0"

    app:layout_constraintStart_toStartOf="@+id/textView_tampilNama"
        app:layout_constraintTop_toTopOf="@+id/tvTampil_Merek"
        app:layout_constraintVertical_bias="0.0" />

<TextView
    android:id="@+id/textView3_TampilModel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginEnd="8dp"
    android:text="@string/modelcar"
    android:textSize="24sp"

    app:layout_constraintBottom_toBottomOf="@+id/tvTampil_Model"
        app:layout_constraintEnd_toStartOf="@+id/tvTampil_Model"

    app:layout_constraintStart_toStartOf="@+id/textView2_TampilMerek"
        app:layout_constraintTop_toTopOf="@+id/tvTampil_Model"
        app:layout_constraintVertical_bias="0.0" />

<TextView
    android:id="@+id/textView4_TampilTahun"
    android:layout_width="wrap_content"
    android:layout_height="30dp"
    android:layout_marginEnd="8dp"
    android:text="@string/tahuncar"
    android:textSize="24sp"

    app:layout_constraintBottom_toBottomOf="@+id/tvTampil_Tahun"
        app:layout_constraintEnd_toStartOf="@+id/tvTampil_Tahun"
```

```

    app:layout_constraintStart_toStartOf="@+id/textView3_TampilModel"
    app:layout_constraintTop_toTopOf="@+id/tvTampil_Tahun" />

    <TextView
        android:id="@+id/textView5_TambilDetail"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:text="@string/tittlecar"
        android:textSize="24sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>

```

Inti dari file ini yaitu untuk mengatur tampilan pada saat ditampilkan detail data dari suatu mobil. Pada file xml ini digunakan Constraint Layout. Terdapat text view yang digunakan untuk menampilkan nameCar, merkCar, modelCar, dan juga yearCar.

3. DetailPresenter

```

package com.example.mypc.aplikasimodul2kel37.ui.detail;

import android.util.Log;

import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;
import com.example.mypc.aplikasimodul2kel37.data.network.RetrofitClient;
import com.google.gson.Gson;
import com.google.gson.JsonArray;
import com.google.gson.JsonObject;
import com.google.gson.reflect.TypeToken;

import java.lang.reflect.Type;
import java.util.List;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class DetailPresenter {
    private final DetailView mView ;
    public DetailPresenter(DetailView detailView) { mView
    =detailView; }

    public void getCarById(DataCar dataCar) {
        RetrofitClient.getInstance()
            .getApi()
            .GetCarById(dataCar.getId())
    }
}

```

```

        .enqueue(new Callback<JSONObject>() {
            @Override
            public void onResponse(Call<JSONObject> call,
Response<JSONObject> response) {
                if (response.isSuccessful()) {
                    JSONObject body = response.body();
                    JSONArray array =
body.get("result").getAsJSONArray();
                    Type type =new
TypeToken<List<DataCar>>().getType();
                    List<DataCar> dataCar = new
Gson().fromJson(array, type);
                    mView.showSuccessCarById(dataCar);
                } else {
                    mView.showErrorCarById("Terdapat
kesalahan, mohon coba kembali");
                }
            }
            @Override
            public void onFailure(Call<JSONObject> call,
Throwable t) {
                Log.d("DATA", t.getMessage());
                mView.showErrorCarById(t.getMessage());
            }
        );
    }
}

```

DetailPresenter digunakan untuk mengambil data getCarById yang diambil dari DataCar. Data yang diambil merupakan data yang sesuai dengan id mobil tersebut yang dipilih. Jika berhasil maka akan ditampilkan data mobil yang dipilih, namun jika gagal maka akan muncul “Terdapat kesalahan, mohon coba kembali”.

4. DetailView

```

package com.example.mypc.aplikasimodul2kel37.ui.detail;

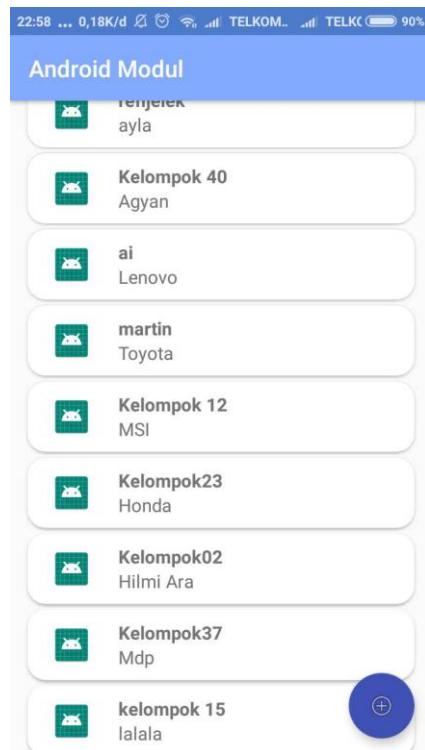
import com.example.mypc.aplikasimodul2kel37.data.model.DataCar;
import java.util.List;

public interface DetailView {
    void showErrorCarById(String message);

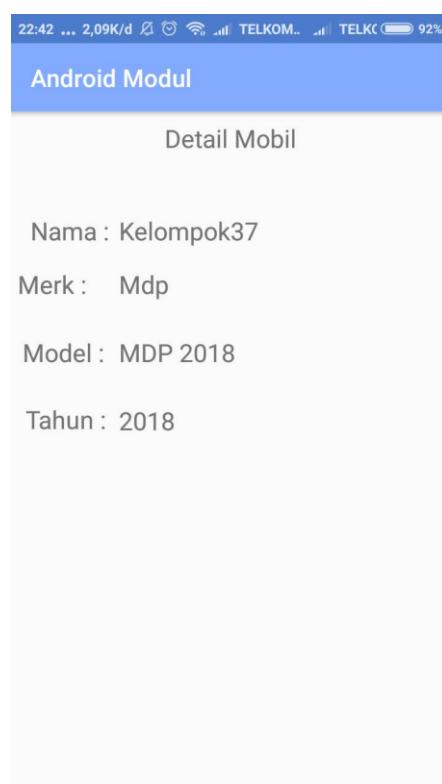
    void showSuccessCarById(List<DataCar> dataCar);
}

```

Pada DetailView terdapat method-method yang akan digunakan pada DetailActivity dan juga DetailPresenter. Berikut adalah hasil tangkapan layar tugas ini,



Gambar 4.43. Tampilan Utama



Gambar 4.44. Tampilan Detail Car

4.7 Kesimpulan

1. Ketika membuat suatu activity baru maka otomatis akan terbuat file .xml yang berfungsi untuk mengatur tampilan halaman tersebut.
2. View hanya berisi inisialisasi method yang digunakan pada activity dan presenter.
3. Dalam menjalankan aplikasi, perangkat android harus terhubung dengan koneksi internet karena aplikasi pada praktikum ini dibuat dengan terhubung dengan database online.
4. Presenter digunakan untuk mengambil data dari luar seperti retrofit, API, dan lain sebagainya.
5. Import class digunakan agar suatu class lain dapat digunakan pada class tersebut yang diinginkan.
6. Intent digunakan untuk pindah Activity.
7. Toast digunakan untuk memberikan pemberitahuan berupa Pop Up.
8. Untuk mempermudah pengaturan dan pengelompokan suatu activity dapat dibuat suatu package.

BAB V

Progressive Web App (PWA) #1

5.1. Tujuan

1. Praktikan dapat memahami konsep dasar teknologi *Progressive Web App*
2. Praktikan dapat mengunggah suatu project PWA ke *hosting firebase*
3. Praktikan dapat memahami konsep dasar tentang *framework Vue.js*
4. Praktikan dapat mengetahui penggunaan Node.js
5. Praktikan dapat memahami konsep dasar penggunaan NPM
6. Praktikan mampu mendeploy ke *smartphone* hasil dari praktikum tentang PWA

5.2. Alat dan Bahan

5.2.1. Laptop



Gambar 5.1 Laptop

Laptop digunakan sebagai media dalam penggerjaan praktikum kali ini. Kita menginstall dan melakukan kegiatan *programming* pada laptop.

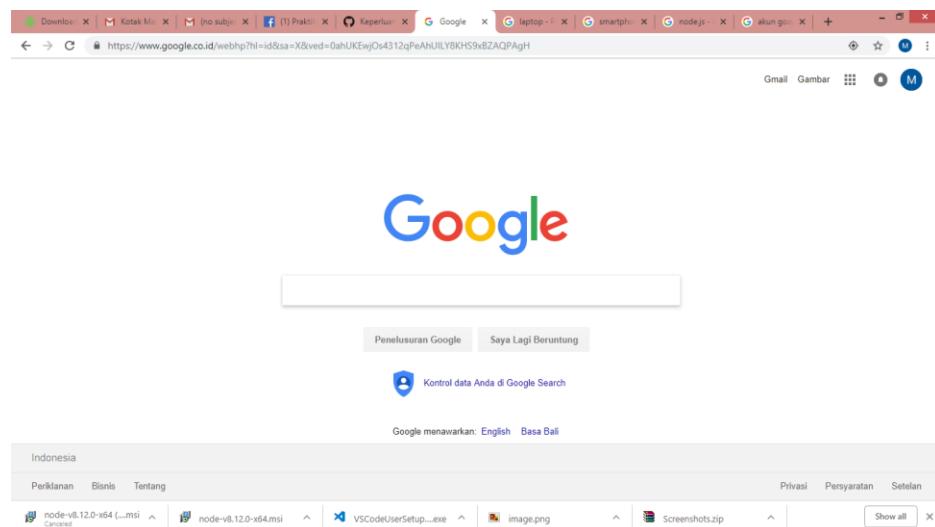
5.2.2. Smartphone



Gambar 5.2 Smartphone

Smartphone digunakan sebagai media untuk melihat hasil akhir tampilan dari program yang telah kita buat. Program yang dibuat nantinya akan *dideploy* ke *smartphone*.

5.2.3. Aplikasi Google Chrome



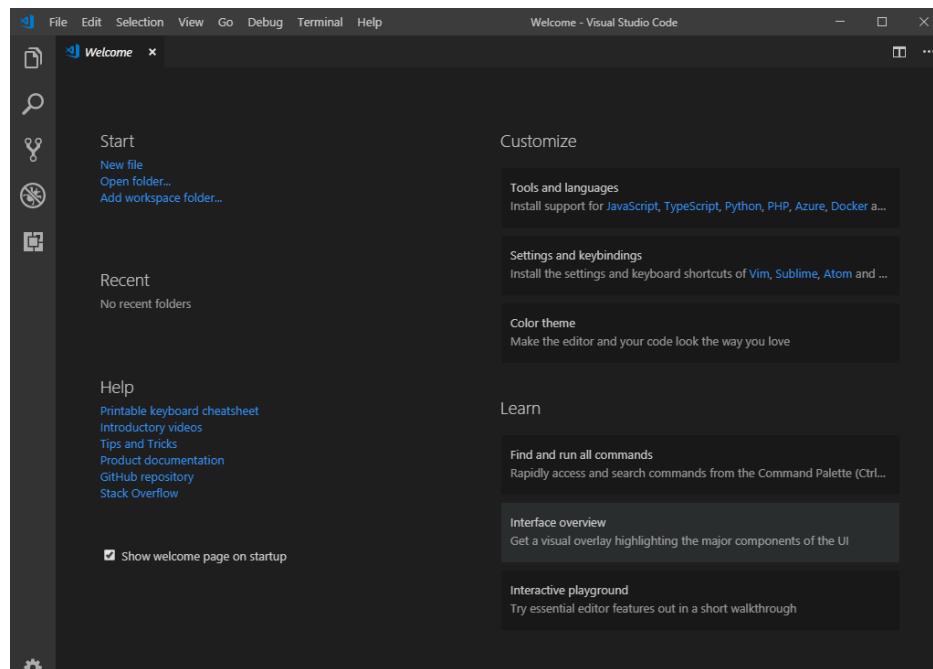
Gambar 5.3 Google Chrome Pada Laptop



Gambar 5.4 Google Chrome Pada Handphone

Google Chrome digunakan untuk kita membuka alamat *hosting* yang telah dibuat sebelumnya. Kita dapat melihat tampilan pada alamat *hosting* yang telah dibuat.

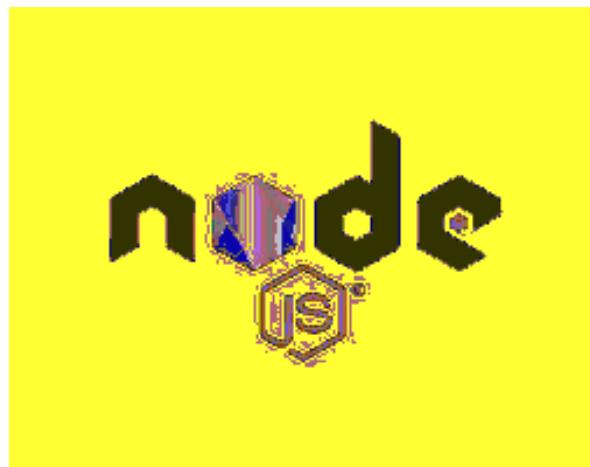
5.2.4. Text Editor



Gambar 5.5 Visual Studio Code

Text Editor yang akan digunakan pada praktikum kali ini yaitu Visual Studio Code. Di sinilah kita akan melakukan kegiatan *coding program* untuk praktikum.

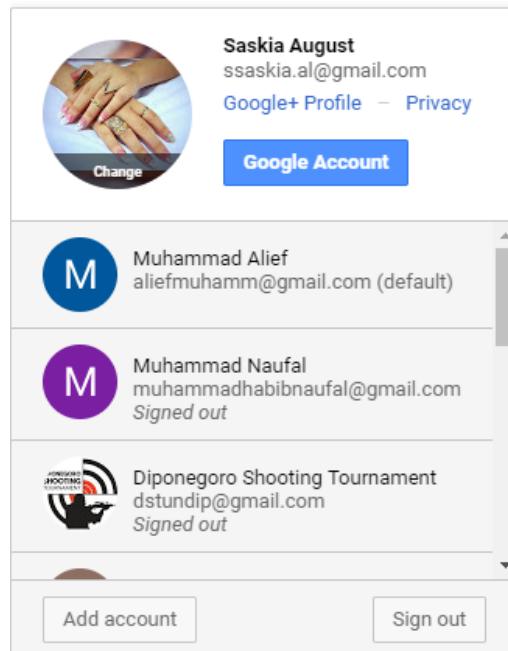
5.2.5. Node.js



Gambar 5.6 Node.js

Kita menggunakan Node.js pada praktikum kali ini sebagai perangkat lunak untuk mengembangkan aplikasi berbasis *web* dan ditulis dalam sintaks bahasa pemrograman JavaScript.

5.2.6. Akun Google aktif



Gambar 5.7 Akun google aktif

Dibutuhkan akun google aktif untuk mendaftar pada firebase dikarenakan pada praktikum ini digunakan firebase untuk hosting konten statis dan konfigurasi ringan untuk PWA yang rumit.

5.3. Dasar Teori

5.3.1. Progressive Web App (PWA)

Progressive Web App adalah pengalaman yang menggabungkan yang terbaik dari *web* dan yang terbaik dari aplikasi. Pengalaman ini bermanfaat untuk pengguna dari kunjungan pertamanya di *tab browser*, tanpa harus melakukan pemasangan. Pengguna secara progresif akan membangun hubungan dengan aplikasi, yang semakin lama semakin kuat. Aplikasi dimuat cepat, bahkan pada jaringan yang tidak stabil, mengirimkan pemberitahuan *push* yang relevan, memiliki ikon pada layar beranda, dan dimuat dengan pengalaman tingkat atas selayar penuh.

PWA memiliki sifat-sifat sebagai berikut :

- **Progresif** - Bekerja untuk setiap pengguna, apa pun pilihan *browser* mereka karena dibangun dengan peningkatan progresif sebagai konsep intinya.
- **Responsif** - Cocok dengan setiap faktor bentuk: perangkat *desktop*, seluler, tablet, atau apa saja yang muncul berikutnya.
- **Konektivitas independen** - Disempurnakan dengan *service worker* agar bisa bekerja *offline* atau pada jaringan berkualitas-rendah.
- **Seperti-Aplikasi** - Terasa seperti sebuah aplikasi untuk pengguna dengan interaksi dan navigasi bergaya-aplikasi karena mereka dibangun di atas model *shell* aplikasi.
- **Segar** - Selalu terkini berkat proses pembaruan *service worker*.
- **Aman** - Disediakan melalui HTTPS untuk mencegah *snooping* dan memastikan materi belum dirusak.
- **Dapat ditemukan** - Dapat diidentifikasi sebagai "aplikasi" berkat manifes W3C dan cakupan registrasi *service worker*, yang memungkinkan mesin telusur untuk menemukannya.
- **Bisa dilibatkan-kembali** - Kemudahan untuk dilibatkan-kembali dengan fitur seperti pemberitahuan *push*.
- **Dapat dipasang** - Memungkinkan pengguna untuk "menyimpan" aplikasi yang mereka anggap paling berguna di layar beranda tanpa kerumitan toko aplikasi.

- **Bisa ditautkan** - Dapat dengan mudah dibagikan melalui URL, tidak memerlukan pemasangan yang rumit.

Sumber: <https://developers.google.com/web/fundamentals/codelabs/your-first-pwapp/?hl=id>

Diakses pada 26 Oktober 2018

5.3.2. Web Server

Server atau *Web server* adalah sebuah *software* yang memberikan layanan berbasis data dan berfungsi menerima permintaan dari HTTP atau HTTPS pada klien yang dikenal dan biasanya kita kenal dengan nama *web browser* (Mozilla Firefox, Google Chrome) dan untuk mengirimkan kembali yang hasilnya dalam bentuk beberapa halaman *web* dan pada umumnya akan berbentuk dokumen HTML.

Fungsi utama *Server* atau *Web server* adalah untuk melakukan atau akan mentransfer berkas permintaan pengguna melalui protokol komunikasi yang telah ditentukan sedemikian rupa. Halaman *web* yang diminta terdiri dari berkas teks, video, gambar, *file* dan banyak lagi. Pemanfaatan *web server* berfungsi untuk mentransfer seluruh aspek pemberkasan dalam sebuah halaman *web* termasuk yang di dalam berupa teks, video, gambar dan banyak lagi.

Salah satu contoh dari *Web Server* adalah Apache. Apache (*Apache Web Server – The HTTP Web Server*) merupakan *web server* yang paling banyak dipergunakan di Internet. Program ini pertama kali didesain untuk sistem operasi lingkungan UNIX. Apache mempunyai program pendukung yang cukup banyak. Hal ini memberikan layanan yang cukup lengkap bagi penggunanya.

Sumber: <https://idcloudhost.com/pengertian-web-server-dan-fungsinya/>

Diakses pada 26 Oktober 2018

5.3.3. Service Worker

Service worker adalah skrip yang dijalankan *browser* Anda di latar belakang, terpisah dari laman *web*, yang membuka pintu ke berbagai fitur yang tidak memerlukan laman *web* atau interaksi pengguna. Saat ini, *service worker* sudah menyertakan berbagai fitur seperti pemberitahuan *push* dan sinkronisasi latar belakang. Di masa mendatang, *service worker* akan mendukung hal-hal lainnya seperti sinkronisasi berkala atau *geofencing*. Fitur inti yang didiskusikan

dalam tutorial adalah kemampuan mencegat dan menangani permintaan jaringan, termasuk mengelola *cache* respons lewat program.

Yang membuat API ini menarik adalah karena memungkinkan Anda mendukung pengalaman *offline*, yang memberikan *developer* kontrol penuh atas pengalaman.

Sebelum *service worker*, ada satu API lain yang memberi pengguna pengalaman *offline* di web, yang disebut AppCache. Masalah utama pada AppCache adalah jumlah *gotcha* yang ada serta fakta bahwa meskipun desain bekerja dengan sangat baik untuk laman aplikasi *web* tunggal, namun ternyata tidak begitu baik untuk situs multi-laman. *Service worker* telah didesain untuk menghindari titik-titik menyulitkan yang sudah umum ini.

Sumber: <https://developers.google.com/web/fundamentals/primers/service-workers/?hl=id>

Diakses pada 26 Oktober 2018

5.3.4. Web App Manifest

Manifes aplikasi web adalah *file JSON* sederhana yang memberikan Anda, *developer*, kemampuan untuk mengontrol bagaimana aplikasi terlihat oleh pengguna di daerah yang mereka harap akan melihat aplikasi (misalnya, layar beranda perangkat seluler), mengarahkan apa yang bisa diluncurkan pengguna, dan menentukan tampilannya pada saat peluncuran.

Manifes aplikasi *web* menyediakan kemampuan untuk menyimpan *bookmark* situs ke layar beranda perangkat. Ketika sebuah situs diluncurkan dengan cara ini:

- Situs akan memiliki ikon dan nama yang unik sehingga pengguna bisa membedakannya dari situs yang lain.
- Situs akan menampilkan sesuatu kepada pengguna selagi sumber daya diunduh atau dipulihkan dari *cache*.
- Situs akan menyediakan karakteristik tampilan *default* ke *browser* untuk menghindari transisi yang terlalu mendadak bila sumber daya situs tersedia.

Sumber: <https://developers.google.com/web/fundamentals/web-app-manifest/?hl=id>

Diakses pada 26 Oktober 2018

5.3.5. Node.js

Node.js adalah perangkat lunak yang didesain untuk mengembangkan aplikasi berbasis *web* dan ditulis dalam sintaks bahasa pemrograman JavaScript. Bila selama ini kita mengenal JavaScript sebagai bahasa pemrograman yang berjalan di sisi *client / browser* saja, maka Node.js ada untuk melengkapi peran JavaScript sehingga bisa juga berlaku sebagai bahasa pemrograman yang berjalan di sisi *server*, seperti halnya PHP, Ruby, Perl, dan sebagainya. Node.js dapat berjalan di sistem operasi Windows, Mac OS X dan Linux tanpa perlu ada perubahan kode program. Node.js memiliki pustaka server HTTP sendiri sehingga memungkinkan untuk menjalankan *server web* tanpa menggunakan program *server web* seperti *Apache* atau *Nginx*.

Untuk mengeksekusi Javascript sebagai bahasa *server* diperlukan *engine* yang cepat dan mempunyai performansi yang bagus. *Engine Javascript* dari Google bernama V8-lah yang dipakai oleh Node.js yang juga merupakan *engine* yang dipakai oleh *browser* Google Chrome.

Berbeda dengan bahasa pemrograman sisi *server* pada umumnya yang bersifat *blocking*, Node.js bersifat *non-blocking*, sebagaimana halnya JavaScript bekerja. Node.js berjalan dengan basis event (*event-driven*). Maksud dari *Blocking* secara sederhana adalah, bahwa suatu kode program akan dijalankan hingga selesai, baru kemudian beralih ke kode program selanjutnya.

Sumber: <https://www.codepolitan.com/mengenal-nodejs-5880234fe9ae3>

Diakses pada 26 Oktober 2018

5.3.6. Node Package Management (npm)

NPM adalah sebuah *manager* paket untuk bahasa pemrograman JavaScript yang menjadi *manager default* untuk lingkungan *runtime* Node.js. Npm terdiri dari *command line client* yang disebut **npm**, dan sebuah *database online* yang berisi paket publik dan paket pribadi berbayar, yang disebut *npm registry*. *Registry* dapat diakses melalui klien, dan ketersediaan paket dapat dicari melalui *website* npm. *Manager* paket dan *registry* dikelola oleh npm, Inc.

NPM dapat mengelola ketergantungan paket dari suatu proyek, serta pemasangan program JavaScript secara global. Ketika digunakan sebagai *manager* paket untuk proyek lokal, npm dapat menginstal semua paket ketergantungan dengan satu perintah melalui *file* package.json. Pada *file* package.json, setiap paket dapat ditentukan versi yang akan dipasang dengan menggunakan skema *semantic versioning*. Hal ini memungkinkan pengembang untuk melakukan pemberharuan paket otomatis dan juga menghindari benturan perubahan yang tidak diinginkan. NPM juga menyediakan alat *version-bumping* untuk memilih paket dengan versi tertentu. Selain package.json, npm juga menyediakan *file* package-lock.json yang bertugas mengunci versi paket yang terpasang pada proyek.

Sumber: [https://id.wikipedia.org/wiki/Npm_\(perangkat_lunak\)](https://id.wikipedia.org/wiki/Npm_(perangkat_lunak))
Diakses pada 26 Oktober 2018

5.3.7. Vue.js

Vue js merupakan salah satu *framework* yang banyak digunakan. Jika anda telah terbiasa dengan menggunakan angular js, maka untuk melakukan migrasi ke vue js tidak lah terlalu susah karena mereka mempunyai kemiripan di bagian sintak, hanya saja angular js memiliki *development stack* yang lebih rumit dibandingkan dengan vue js. Bagi anda yang penasaran dengan angular js, silahkan lihat di artikel belajar angular js. Vue js juga menggunakan fitur *data binding* sama seperti angular js. Jika di angular js kita menggunakan fungsi *controller*, maka di vue js kita akan menggunakan pendekatan dom yaitu kita akan menggunakan *id*, *name* atau *class* dari suatu *component* html.

Sumber: <https://rizkimufrizal.github.io/belajar-vue-js/>
Diakses pada 26 Oktober 2018

5.3.8. Firebase

Firebase Hosting adalah layanan *hosting* konten *web* yang berkelas produksi untuk *developer*. Hanya dengan satu perintah, Anda dapat menerapkan aplikasi *web* serta menyajikan konten dinamis dan statis ke jaringan penayangan konten (CDN) global dengan cepat dan mudah.

Firebase Hosting dibuat untuk *developer web modern*. Dengan hadirnya *framework* JavaScript *front-end* seperti Angular dan fitur generator statis seperti

Jekyll, situs dan aplikasi kini jadi lebih tangguh daripada sebelumnya. Jika Anda menerapkan halaman landing aplikasi yang sederhana atau Progressive Web App (PWA) yang kompleks, Hosting menyediakan infrastruktur, fitur, dan rangkaian sarana yang disesuaikan untuk menerapkan serta mengelola situs dan aplikasi.

Hosting menambahkan *subdomain* ke *project* Anda di *domain* firebaseapp.com. Dengan *Firebase CLI*, Anda dapat menerapkan *file* dari direktori lokal di komputer Anda ke *server Hosting*. Selain menyajikan konten statis, Anda dapat menggunakan *Cloud Functions for Firebase* untuk menyajikan konten dinamis di situs Anda. Semua konten disajikan melalui koneksi SSL dari *server edge* terdekat di CDN global kami.

Selain *hosting* konten, *Firebase Hosting* menawarkan opsi konfigurasi *hosting* yang ringan kepada Anda untuk membuat *Progressive Web App* yang canggih. Anda dapat menulis ulang URL untuk perutean sisi klien atau menyiapkan header kustom dengan mudah.

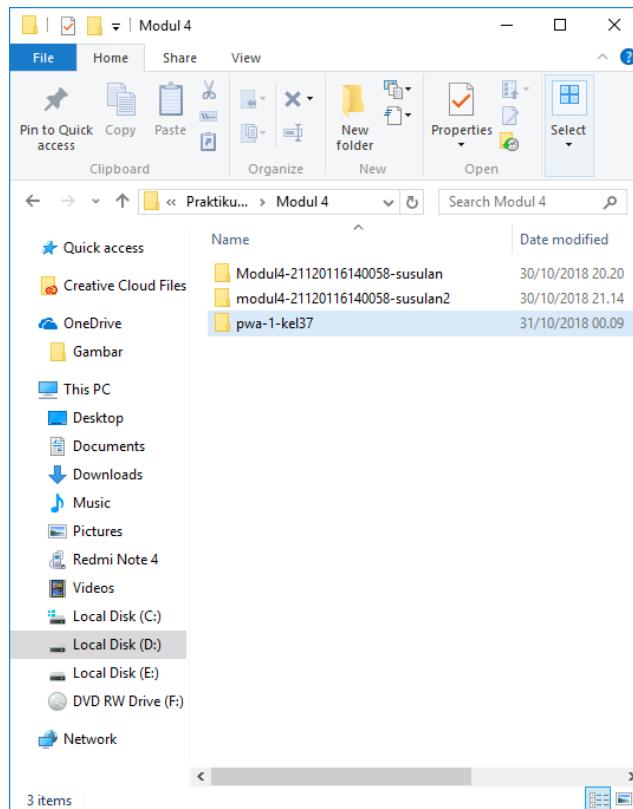
Sumber: <https://firebase.google.com/docs/hosting/?hl=id>

Diakses pada 26 Oktober 2018

5.4. Langkah Percobaan

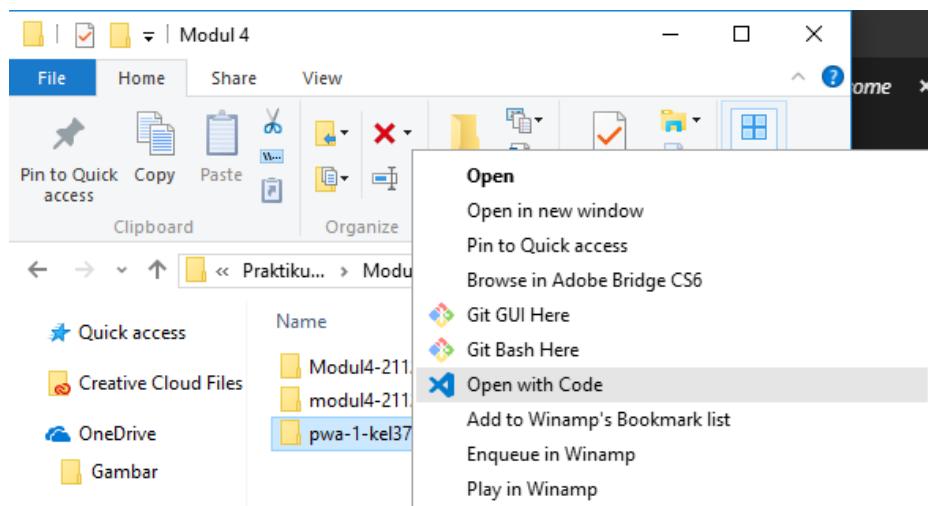
5.4.1. Pembuatan Halaman Website

- Buat *folder* beri nama “pwa-1-kel37”, untuk penempatan *foldernya* bebas.



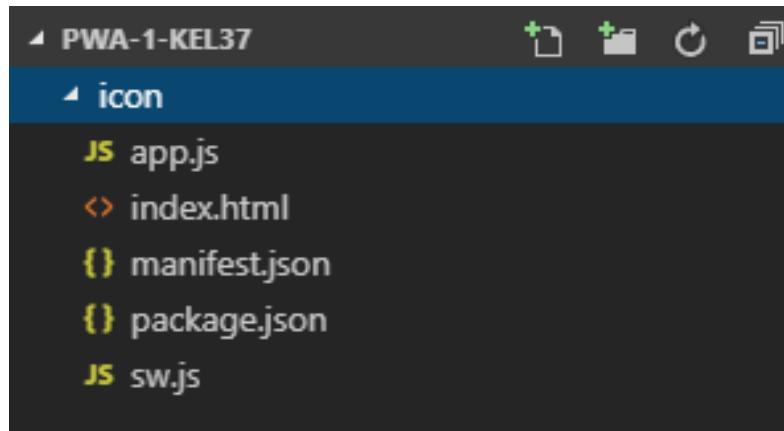
Gambar 5.8 Folder pwa-1-kel37

- Setelah itu klik kanan *folder* tersebut dan pilih *Open with Code*, untuk membuka VS Code beserta *foldernya*.



Gambar 5.9 Membuka folder dengan VS Code

3. Setelah itu buatlah 5 *file* dan 1 *folder* seperti gambar di bawah.



Gambar 5.10 *Folder* dan *File* yang dibuat pada Pwa-1-kel37

4. Pada *file package.json* diberi kode seperti ini.

```
{
  "name": "pwa-1-kel37",
  "version": "1.0.0",
  "description": "Praktikum MDP PWA Jilid 1",
  "main": "index.js",
  "scripts": {
    "start": "http-server -c-1 -p 8888"
  },
  "keywords": [
    "pwa"
  ],
  "author": "Asisten MDP PWA 1",
  "license": "UNLICENSED",
  "devDependencies": {
    "http-server": "^0.10.0"
  }
}
```

5. Setelah itu, isikan *file index.html* dengan kode berikut:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <title>Modul PWA</title>
  <meta name="viewport" content="width=device-width,
initial-scale=1">
  <link rel="stylesheet"
 href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">
  <link rel="shortcut icon" href="icon/192.ico"
 type="image/x-icon" />
  <link rel="icon"
 href="https://upload.wikimedia.org/wikipedia/commons/f/f2/Lam
bang Kota Semarang.png" sizes="32x32" />
```

```

<!-- buat manifest json -->
</head>
<body>

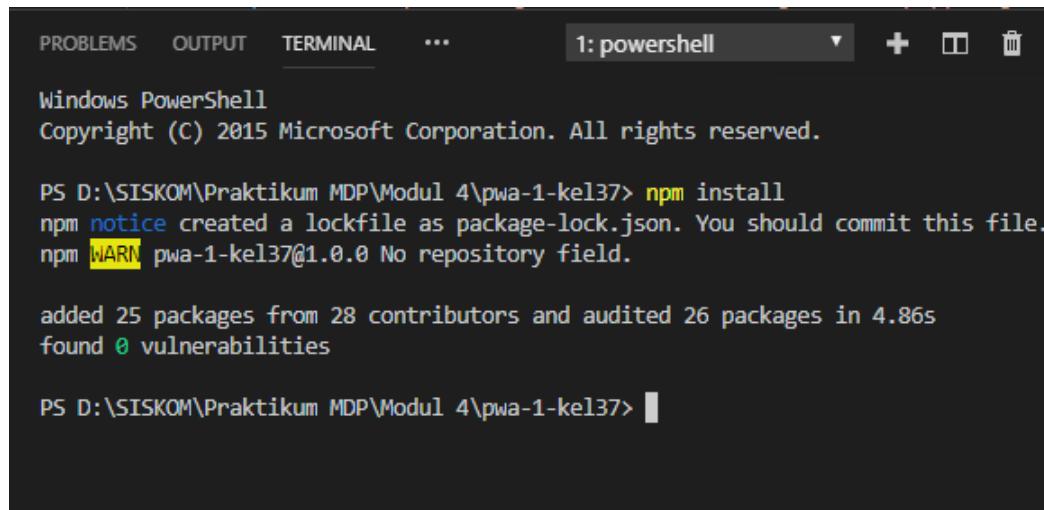
<!-- untuk program utama -->

</body>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
    <script
src="https://unpkg.com/axios/dist/axios.min.js"></script>
    <script src="app.js"></script>
    <!-- untuk service worker -->

</html>

```

- Setelah itu kita buka *terminal* pada *VS Code*, yaitu dengan menekan **Ctrl+`**, setelah itu kita ketikan perintah **npm install**. Pastikan sudah menginstall package npm di *visual studio code*.



The screenshot shows the VS Code interface with the terminal tab selected. The terminal window title is "1: powershell". The output shows the results of an npm install command:

```

PROBLEMS OUTPUT TERMINAL ...
1: powershell + -x -d -l

Windows PowerShell
Copyright (C) 2015 Microsoft Corporation. All rights reserved.

PS D:\SISKOM\Praktikum MDP\Modul 4\pwa-1-kel37> npm install
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN pwa-1-kel37@1.0.0 No repository field.

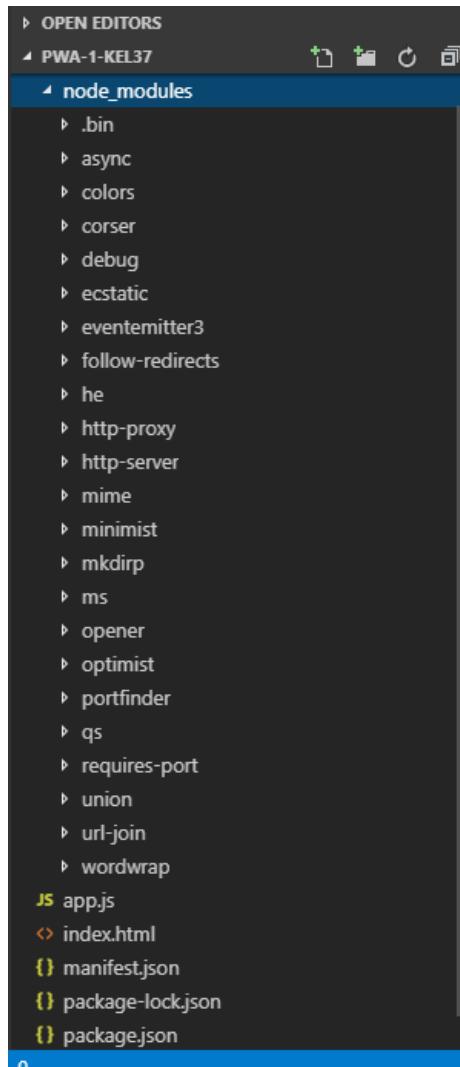
added 25 packages from 28 contributors and audited 26 packages in 4.86s
found 0 vulnerabilities

PS D:\SISKOM\Praktikum MDP\Modul 4\pwa-1-kel37>

```

Gambar 5.11 *Install NPM menggunakan terminal VSCode*

7. Setelah itu akan muncul *folder node_module* dan **package-lock.json**



Gambar 5.12 node_modules dan package-lock.json

8. Setelah itu buka *file app.js* dan isikan dengan kode berikut

```
new Vue({
  el:'#app',
  data:{
    kodepos : [],search : '',
    awal:0,akhir:10,halaman:1,tampil:true //deklarasi variabel
  }, //1 halaman menampilkan 10 data
  created(){ //method mengambil data
    this.getData(); //untuk mengambil data
    console.log(this.awal) //dari variabel awal
  },
  methods:{
    next(){ //ketika menekan tombol next
      this.awal=this.awal+10 //menampilkan 10 data selanjutnya
      this.akhir=this.akhir+10
    }
  }
})
```

```

        this.halaman=this.halaman+1 //halaman +1
    },
    prev(){ //ketika menekan tombol prev
        this.awal=this.awal-10 //menampilkan 10 data
sebelumnya
        this.akhir=this.akhir-10
        this.halaman=this.halaman-1 //halaman -1
    },
    getData(){ //method untuk mengambil data
        var kd=this;
        axios.get('https://kodepos-
2d475.firebaseio.com/kota_kab/k110.json?print=pretty').then(function(response){ //link untuk kode pos semarang
            kd.kodepos=response.data;
        })
    },
    computed:{
        cari(){ //method untuk mencari
            return this.kodepos.filter(pos =>{
                return
pos.kecamatan.toLowerCase().match(this.search.toLowerCase());
//mengembalikan nilai kode pos
            });
        }
    }
);
}

```

9. Setelah kita mengisi app.js, selanjutnya kita melengkapi kode pada *file index.html* dengan kode berikut diletakkan dibawah tulisan **<!-- untuk program utama -->**

```

<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Modul PWA</title>
    <meta name="viewport" content="width=device-width,
initial-scale=1">
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">
            <link rel="shortcut icon" href="icon/192.ico" type="image/x-icon" />
            <link rel="manifest" href="manifest.json">
            <link rel="icon" href="https://upload.wikimedia.org/wikipedia/commons/f/f2/Lambang_Kota_Semarang.png" sizes="32x32" />
<!-- buat manifest json -->
</head>
<body>

<div class="container-fluid">

```

```

<div id="app">
    <center>
        <div class="card bg-warning mb-3">
            <div class="card-body">
                
                <br><br>
                <h5 class="card-title">Daftar Kode POS Semarang</h5>
                <p class="card-text">Berikut adalah semua kode pos yang ada di Kab. Semarang dan Kota Semarang</p>
                <p>Kurang lebih datanya ada {{kodepos.length}}</p>
                <a href="#" v-on:click="tampil = !tampil" v-show="tampil" class="btn btn-primary">Tampilan</a>
            </div>
        </div>
        <div v-show="!tampil">
            <div class="row">
                <div class="col-sm-4 col-md-4 col-lg-4 col-4">
                    <button class="btn btn-primary btn-sm" v-show="awal>0" v-on:click="prev()">Balek Wae</button>
                </div>
                <div class="col-sm-4 col-md col-lg-4 col-4">
                    <span>Halaman : {{halaman}}</span>
                </div>
                <div class="col-sm-4 col-md-4 col-lg-4 col-4">
                    <button class="btn btn-primary btn-sm" v-show="akhir <= kodepos.length" v-on:click="next()">Lanjoot</button>
                </div>
            </div>
            <div class="row" style="margin:20px">
                <div class="col-sm-3"></div>
                <div class="col-sm-6">
                    <input type="text" v-model="search" class="form-control" placeholder="Pencarian berdasarkan Kecamatan">
                </div>
            </div>
        </div>
    </center>
    <div v-show="!tampil" class="table-responsive container">
        <table id="example" class="table table-hover table-bordered">
            <tr>
                <th>Kecamatan</th>
                <th>Kelurahan</th>
                <th>Kode POS</th>
            </tr>
            <tr v-for="pos in cari.slice(awal, akhir)">
                <td style="width:40%">{{pos.kecamatan}}</td>
                <td style="width:40%">{{pos.kelurahan}}</td>
                <td style="width:20%">{{pos.kodepos}}</td>
            </tr>
        </table>
    </div>

```

```

        </tr>
    </table>
</div>
</div>

</body>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"></script>
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
<script
src="https://unpkg.com/axios/dist/axios.min.js"></script>
<script src="app.js"></script>

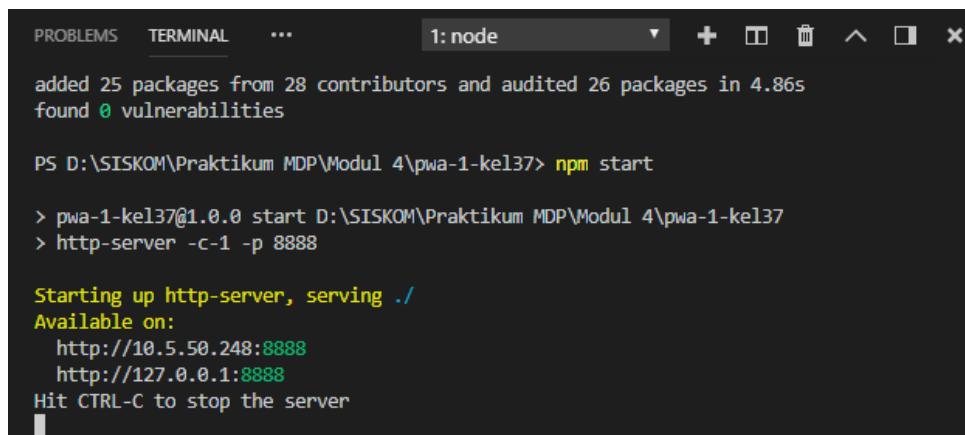
<script src="sw.js"></script>

<!-- untuk service worker -->

</html>

```

10. Buka *terminal* kembali, dan ketikkan perintah ***npm start***



```

PROBLEMS TERMINAL ...
1: node + □ ^ □ ×
added 25 packages from 28 contributors and audited 26 packages in 4.86s
found 0 vulnerabilities

PS D:\SISKOM\Praktikum MDP\Modul 4\pwa-1-kel37> npm start

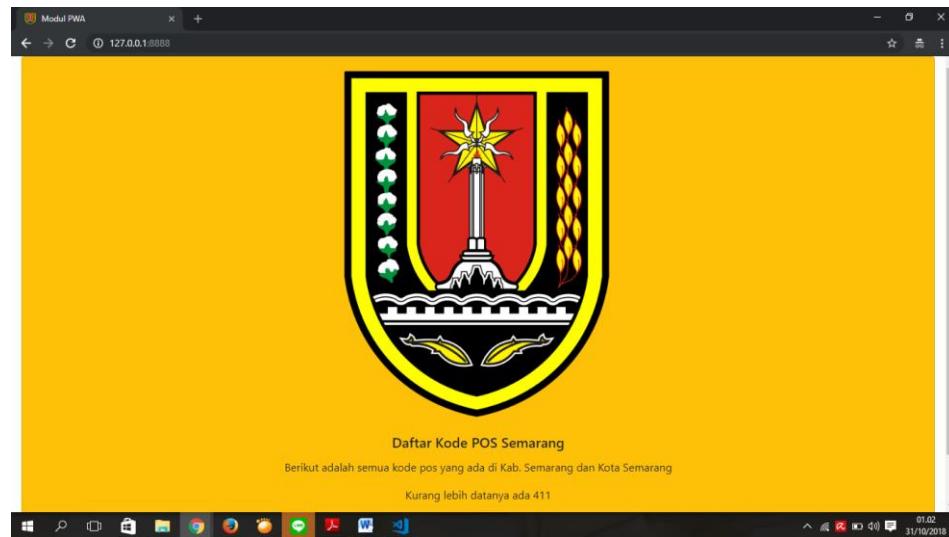
> pwa-1-kel37@1.0.0 start D:\SISKOM\Praktikum MDP\Modul 4\pwa-1-kel37
> http-server -c-1 -p 8888

Starting up http-server, serving .
Available on:
  http://10.5.50.248:8888
  http://127.0.0.1:8888
Hit CTRL-C to stop the server

```

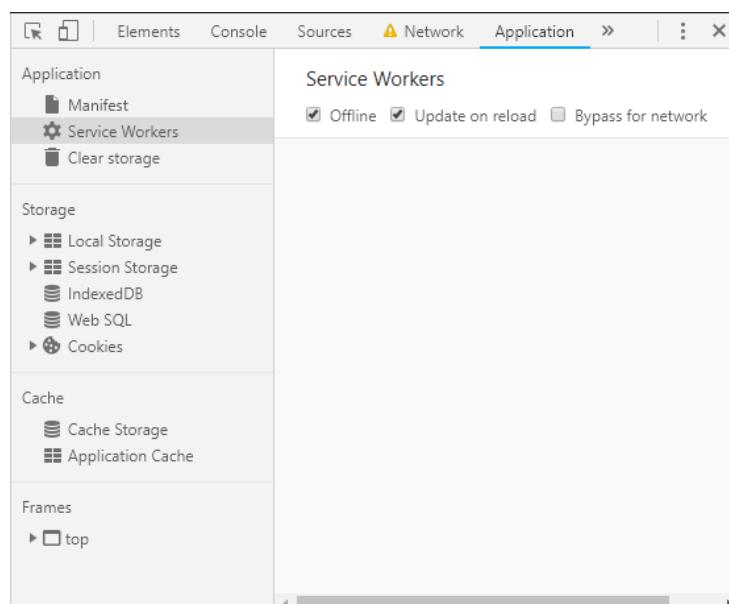
Gambar 5.13 Mengetikkan perintah *npm start*

Akan ada 2 *link*, kita pilih yang 127.0.0.1:8888, setelah itu kita masukan di *web browser* google chrome. Hasilnya seperti ini



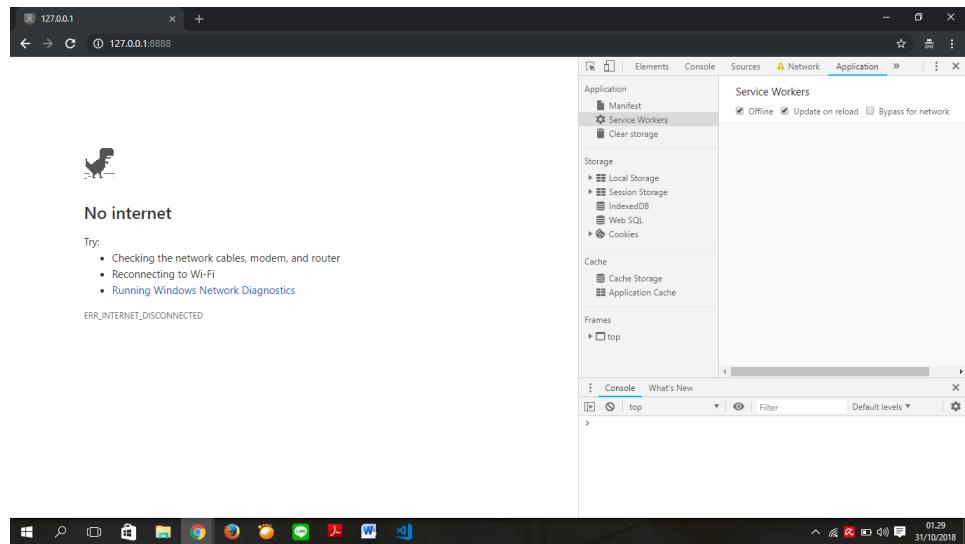
Gambar 5.14 Tampilan yang muncul di *browser*

11. Setelah berhasil menampilkan halaman seperti gambar diatas, selanjutnya kita buka *inspect element*, dengan mengetikan **ctrl+shift+I** atau klik kanan pilih *inspect element*
12. Setelah sampai di *inspect element*, kita pilih tab **application** dan pilih **service worker**. Kita beri centang pada *offline*.



Gambar 5.15 Halaman *inspect elements*

Setelah kita beri centang pada *offline*, reload halaman webnya maka akan muncul seperti ini.



Gambar 5.16 Halaman web saat *offline*

13. Setelah itu hilangkan centang pada *offline* dan *reload* kembali.
14. Buka file **sw.js** dan isikan kode berikut.

```

self.addEventListener('install', function (event) {
    console.log('SW Installed');
    event.waitUntil(
        caches.open('static')
            .then(function (cache) {
                cache.addAll([
                    '/',
                    '/index.html',
                    '/app.js',
                    '/manifest.json',

                    'https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css',
                    'https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js',
                    'https://cdn.jsdelivr.net/npm/popper.js@1.14.3/umd/popper.min.js',
                    'https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js',
                    'https://cdn.jsdelivr.net/npm/vue/dist/vue.js',
                    'https://unpkg.com/axios/dist/axios.min.js',
                    'https://kodepos-2d475.firebaseio.com/kota_kab/k110.json?print=pretty',
                    'https://upload.wikimedia.org/wikipedia/commons/f/f2/Lambang_Kota_Semarang.png'
                ]);
            })
    );
})

```

```

        );
    });

self.addEventListener('activate', function () {
    console.log('SW activated');
});

self.addEventListener('fetch', function (e) {
    console.log('[ServiceWorker] Fetch', e.request.url);
    e.respondWith(
        caches.match(e.request).then(function (response) {
            return response || fetch(e.request);
        })
    );
});
}
);

```

Dan buka **manifest.json** dan masukkan kode berikut

```
{
    "name": "Pos Semarang",
    "short_name": "Kode Pos Semarang",
    "theme_color": "#ffc107",
    "background_color": "#ffc107",
    "display": "standalone",
    "orientation": "portrait",
    "scope": ".",
    "start_url": ".",
    "icons" : [
        {
            "src": "/icon/32.ico",
            "sizes": "32x32",
            "type": "image/ico"
        },
        {
            "src": "/icon/96.ico",
            "sizes": "96x96",
            "type": "image/ico"
        },
        {
            "src": "/icon/128.ico",
            "sizes": "128x128",
            "type": "image/ico"
        },
        {
            "src": "/icon/192.ico",
            "sizes": "192x192",
            "type": "image/ico"
        },
        {
            "src": "/icon/256.ico",
            "sizes": "256x256",
            "type": "image/ico"
        }
    ]
}
```

Setelah menambahkan kode tersebut, jangan lupa kita tambahkan kode berikut pada **index.html**

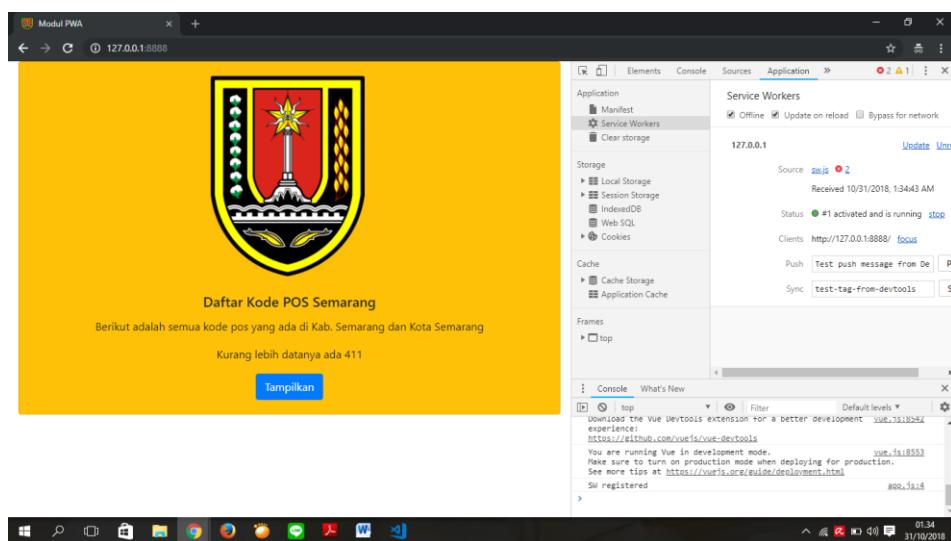
```
<link rel="manifest" href="manifest.json">
<script src="sw.js"></script>
```

Diletakan sebagaimana terlihat pada **index.html**

15. Tambahkan kode berikut pada **app.js**

```
if ('serviceWorker' in navigator) {
    navigator.serviceWorker.register('/sw.js')
        .then(function() {
            console.log('SW registered');
        });
}
```

16. Setelah itu di-reload terlebih dahulu dan buka inspect elemen, dan buka tab *application* dan lakukan seperti langkah 12 dan lihat perbedaan.

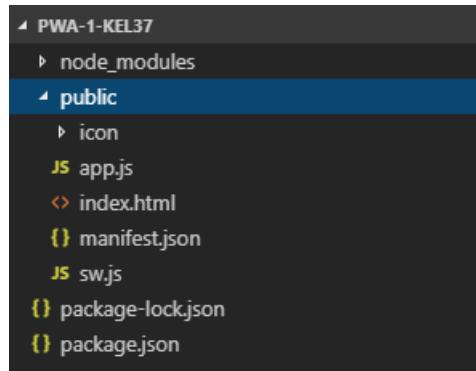


Gambar 5.17 Halaman *web* setelah ditambahkan *service worker*

17. Setelah kita menambahkan *service worker* dan *manifest*, maka halaman *web* tetap bejalan.

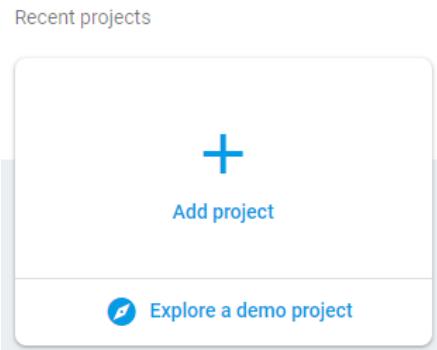
5.4.2. Deploy Halaman Website ke Firebase

1. Setelah berhasil di langkah sebelumnya, pada praktikum ini kita *deploy* dengan menggunakan *firebase*.
2. Ubah struktur *folder* proyek kalian menjadi seperti gambar dibawah



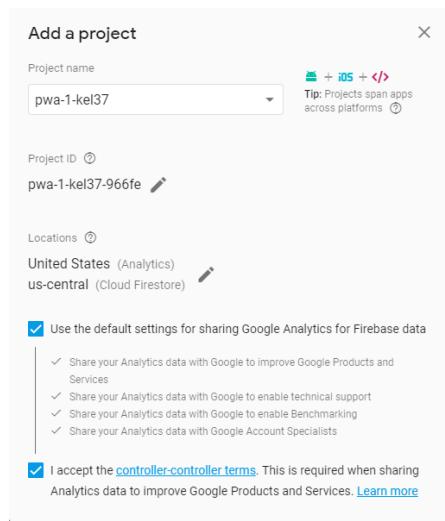
Gambar 5.18 Struktur *folder* yang baru

3. Buka link ini <https://firebase.google.com/> dan jangan lupa *login* terlebih dahulu.
4. Tambahkan proyek baru



Gambar 5.19 Menambahkan proyek baru

5. Nama proyeknya **pwa-1-kel37**



Gambar 5.20 Membuat proyek baru bernama pwa-1-kel37

6. Buka terminal dan masukkan perintah berikut untuk menginstall *firebase*

```
npm install -g firebase-tools
```

```
PS D:\SISKOM\Praktikum MDP\Modul 4\pwa-1-kel37> npm install -g firebase-tools
C:\Users\ASUS\AppData\Roaming\npm\firebase -> C:\Users\ASUS\AppData\Roaming\npm\node
_node_modules\firebase-tools\lib\bin\firebase.js
+ firebase-tools@6.0.0
updated 1 package in 39.86s
PS D:\SISKOM\Praktikum MDP\Modul 4\pwa-1-kel37> █
```

Gambar 5.21 Menginstall firebase

7. Setelah itu *login* ke *firebase* dengan mengetikan perintah berikut

firebase login

```
PROBLEMS TERMINAL ...
1: powershell + - x

PS D:\SISKOM\Praktikum MDP\Modul 4\pwa-1-kel37> firebase login
? Allow Firebase to collect anonymous CLI usage and error reporting information? Ye
s

Visit this URL on any device to log in:
https://accounts.google.com/o/oauth2/auth?client_id=563584335869-fgrhgmd47bqnekij5i8
b5pr03ho849e6.apps.googleusercontent.com&scope=email%20openid%20https%3A%2F%2Fwww.go
ogleapis.com%2Fauth%2Fcloudplatformprojects.readonly%20https%3A%2F%2Fwww.googleapis.c
om%2Fauth%2Ffirebase%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fcloud-platform&res
ponse_type=code&state=1029124400&redirect_uri=http%3A%2F%2Flocalhost%3A9005

Waiting for authentication...

+ Success! Logged in as ssaskia.al@gmail.com
```

Gambar 5.22 Berhasil *login* ke *firebase* dengan akun Gmail

8. Langkah selanjutnya kita mengetikan perintah **firebase init** , untuk menginisialisasi proyek yang akan *dideploy* ke *firebase*

Gambar 5.23 Menginisialisasi proyek yang akan dideploy ke *firebase*

9. Jika tampilan dari *terminal* seperti gambar diatas, pada saat pertanyaan “**Are you ready to proceed ?**” kita ketikan huruf “Y“.

```
You're about to initialize a Firebase project in this directory:  
D:\SISKOM\Praktikum MDP\Modul 4\pwa-1-kel37  
? Are you ready to proceed? Yes
```

Gambar 5.24 Menyetujui proses inisialisasi

10. Setelah kita menekan huruf “Y”, akan muncul tampilan untuk memilih fitur yang akan digunakan, kita pilih “**Hosting: Configure and Deploy Firebase Hosting sites**” dengan cara menekan “Space” setelah itu tekan “Enter”

```
? Which Firebase CLI features do you want to setup for this folder? Press Space to
select features, then Enter to confirm your choices.
( ) Database: Deploy Firebase Realtime Database Rules
( ) Firestore: Deploy rules and create indexes for Firestore
( ) Functions: Configure and deploy Cloud Functions
>(*) Hosting: Configure and deploy Firebase Hosting sites
( ) Storage: Deploy Cloud Storage security rules
```

Gambar 5.25 Menentukan fitur yang akan digunakan

11. Setelah itu akan muncul semua projek yang kita buat pada firebase, kita pilih projek “**pwa-1-kel37**” terus tekan enter.

```
== Project Setup

First, let's associate this project directory with a Firebase project.
You can create multiple project aliases by running firebase use --add,
but for now we'll just set up a default project.

? Select a default Firebase project for this directory:
[don't setup a default project]
modul4-21120116140058-susulan (Modul4-21120116140058-susulan)
modul4-21120116140058-susulan2 (modul4-21120116140058-susulan2)
> pwa-1-kel37-966fe (pwa-1-kel37)
[create a new project]
```

Gambar 5.26 Menentukan proyek *firebase* yang akan digunakan

12. Akan ada 3 pertanyaan yaitu:

- *What do you want to use as your public directory ?* kita tekan **enter** saja
- *Configure as a single-page app (rewrite all urls to /index.html)?* kita tekan **N**
- *File public/index.html already exists. Overwrite ?* kita tekan **N**

Setelah itu kita tunggu saja.

```
will contain Hosting assets to be uploaded with firebase deploy. If you
have a build process for your assets, use your build's output directory.

? What do you want to use as your public directory? public
? Configure as a single-page app (rewrite all urls to /index.html)? No
+ Wrote public/404.html
? File public/index.html already exists. Overwrite? No
i Skipping write of public/index.html

i Writing configuration info to firebase.json...
i Writing project information to .firebaserc...
i Writing gitignore file to .gitignore...

+ Firebase initialization complete!
PS D:\SISKOM\Praktikum MDP\Modul 4\pwa-1-kel37>
```

Gambar 5.27 Menjawab pertanyaan yang muncul

13. Setelah selesai, kita tuliskan perintah “firebase deploy” pada terminal dan tunggu sampai tampilan terminal seperti pada gambar dibawah.

```
PS D:\SISKOM\Praktikum MDP\Modul 4\pwa-1-kel37> firebase deploy
=== Deploying to 'pwa-1-kel37-966fe'...

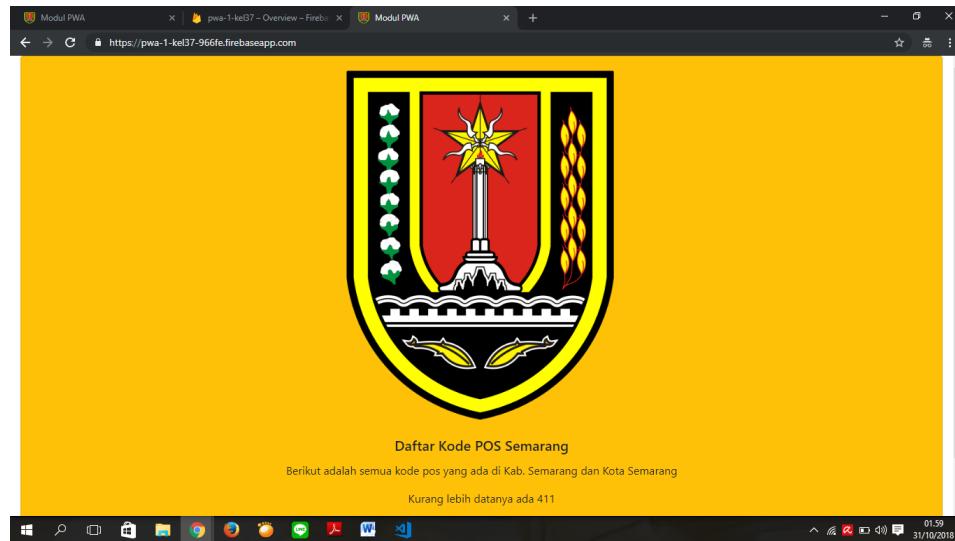
i  deploying hosting
i  hosting[pwa-1-kel37-966fe]: beginning deploy...
i  hosting[pwa-1-kel37-966fe]: found 5 files in public
+  hosting[pwa-1-kel37-966fe]: file upload complete
i  hosting[pwa-1-kel37-966fe]: finalizing version...
+  hosting[pwa-1-kel37-966fe]: version finalized
i  hosting[pwa-1-kel37-966fe]: releasing new version...
+  hosting[pwa-1-kel37-966fe]: release complete

+ Deploy complete!

Project Console: https://console.firebaseio.google.com/project/pwa-1-kel37-966fe/overview
Hosting URL: https://pwa-1-kel37-966fe.firebaseioapp.com
PS D:\SISKOM\Praktikum MDP\Modul 4\pwa-1-kel37>
```

Gambar 5.28 Deploy proyek ke *firebase*

14. Setelah berhasil, akan muncul link, di *Hosting URL*, kita langsung menuju ke halaman tersebut, maka kita sudah berhasil mendeploy halaman *web* yang kita buat.



Gambar 5.29 Tampilan halaman *web* yang telah berhasil dideploy ke *firebase*

5.4.3. Website ke Mobile

Untuk menjadikan halaman *website* ini menjadi aplikasi *mobile* caranya yaitu:

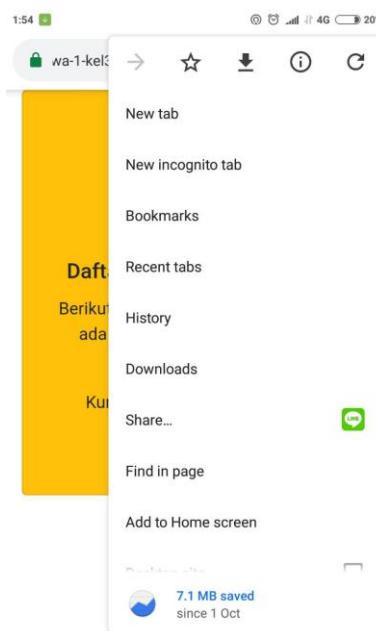
1. Buka *Smartphone* kalian yang sudah diinstal aplikasi *google chrome*

2. Buka alamat *hosting* yang sudah dibuat di *google chrome*



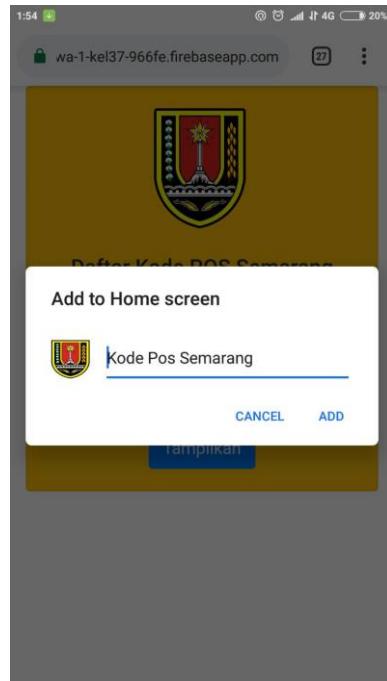
Gambar 5.30 Tampilan halaman *web hosting* pada *smartphone*

3. Setelah itu klik tombol kanan pojok pada *google chrome*, terus pilih “Tambahkan ke Layar Utama”

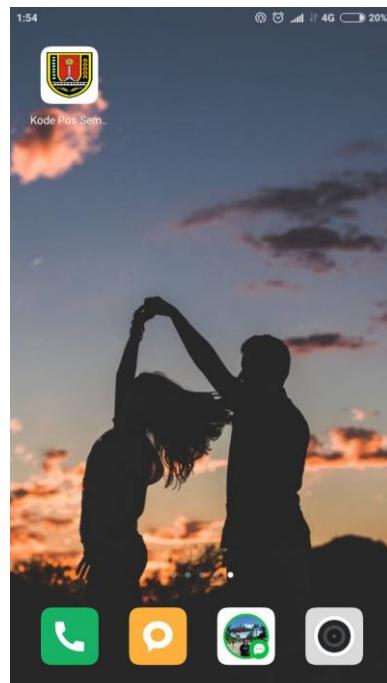


Gambar 5.31 Add Webpage to Home Screen

4. Akan muncul *pop up*, kita pilih tambahkan, maka aplikasi ini akan muncul pada *homescreen* di *smartphone* kalian, silahkan dicoba

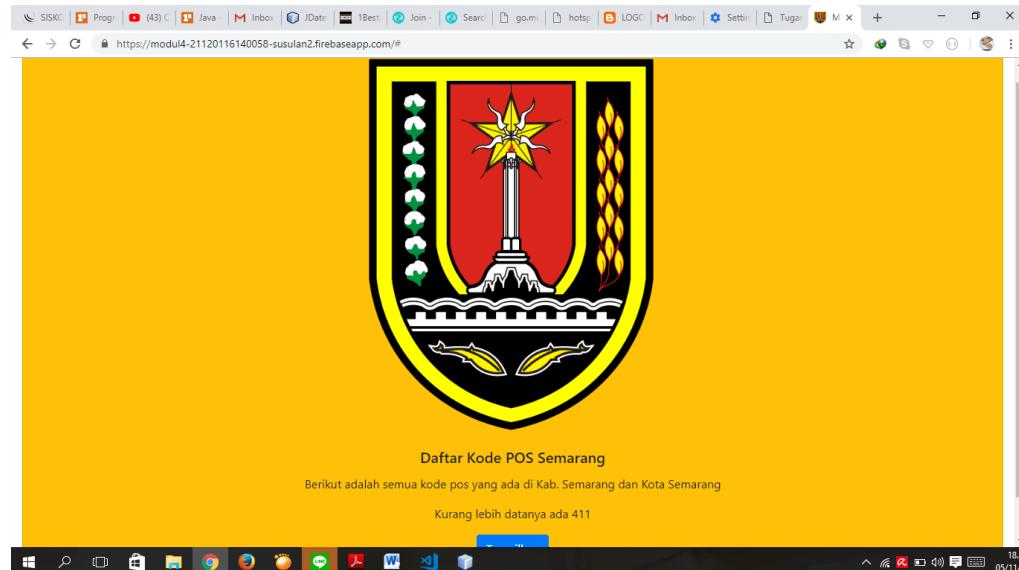


Gambar 5.32 *Pop Up* yang muncul untuk menambahkan ke layar utama

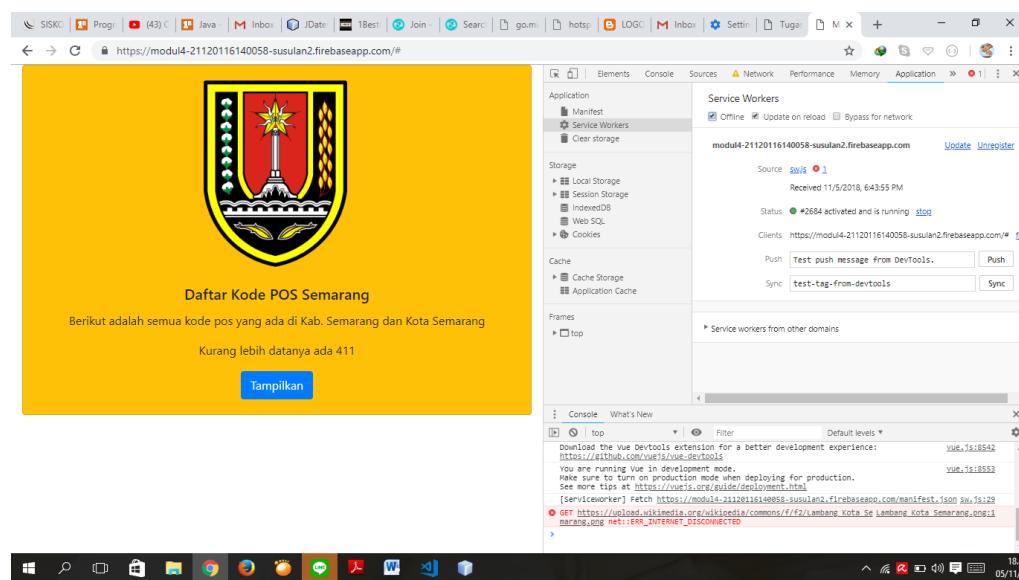


Gambar 5.33 Tampilan *shortcut webpage* pada layar utama

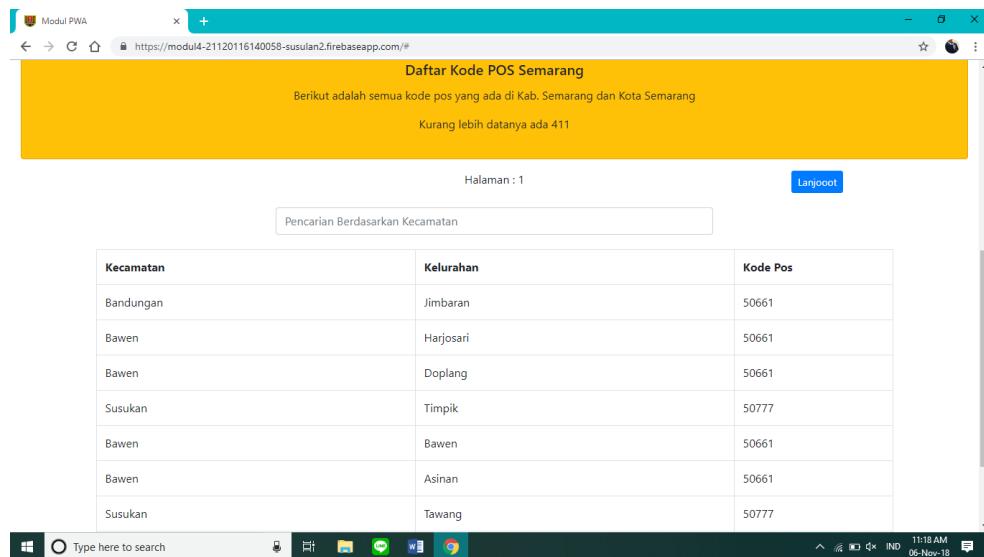
5.5 Analisa Hasil Percobaan



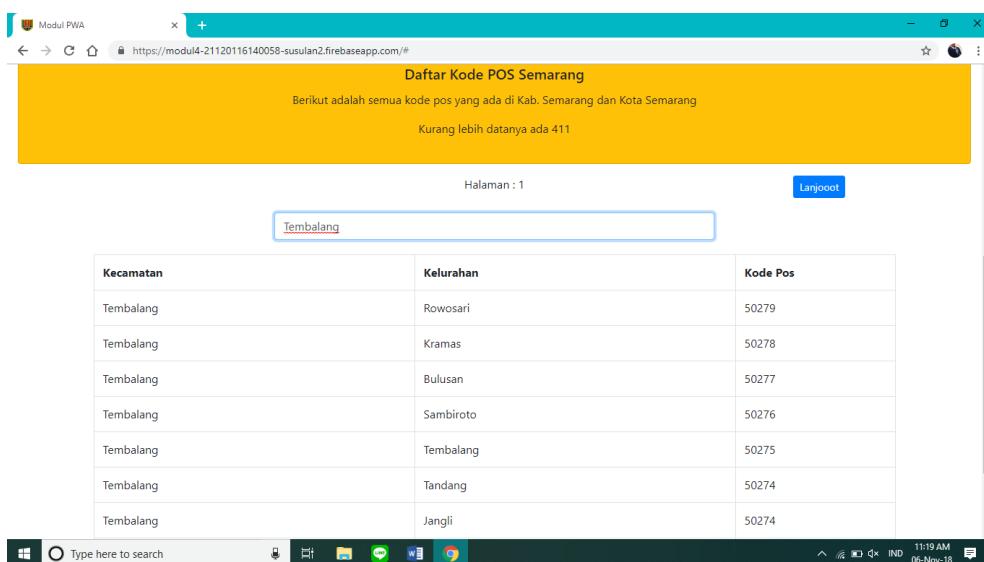
Gambar 5.34 Tampilan *project* pada web



Gambar 5.35 Tampilan *project* pada web saat kondisi Offline



Gambar 5.36 Tampilan *project* setelah tombol Tampilkan ditekan



Gambar 5.37 Tampilan *project* saat melakukan pencarian data



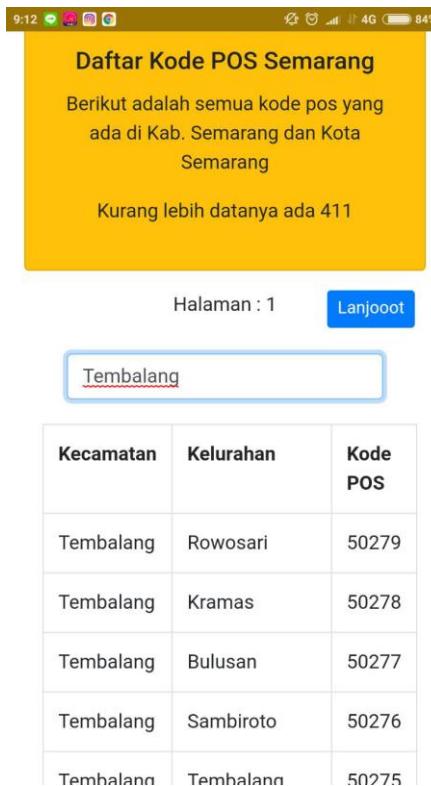
Gambar 5.38 Tampilan *splash screen project* pada *mobile*



Pencarian berdasarkan Kecamat:

Kecamatan	Kelurahan	Kode POS
Bandungan	Jimbaran	50661
Bawen	Hariosari	50661

Gambar 5.39 Tampilan *project* pada *mobile*



Gambar 5.40 Tampilan project pada saat melakukan pencarian pada mobile

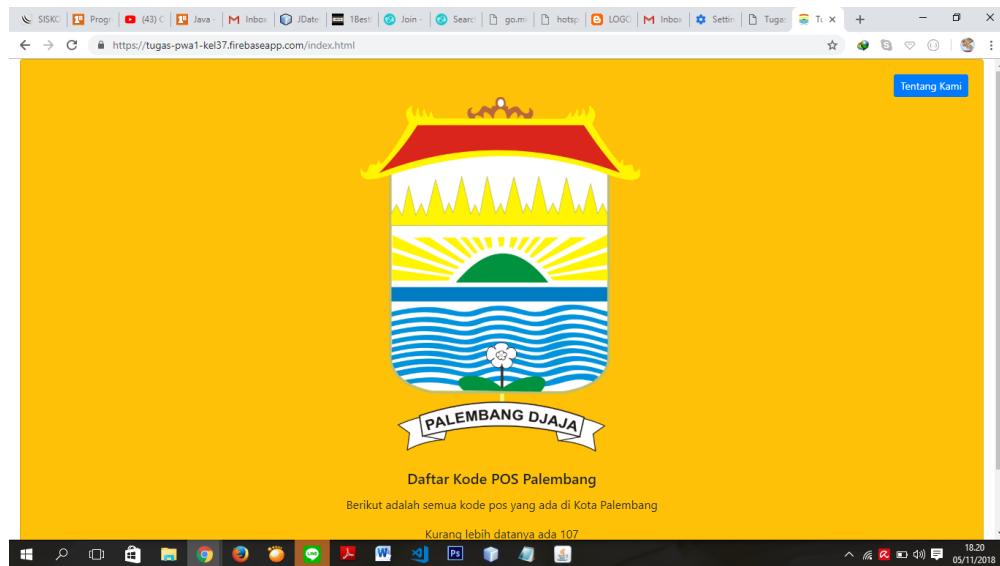
```
{
  "name": "Pos Semarang",
  "short_name": "Kode Pos Semarang",
  "theme_color": "#ffc107",
  "background_color": "#ffc107",
  "display": "standalone",
  "orientation": "portrait",
  "scope": ".",
  "start_url": ".",
  "icons" : [
    {
      "src": "/icon/32.ico",
      "sizes": "32x32",
      "type": "image/ico"
    },
    {
      "src": "/icon/96.ico",
      "sizes": "96x96",
      "type": "image/ico"
    },
    {
      "src": "/icon/128.ico",
      "sizes": "128x128",
      "type": "image/ico"
    },
    {
      "src": "/icon/192.ico",
      "sizes": "192x192",
      "type": "image/ico"
    }
  ]
}
```

```
        "type": "image/ico"
    },
{
    "src": "/icon/256.ico",
    "sizes": "256x256",
    "type": "image/ico"
}
]
```

Manifest.json digunakan untuk menjadikan link yang telah kita buat, menjadi aplikasi pada handphone. Jika kita membuat *progressive web apps*, kita membutuhkan manifest.json, manifest.json merupakan file json dengan semua kebutuhan informasi yang dibutuhkan untuk menyimpan web app ke home screen handphone pengguna. Pada file manifest.json berisi mengenai *user experience* seperti splash screen, theme color, icon, dan lainnya. Pada splash screen diberi nama Pos Semarang, judulnya Kode Pos Semarang, theme color dan background color adalah #ffc107 (warna kuning), orientasi yang digunakan adalah portrait, lalu memanggil icon-icon yang akan digunakan, disini menggunakan 5 icon.

<https://modul4-21120116140058-susulan2.firebaseioapp.com/#>

5.6 Tugas

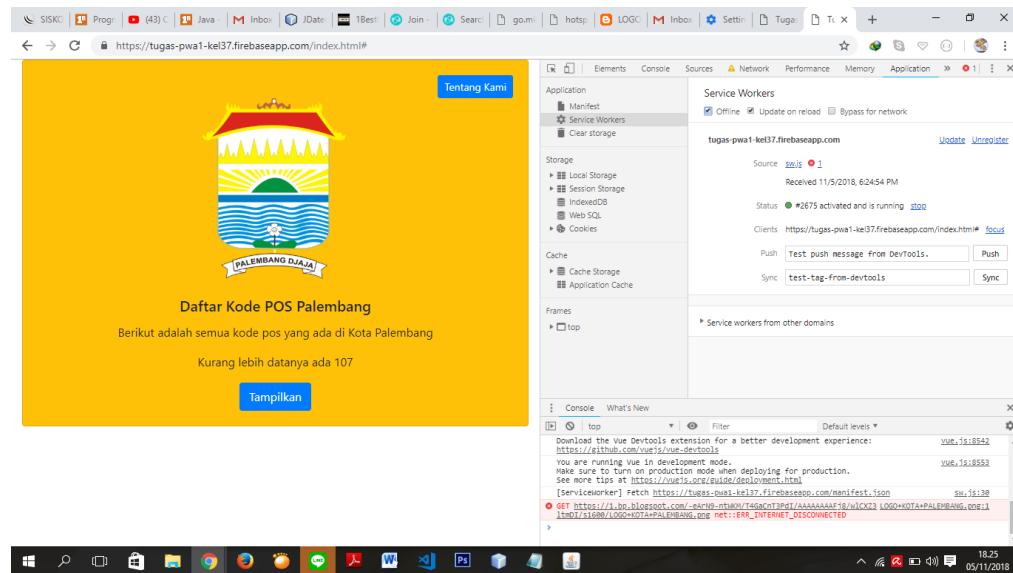


Gambar 5.41 Tampilan *project* tugas pada web

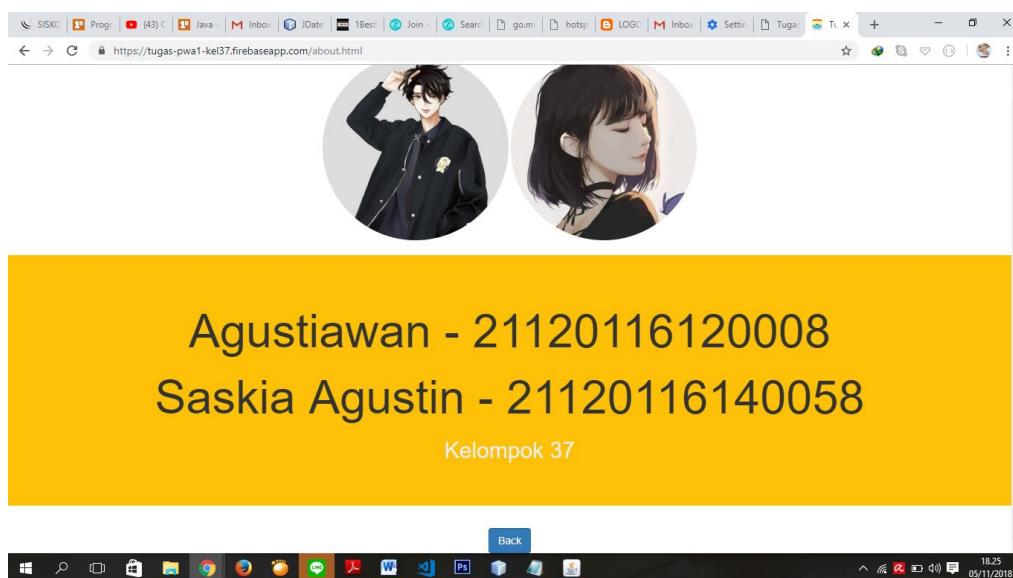
The screenshot shows a web browser window with the URL <https://tugas-pwa1-kel37.firebaseio.com/index.html#>. The page displays a table titled "Pencarian Berdasarkan Kecamatan". The table has three columns: "Kecamatan", "Kelurahan", and "Kode Pos". The data is as follows:

Kecamatan	Kelurahan	Kode Pos
Paju	Talang Putri	30268
Paju	Paju Iir	30268
Paju	Talang Bubuk	30268
Paju	Bagus Kuning	30268
Paju	Komperta	30268
Seberang Ulu II	11 Ulu	30267
Paju	Paju Darat	30267
Paju	Paju Ulu	30266
Seberang Ulu II	16 Ulu	30265
Seberang Ulu II	13 Ulu	30263
Seberang Ulu II	14 Ulu	30264
Seberang Ulu II	Tangga Takat	30264
Seberang Ulu II	Sentosa	30261
Seberang Ulu II	12 Ulu	30262
Kertapati	Ogan Baru	30258

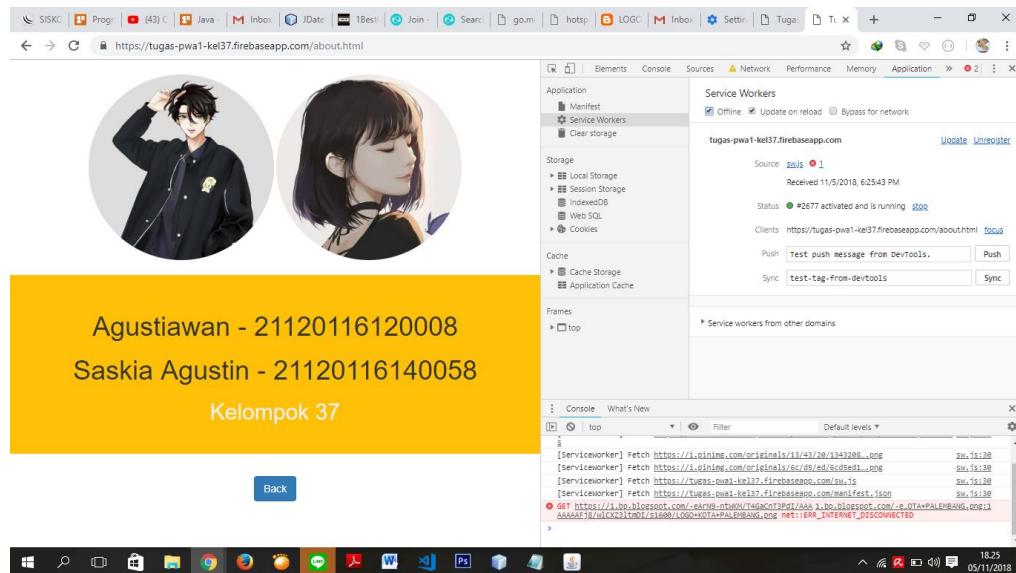
Gambar 5.42 Tampilan *project* tugas pada web daftar kode pos berjumlah 15



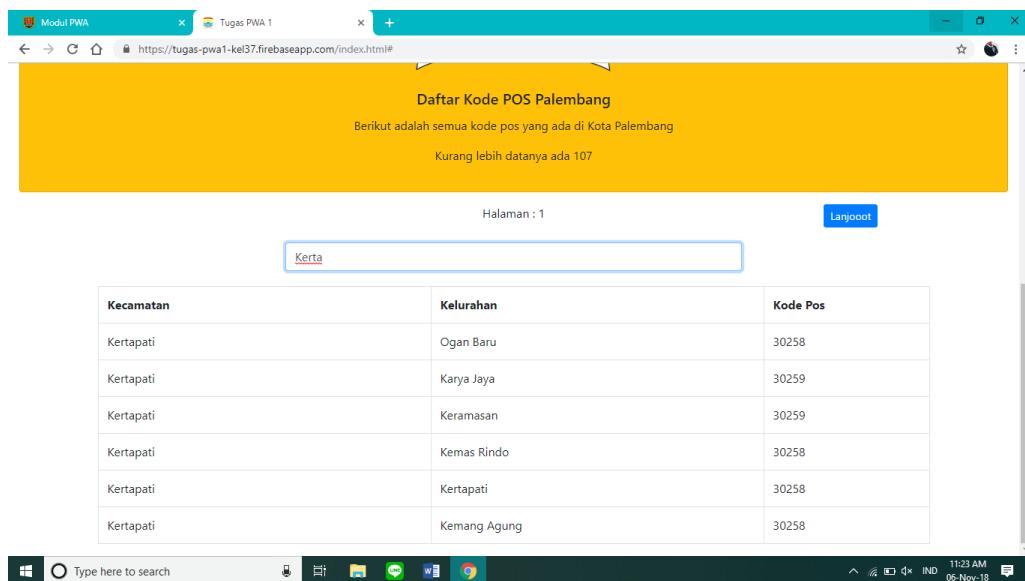
Gambar 5.43 Tampilan *project tugas* pada web saat *Offline*



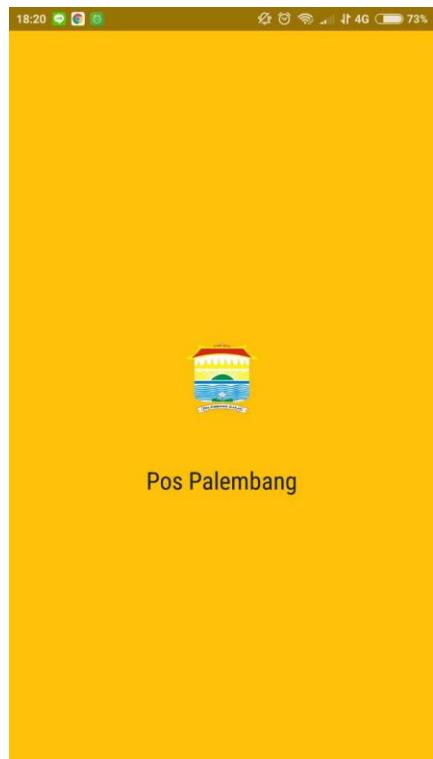
Gambar 5.44 Tampilan *project tugas* halaman *About* pada web



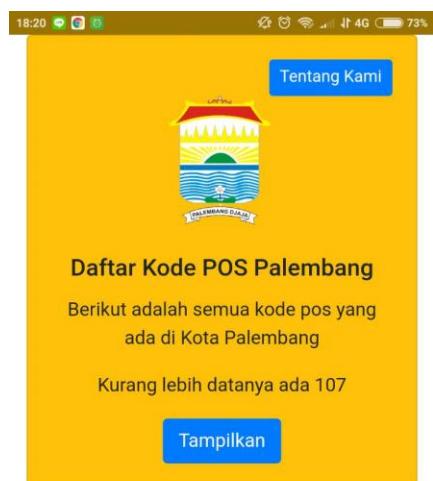
Gambar 5.45 Tampilan project tugas halaman *About* pada web saat *Offline*



Gambar 5.46 Tampilan project saat melakukan pencarian pada web



Gambar 5.47 Tampilan *splash screen project* tugas pada *mobile*



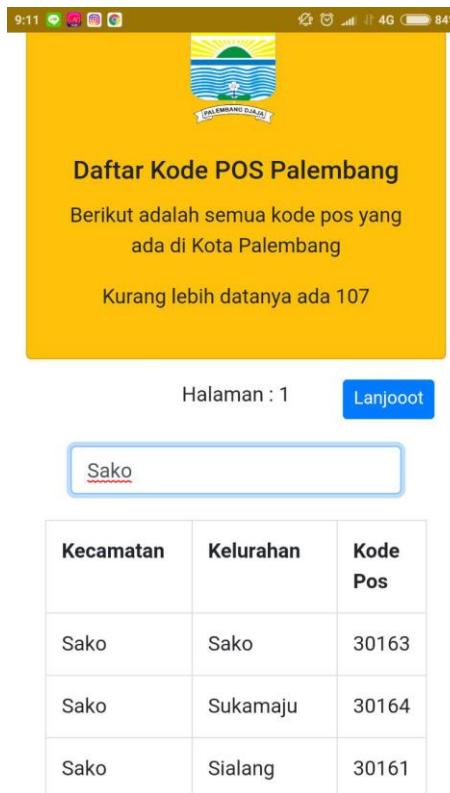
Gambar 5.48 Tampilan *project tugas* pada *mobile*



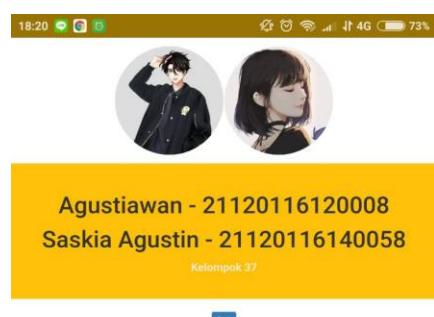
Pencarian Berdasarkan Kecamat:

Kecamatan	Kelurahan	Kode Pos
Plaju	Talang Putri	30268
Plaju	Plaju Ilir	30268
Plaiu	Taland	30268

Gambar 5.49 Tampilan project tugas pada mobile



Gambar 5.50 Tampilan *project* tugas saat melakukan pencarian pada *mobile*



Gambar 5.51 Tampilan *project* tugas halaman *About* pada *mobile*

```

<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Tugas PWA 1</title>
    <meta name="viewport" content="width=device-width, initial-
scale=1">
        <link rel="manifest" href="manifest.json">
            <link
                href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
                rel="stylesheet"/>
        <link rel="shortcut icon" href="icon/192.ico" type="image/x-
icon" />
        <link rel="icon" href="https://1.bp.blogspot.com/-eArN9-
ntWKM/T4GaCnT3PdI/AAAAAAAFAj8/wlCXZ3ltmDI/s1600/LOGO+KOTA+PALEMB
ANG.png" sizes="32x32" />
    <!-- buat manifest json -->
</head>
<body>

<!-- untuk program utama -->
<div class="container-fluid">
    <div id="app" >
        <center>
            <div class="card bg-warning mb-3">
                <div class="card-body">
                    <div align="right">
                        <a href="about.html" class="btn btn-primary
btn-sm">Tentang Kami</a>
                    </div>
                    
                    <br><br>
                    <h5 class="card-title">Daftar Kode POS
Palembang</h5>
                    <p class="card-text">Berikut adalah semua kode
pos yang ada di Kota Palembang</p>
                    <p>Kurang lebih datanya ada
{{kodepos.length}}</p>
                    <a href="#" v-on:click="tampil = !tampil" v-
show="tampil" class="btn btn-primary">Tampilkan</a>
                </div>
            </div>
            <div v-show="!tampil">
                <div class="row">
                    <div class="col-sm-4 col-md-4 col-lg-4 col-4">
                        <button class="btn btn-primary btn-sm" v-
show="awal>0" v-on:click="prev()">Balek Bae</button>
                    </div>
                    <div class="col-sm-4 col-md-4 col-lg-4 col-4">
                        <span>Halaman : {{halaman}}</span>
                    </div>
                </div>
            </div>
        </div>
    </div>
</body>

```

```

        <div class="col-sm-4 col-md-4 col-lg-4 col-4">
            <button class="btn btn-primary btn-sm" v-
show="akhir" <= kodepos.length" v-
on:click="next()">Lanjoot</button>
        </div>
    </div>
    <div class="row" style="margin:20px">
        <div class="col-sm-3"></div>
        <div class="col-sm-6">
            <input type="text" v-model="search"
class="form-control" placeholder="Pencarian Berdasarkan
Kecamatan">
        </div>
    </div>
    </center>
    <div v-show="!tampil" class="table-responsive container">
        <table id="example" class="table table-hover table-
bordered">
            <tr>
                <th>Kecamatan</th>
                <th>Kelurahan</th>
                <th>Kode Pos</th>
            </tr>
            <tr v-for="pos in cari.slice(awal,akhir)">
                <td style="width:40%">{{pos.kecamatan}}</td>
                <td style="width:40%">{{pos.kelurahan}}</td>
                <td style="width:20%">{{pos.kodepos}}</td>
            </tr>
        </table>
    </div>
    </div>
</div>
</body>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.mi
n.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/
popper.min.js"></script>
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap
.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
<script
src="https://unpkg.com/axios/dist/axios.min.js"></script>
<script src="app.js"></script>
<script src="sw.js"></script>
<!-- untuk service worker -->

</html>

```

Index.html berisi sintaks-sintaks yang mengatur tentang segalanya yang akan ditampilkan saat *link* diakses. Pada file index.html ini di bagian *Head* ditentukan

title saat *link project* diakses yaitu Tugas PWA 1. Kemudian diatur *icon* yang akan muncul yaitu gambar yang sesuai dengan *link* yang dipanggil, dimasukkan juga manifest.json agar dapat berjalan pada saat halaman awal ini diakses. Pada bagian *body* diatur susunan serta tampilan yang diinginkan. Dibuat *button* “About” yang apabila diklik akan mengakses halaman about.html terletak di sebelah kanan. Pada tampilan awal ini dimasukkan gambar lambang dari Kota Palembang sesuai dengan *link* yang dimasukkan dengan pengaturan besar 30%, terletak di bagian atas. Berikutnya ada tulisan ‘Daftar Kode Pos Palembang’ di bawah gambar, lalu di bawahnya ada tulisan lagi yaitu ‘Berikut adalah semua kode pos yang ada di Kota Palembang’, di bawahnya lagi tertulis ‘Kurang lebih datanya ada ‘ yang di akhirnya terdapat jumlah data kodepos yang ada. Di bawah tulisan tersebut ada *button* tampilan yang apabila diklik akan memunculkan tabel daftar kode pos yang ada. Setelah tabel dimunculkan akan ada keterangan halaman berapa yang tengah ditampilkan, *search box* untuk melakukan pencarian berdasarkan kecamatan, terdapat *button* Lanjoot untuk pindah ke halaman berikutnya, dan akan muncul *button* Balek Bae di halaman setelah halaman 1 untuk kembali ke halaman sebelumnya. Tabel diatur memiliki 3 kolom yaitu, Kecamata, Kelurahan, dan Kode Pos. Pada bagian akhir dilakukan pemanggilan beberapa *link* dan file app.js serta sw.js untuk mengkoneksikan halaman index.html dengan *link* dan isi dari file app.js serta sw.js untuk mengaktifkan *service worker* agar isi *link* tetap tampil saat koneksi dimatikan.

```
if ('serviceWorker' in navigator) {
    navigator.serviceWorker.register('/sw.js')
        .then(function () {
            console.log('SW registered');
        });
}
new Vue({
    el : '#app',
    data:{
        kodepos : [],search : '', awal : 0, akhir : 15, halaman : 1,tampil:true
    },
    created(){
        this.getData();
        console.log(this.awal)
    },
    methods:{
        next() {
    
```

```

        this.awal = this.awal+=15
        this.akhir = this.akhir+=15
        this.halaman = this.halaman+=1
    },
    prev() {
        this.awal = this.awal -= 15
        this.akhir = this.akhir -= 15
        this.halaman = this.halaman -= 1
    },
    getData() {
        var kd = this;
        axios.get('https://kodepos-
2d475.firebaseio.com/kota_kab/k442.json?print=pretty').then(function(response) {
            kd.kodepos = response.data;
        })
    },
    computed:{
        cari(){
            return this.kodepos.filter(pos =>{
                return
pos.kecamatan.toLowerCase().match(this.search.toLowerCase());
            });
        }
    }
);
}

```

App.js mengatur pemanggilan-pemanggilan data yang akan dimasukkan dalam *project* ini. Diaktifkan *service worker* dengan pemanggilan file sw.js. Data yang akan ditampilkan pada tabel yang diatur pada index.html diatur akan menampilkan sebanyak 15 data kode pos saja pada setiap halaman, kecuali di bagian akhir nantinya. Data kode pos yang ditampilkan dari data pada https://kodepos-2d475.firebaseio.com/kota_kab/k442.json?print=pretty.

Kemudian untuk fungsi cari ini merupakan fungsi pada *search box* pada halaman index.html diatur agar mencari kodepos berdasarkan kecamatannya.

```
{
  "name": "tugas-pwa1-kel37",
  "version": "1.0.0",
  "description": "Praktikum MDP PWA Jilid 1",
  "main": "index.js",
  "scripts": {
    "start": "http-server -c-1 -p 8888"
  },
  "keywords": [
    "pwa"
  ],
  "author": "Asisten MDP PWA 1",
  "license": "UNLICENSED",
  "devDependencies": {
    "http-server": "^0.10.0"
  }
}
```

```

    }
}
```

Package.json merupakan deskripsi mengenai *project* yang akan kita buat. Berisi nama *project*, versi, deskripsi *project*, dan halaman utama yang diakses yaitu index.js, serta tertera *author* dari *project* yaitu, Asisten MDP PWA 1.

```
{
  "name": "Pos Palembang",
  "short_name": "Kode Pos Palembang",
  "theme_color": "#ffc107",
  "background_color": "#ffc107",
  "display": "standalone",
  "orientation": "portrait",
  "scope": ".",
  "start_url": ".",
  "icons" : [
    {
      "src": "/icon/32.ico",
      "sizes": "32x32",
      "type": "image/ico"
    },
    {
      "src": "/icon/96.ico",
      "sizes": "96x96",
      "type": "image/ico"
    },
    {
      "src": "/icon/128.ico",
      "sizes": "128x128",
      "type": "image/ico"
    },
    {
      "src": "/icon/192.ico",
      "sizes": "192x192",
      "type": "image/ico"
    },
    {
      "src": "/icon/256.ico",
      "sizes": "256x256",
      "type": "image/ico"
    }
  ]
}
```

Manifest.json digunakan untuk menjadikan link yang telah kita buat, menjadi aplikasi pada handphone. Jika kita membuat *progressive web apps*, kita membutuhkan manifest.json, manifest.json merupakan file json dengan semua kebutuhan informasi yang dibutuhkan untuk menyimpan web app ke home screen handphone pengguna. Pada file manifest.json berisi mengenai *user experience* seperti splash screen, theme color, icon, dan lainnya. Pada splash screen diberi

nama Pos Palembang, judulnya Kode Pos Palembang, theme color dan background color adalah #ffc107 (warna kuning), orientasi yang digunakan adalah portrait, lalu memanggil icon-icon yang akan digunakan, disini menggunakan 5 icon.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Tugas PWA Modul 1</title>
        <link rel="manifest" href="manifest.json">
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
        <link rel="icon" href="https://1.bp.blogspot.com/-eArN9-ntWKM/T4GaCnT3PdI/AAAAAAAFAj8/wlCXZ3ltmDI/s1600/LOGO+KOTA+PALEMBANG.png" sizes="32x32" />        <style type="text/css">
            img {
                height: 250px;
                width: 250px;
                margin-top: 20px;
                margin-bottom: 20px;
            }
            .jumbotron {
                background-color: #ffc107;
            }
            .jumbotron * {
                color: #eee;
            }
        </style>
    </head>
<body>

    <div class="text-center">
        
        
    </div>

    <div class="jumbotron text-center">
        <h1>Agustiawan - 21120116120008</h1>
        <h1>Saskia Agustin - 21120116140058</h1>
        <h2>Kelompok 37</h2>

    </div>
    <div class="text-center">
        <a href="index.html" class="btn btn-primary">Back</a>
    </div>

</body>
<script src="sw.js"></script>
</html>
```

About.html merupakan halaman yang memiliki fungsi yang sama dengan index.html. Pada *file* ini diatur halaman yang akan menampilkan tentang praktikan dari kelompok 37 praktikum MDP 2018. Memiliki *title head* yaitu, Tugas PWA Modul 1 dengan *icon* berupa lambang kota Palembang yang dipanggil berdasarkan *link* yang telah dimasukkan dengan ukuran *height*, *width*, *margin-top*, dan *margin-bottom* yang telah diatur serta pemilihan warna *background* juga diatur di dalamnya. Pada bagian *body* diatur pada posisi tengah yaitu, gambar yang melambangkan kedua praktikan dari kelompok 37. Berikutnya di bawah gambar diatur tulisan besar pada bagian tengah yaitu, nama dan NIM dari praktikan. Di bawahnya terdapat *button ‘back’* yang berfungsi untuk kembali ke halaman index.html. Pada bagian akhir dimasukkan sintaks `<script src="sw.js"></script>` untuk mengaktifkan *service worker* agar halaman tetap dapat tampil walau koneksi mati.

<https://tugas-pwaI-kel37.firebaseio.com/index.html>

5.7 Kesimpulan

1. *Service Worker* berfungsi untuk menyimpan *cache* sehingga *project* tetap dapat ditampilkan ketika koneksi mati.
2. *Vue.js* digunakan untuk membangun antarmuka.
3. *Manifest.json* befungsi untuk menyimpan *web app* ke *home screen handphone* pengguna.
4. *Firebase* digunakan untuk memberi *hosting* pada *project* yang telah dibuat.
5. *Node.js* digunakan untuk menjalankan *web server*.
6. PWA berguna untuk membuat aplikasi pada *web* dan dapat *dideploy* pada Android dengan penyajian antarmuka yang bagus.
7. Untuk beberapa kasus *project* harus diakses dengan Google Chrome dalam mode penyamaran.
8. *Link* yang dimasukkan sebaiknya dalam https, karena terkadang terjadi *error* saat menggunakan http.

BAB VI

PWA #2

6.1 Tujuan

1. Praktikan mengetahui apa itu *Progressive Web Application*.
2. Praktikan memahami kegunaan dari *Progressive Web Application*.
3. Praktikan mengetahui cara mengimplementasikan *Progressive Web Application*.
4. Praktikan dapat memahami cara kerja dari *Progressive Web Application*.
5. Praktikan dapat mengimplementasikan berbagai *API* (*Application Programming Interface*) pada *Progressive Web Application*.
6. Praktikan dapat mengetahui berbagai macam jenis *API* (*Application Programming Interface*) untuk berbagai *platform*.

6.2. Alat dan Bahan

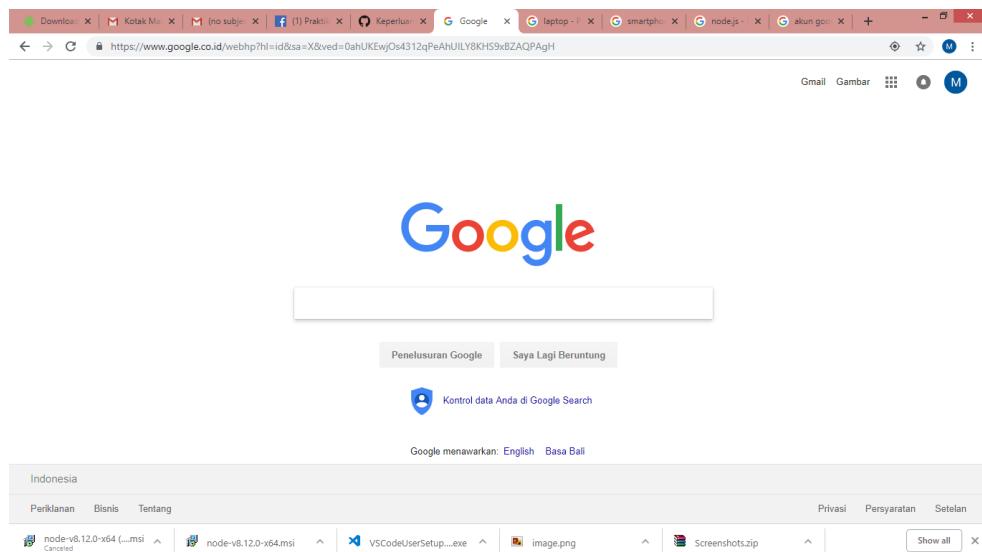
1. PC / Notebook



Gambar 6.1 Laptop

Laptop digunakan sebagai media dalam pengerjaan praktikum kali ini. Kita menginstall dan melakukan kegiatan *programming* pada *laptop*.

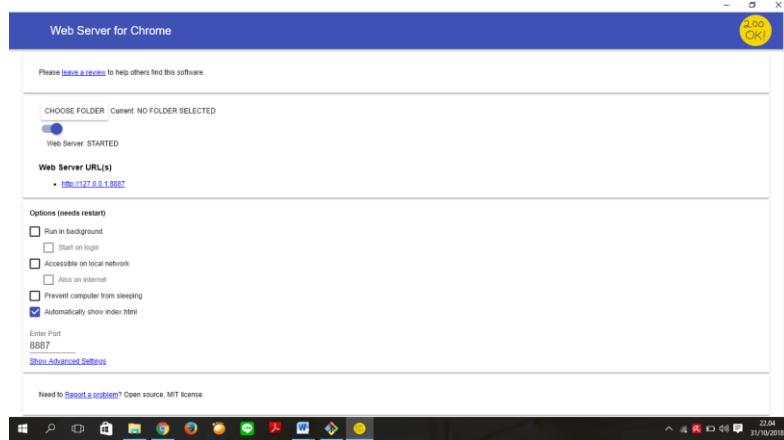
2. Google Chrome (ver 52 atau diatasnya)



Gambar 6.2 Google Chrome Pada Laptop

Google Chrome digunakan untuk kita membuka alamat *hosting* yang telah dibuat sebelumnya. Kita dapat melihat tampilan pada alamat *hosting* yang telah dibuat.

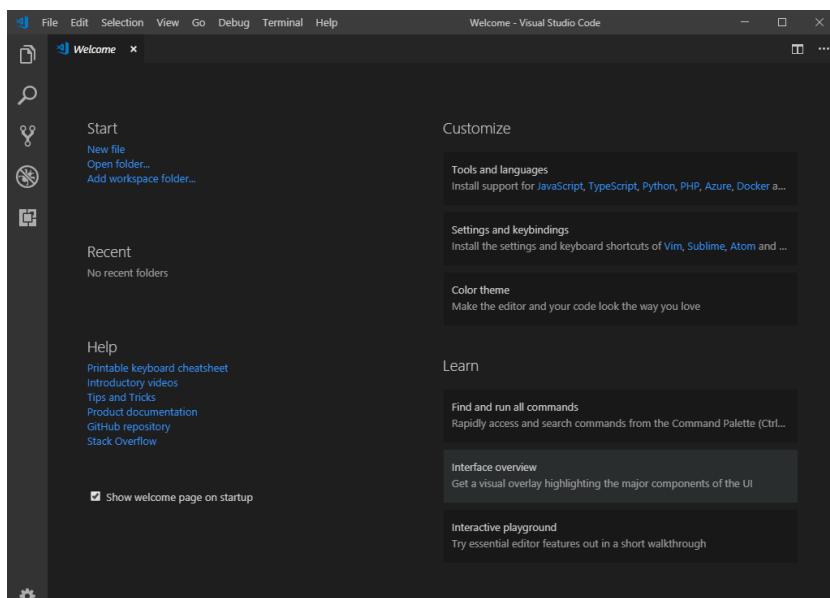
3. Web Server for Chrome (ekstensi google chrome)



Gambar 6.3 Web Server for Chrome

Pada praktikum kali ini digunakan *Web Server for Chrome* sebagai *software* yang memberikan layanan berbasis data dan berfungsi menerima permintaan dari HTTP atau HTTPS pada klien yang dikenal dan biasanya kita kenal dengan nama *web browser* (Mozilla Firefox, Google Chrome) dan untuk mengirimkan kembali yang hasilnya dalam bentuk beberapa halaman *web* dan pada umumnya akan berbentuk dokumen HTML.

4. Notepad++ / Sublime text 3/Visual Studio Code (Text editor)



Gambar 6.4 Visual Studio Code

Text Editor yang akan digunakan pada praktikum kali ini yaitu Visual Studio Code. Di sinilah kita akan melakukan kegiatan *coding program* untuk praktikum.

6.3 Dasar Teori

6.3.1 Progressive Web Application

Progressive Web App adalah pengalaman yang menggabungkan yang terbaik dari *web* dan yang terbaik dari aplikasi. Pengalaman ini bermanfaat untuk pengguna dari kunjungan pertamanya di *tab browser*, tanpa harus melakukan pemasangan. Pengguna secara progresif akan membangun hubungan dengan aplikasi, yang semakin lama semakin kuat. Aplikasi dimuat cepat, bahkan pada jaringan yang tidak stabil, mengirimkan pemberitahuan *push* yang relevan, memiliki ikon pada layar beranda, dan dimuat dengan pengalaman tingkat atas selayar penuh.

PWA memiliki sifat-sifat sebagai berikut :

- **Progresif** - Bekerja untuk setiap pengguna, apa pun pilihan *browser* mereka karena dibangun dengan peningkatan progresif sebagai konsep intinya.
- **Responsif** - Cocok dengan setiap faktor bentuk: perangkat *desktop*, seluler, tablet, atau apa saja yang muncul berikutnya.
- **Konektivitas independen** - Disempurnakan dengan *service worker* agar bisa bekerja offline atau pada jaringan berkualitas-rendah.
- **Seperti-Aplikasi** - Terasa seperti sebuah aplikasi untuk pengguna dengan interaksi dan navigasi bergaya-aplikasi karena mereka dibangun di atas model *shell* aplikasi.
- **Segar** - Selalu terkini berkat proses pembaruan *service worker*.
- **Aman** - Disediakan melalui HTTPS untuk mencegah *snooping* dan memastikan materi belum dirusak.
- **Dapat ditemukan** - Dapat diidentifikasi sebagai "aplikasi" berkat manifes W3C dan cakupan registrasi *service worker*, yang memungkinkan mesin telusur untuk menemukannya.
- **Bisa dilibatkan-kembali** - Kemudahan untuk dilibatkan-kembali dengan fitur seperti pemberitahuan push.
- **Dapat dipasang** - Memungkinkan pengguna untuk "menyimpan" aplikasi yang mereka anggap paling berguna di layar beranda tanpa kerumitan toko aplikasi.

- **Bisa ditautkan** - Dapat dengan mudah dibagikan melalui URL, tidak memerlukan pemasangan yang rumit.

Sumber: <https://developers.google.com/web/fundamentals/codelabs/your-first-pwapp/?hl=id>

6.3.2 Service Worker

Service worker adalah skrip yang dijalankan *browser* Anda di latar belakang, terpisah dari laman *web*, yang membuka pintu ke berbagai fitur yang tidak memerlukan laman *web* atau interaksi pengguna. Saat ini, *service worker* sudah menyertakan berbagai fitur seperti pemberitahuan *push* dan sinkronisasi latar belakang. Di masa mendatang, *service worker* akan mendukung hal-hal lainnya seperti sinkronisasi berkala atau *geofencing*. Fitur inti yang didiskusikan dalam tutorial adalah kemampuan mencegat dan menangani permintaan jaringan, termasuk mengelola *cache* respons lewat program.

Yang membuat API ini menarik adalah karena memungkinkan Anda mendukung pengalaman *offline*, yang memberikan *developer* kontrol penuh atas pengalaman.

Sebelum *service worker*, ada satu API lain yang memberi pengguna pengalaman *offline* di web, yang disebut AppCache. Masalah utama pada AppCache adalah jumlah *gotcha* yang ada serta fakta bahwa meskipun desain bekerja dengan sangat baik untuk laman aplikasi *web* tunggal, namun ternyata tidak begitu baik untuk situs multi-laman. *Service worker* telah didesain untuk menghindari titik-titik menyulitkan yang sudah umum ini.

Sumber: <https://developers.google.com/web/fundamentals/primers/service-workers/?hl=id>

6.3.3 Web Server

Server atau *Web server* adalah sebuah *software* yang memberikan layanan berbasis data dan berfungsi menerima permintaan dari HTTP atau HTTPS pada klien yang dikenal dan biasanya kita kenal dengan nama *web browser* (Mozilla Firefox, Google Chrome) dan untuk mengirimkan kembali yang hasilnya dalam bentuk beberapa halaman *web* dan pada umumnya akan berbentuk dokumen HTML.

Fungsi utama *Server* atau *Web server* adalah untuk melakukan atau akan mentransfer berkas permintaan pengguna melalui protokol komunikasi yang telah

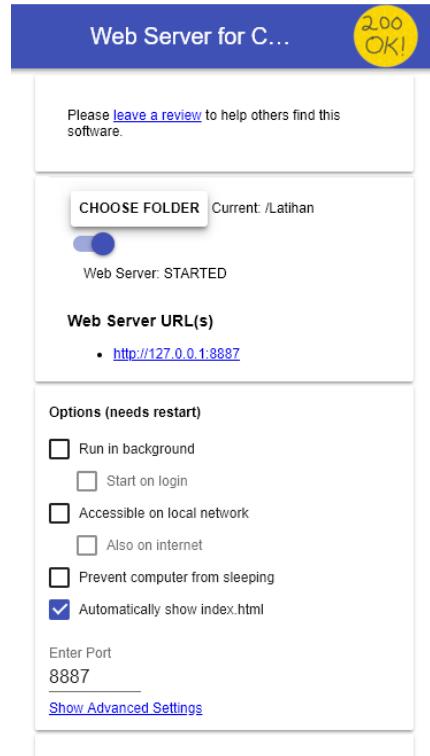
ditetukan sedemikian rupa. Halaman *web* yang diminta terdiri dari berkas teks, video, gambar, *file* dan banyak lagi. Pemanfaatan *web server* berfungsi untuk mentransfer seluruh aspek pemberkasan dalam sebuah halaman *web* termasuk yang di dalam berupa teks, video, gambar dan banyak lagi.

Salah satu contoh dari *Web Server* adalah Apache. Apache (*Apache Web Server – The HTTP Web Server*) merupakan *web server* yang paling banyak dipergunakan di Internet. Program ini pertama kali didesain untuk sistem operasi lingkungan UNIX. Apache mempunyai program pendukung yang cukup banyak. Hal ini memberikan layanan yang cukup lengkap bagi penggunanya.

Sumber: <https://idcloudhost.com/pengertian-web-server-dan-fungsinya/>

6.4 Langkah Percobaan

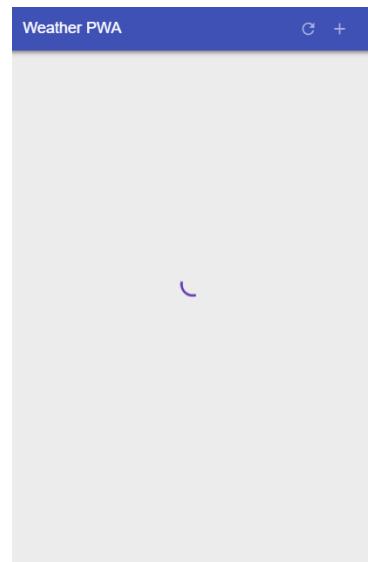
1. Menyalakan dan mengkonfigurasi Web Server for Chrome.



Gambar 6.5 Memilih *folder* Latihan pada *Web Server for Chrome*

Choose folder lalu pilih *folder* Latihan yang telah *didownload* sebelumnya, kemudian centang *Automatically show index.html*. Lalu nyalakan *web server*.

2. Buka link dari *web server* dan lihat tampilannya.



Gambar 6.6 Tampilan *link* dari *web server*

3. Untuk melihat bagaimana data cuaca palsu dirender, hilangkan tanda komentar pada baris berikut di bagian bawah file `index.html`

```
<!--<script src="scripts/app.js" async></script>-->
```

Berikutnya, hilangkan tanda komentar pada baris berikut di bagian bawah file `app.js`

```
// app.updateForecastCard(initialWeatherForecast);
```

Muat ulang aplikasi. Hasilnya akan menjadi kartu prakiraan cuaca yang terformat dengan baik (menggunakan data palsu yang telah dibuat) dengan spinner dinonaktifkan, seperti ini:



Gambar 6.7 Hasil tampilan *link* dari *web server* setelah diedit dan dimuat ulang

4. Tambahkan kode yang diperlukan untuk menyimpan preferensi pengguna. Temukan komentar TODO berikut dalam `app.js`

```
// TODO add saveSelectedCities function here
```

Dan tambahkan kode berikut di bawah komentar.

```
// Save list of cities to localStorage.
app.saveSelectedCities = function() {
    var selectedCities = JSON.stringify(app.selectedCities);
    localStorage.selectedCities = selectedCities;
};
```

```
// TODO add saveSelectedCities function here
// Save list of cities to localStorage.
app.saveSelectedCities = function() {
    var selectedCities = JSON.stringify(app.selectedCities);
    localStorage.selectedCities = selectedCities;
};
```

Gambar 6.8 Penambahan kode di bawah // TODO add saveSelectedCities
function here

5. Berikutnya, tambahkan kode *startup* untuk memeriksa apakah pengguna memiliki kota yang disimpan dan merender kota tersebut, atau menggunakan data yang dimasukkan. Temukan komentar berikut:

```
// TODO add startup code here
```

Dan tambahkan kode berikut di bawah komentar.

```
app.selectedCities = localStorage.selectedCities;
if (app.selectedCities) {
    app.selectedCities = JSON.parse(app.selectedCities);
    app.selectedCities.forEach(function(city) {
        app.getForecast(city.key, city.label);
    });
} else {
    /* The user is using the app for the first time, or the
    user has not
     * saved any cities, so show the user some fake data. A
    real app in this
     * scenario could guess the user's location via IP
    lookup and then inject
     * that data into the page.
    */
    app.updateForecastCard(initialWeatherForecast);
    app.selectedCities = [
        {key: initialWeatherForecast.key, label:
    initialWeatherForecast.label}
    ];
    app.saveSelectedCities();
}
```

```
// TODO add startup code here
app.selectedCities = localStorage.selectedCities;
if (app.selectedCities) {
    app.selectedCities = JSON.parse(app.selectedCities);
    app.selectedCities.forEach(function(city) {
        app.getForecast(city.key, city.label);
    });
} else {
    /* The user is using the app for the first time, or the user has not
    * saved any cities, so show the user some fake data. A real app in this
    * scenario could guess the user's location via IP lookup and then inject
    * that data into the page.
    */
    app.updateForecastCard(initialWeatherForecast);
    app.selectedCities = [
        {key: initialWeatherForecast.key, label: initialWeatherForecast.label}
    ];
    app.saveSelectedCities();
}
```

Gambar 6.9 Penambahan kode di bawah // TODO add startup code here

Kode *startup* memeriksa bila ada kota yang disimpan dalam penyimpanan lokal. Jika ada, itu akan mem-parse data penyimpanan lokal dan kemudian menampilkan kartu prakiraan untuk masing-masing kota yang disimpan. Jika tidak, kode *startup* akan menggunakan data prakiraan palsu dan menyimpannya sebagai kota *default*.

6. Mengubah penanganan tombol "add city" untuk menyimpan kota yang dipilih ke penyimpanan lokal. Cari `butAddCity` pada `app.js` lalu ubah menjadi seperti berikut.

```
document.getElementById('butAddCity').addEventListener('click', function() {
    // Add the newly selected city
    var select = document.getElementById('selectCityToAdd');
    var selected = select.options[selected.selectedIndex];
    var key = selected.value;
    var label = selected.textContent;
    if (!app.selectedCities) {
        app.selectedCities = [];
    }
    app.getForecast(key, label);
    app.selectedCities.push({key: key, label: label});
    app.saveSelectedCities();
    app.toggleAddDialog(false);
});
```

```
document.getElementById('butAddCity').addEventListener('click', function() {
    // Add the newly selected city
    var select = document.getElementById('selectCityToAdd');
    var selected = select.options[selected.selectedIndex];
    var key = selected.value;
    var label = selected.textContent;
    if (!app.selectedCities) {
        app.selectedCities = [];
    }
    app.getForecast(key, label);
    app.selectedCities.push({key: key, label: label});
    app.saveSelectedCities();
    app.toggleAddDialog(false);
});
```

Gambar 6.10 Mengubah kode untuk penanganan tombol *add city*

7. Selanjutnya mendaftarkan *service worker*. Temukan komentar berikut pada `app.js`

```
// TODO add service worker code here
```

Tambahkan kode berikut setelah komentar tersebut

```
if ('serviceWorker' in navigator) {
    navigator.serviceWorker
```

```

    .register('./service-worker.js')
        .then(function() { console.log('Service Worker Registered'); });
}

```

```

// TODO add service worker code here
if ('serviceWorker' in navigator) {
    navigator.serviceWorker
        .register('./service-worker.js')
        .then(function() { console.log('Service Worker Registered'); });
}
}());

```

Gambar 6.11 Penambahan kode di bawah // TODO add service worker code here

- Buatlah *file* bernama `service-worker.js` pada *folder* latihan lalu tambahkan kode berikut.

```

var cacheName = 'weatherPWA-step-6-1';
var filesToCache = [];

self.addEventListener('install', function(e) {
    console.log('[ServiceWorker] Install');
    e.waitUntil(
        caches.open(cacheName).then(function(cache) {
            console.log('[ServiceWorker] Caching app shell');
            return cache.addAll(filesToCache);
        })
    );
});

```

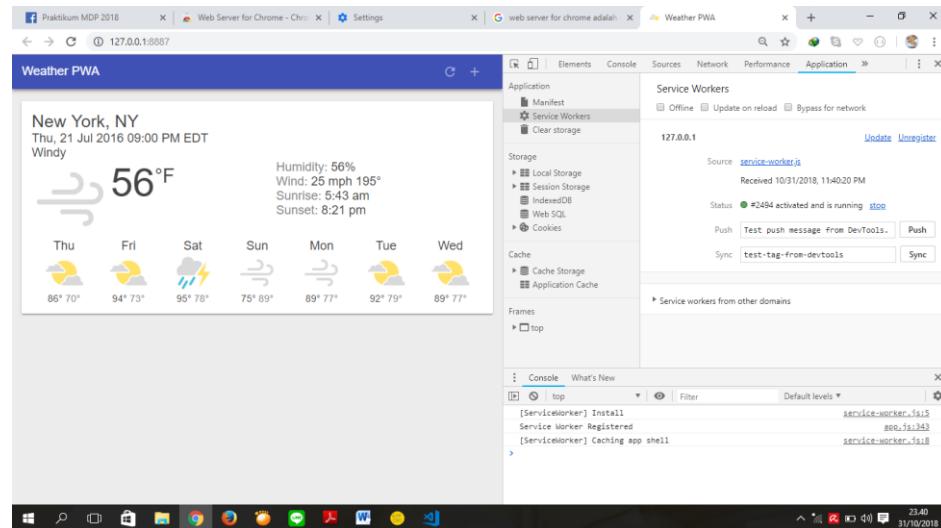
```

index.html JS appjs JS service-workerjs ×
1 var cacheName = 'weatherPWA-step-6-1';
2 var filesToCache = [];
3
4 self.addEventListener('install', function(e) {
5     console.log('[ServiceWorker] Install');
6     e.waitUntil(
7         caches.open(cacheName).then(function(cache) {
8             console.log('[ServiceWorker] Caching app shell');
9             return cache.addAll(filesToCache);
10        })
11    );
12 });
13 |

```

Gambar 6.12 Membuat *file* service-worker.js

9. Buka Dev Tools pada Google chrome (Ctrl + Shift + i) saat membuka page latihan. Jika berhasil maka tampilan pada *Service Workers* akan seperti ini.



Gambar 6.13 Tampilan *Dev Tools* pada Google Chrome setelah ditambah kode pada program

10. Tambahkan *event listener activate* di atas *event listener install* dalam file *service-worker.js*

```
self.addEventListener('activate', function(e) {
    console.log('[ServiceWorker] Activate');
    e.waitUntil(
        caches.keys().then(function(keyList) {
            return Promise.all(keyList.map(function(key) {
                if (key !== cacheName) {
                    console.log('[ServiceWorker] Removing old
cache', key);
                    return caches.delete(key);
                }
            }));
        })
    );
    return self.clients.claim();
});
```

11. Pada bagian atas file *service-worker.js*, ganti `var filesToCache = [];` dengan kode di bawah ini:

```
var filesToCache = [
  '/',
  '/index.html',
  '/scripts/app.js',
  '/styles/inline.css',
  '/images/clear.png',
  '/images/cloudy-scattered-showers.png',
  '/images/cloudy.png',
  '/images/fog.png',
```

```

    '/images/ic_add_white_24px.svg',
    '/images/ic_refresh_white_24px.svg',
    '/images/partly-cloudy.png',
    '/images/rain.png',
    '/images/scattered-showers.png',
    '/images/sleet.png',
    '/images/snow.png',
    '/images/thunderstorm.png',
    '/images/wind.png'
];

```

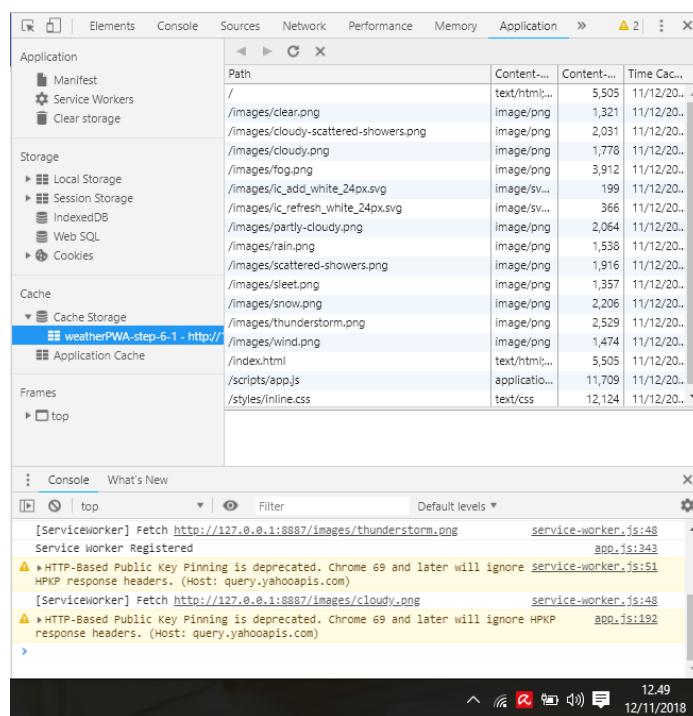
12. Sediakan *shell* aplikasi dari *cache*. Tambahkan kode berikut ke bagian bawah file *service-worker.js*

```

self.addEventListener('fetch', function(e) {
  console.log('[ServiceWorker] Fetch', e.request.url);
  e.respondWith(
    caches.match(e.request).then(function(response) {
      return response || fetch(e.request);
    })
  );
});

```

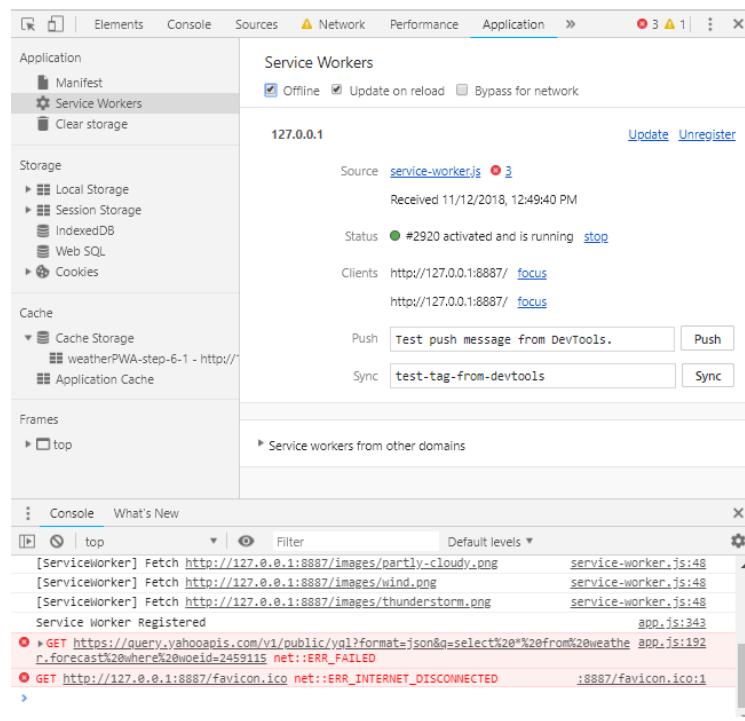
13. Muat ulang laman Anda lalu buka panel *Cache Storage* pada panel *Application* dari DevTools, lalu lihat apakah tampilan sudah seperti ini.



Gambar 6.14 Hasil pada *Cache Storage* setelah dimuat ulang

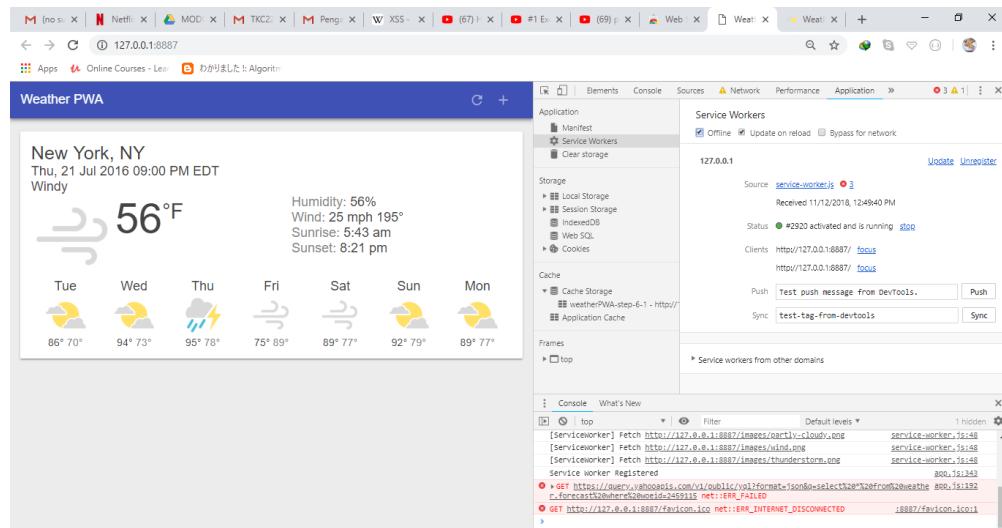
Jika sudah seperti itu maka *resource* yang digunakan telah berhasil di *cache*.

14. Sekarang menguji mode *offline*. Kembali ke panel *Service Worker* dari DevTools dan aktifkan kotak centang *Offline*. Setelah mengaktifkannya, Anda akan melihat ikon peringatan kecil berwarna kuning di sebelah *tab panel Network*. Ini menunjukkan bahwa kondisi telah *offline*.



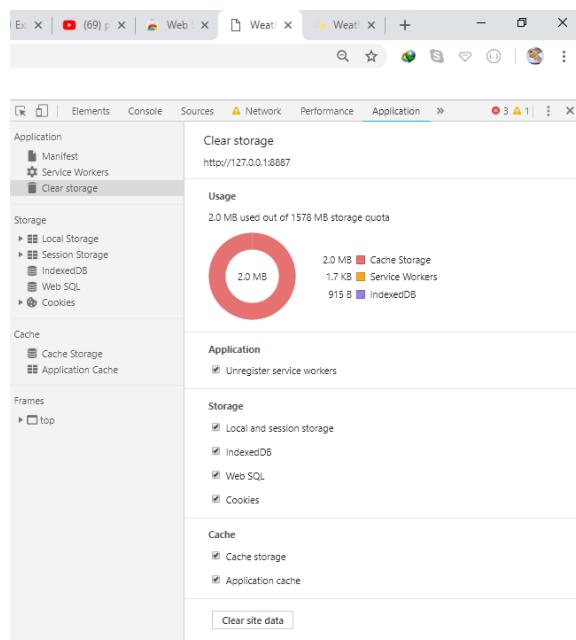
Gambar 6.15 Mengaktifkan mode *offline* pada *Service Worker*

15. *Reload* halaman dan lihat apakah yang akan ditampilkan dalam keadaan *offline*.



Gambar 6.16 Hasil yang ditampilkan pada mode *offline*

16. Selanjutnya adalah membuat *service worker* untuk meng-cache data prakiraan cuaca. Pertama hapus semua data yang tersimpan (localStorage, data indexedDB, file cache) dan membuang *service worker*, gunakan panel *Clear storage* di tab *Application* lalu klik *Clear site data*.



Gambar 6.17 Menghapus semua data yang tersimpan

17. Buka file `service-worker.js` lalu tambahkan kode berikut ke bagian atas dari file tersebut.

```
var dataCacheName = 'weatherData-v1';
```

18. Perbarui `if (key !== cacheName) {` dari event listener `activate` sehingga tidak menghapus *cache data* ketika membersihkan *cache shell* aplikasi.

```
if (key !== cacheName && key !== dataCacheName) {
```

19. Perbarui event listener `fetch` untuk menangani permintaan ke data API secara terpisah dari permintaan lainnya.

```
self.addEventListener('fetch', function(e) {
  console.log('[Service Worker] Fetch', e.request.url);
  var dataUrl = 'https://query.yahooapis.com/v1/public/yql';
  if (e.request.url.indexOf(dataUrl) > -1) {
    /*
     * When the request URL contains dataUrl, the app is
     * asking for fresh
     * weather data. In this case, the service worker always
     * goes to the
```

```

        * network and then caches the response. This is called
the "Cache then
        * network" strategy:
        */
e.respondWith(
    caches.open(dataCacheName).then(function(cache) {
        return fetch(e.request).then(function(response) {
            cache.put(e.request.url, response.clone());
            return response;
        });
    })
);
} else {
/*
    * The app is asking for app shell files. In this
scenario the app uses the
    * "Cache, falling back to the network" offline
strategy:
    */
e.respondWith(
    caches.match(e.request).then(function(response) {
        return response || fetch(e.request);
    })
);
}
);
}
);

```

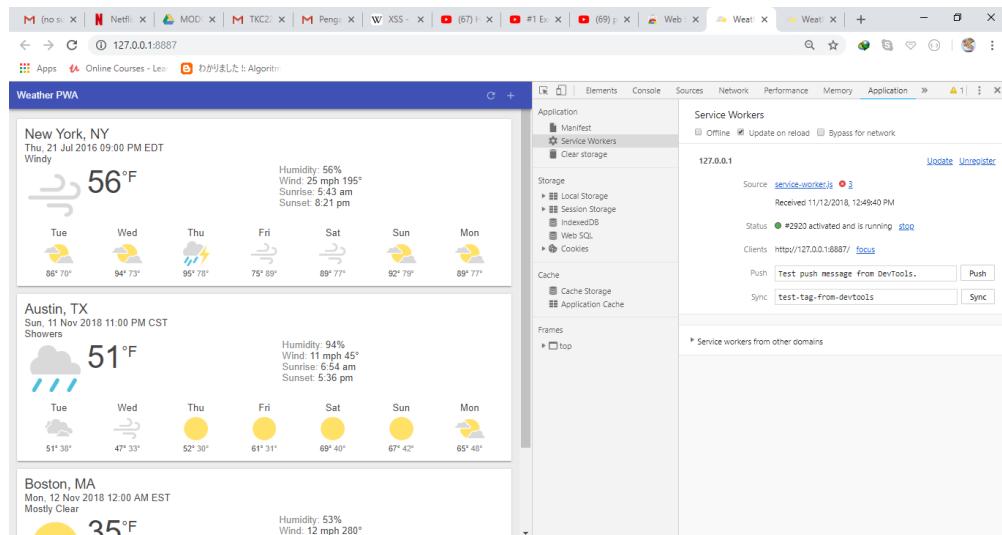
20. Berikutnya buka `app.js`, kita harus memeriksa apakah objek `caches` ada dan meminta data terbaru dari situ. Temukan komentar `TODO add cache logic here` di `app.getForecast()`, dan kemudian tambahkan kode berikut di bawah komentar tersebut.

```

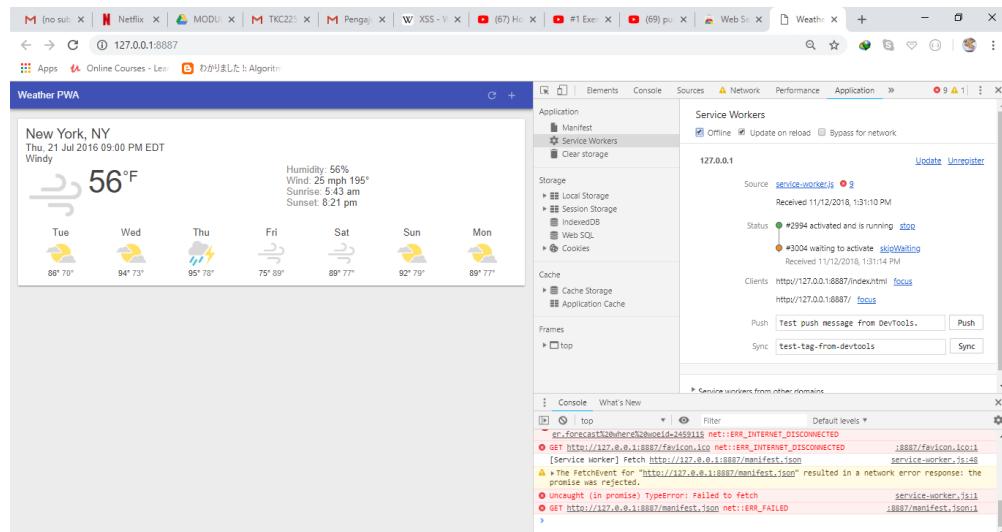
if ('caches' in window) {
    /*
        * Check if the service worker has already cached this
city's weather
        * data. If the service worker has the data, then
display the cached
        * data while the app fetches the latest data.
    */
    caches.match(url).then(function(response) {
        if (response) {
            response.json().then(function
updateFromCache(json) {
                var results = json.query.results;
                results.key = key;
                results.label = label;
                results.created = json.query.created;
                app.updateForecastCard(results);
            });
        }
    });
}

```

21. Lakukan Pengujian dengan menambahkan kota baru pada prakiraan cuaca, lalu centang *Offline* pada *Service workers* di Dev Tools Google Chrome, lalu coba *reload* halaman tersebut.



Gambar 6.18 Menambahkan kota baru saat mode *Online*



Gambar 6.19 Tampilan saat mode *Offline*

22. Buat file bernama `manifest.json` di folder Latihan dan masukkan kode berikut.

```
{
  "name": "Weather",
  "short_name": "Weather",
  "icons": [
    {
      "src": "images/icons/icon-128x128.png",
      "sizes": "128x128",
      "type": "image/png"
    }
  ]
}
```

```

    "src": "images/icons/icon-144x144.png",
    "sizes": "144x144",
    "type": "image/png"
  },
  {
    "src": "images/icons/icon-152x152.png",
    "sizes": "152x152",
    "type": "image/png"
  },
  {
    "src": "images/icons/icon-192x192.png",
    "sizes": "192x192",
    "type": "image/png"
  },
  {
    "src": "images/icons/icon-256x256.png",
    "sizes": "256x256",
    "type": "image/png"
  ],
  "start_url": "/index.html",
  "display": "standalone",
  "background_color": "#3E4EB8",
  "theme_color": "#2F3BA2"
}

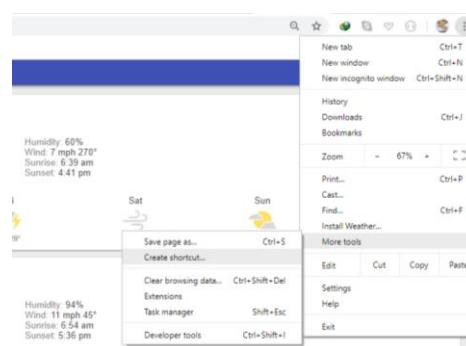
```

23. Sekarang tambahkan baris berikut ke bagian bawah elemen `<head>` dalam file `index.html`

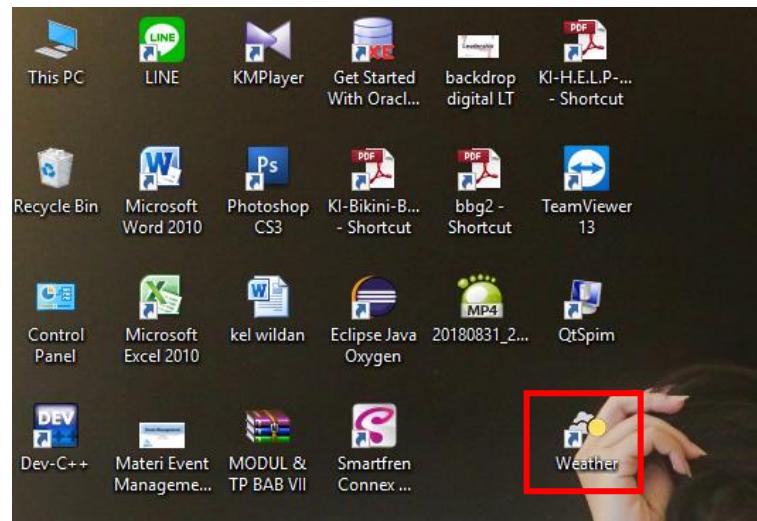
```
<link rel="manifest" href="/manifest.json">
```

24. Run aplikasi pada google chrome lalu buka Developer tools (Ctrl + Shift + i) lalu klik *tab application* dan klik *manifest*, pada bagian *identity* klik *Add to homescreen* lalu *reload page*, maka akan muncul *pop up* pada bagian atas google chrome, klik *Add* pada *pop up* tersebut dan lihat hasilnya.

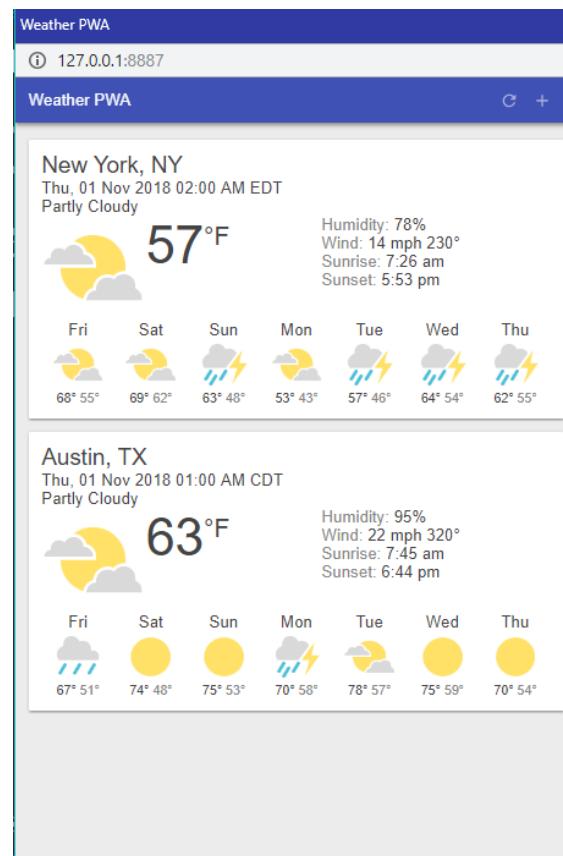
Untuk beberapa kasus tidak dapat menambahkan dengan cara ini lakukan dengan cara memilih *Create Shortcut* pada menu *More tools* dalam *Customize and Control Google Chrome*.



Gambar 6.20 Membuat *shortcut* dari PWA cuaca



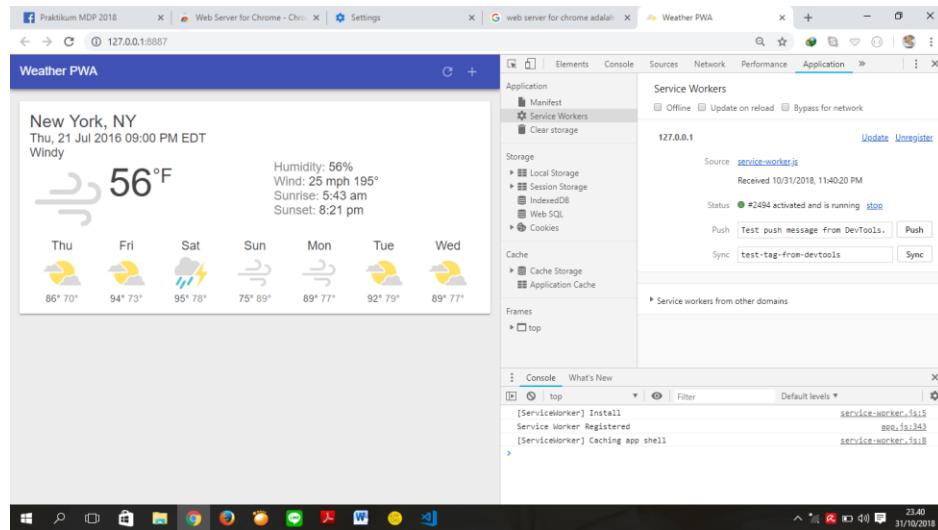
Gambar 6.21 Shortcut pada homescreen



Gambar 6.22 Tampilan aplikasi ketika dibuka

6.5 Analisa Percobaan

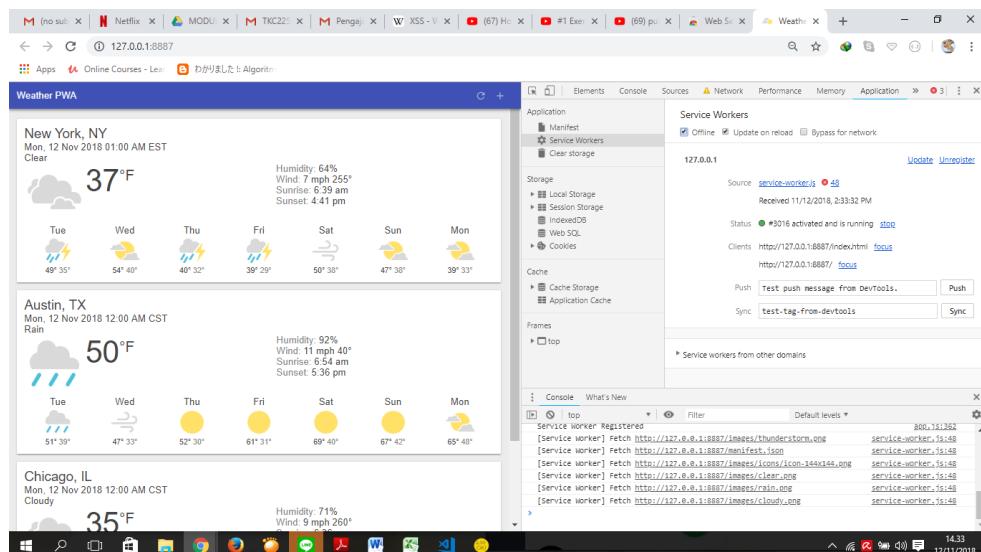
6.5.1 Percobaan menampilkan fake data saat offline



Gambar 6.23 Penampilan *fake data* pada saat *offline*

Pada percobaan ini, kita menggunakan *fake data*, atau data yang tidak *real* pada *data weather*, jadi data ini merupakan *data default* ketika *web server* tidak mempunya *cache data* ketika mode *offline* diaktifkan dan *service-worker* akan mengarahkan pada *data default* yang dapat disetting melalui *scripts file app.js* yang dibuat dengan sebuah *variable* bernama `initialWeatherForecast`, disitu data berisikan pengaturan dalam membuat *data default fake* yang akan digunakan.

6.5.2 Percobaan menampilkan data terakhir saat offline



Gambar 6.24 Penampilan data terakhir pada saat *offline*

Pada percobaan kali ini, kita menggunakan data terakhir yang ditampilkan sebelum web server memasuki mode offline, jadi data yang terakhir ada pada weather yang merupakan data realtime, akan tetap bisa dilihat karena cache storage yang menyimpan data tersebut masih bisa digunakan ketika memasuki mode offline, hal ini juga merupakan fungsi dari cache storage yang mana dapat menyimpan serpihan data yang nantinya dapat digunakan lagi jika tidak terkoneksi pada internet.

6.5.3 Membuat manifest

```
{
  "name": "Weather",
  "short_name": "Weather",
  "icons": [
    {
      "src": "images/icons/icon-128x128.png",
      "sizes": "128x128",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-144x144.png",
      "sizes": "144x144",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-152x152.png",
      "sizes": "152x152",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-192x192.png",
      "sizes": "192x192",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-256x256.png",
      "sizes": "256x256",
      "type": "image/png"
    }
  ],
  "start_url": "/index.html",
  "display": "standalone",
  "background_color": "#3E4EB8",
  "theme_color": "#2F3BA2"
}
```

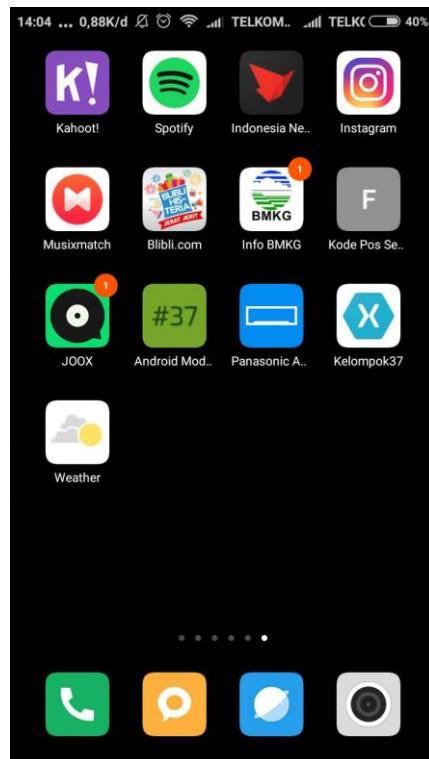
Manifest.json merupakan *file* berekstensi json, yaitu Bahasa pemrograman kekinian yang digunakan, dan butuh *library* penerjemah untuk menerjemahkan Bahasa json. *File* manifest.json pada proyek kali ini berfungsi sebagai *file* yang membuat suatu tampilan pada *mobile* saat *icon* pada *homescreen* di tekan. Karena pada dasarnya proyek ini merupakan proyek html yang diperuntukan untuk *web*,

namun agar tampilan pada *mobile* bisa lebih di tolerir, maka fungsi manifest.json ini lah yang berperan dalam pembuatan *splashscreen* awal program saat dibuka di *mobile* serta memberikan kemampuan untuk mengontrol bagaimana aplikasi terlihat oleh *user*.

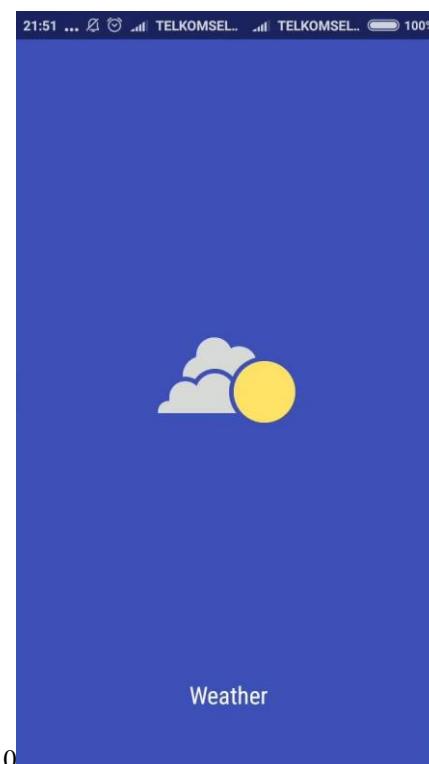
Pada *source code* diatas, terdapat beberapa pendefinisian, diawali dengan “*name*” yang mana berfungsi sebagai nama yang akan ditampilkan pada tampilan awal *splashscreen* program, kemudian “*short_name*” berfungsi sebagai nama yang diberikan pada *homescreen* program, “*theme_color*” berfungsi sebagai pemberi warna utama pada halaman dan untuk kode warna yang digunakan adalah #2F3BA2 yang mana berwarna biru dongker, dan “*background_color*” juga merupakan tampilan awal yang mana *backgroundnya* memiliki jenis *style color* atau berupa warna biru keputihan atau dengan kode warna #3E4EB8.

Untuk “*display*” diberikan jenis “*standalone*” agar jenis tampilan tidak seperti *web browser* dan lebih mengarah pada tampilan aplikasi *mobile* sungguhan, jika *user* ingin tampilan sesuai dengan *web browser*, bisa mengubah jenis *display* menjadi “*browser*”. Lalu “*start_url*” berfungsi sebagai halaman mana yang akan diakses pertama oleh aplikasi.

Dan yang terakhir adalah “*icons*” yang berfungsi untuk menentukan *icon* yang akan dipakai aplikasi, pada proyek ini masing-masing *icon* dipersiapkan dengan ditentukan “*src*” atau asal alamat *icon* tersebut, “*sizes*” untuk berapa besar ukuran *icon* yang digunakan, dan “*type*” merupakan tipe dari gambar yang digunakan.



Gambar 6.25 Hasil *deploy* aplikasi pada *icon homescreen*



Gambar 6.26 Tampilan *splash screen* aplikasi pada *mobile*



Gambar 6.27 Tampilan *index* aplikasi pada *mobile*

(Link Github Tugas: <https://github.com/saskiaugust/kel37-pwa2>)

6.6 Tugas

Pada percobaan tugas, diminta menambahkan *data weather* pada aplikasi dengan 3 buah kota yang ada di Indonesia dan memiliki data yang *realtime* atau tidak *fake*, kemudian diminta untuk membuat *footer* pada tampilan aplikasi, dan yang terakhir mengubah logo dan tampilan *splashscreen* pada *manifest*. Oleh karena itu dibutuhkan *source code* berikut pada app.js, index.html, dan manifest.json.

Adapun *source code* pada app.js adalah sebagai berikut:

```
(function() {
    'use strict';

    var app = {
        isLoading: true,
        visibleCards: {},
        selectedCities: [],
        spinner: document.querySelector('.loader'),
        cardTemplate: document.querySelector('.cardTemplate'),
        container: document.querySelector('.main'),
        addDialog: document.querySelector('.dialog-container'),
        daysOfWeek: ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat',
        'Sun']
    };

    document.getElementById('butRefresh').addEventListener('click',
    function() {
        // Refresh all of the forecasts
        app.updateForecasts();
    });

    document.getElementById('butAdd').addEventListener('click',
    function() {
        // Open/show the add new city dialog
        app.toggleAddDialog(true);
    });

    document.getElementById('butAddCity').addEventListener('click',
    function() {
        // Add the newly selected city
        var select = document.getElementById('selectCityToAdd');
        var selected = select.options[select.selectedIndex];
        var key = selected.value;
        var label = selected.textContent;
        if (!app.selectedCities) {
            app.selectedCities = [];
        }
        app.getForecast(key, label);
        app.selectedCities.push({key: key, label: label});
        app.saveSelectedCities();
        app.toggleAddDialog(false);
    });
});
```

```

});;

document.getElementById('butAddCancel').addEventListener('click'
, function() {
    // Close the add new city dialog
    app.toggleAddDialog(false);
});

// Toggles the visibility of the add new city dialog.
app.toggleAddDialog = function(visible) {
    if (visible) {
        app.addDialog.classList.add('dialog-container--visible');
    } else {
        app.addDialog.classList.remove('dialog-container--visible');
    }
};

app.toggleDelDialog = function(visible) {
    if (visible) {
        app.addDialog.classList.remove('dialog-container--visible');
    } else {
        app.addDialog.classList.remove('dialog-container--visible');
    }
};

// Updates a weather card with the latest weather forecast. If
the card
// doesn't already exist, it's cloned from the template.
app.updateForecastCard = function(data) {
    var dataLastUpdated = new Date(data.created);
    var sunrise = data.channel.astronomy.sunrise;
    var sunset = data.channel.astronomy.sunset;
    var current = data.channel.item.condition;
    var humidity = data.channel.atmosphere.humidity;
    var wind = data.channel.wind;

    var card = app.visibleCards[data.key];
    if (!card) {
        card = app.cardTemplate.cloneNode(true);
        card.classList.remove('cardTemplate');
        card.querySelector('.location').textContent = data.label;
        card.removeAttribute('hidden');
        app.container.appendChild(card);
        app.visibleCards[data.key] = card;
    }

    // Verifies the data provide is newer than what's already
visible
    // on the card, if it's not bail, if it is, continue and
update the
    // time saved in the card
    var cardLastUpdatedElem = card.querySelector('.card-last-
```

```

updated');
var cardLastUpdated = cardLastUpdatedElem.textContent;
if (cardLastUpdated) {
    cardLastUpdated = new Date(cardLastUpdated);
    // Bail if the card has more recent data than the data
    if (dataLastUpdated.getTime() < cardLastUpdated.getTime())
{
    return;
}
}
cardLastUpdatedElem.textContent = data.created;

card.querySelector('.description').textContent =
current.text;
card.querySelector('.date').textContent = current.date;
card.querySelector('.current
.icon').classList.add(app.getIconClass(current.code));
card.querySelector('.current .temperature
.value').textContent =
Math.round(current.temp);
card.querySelector('.current .sunrise').textContent =
sunrise;
card.querySelector('.current .sunset').textContent = sunset;
card.querySelector('.current .humidity').textContent =
Math.round(humidity) + '%';
card.querySelector('.current .wind .value').textContent =
Math.round(wind.speed);
card.querySelector('.current .wind .direction').textContent =
wind.direction;
var nextDays = card.querySelectorAll('.future .oneday');
var today = new Date();
today = today.getDay();
for (var i = 0; i < 7; i++) {
    var nextDay = nextDays[i];
    var daily = data.channel.item.forecast[i];
    if (daily && nextDay) {
        nextDay.querySelector('.date').textContent =
app.daysOfWeek[(i + today) % 7];

nextDay.querySelector('.icon').classList.add(app.getIconClass(da
ily.code));
        nextDay.querySelector('.temp-high .value').textContent =
Math.round(daily.high);
        nextDay.querySelector('.temp-low .value').textContent =
Math.round(daily.low);
    }
}
if (app.isLoading) {
    app.spinner.setAttribute('hidden', true);
    app.container.removeAttribute('hidden');
    app.isLoading = false;
}
};

/*
 * Gets a forecast for a specific city and updates the card

```

```

with the data.
    * getForecast() first checks if the weather data is in the
cache. If so,
        * then it gets that data and populates the card with the
cached data.
        * Then, getForecast() goes to the network for fresh data. If
the network
            * request goes through, then the card gets updated a second
time with the
                * freshest data.
            */
    app.getForecast = function(key, label) {
        var statement = 'select * from weather.forecast where
woeid=' + key;
        var url =
'https://query.yahooapis.com/v1/public/yql?format=json&q=' +
statement;
        // TODO add cache logic here
        if ('caches' in window) {
            /*
                * Check if the service worker has already cached this
city's weather
                    * data. If the service worker has the data, then display
the cached
                        * data while the app fetches the latest data.
                    */
            caches.match(url).then(function(response) {
                if (response) {
                    response.json().then(function updateFromCache(json) {
                        var results = json.query.results;
                        results.key = key;
                        results.label = label;
                        results.created = json.query.created;
                        app.updateForecastCard(results);
                    });
                }
            });
        }
    }

    // Fetch the latest data.
    var request = new XMLHttpRequest();
    request.onreadystatechange = function() {
        if (request.readyState === XMLHttpRequest.DONE) {
            if (request.status === 200) {
                var response = JSON.parse(request.response);
                var results = response.query.results;
                results.key = key;
                results.label = label;
                results.created = response.query.created;
                app.updateForecastCard(results);
            }
        } else {
            // Return the initial weather forecast since no data is
available.
            app.updateForecastCard(initialWeatherForecast);
        }
    }
}

```

```

};

request.open('GET', url);
request.send();
};

// Iterate all of the cards and attempt to get the latest
forecast data
app.updateForecasts = function() {
  var keys = Object.keys(app.visibleCards);
  keys.forEach(function(key) {
    app.getForecast(key);
  });
};

// TODO add saveSelectedCities function here
// Save list of cities to localStorage.
app.saveSelectedCities = function() {
  var selectedCities = JSON.stringify(app.selectedCities);
  localStorage.selectedCities = selectedCities;
};

app.getIconClass = function(weatherCode) {
  // Weather codes:
https://developer.yahoo.com/weather/documentation.html#codes
  weatherCode = parseInt(weatherCode);
  switch (weatherCode) {
    case 25: // cold
    case 32: // sunny
    case 33: // fair (night)
    case 34: // fair (day)
    case 36: // hot
    case 3200: // not available
      return 'clear-day';
    case 0: // tornado
    case 1: // tropical storm
    case 2: // hurricane
    case 6: // mixed rain and sleet
    case 8: // freezing drizzle
    case 9: // drizzle
    case 10: // freezing rain
    case 11: // showers
    case 12: // showers
    case 17: // hail
    case 35: // mixed rain and hail
    case 40: // scattered showers
      return 'rain';
    case 3: // severe thunderstorms
    case 4: // thunderstorms
    case 37: // isolated thunderstorms
    case 38: // scattered thunderstorms
    case 39: // scattered thunderstorms (not a typo)
    case 45: // thundershowers
    case 47: // isolated thundershowers
      return 'thunderstorms';
    case 5: // mixed rain and snow
  }
}

```

```

        case 7: // mixed snow and sleet
        case 13: // snow flurries
        case 14: // light snow showers
        case 16: // snow
        case 18: // sleet
        case 41: // heavy snow
        case 42: // scattered snow showers
        case 43: // heavy snow
        case 46: // snow showers
            return 'snow';
        case 15: // blowing snow
        case 19: // dust
        case 20: // foggy
        case 21: // haze
        case 22: // smoky
            return 'fog';
        case 24: // windy
        case 23: // blustery
            return 'windy';
        case 26: // cloudy
        case 27: // mostly cloudy (night)
        case 28: // mostly cloudy (day)
        case 31: // clear (night)
            return 'cloudy';
        case 29: // partly cloudy (night)
        case 30: // partly cloudy (day)
        case 44: // partly cloudy
            return 'partly-cloudy-day';
    }
};

/*
 * Fake weather data that is presented when the user first
uses the app,
 * or when the user has not saved any cities. See startup code
for more
 * discussion.
 */
var initialWeatherForecast = {
    key: '2459115',
    label: 'New York, NY',
    created: '2016-07-22T01:00:00Z',
    channel: {
        astronomy: {
            sunrise: "5:43 am",
            sunset: "8:21 pm"
        },
        item: {
            condition: {
                text: "Windy",
                date: "Thu, 21 Jul 2016 09:00 PM EDT",
                temp: 56,
                code: 24
            },
            forecast: [
                {code: 44, high: 86, low: 70},

```

```

        {code: 44, high: 94, low: 73},
        {code: 4, high: 95, low: 78},
        {code: 24, high: 75, low: 89},
        {code: 24, high: 89, low: 77},
        {code: 44, high: 92, low: 79},
        {code: 44, high: 89, low: 77}
    ]
},
atmosphere: {
    humidity: 56
},
wind: {
    speed: 25,
    direction: 195
}
}
};

// TODO uncomment line below to test app with fake data
app.updateForecastCard(initialWeatherForecast);

// TODO add startup code here
app.selectedCities = localStorage.selectedCities;
if (app.selectedCities) {
    app.selectedCities = JSON.parse(app.selectedCities);
    app.selectedCities.forEach(function(city) {
        app.getForecast(city.key, city.label);
    });
} else {
    /* The user is using the app for the first time, or the user
has not
     * saved any cities, so show the user some fake data. A real
app in this
     * scenario could guess the user's location via IP lookup
and then inject
     * that data into the page.
    */
    app.updateForecastCard(initialWeatherForecast);
    app.selectedCities = [
        {key: initialWeatherForecast.key, label:
initialWeatherForecast.label}
    ];
    app.saveSelectedCities();
}

// TODO add service worker code here
if ('serviceWorker' in navigator) {
    navigator.serviceWorker
        .register('./service-worker.js')
        .then(function() { console.log('Service Worker
Registered'); });
}

})();

```

File app.js merupakan file tempat dimana segala fungsi dibuat dan diberikan deskripsi, pada file app.js juga terdapat banyak fungsi yang memiliki kegunaan masing masing, dapat dimisalkan mulai dari *updateForecasts* berupakan fungsi yang berguna untuk me *refresh* data *weather* yang *realtime*, kemudian *butAddCity* untuk fungsi menambah daftar kota yang hendak dilihat data *weathernya*, lalu ada *weather code* yang berfungsi sebagai pendeskripsian kode *weather* pada setiap kondisinya. Kemudian diakhir *source code* juga terdapat pengaturan *service-worker* yang digunakan saat *web server* memasuki mode *offline*. Kemudian dilanjutkan dengan source index.html sebagai berikut:

```
<!DOCTYPE html>
<html>
<head>
    <link rel="manifest" href="/manifest.json">
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <link rel="canonical" href="https://weather-pwa-
sample.firebaseio.com/final/">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Weather PWA</title>
    <link rel="stylesheet" type="text/css"
    href="styles/inline.css">

    <!-- TODO add manifest here -->

</head>
<body>

    <header class="header">
        <h1 class="header_title">Weather PWA</h1>
        <button id="butRefresh" class="headerButton" aria-
label="Refresh"></button>
        <button id="butAdd" class="headerButton" aria-
label="Add"></button>
    </header>

    <main class="main">

        <div class="card cardTemplate weather-forecast" hidden>
            <!-- <button id="butDel" class="cardButton" aria-
label="Delete"></button> -->
            <div class="city-key" hidden></div>
            <div class="card-last-updated" hidden></div>
            <div class="location"></div>
            <div class="date"></div>

            <div class="description"></div>
            <div class="current">
                <div class="visual">
```

```
<div class="icon"></div>
<div class="temperature">
    <span class="value"></span><span
class="scale">°F</span>
</div>
<div class="description">
    <div class="humidity"></div>
    <div class="wind">
        <span class="value"></span>
        <span class="scale">mph</span>
        <span class="direction"></span>°
    </div>
    <div class="sunrise"></div>
    <div class="sunset"></div>
</div>
<div class="future">
    <div class="oneday">
        <div class="date"></div>
        <div class="icon"></div>
        <div class="temp-high">
            <span class="value"></span>°
        </div>
        <div class="temp-low">
            <span class="value"></span>°
        </div>
    </div>
    <div class="oneday">
        <div class="date"></div>
        <div class="icon"></div>
        <div class="temp-high">
            <span class="value"></span>°
        </div>
        <div class="temp-low">
            <span class="value"></span>°
        </div>
    </div>
    <div class="oneday">
        <div class="date"></div>
        <div class="icon"></div>
        <div class="temp-high">
            <span class="value"></span>°
        </div>
        <div class="temp-low">
            <span class="value"></span>°
        </div>
    </div>
    <div class="oneday">
        <div class="date"></div>
        <div class="icon"></div>
        <div class="temp-high">
            <span class="value"></span>°
        </div>
        <div class="temp-low">
            <span class="value"></span>°
        </div>
    </div>
    <div class="oneday">
        <div class="date"></div>
        <div class="icon"></div>
        <div class="temp-high">
            <span class="value"></span>°
        </div>
        <div class="temp-low">
            <span class="value"></span>°
        </div>
    </div>
</div>
```

```

        </div>
    </div>
    <div class="oneday">
        <div class="date"></div>
        <div class="icon"></div>
        <div class="temp-high">
            <span class="value"></span>°
        </div>
        <div class="temp-low">
            <span class="value"></span>°
        </div>
    </div>
    <div class="oneday">
        <div class="date"></div>
        <div class="icon"></div>
        <div class="temp-high">
            <span class="value"></span>°
        </div>
        <div class="temp-low">
            <span class="value"></span>°
        </div>
    </div>
    <div class="oneday">
        <div class="date"></div>
        <div class="icon"></div>
        <div class="temp-high">
            <span class="value"></span>°
        </div>
        <div class="temp-low">
            <span class="value"></span>°
        </div>
    </div>
</main>

<div class="dialog-container">
    <div class="dialog">

        <div class="dialog-title">Add new city</div>
        <div class="dialog-body">
            <select id="selectCityToAdd">
                <!-- Values map to Yahoo Weather API Where On Earth
Identifiers (WOEIDs). -->

https://developer.yahoo.com/weather/documentation.html#req
                <option value="2357536">Austin, TX</option>
                <option value="2367105">Boston, MA</option>
                <option value="2379574">Chicago, IL</option>
                <option value="2459115">New York, NY</option>
                <option value="2475687">Portland, OR</option>
                <option value="2487956">San Francisco, CA</option>
                <option value="2490383">Seattle, WA</option>
                <option value="1047592">Kendal, INA</option>
                <option value="56000422">Tegal, INA</option>
                <option value="1048324">Semarang, INA</option>
            </select>
        </div>
    </div>
</div>

```

```

        </select>
    </div>
    <div class="dialog-buttons">
        <button id="butAddCity" class="button">Add</button>
        <button id="butAddCancel" class="button">Cancel</button>
    </div>
</div>

<div class="loader">
    <svg viewBox="0 0 32 32" width="32" height="32">
        <circle id="spinner" cx="16" cy="16" r="14"
fill="none"></circle>
    </svg>
</div>

<!-- Uncomment the line below when ready to test with fake
data -->
<script src="scripts/app.js" async></script>
<div id="footer">
    <center>
        Modul 5 Praktikum MDP 2018 - Created by : Agus dan
        Saskia (Kelompok 37)
        Copyright © 2018
    </center>
</div>

</body>
</html>

```

Index.html merupakan file berjenis html yang berisikan pengaturan apa saja yang akan ditampilkan sebagai *interface* yang akan dilihat pertama kali oleh *user*, index.html pada percobaan kali ini berisikan tentang pengaturan *background*, gambar - gambar yang hendak ditampilkan, pengaturan tampilan data dari data yang diambil pada suatu sumber dan mengolahnya, dan pada percobaan tugas, diambil data dari *Yahoo Weather API* yang bersumber dari <https://developer.yahoo.com/weather/documentation.html#req> yang kemudian kita olah datanya dan menampilkannya dengan *template* yang sudah disetting sebelumnya, pada dasarnya penggunaan *weather API* ini diberikan kode lokasi untuk setiap kota di dunia, jadi pada saat diakses kode tersebut, maka lokasinya akan menyesuaikan dengan data *weather* lokasi *Yahoo*, kemudian terdapat *button delete* yang belum bisa difungsikan karena masih dalam tahap pengembangan, yang nantinya akan berfungsi sebagai penghapus data *weather* kota yang dipilih untuk dihapus pada halaman utama. Adapun juga membuat dan mengatur

tampilan footer di index.html ini yang pada percobaan kali ini digunakan sedikit fitur css yang mana file css yang dapat diinput pada folder *scripts*.

Kemudian pada index.html juga terdapat tampilan pada saat ingin menambahkan daftar kota yang hendak dilihat data *weather*nya, diketahui data yang terdapat pada index.html sebanyak 11 kota, adapun pada percobaan tugas diminta untuk menambahkan 3 kota yang ada di Indonesia, oleh karena itu ditambahkan Kendal, Tegal, dan Semarang yang masing-masing memiliki kode lokasi 1047592, 56000422, dan 1048324. Yang nantinya kode lokasi ini lah yang akan menjadi patokan bagi *API Weather Yahoo* dalam memberikan data weather suatu kota. Lalu ada juga source code dari manifest.json sebagai berikut:

```
{
  "name": "Kelompok 37",
  "short_name": "Weather",
  "icons": [
    {
      "src": "images/icons/icon-128x128.png",
      "sizes": "128x128",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-144x144.png",
      "sizes": "144x144",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-152x152.png",
      "sizes": "152x152",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-192x192.png",
      "sizes": "192x192",
      "type": "image/png"
    },
    {
      "src": "images/icons/icon-256x256.png",
      "sizes": "256x256",
      "type": "image/png"
    }
  ],
  "start_url": "/index.html",
  "display": "standalone",
  "background_color": "#7f7fbf",
  "theme_color": "#ae5daf"
}
```

Manifest.json merupakan file berekstensi json, yaitu Bahasa pemrograman kekinian yang digunakan, dan butuh *library* penerjemah untuk menerjemahkan Bahasa json. File manifest.json pada proyek kali ini berfungsi sebagai file yang membuat suatu tampilan pada mobile saat *icon* pada *homescreen* di tekan. Karena pada dasarnya proyek ini merupakan proyek html yang diperuntukan untuk web,

namun agar tampilan pada mobile bisa lebih di tolerir, maka fungsi manifest.json ini lah yang berperan dalam pembuatan *splashscreen* awal program saat dibuka di *mobile* serta memberikan kemampuan untuk mengontrol bagaimana aplikasi terlihat oleh *user*.

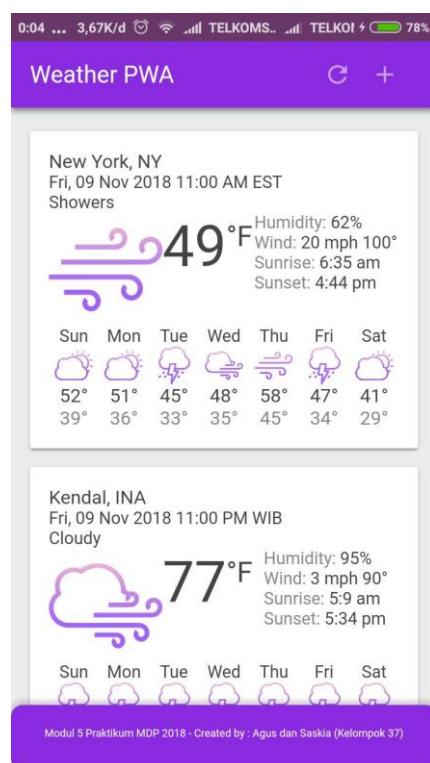
Pada *source code* diatas, terdapat beberapa pendefinisian, diawali dengan “*name*” yang mana berfungsi sebagai nama yang akan ditampilkan pada tampilan awal *splashscreen* program, kemudian “*short_name*” berfungsi sebagai nama yang diberikan pada *homescreen* program, “*theme_color*” berfungsi sebagai pemberi warna utama pada halaman dan untuk kode warna yang digunakan adalah #ae5daf yang mana berwarna werah pucat, dan “*background_color*” juga merupakan tampilan awal yang mana backgroundnya memiliki jenis *style color* atau berupa warna putih berkode #7f7fbf, yang pada percobaan kali ini digunakan warna sebagai background.

Untuk “*display*” diberikan jenis “*standalone*” agar jenis tampilan tidak seperti web browser dan lebih mengarah pada tampilan aplikasi *mobile* sungguhan, jika *user* ingin tampilan sesuai dengan web browser, bisa mengubah jenis *display* menjadi “*browser*”. Lalu “*start_url*” berfungsi sebagai halaman mana yang akan diakses pertama oleh aplikasi.

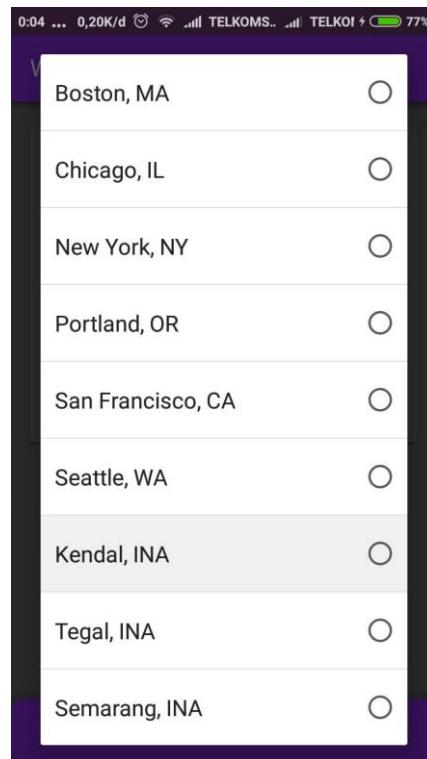
Dan yang terakhir adalah “*icons*” yang berfungsi untuk mensettings icon yang akan dipakai aplikasi, pada proyek ini masing-masing icon dipersiapkan dengan ditentukan “*src*” atau asal alamat icon tersebut, “*sizes*” untuk berapa besar ukuran icon yang digunakan, dan “*type*” merupakan tipe dari gambar yang digunakan.



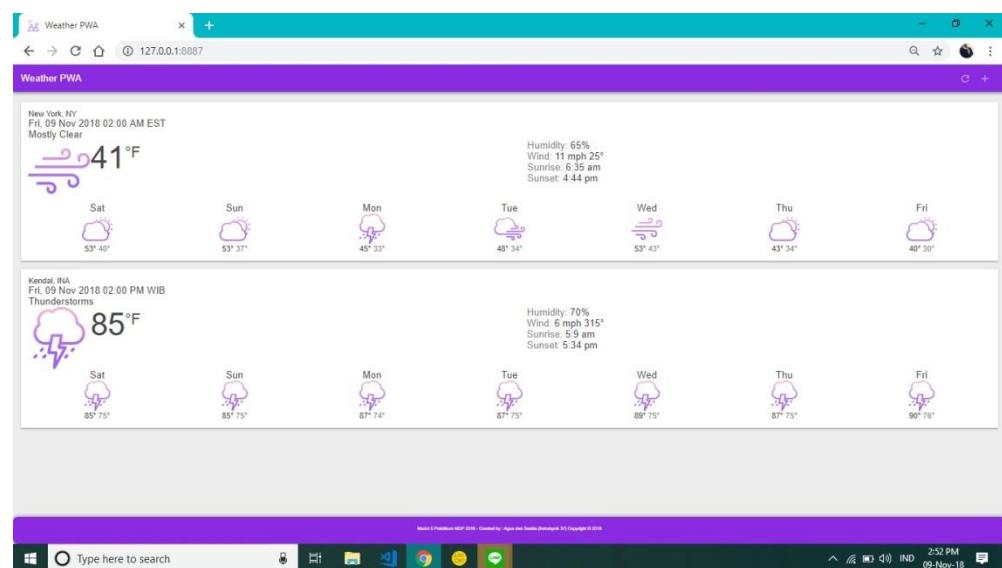
Gambar 6.28 Tampilan *splash screen* aplikasi pada *mobile*



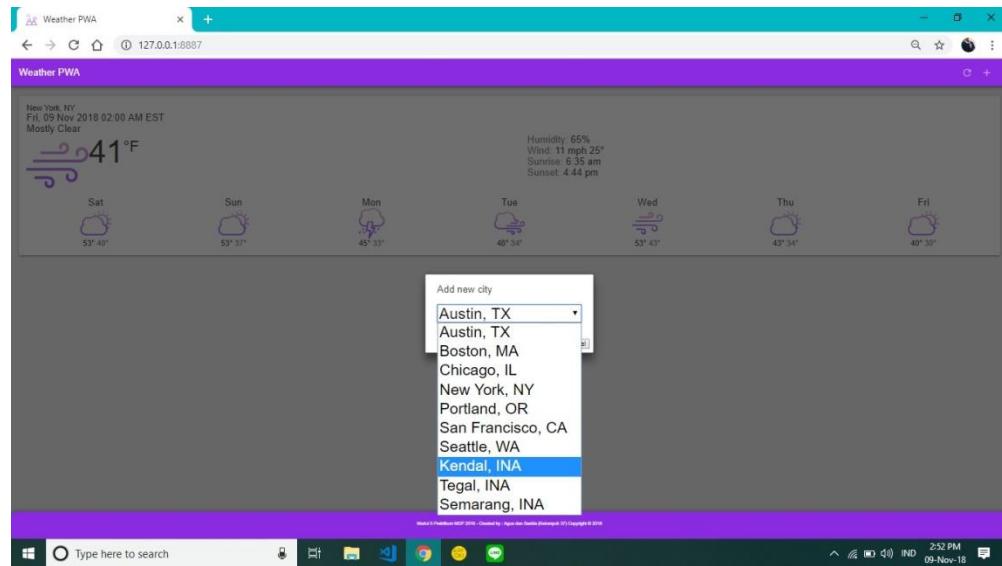
Gambar 6.29 Tampilan data *weather real time* beserta *footer*



Gambar 6.30 Tampilan pilihan kota



Gambar 6.31 Tampilan data *weather real time* pada browser



Gambar 6.32 Tampilan pilihan kota

(Link github tugas: <https://github.com/saskiaugust/kel37-pwa2>)

6.7 Kesimpulan

1. Index.html berfungsi sebagai file yang bertanggung jawab atas sebagal proses pembentukan *interface* utama.
2. Manifest.json merupakan file yang berfungsi membuat tampilan proyek pada PWA terlihat lebih baik dan tidak berbasis web ketika ditampilkan lewat *mobile*.
3. App.js merupakan file yang bertanggung jawab dalam teknis menampilkan data pada proyek PWA.
4. Penggunaan PWA dalam pembuatan aplikasi *mobile* tergolong lebih ringan dan dinamis.
5. Aplikasi berbasis PWA cenderung lebih mudah dalam pengelolaan data pada aplikasi PWA sama seperti basis web.
6. Dalam pembuatan aplikasi PWA, dapat menggunakan fitur *service-worker* jika hendak menjalankan aplikasi secara *offline*.
7. Kostumisasi aplikasi berbasis PWA cenderung lebih mudah karena menggunakan basis *Webpage* (php/html/dll).
8. *Cache Storage* dapat digunakan kembali dalam menampilkan *interface* jika aplikasi PWA sedang tidak terhubung koneksi internet.
9. Pengembangan aplikasi PWA saat ini sudah dapat diintegrasikan dengan berbagai jenis *API (Application Programming Interface)*.
10. Aplikasi berbasis PWA didukung oleh *Google Firebase* dalam hal *deploying* terhadap *mobile*, sehingga dapat lebih membuat tampilan pada *mobile* terlihat baik.

BAB VII

SWIFT FOR iOS

7.1 Tujuan

1. Praktikan mampu mengenal bahasa pemrograman *swift*
2. Praktikan mampu membuat program sederhana menggunakan Xcode
3. Praktikan dapat mengetahui fitur-fitur yang ada pada Xcode
4. Praktikan mampu membuat aplikasi yang dapat jalan di semua perangkat iPhone
5. Praktikan mampu menggunakan *segue*
6. Praktikan mampu menggunakan *constraints*

7.2 Dasar Teori

7.2.1 Pengenalan iOS

iOS adalah sistem operasi perangkat bergerak yang dikembangkan dan didistribusikan oleh Apple Inc. Sistem operasi ini pertama diluncurkan tahun 2007 untuk iPhone dan iPod Touch, dan telah dikembangkan untuk mendukung perangkat Apple lainnya seperti iPad dan Apple TV. Tidak seperti Windows Phone (Windows CE) Microsoft dan Android Google, Apple tidak melisensikan iOS untuk dapat diinstal di perangkat keras non-Apple. Pada 12 September 2012, App Store Apple berisi lebih dari 700.000 aplikasi iOS, yang secara kolektif telah diunduh lebih dari 30 miliar kali. SO ini memiliki pangsa pasar 14,9% untuk unit sistem operasi perangkat bergerak telefon cerdas yang dijual pada kuartal ketiga 2012, terbanyak setelah Android Google. Pada bulan Juni 2012, iOS mencakup 65% konsumsi data web perangkat bergerak (termasuk di iPod Touch dan iPad). Pada pertengahan 2012, terdapat 410 juta perangkat bergerak yang diaktifkan. Menurut Apple pada tanggal 12 September 2012, 400 juta perangkat bergerak iOS telah dijual sepanjang bulan Juni 2012.

Antarmuka pengguna iOS didasarkan pada konsep manipulasi langsung menggunakan gerakan multisentuh. Elemen kontrol antarmukanya meliputi slider, switch, dan tombol. Interaksi dengan SO ini mencakup gerakan seperti geser, sentuh, jepit, dan jepit buka, masing - masing memiliki arti tersendiri dalam konteks iOS dan antarmuka multi sentuhnya. Akselerometer internalnya dipakai oleh sejumlah aplikasi agar bisa merespon terhadap pengguncangan alat (misalnya membatalkan tindakan) atau memutaranya dalam tiga dimensi (misalnya beralih dari mode potret ke lanskap).iOS diturunkan dari OS X, yang memiliki fondasi Darwin dan karena itu iOS merupakan sistem operasi Unix.



Gambar 7.1 iOS 10

(Sumber : Wikipedia.org)

7.2.2 Swift

Swift adalah bahasa pemrograman objek fungsional untuk pengembangan iOS dan OS X yang dibuat oleh Apple. Didesain untuk berdampingan dengan Objective-C dan menghindari program dari kode yang salah. Swift diperkenalkan oleh Apple pada acara tahunan WWDC (*World Wide Developer Conference*) 2014. Bahasa ini dibangun dengan compiler LLVM yang termasuk dalam Xcode 6 beta. Sebuah buku berisi 500 halaman petunjuk, *The Swift Programming Language*, dirilis bersamaan dengan WWDC, dan tersedia secara gratis di *iBooks*. Pada tanggal 2 Juni 2014, aplikasi WWDC menjadi yang pertama dirilis ke publik yang ditulis dengan Swift. Pada tahun 2014 juga, Swift diupdate menjadi Swift 1.2 yang membawa beberapa perbaikan. Pada tahun 2015 diluncurkan Swift 3 sebagai pengembangan lebih lanjut dari Swift 1.2. Pada tahun tersebut pula Swift menjadi Bahasa pemrograman *open source* dan tersedia dibawah *Apache Liscence*. Pada tahun 2016 bersamaan dengan diluncurnya MacOS Sierra, iOS 10, dan WatchOS 3, Swift kembali diupdate menjadi Swift 3 dengan IDE nya menjadi XCode



Gambar 7.2 Logo Swift

(Sumber : Wikipedia.org)

7.2.3 Xcode

Xcode adalah *IDE* yang lengkap untuk membangun aplikasi MacOS, iOS, WatchOS, dan tvOS, dengan adanya Xcode, sebuah aplikasi yang dapat membantu pengembang / developer dapat melakukan hal lebih dari sebelumnya yaitu lebih cepat, efisien, dan aplikasi yang dihasilkan berkualitas tinggi. Xcode dapat otomatis mengkonfigurasi aplikasi kita untuk menggunakan layanan terbaru Apple dan mengelola banyak gambar kita dalam katalog asset terpadu, dan membantu kita dalam merancang sebuah aplikasi yang terlihat cantik pada iOS, watchOS, tvOS atau macOS. Dengan adanya Xcode pengembang sangat mudah untuk membuat aplikasi dan menjalankan tes/simulasi, menganalisis kode, dan memantau kinerja yang dilakukan.

Saat ini versi Xcode yang terbaru adalah Xcode 8.0 yang sudah terintegrasi dengan SDK dan Simulator dari seluruh OS yang dimiliki oleh Apple. Xcode tersedia gratis bagi pengguna macOS dan dapat diunduh dari Mac App Store.



Gambar 7.3 Logo Xcode

(Sumber : Modul Praktikum MDP 2016)

7.3 Langkah Kerja

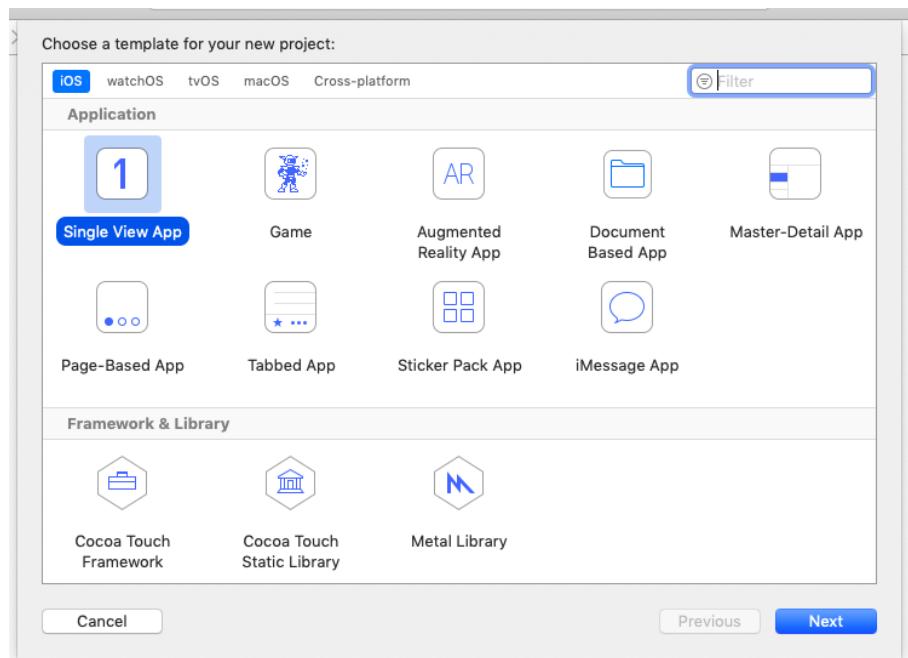
Pada percobaan ini akan dibuat aplikasi *Random Number Generator* dengan melakukan *Login* pada halaman pertama. Aplikasi ini menggunakan metode *random number generator*, memakai fungsi dalam Xcode.

1. Membuka aplikasi Xcode



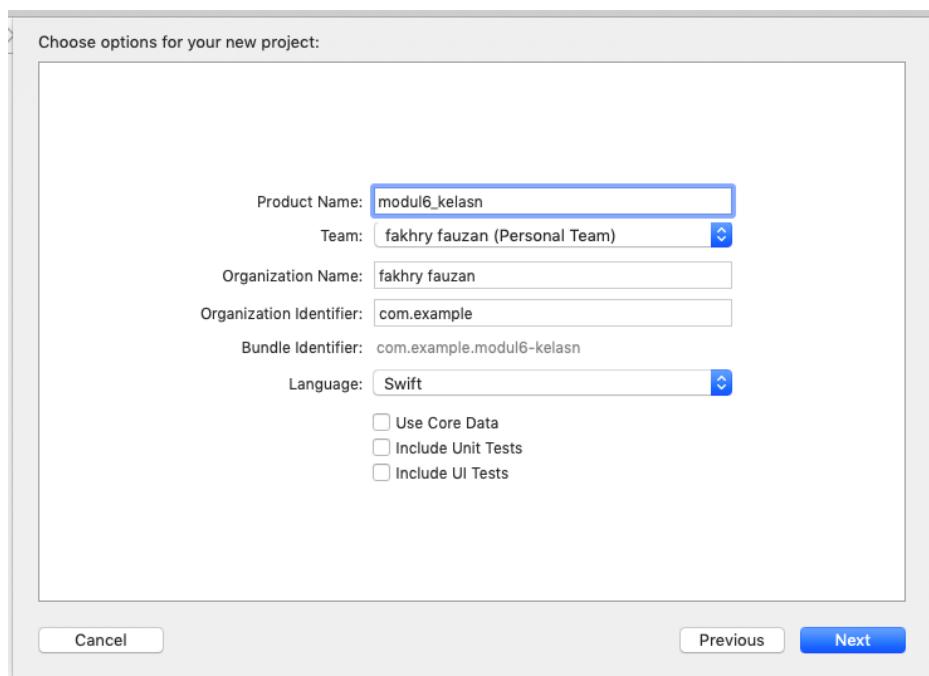
Gambar 7.4 Tampilan Awal Xcode

2. Pilih *Create a new Xcode project*
 - a. Memilih *template* untuk *project* baru, dalam hal ini pilih *Single View Application*



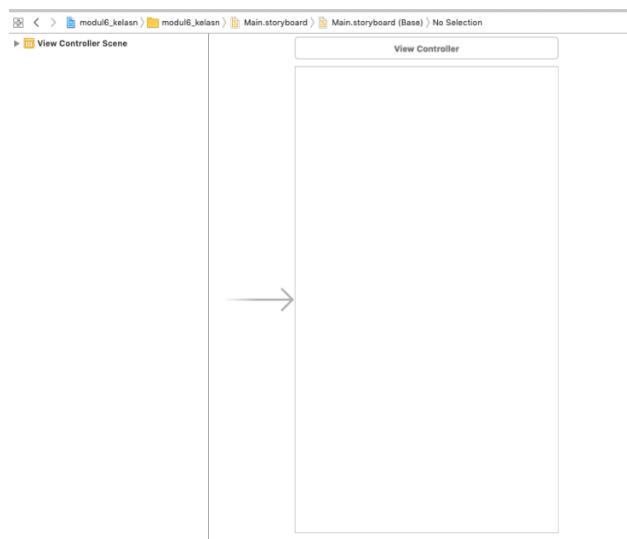
Gambar 7.5 Memilih *Single View App*

- b. Memberikan nama *project* yang akan dibuat, lalu pilih *device iPhone*



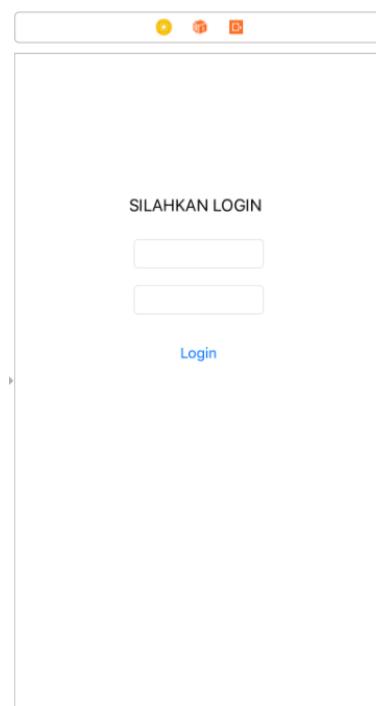
Gambar 7.6 Memberi Nama *Project*

c. Pilih Main.storyboard



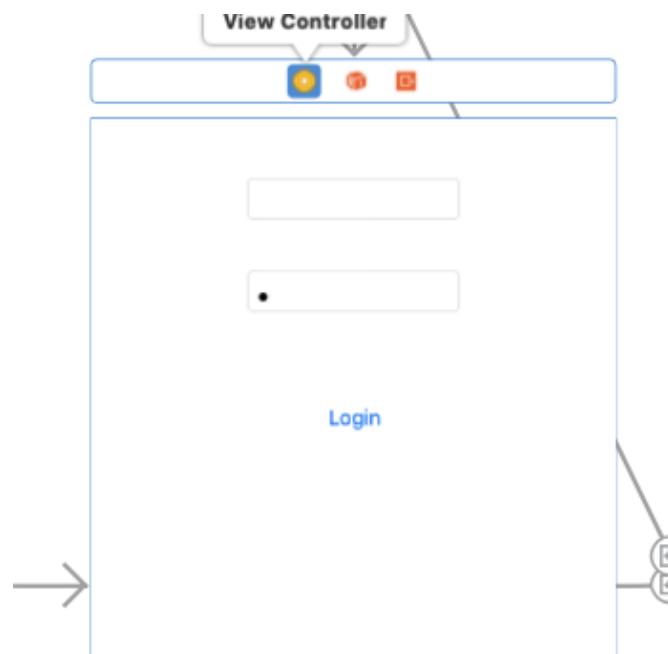
Gambar 7.7 Memilih *Main Storyboard*

3. *Drag and Drop Label, Text Field, dan Button* sesuai kebutuhan dan estetika



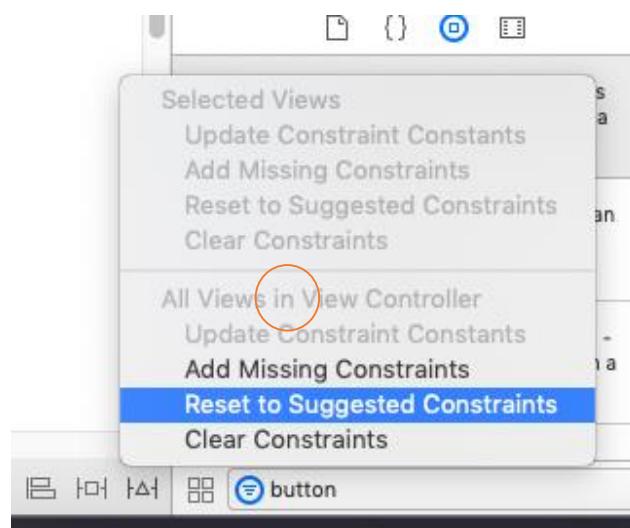
Gambar 7.8 Membuat Tampilan Aplikasi

4. Tekan *ViewController* pada *View*



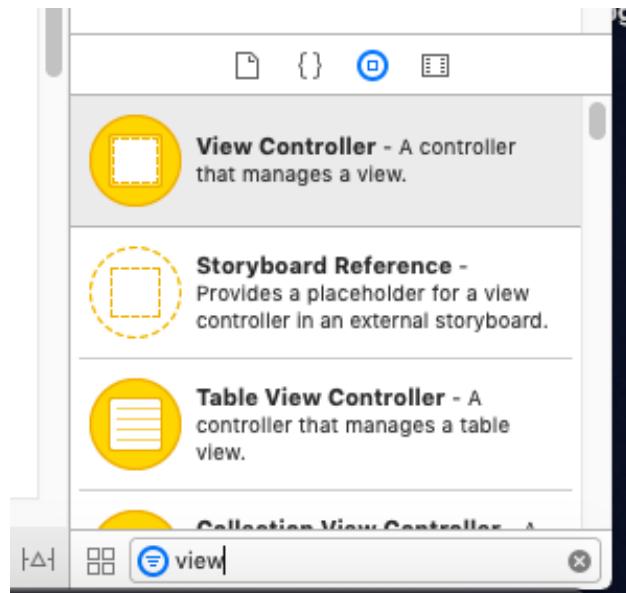
Gambar 7.9 Menekan ViewController pada View

5. Tekan *Resolve Auto Layout Issues*, dan tekan *Reset to Suggested Constraints* untuk melakukan *constraints* otomatis



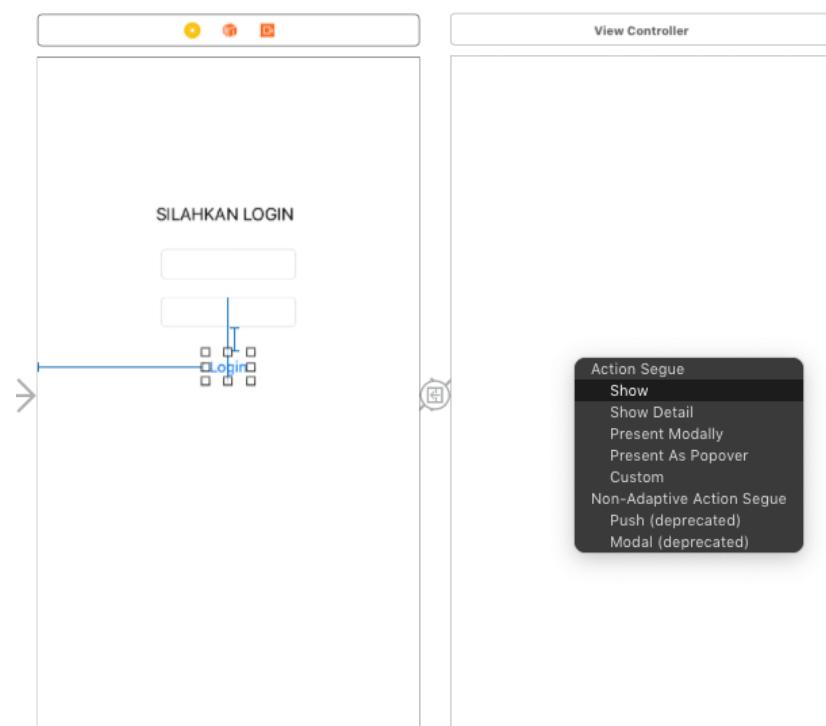
Gambar 7.10 Melakukan *Constraint* Otomatis

6. Tambahkan *view*



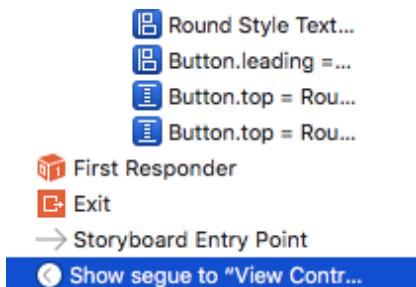
Gambar 7.11 Menambahkan *Segue*

7. Buat *segue*, *segue* berfungsi sebagai arahan disaat pemindahan *view*.

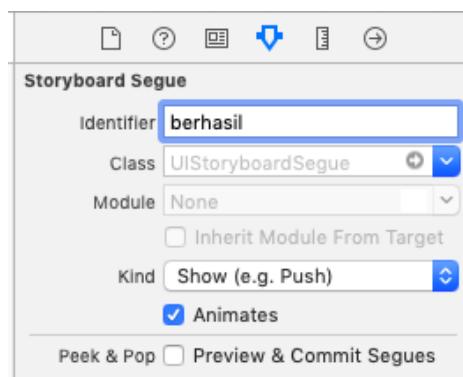


Gambar 7.12 Membuat *Segue*

8. Berinama *segue* untuk *view* pertama

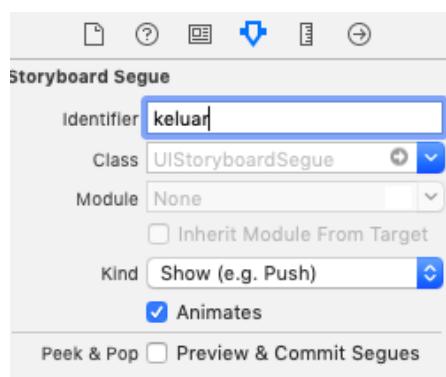


Gambar 7.13 Show Segue



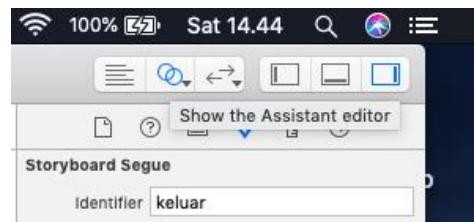
Gambar 7.14 Memberi nama *view* pertama

9. Begitu pula pada *view* kedua



Gambar 7.15 Memberi nama *view* kedua

10. Tekan *show the assistant editor*

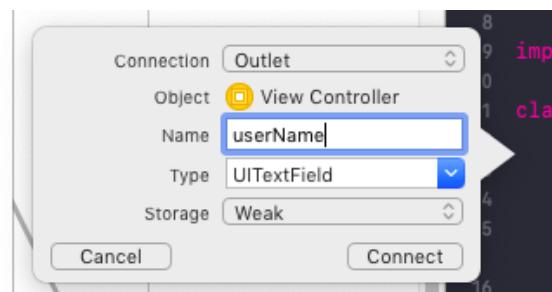


Gambar 7.16 Menekan *Show the Assistant Editor*

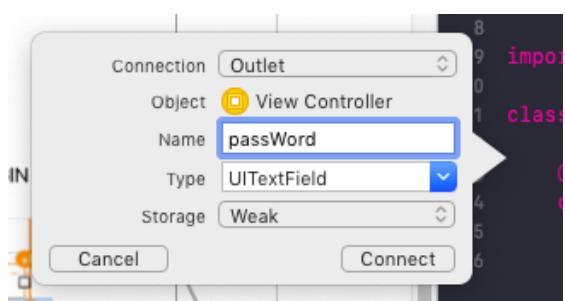
11. Koneksikan semua *widget* dengan halaman viewcontroller.swift. Masukkan *button*, dan *textfield* ke *source code* dengan *drag* sambil menahan tombol *control* dari *main storyboard* ke halaman kode. Lalu beri nama *variable* bagi setiap objek.



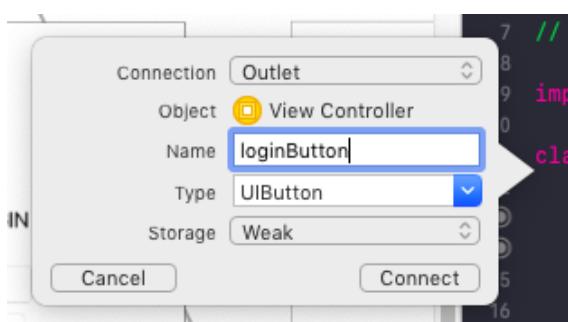
Gambar 7.17 Menghubungkan Widget dengan halaman viewcontroller.swift



Gambar 7.18 Memberi nama *TextField1*

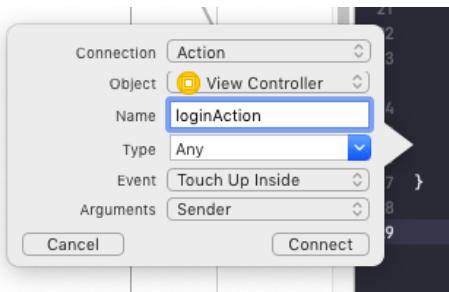


Gambar 7.19 Memberi nama *TextField2*



Gambar 7.20 Memberi nama *Button*

12. Pada *LoginAction* pilih *Action* pada *Connection*



Gambar 7.21 Memberi nama *Action*

Tambahkan *code* berikut di dalam *loginAction()* ,

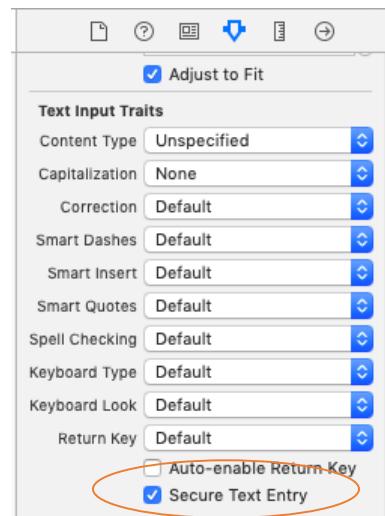
```

let user = "mdp"
let pass = "123456"

        if (userName.text == user && passWord.text == pass) {
            performSegue(withIdentifier: "berhasil",
sender: self)
        }

        else if (userName.text == "" && passWord.text == "") {
            let alert = UIAlertController(title: "Login Gagal", message: "Username/Password Kosong", preferredStyle: UIAlertControllerStyle.alert)
            alert.addAction(UIAlertAction(title: "OK", style: UIAlertActionStyle.cancel, handler: nil))
            self.present(alert, animated: true)
        }
        else {
            let alert = UIAlertController(title: "Login Gagal", message: "Username/Password Salah", preferredStyle: UIAlertControllerStyle.alert)
            alert.addAction(UIAlertAction(title: "OK", style: UIAlertActionStyle.cancel, handler: nil))
            self.present(alert, animated: true)
        }
    }
}

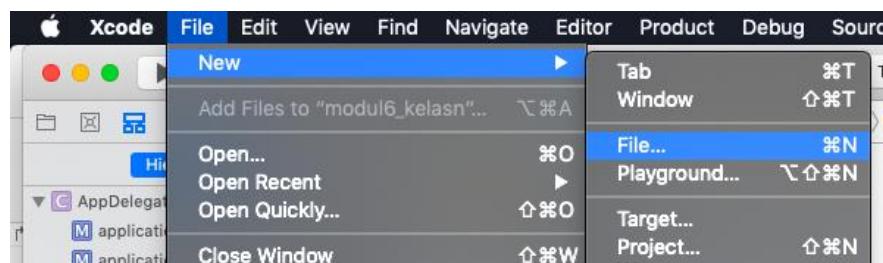
```



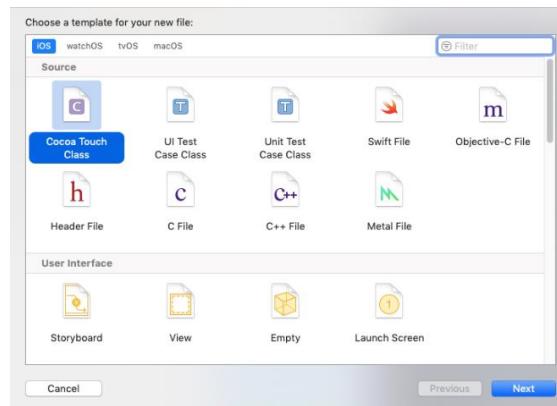
Gambar 7.22 Secure Text Entry

Klik *textfield* ke 2 klik *show the attributes inspector*, dan centang *Secure Text Entry*, guna men-dots *text* yang masuk.

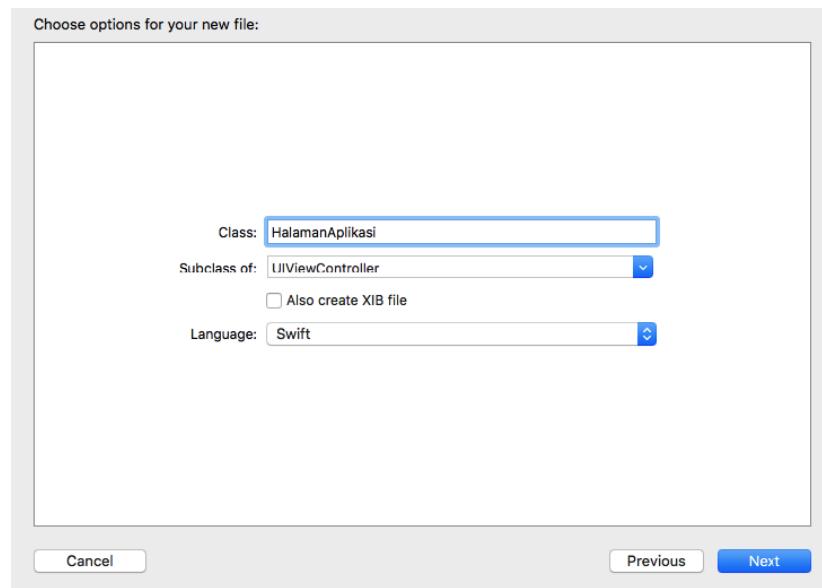
13. Buat file baru, Pilih *Swift File*, beri nama HalamanAplikasi, Ganti *Foundation* menjadi *UIKit* di HalamanAplikasi.swift



Gambar 7.23 New File



Gambar 7.24 Memilih Cocoa Touch Class

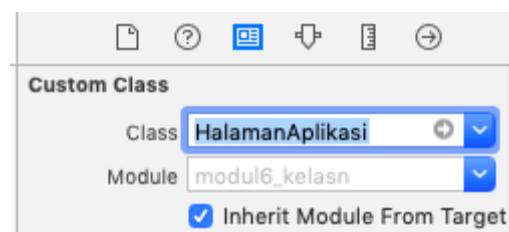


Gambar 7.25 Memberi Nama HalamanAplikasi

14. Pilih *view* ke dua dan pilih *class* HalamanAplikasi



Gambar 7.26 Memilih HalamanAplikasi (1)



Gambar 7.27 Memilih HalamanAplikasi (2)

15. Pilih HalamanAplikasi.swift



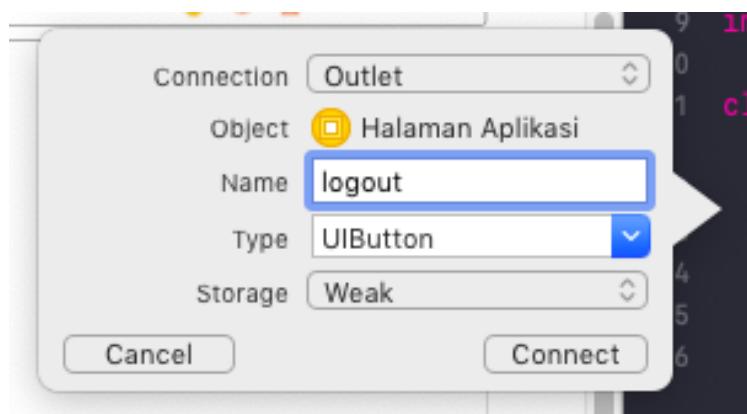
Gambar 7.28 Memilih HamalanAplikasi.swift

Dan pastikan yang terbuka adalah HalamanAplikasi.swift

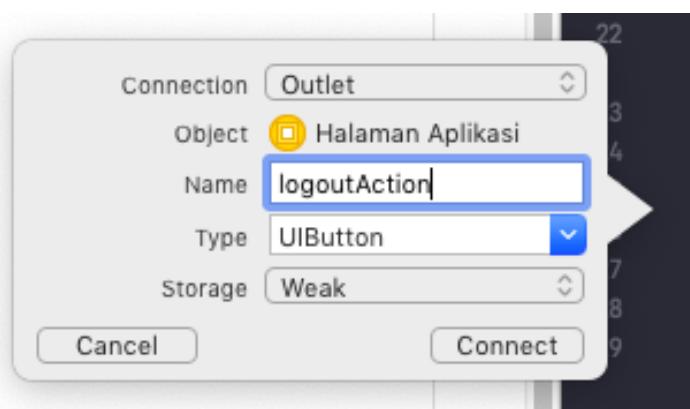
16. Tambahkan *Function* pada *Logout*



Gambar 7.29 Memasukkan Fungsi Tombol *Logout* ke *Code*



Gambar 7.30 Membuat *button logout*



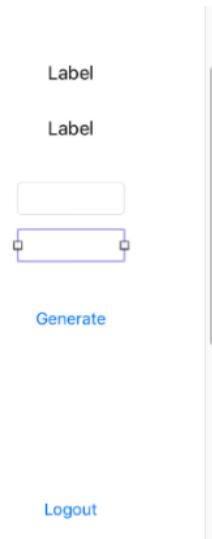
Gambar 7.31 Membuat *LogoutAction*

Di *function logoutAction* tambahkan *code* berikut

```
        performSegue(withIdentifier: "keluar", sender:
self)
```

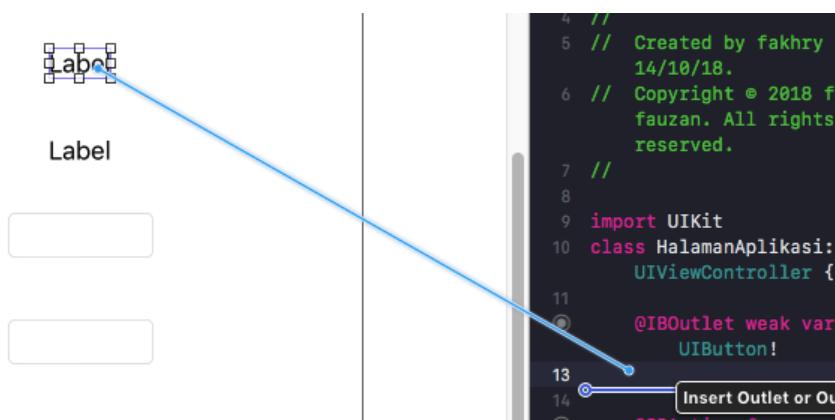
17. *Clean* dan *run program*

18. *Drag and Drop Label, Text Field, dan Button* sesuai kebutuhan dan estetika

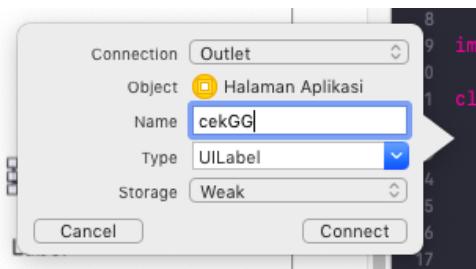


Gambar 7.32 Membuat Tampilan Halaman *Generate Number*

19. Masukkan *label*, *button*, dan *textfield* ke *source code* dengan *drag* sambil menahan tombol *control* dari *main storyboard* ke halaman kode. Lalu beri nama *variable* bagi setiap objek.



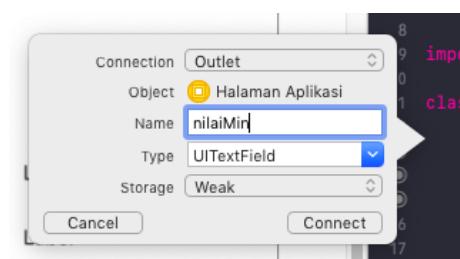
Gambar 7.33 Memasukkan komponen ke dalam *source code*



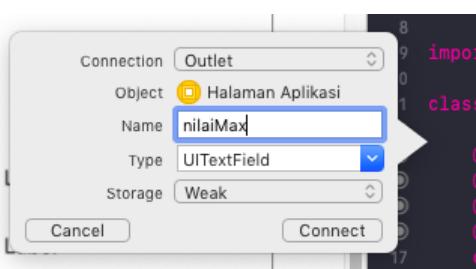
Gambar 7.34 memberi nama variabel cekGG



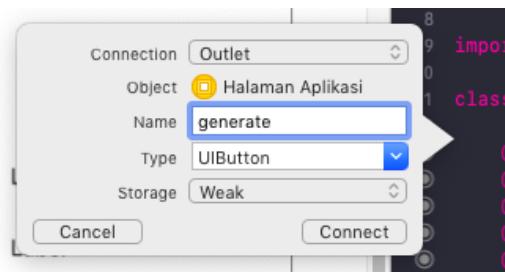
Gambar 7.35 memberi nama variabel hasil



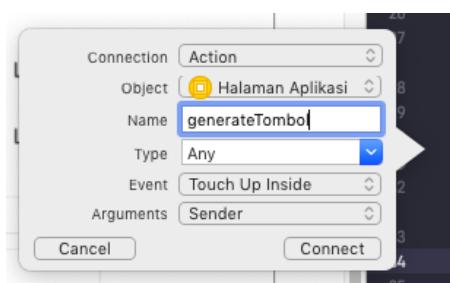
Gambar 7.36 memberi nama variabel nilaiMin



Gambar 7.37 memberi nama variabel nilaiMax



Gambar 7.38 memberi nama variabel generate



Gambar 7.39 memberi nama action generateTombol

Keterangan :

Label pertama = cekGG

Label kedua = hasil

Textfield pertama = nilaiMin

Textfield kedua = nilaiMax

Button = generate

Button Action = generateTombol

20. Masukkan *source code* seperti di bawah ini

```
var hasilakhir = Int()
override func viewDidLoad() {
    super.viewDidLoad()
    nilaiMin.text = "0"
    nilaiMax.text = "100"
    hasil.text = "Hasil"
    cekGG.text = "Genap / Ganjil ?"
}
```

21. Buat *function* berikut

```
func ganjilgenap(){
    if hasilakhir % 2 == 0 {
        cekGG.text = "Genap"
    }
}
```

```
        else{
            cekGG.text = "Ganjil"
        }
    }
```

22. Tambahkan *code* berikut di dalam generateTombol()

```
let nilai = UInt32((Int(nilaiMax.text!)!
Int(nilaiMin.text!)!) + 1)
let random = arc4random_uniform(nilai)
hasilakhir = Int(random)+Int(nilaiMin.text!)!
hasil.text = String(hasilakhir);
ganjilgenap()
```

23. *Run program* yang telah dibuat pada *Simulator*

SILAHKAN LOGIN

Login

Gambar 7.40 Tampilan *Login*

Carrier 3:30 PM

Ganjil

51

[Generate](#)

[Logout](#)

Gambar 7.41 Tampilan *Number Generator*

7.4 Hasil Percobaan dan Analisa

```
let user = "mdp"
let pass = "123456"

    if (userName.text == user && passWord.text == pass) {
        performSegue(withIdentifier: "berhasil", sender:
self)
    }

    else if (userName.text == "" && passWord.text == "") {
        let alert = UIAlertController(title: "Login Gagal",
message: "Username/Password Kosong", preferredStyle:
UIAlertControllerStyle.alert)
        alert.addAction(UIAlertAction(title: "OK", style:
UIAlertActionStyle.cancel, handler: nil))
        self.present(alert, animated: true)
    }
    else {
        let alert = UIAlertController(title: "Login Gagal",
message: "Username/Password Salah", preferredStyle:
UIAlertControllerStyle.alert)
        alert.addAction(UIAlertAction(title: "OK", style:
UIAlertActionStyle.cancel, handler: nil))
        self.present(alert, animated: true)
    }
}
```

Source code diatas merupakan *source code Login*, dimana pada *source code* tersebut berisi *username* “mdp” dan *password* “123456” untuk melakukan proses *login*. Jika autentikasi berhasil maka akan dilakukan *segue* “berhasil” yang mana merupakan perpindahan halaman dari halaman *Login* ke HalamanAplikasi. Ketika *username* dan *password* yang dimasukkan salah maka akan muncul pesan “*Username/Password Salah*” dan ketika *username* dan *password* yang tidak dimasukkan maka akan muncul pesan “*Username/Password Kosong*”.

```
PerformSegueWithIdentifier("keluar", sender: self)
```

Source code diatas merupakan *source code Logout*. Pada *source code* ini jika *user* menekan tombol *Logout* maka akan dilakukan *segue* “*keluar*” yang mana halaman akan berpindah dari HalamanAplikasi ke halaman *Login*.

```
var hasilakhir = Int()
override func viewDidLoad() {
    super.viewDidLoad()
    nilaiMin.text = "0"
```

```

nilaiMax.text = "100"
hasil.text = "Hasil"
cekGG.text = "Genap / Ganjil ?"
}

```

Pada *source code* diatas, diberikan nilai awal pada *label* yang ada pada HalamanAplikasi. Nilai minimal (nilaiMin) diatur dengan nilai awal 0 dan untuk nilai maksimal (nilaiMax) diatur dengan nilai awal 100. Hasil memiliki nilai awal “Hasil” dan cekGG memiliki nilai awal “Genap / Ganjil?”

```

let nilai = UInt32((Int(nilaiMax.text!)! - Int(nilaiMin.text!)!)+1)
    let random = arc4random_uniform(nilai)
    hasilakhir = Int(random)+Int(nilaiMin.text!)!
    hasil.text = String(hasilakhir);
    ganjilgenap()
}

```

Untuk melakukan *generate* nomor menggunakan *source code* diatas. Ketika tombol *generate* ditekan maka akan diambil nilai dari nilaiMax – nilaiMin +1. Agar nilai yang dihasilkan menjadi *random* dengan menggunakan `arc4random_uniform` dari nilai yang telah didapat sebelumnya. Hasilakhir berupa nomor *random* dijumlahkan dengan *integer* dari nilaiMin tadi. Kemudian hasilakhir dimasukkan pada label hasil. Kemudian dilakukan *method* `ganjilgenap()`.

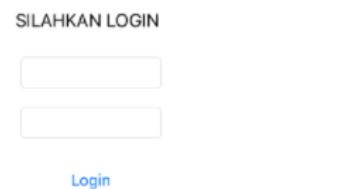
```

func ganjilgenap(){
    if hasilakhir % 2 == 0 {
        cekGG.text = "Genap"
    }
    else{
        cekGG.text = "Ganjil"
    }
}

```

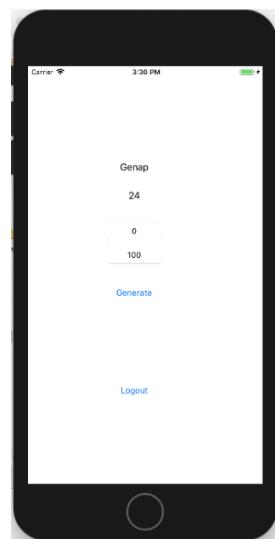
Method `ganjilgenap()` digunakan untuk melakukan *set* label dengan nilai “Genap” atau “Ganjil” sesuai dengan nilai dari hasil akhir di mod 2. Jika hasilakhir mod 2 == 0 maka cekGG akan bernilai “Genap” dan jika hasilakhir mod 2 == 1 maka cekGG akan bernilai “Ganjil”.

Berikut adalah hasil tangkapan layar mengenai aplikasi ini,



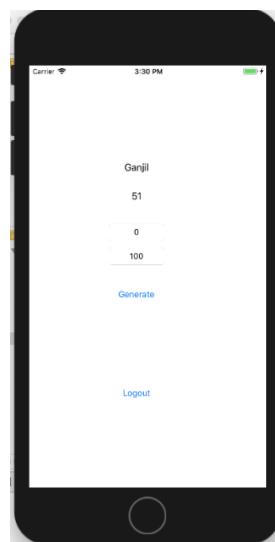
Gambar 7.42 Tampilan Program pada Halaman Login

Gambar diatas merupakan tampilan halaman *login*, dimana *Textfield* pertama (yang atas) digunakan untuk memasukkan *username* dan *Textfield* kedua (yang bawah) digunakan untuk memasukkan *password*. Setelah memasukkan *username* dan *password* kemudian menekan *button Login* untuk masuk ke Halaman Aplikasi.



Gambar 7.43 Tampilan Program pada Halaman Aplikasi saat genap

Gambar diatas adalah tampilan program saat melakukan *generate random* nomor dari nilai minimalnya 0 dan nilai maksimalnya 100. Nilai *random* yang keluar adalah 24. Karena 24 mod 2 adalah 0 maka diatas angka 24 tersebut tertulis “Genap”.



Gambar 7.44 Tampilan Program pada HalamanAplikasi saat ganjil

Gambar diatas adalah tampilan program saat melakukan *generate random* nomor dari nilai minimalnya 0 dan nilai maksimalnya 100. Nilai *random* yang keluar adalah 52. Karena 52 mod 2 adalah 1 maka diatas angka 52 tersebut tertulis “Genap”.

7.5 Kesimpulan

1. Inisialisasi pada Xcode dapat dilakukan dengan melakukan *drag and drop*.
2. Bahasa pemrograman *swift* pada iMac dan Mac mini memiliki sedikit perbedaan.
3. Untuk menghubungkan antara *segue* satu dengan *segue* lainnya hanya perlu dilakukan menarik *segue* tersebut ke halaman yang ingin dituju.
4. Untuk melakukan pengujian aplikasi menggunakan *simulator*.
5. Untuk menambahkan komponen seperti *Label*, *TextField*, maupun *Button* dapat dilakukan secara *drag and drop*
6. Pemberian nama *segue* dapat dilakukan dengan membuka *View Controller* terlebih dahulu.
7. Sebelum dilakukan *run* program, terlebih dahulu dilakukan *clean*.
8. *Connection Action* digunakan memberikan perintah jika fitur diberi aksi seperti menekan *Button*.

BAB VIII

PENUTUP

8.1. KESIMPULAN

Dari praktikum kali ini dapat disimpulkan bahwa :

1. Praktikum Pemrograman Perangkat Bergerak membahas tentang pengembangan aplikasi pada mobile device berdasarkan IDE dan juga mobile device sasaran seperti Xamarin untuk pengembangan aplikasi cross platform, Android Studio untuk Android device, PWA, dan juga Xcode untuk membangun aplikasi untuk iOS device.
2. Xamarin memiliki kelebihan yaitu ketika user melakukan pemrograman maka aplikasi yang dihasilkan dapat dijalankan tidak hanya pada satu OS saja tetapi bisa juga untuk OS yang lain seperti Android, iOS, dan juga Windows.
3. Xamarin merupakan *platform* untuk pengembangan aplikasi *cross platform* di mana pengembangannya menggunakan bahasa C# dan .NET framework.
4. Software Development Kit diperlukan ketika user ingin melakukan pemrograman menggunakan Android Studio. SDK berfungsi sebagai satu set alat pengembangan yang digunakan untuk mengembangkan aplikasi.
5. Shared Preferences mirip seperti database yang lainnya namun digunakan untuk menyimpan data dengan skala kecil dan sederhana seperti username dan password user.
6. Pada *Progressive Web App (PWA)*, *Service Worker* berfungsi untuk menyimpan *cache* sehingga *project* tetap dapat ditampilkan ketika koneksi mati.
7. Aplikasi berbasis PWA didukung oleh *Google Firebase* dalam hal *deploying* terhadap *mobile*, sehingga dapat lebih membuat tampilan pada *mobile* terlihat baik.
8. Xcode adalah IDE yang lengkap untuk membangun aplikasi MacOS, iOS, WatchOS, dan tvOS. Bahasa pemrograman untuk mengembangkan aplikasi berbasis iOS yaitu bahasa Swift.

8.2. SARAN

1. Modul yang akan digunakan untuk praktikum sebaiknya dicek ulang lagi supaya proses praktikum dapat berjalan lebih lancar.
2. Jam mulai praktikum sering tidak sesuai dengan jadwal, seharusnya kalau memang sudah masuk waktu praktikum disegerakan untuk memulai praktikum agar waktu lebih efisien.
3. Software atau Development Tools praktikum sebaiknya diberikan terlebih dahulu kepada praktikan untuk dipelajari, agar praktikum dapat berjalan lancar dan mengurangi sedikit hambatan.
4. Perlunya kerjasama yang baik dengan teman satu kelompok agar ilmu yang didapat saat praktikum lebih maksimal.

DAFTAR PUSTAKA

- <http://www.jadibaru.com/android/pengenalan-android-studio-2>
- <https://blog.dicoding.com/android-module-5-android-layout/>
- <https://developers.google.com/web/fundamentals/codelabs/your-first-pwapp/?hl=id>
- <https://idcloudhost.com/pengertian-web-server-dan-fungsinya/>
- <https://developers.google.com/web/fundamentals/primers/service-workers/?hl=id>
- <https://developers.google.com/web/fundamentals/web-app-manifest/?hl=id>
- <https://www.codepolitan.com/mengenal-nodejs-5880234fe9ae3>
- [https://id.wikipedia.org/wiki/Npm_\(perangkat_lunak\)](https://id.wikipedia.org/wiki/Npm_(perangkat_lunak))
- <https://rizkimufrizal.github.io/belajar-vue-js/>
- <https://firebase.google.com/docs/hosting/?hl=id>
- <https://idcloudhost.com/pengertian-web-server-dan-fungsinya/>

LAMPIRAN I (Lembar Asistensi)

Laboratorium Rekayasa Perangkat Lunak**Teknik Komputer Universitas Diponegoro**D204 Gedung Kuliah Bersama Fakultas Teknik
Universitas Diponegoro Semarang**LEMBAR ASISTENSI**

PRAKTIKUM PEMROGRAMAN PERANGKAT BERGERAK 2018

Judul Praktikum : Xamarin
 Tanggal Praktikum : 18 Oktober 2018
 Nama Asisten : 1. Raisya Rahmah Noor (21120115140065)
 2. Najakhul Fahmi (21120115120035)
 Nama Praktikan : Agustiawan (21120116120008)
 Kelompok : 37



No.	Hari/Tanggal/Jam	Keterangan	TTD 1	TTD 2
1.		Rev I	Rasy	Mift
2.		Rev II	Rasy	Mift
3.		ACC	Rasy	Mift
4.				

Mengetahui,
Koordinator Praktikum

Pramudya Erviansyah
21120115130055

Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik
Universitas Diponegoro Semarang



LEMBAR ASISTENSI

PRAKTIKUM PEMROGRAMAN PERANGKAT BERGERAK 2018

Judul Praktikum : Android Studio #1
 Tanggal Praktikum : 20 Oktober 2018
 Nama Asisten : 1. Muhammad Adinugroho (21120115130063)
 2. Bima Fajar Setiawan (21120115140088)
 Nama Praktikan : Agustiawan (21120116120008)
 Kelompok : 37



No.	Hari/Tanggal/Jam	Keterangan	TTD 1	TTD 2
1.		Reu I		
2.		Reu II		
3.		Acc		
4.				

Mengetahui,
Koordinator Praktikum

Pramudya Erviansyah
21120115130055

Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik
Universitas Diponegoro Semarang



LEMBAR ASISTENSI

PRAKTIKUM PEMROGRAMAN PERANGKAT BERGERAK 2018

Judul Praktikum : Android Studio #2
 Tanggal Praktikum : 25 Oktober 2018
 Nama Asisten : 1. Pramudya Erviansyah (21120115130055)
 2. Tito Anugerah M. (21120115130049)
 Nama Praktikan : Agustiawan (21120116120008)
 Kelompok : 37



No.	Hari/Tanggal/Jam	Keterangan	TTD 1	TTD 2
1.		AC	✓	✓
2.				
3.				
4.				

Mengetahui,

Koordinator Praktikum

Pramudya Erviansyah
21120115130055

Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik
Universitas Diponegoro Semarang



LEMBAR ASISTENSI

PRAKTIKUM PEMROGRAMAN PERANGKAT BERGERAK 2018

Judul Praktikum : PWA #1
 Tanggal Praktikum : 27 Oktober 2018
 Nama Asisten : 1. Muchammad Dwi C. N. (21120115120039)
 2. Pandu Kent Ellian (21120115120009)
 Nama Praktikan : Agustiawan (21120116120008)
 Kelompok : 37



No.	Hari/Tanggal/Jam	Keterangan	TTD 1	TTD 2
1.		Rev 1	✓	✓
2.		Rev 2	✓	✓
3.		ALC	✓	✓
4.				

Mengetahui,
 Koordinator Praktikum

Pramudya Erviansyah
21120115130055

Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik
Universitas Diponegoro Semarang



LEMBAR ASISTENSI

PRAKTIKUM PEMROGRAMAN PERANGKAT BERGERAK 2018

Judul Praktikum : PWA #2
 Tanggal Praktikum : 1 November 2018
 Nama Asisten : 1. Lian Cahyo Wijaya (21120115120001)
 2. Pradipta Sekar A. P. W. (21120115120030)
 Nama Praktikan : Agustiawan (21120116120008)
 Kelompok : 37



No.	Hari/Tanggal/Jam	Keterangan	TTD 1	TTD 2
1.		Re VI		
2.		ACC		
3.				
4.				

Mengetahui,
 Koordinator Praktikum

Pramudya Erviansyah
 21120115130055

Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik
Universitas Diponegoro Semarang



LEMBAR ASISTENSI

PRAKTIKUM PEMROGRAMAN PERANGKAT BERGERAK 2018

Judul Praktikum : iOS
 Tanggal Praktikum : 3 November 2018
 Nama Asisten : 1. Dhesti Rosytawati R. (21120115120033)
 2. Fakhry Fauzan (21120115140062)
 Nama Praktikan : Agustiawan (21120116120008)
 Kelompok : 37



No.	Hari/Tanggal/Jam	Keterangan	TTD 1	TTD 2
1.		Rev	A	B
2.		ACC	A	B
3.				
4.				

Mengetahui,
 Koordinator Praktikum

Pramudya Erviansyah
21120115130055

Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik
Universitas Diponegoro Semarang



LEMBAR ASISTENSI ASISTENSI TUGAS AKHIR

PRAKTIKUM PEMROGRAMAN PERANGKAT BERGERAK 2018

Judul Tugas Akhir : Kuis Brilian
Nama Asisten : Bima Fajar Setiawan (21120115140088)
Nama Praktikan : Agustiawan (21120116120008)



No.	Hari/Tanggal/Jam	Keterangan	Tanda Tangan
1.	19/11/2018	Perdona	
2.	19/11/2018	ACC	
3.			
4.			

Mengetahui,
Koordinator Praktikum

Pramudya Erviansyah
21120115130055

LAMPIRAN II (Tugas Pendahuluan)

Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik
Universitas Diponegoro Semarang


**REKAYASA
PERANGKAT
LUNAK**

Praktikum Pemrograman Perangkat Bergerak
 BAB III
 Android Studio v.1

Nama : Agustiawan
 NIM : 2112016120008
 Kelompok : 37

90

1. Android Studio menggunakan metode native karena dibangun dengan bahasa pemrograman yang spesifik untuk platform tertentu.
 2. SDK (Software Development Kit) merupakan satu set perktos pengembangan software yang digunakan untuk mengembangkan atau membuat aplikasi untuk paket software tertentu. Software framework, hardware platform, sistem komputer, konsol video game, sistem operasi, atau platform sejenis lainnya.
 Kegunaan pada Android Studio adalah untuk menambahkan fitur-fitur yang dapat kita gunakan pada aplikasi yang akan kita buat seperti Gyroscope, accelerometer, GPS, dan sebagainya.
 3. Gradle adalah build environment sebagai tool untuk menyederhanakan pembangunan proyek secara terstruktur seperti halnya mengkompilasi source code yang kita buat menjadi kode biner dan memaketkan dalam paket biner, menjalankan beberapa test dan unit-unit yang berkaitan dengan keperluan eksekusi development secara otomatis.
 4. Jenis-jenis Layout

- a. Constraint Layout, merupakan tampilan grup yang tersedia di pustaka Constraint Layout.
- b. Linear Layout, otomatis menampilkan komponen-komponen didalamnya secara horizontal atau vertikal.
- c. Relative Layout, merupakan layout yang menampilkan elemen-elemen view dalam posisi yang relatif.
- d. Grid Layout, otomatis memberikan kemudahan dengan mengatur komponen didalamnya ke dalam bentuk grid.
- e. Frame Layout, adalah layout yang poling sekrhona. Layout ini akan membuat komponen yang ada didalamnya menjadi menutup atau saling menutupi satu dengan yang lainnya.

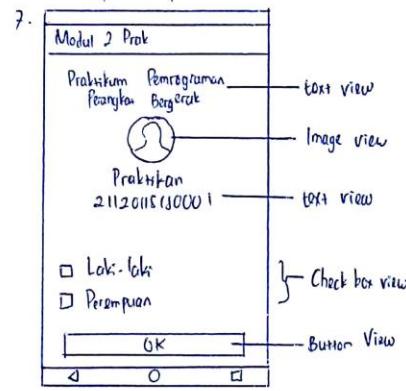
Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik
Universitas Diponegoro Semarang



5. Intent adalah sebuah kelas dalam programming Android yang berfungsi untuk perpindahan halaman. Intent juga merupakan suatu objek yang terdapat dalam suatu activity dimana objek tersebut dapat berkomunikasi dengan activity yang lain. Intent dibagi menjadi 2, yaitu:
 - Explicit Intent, berfungsi untuk mengakses komponen-komponen dalam satu aplikasi yang sama. Misalnya seperti + berpindah Activity.
 - Implicit Intent, berfungsi untuk memanggil fungsi activity yang sudah ada di jaringan internal android seperti dial number, open browser, dan yang lainnya.
6. SQLite merupakan sistem manajemen database relasional yang dibangun dalam sebuah library bahasa pemrograman C. Perbedaan antara SQLite dengan Shared Preferences adalah SQLite berupa file tunggal atau beberapa file saja. Sedangkan shared preferences adalah berisi data privat pada file internal aplikasi dengan penerapan key-value.



- TextView digunakan untuk menampilkan kata / kalimat
- ImageView digunakan untuk menampilkan gambar / ikon
- Button View digunakan untuk menampilkan tombol
- Check box view digunakan untuk memilih satu opsi dari beberapa opsi yang disediakan

Laboratorium Rekayasa Perangkat Lunak

Teknik Komputer Universitas Diponegoro

D204 Gedung Kuliah Bersama Fakultas Teknik
Universitas Diponegoro Semarang



Praktikum Pengembangan Perangkat Lunak
BAB VI
PWA N2

Nama : Agustriawan

NIM : 21120116120008

Kelompok : 37

1. PWA adalah sebuah aplikasi web yang menggunakan beberapa teknologi modern yang membuat user experience seperti menggunakan Native App (Android/iOS)
2. Cara kerja service worker aplikasi kita akan diatur oleh service workers agar bisa berjalan offline terlebih dahulu, mengambil data dari cache jika request ke jaringan gagal.
3. Keunggulan dari PWA adalah dapat akses ke fitur device (Kamera, GPS, ds), dapat berjalan saat offline, dan memiliki jangkauan yang luas.
4. Contoh website yang sudah menggunakan PWA adalah freelancer.com, bultrysizephoto.com, homecrowd.io, lamplight.dev, github.io/compass/, usecubes.com, duolingo.com, gitter.im theoutline.com
5. Cara memambalikan asset:
 - a. Pertama mendaftarkan service worker pada situs yang kita buat.
 - b. Kedua, buka situs anda maka akan terpilih "install", ini ditunjukkan untuk menyimpan asset pada situs.
 - c. Isikan kode pada service worker
6. Yang terakhir adalah memprogram service worker kita untuk mengambilan cegatan permintaan ke setiap sumber daya ini dan menggunakan objek "cacher" untuk mengambilan versi yang tersimpan secara lokal dari masing-masing sumber daya.

Makalah Tugas Akhir
 Praktikum Pemrograman Perangkat Bergerak
 "Aplikasi Kuis Brilian"
 Berbasis Android

AGUSTIAWAN (21120116120008)
 Departemen Teknik Komputer, Fakultas Teknik, Universitas Diponegoro
 Jl. Prof. Soedarto, S.H., Tembalang, Semarang, Indonesia

Abstrak – Aplikasi Kuis Brilian berbasis Android ini merupakan aplikasi dimana pengguna dapat melakukan kuis mengenai pertanyaan yang tersedia. Pengguna dapat memilih satu (1) jawaban dari tiga (3) pilihan jawaban yang tersedia. Setiap sesi kuis terdapat lima (5) pertanyaan dengan skor maksimal yaitu lima (5).

Kata kunci : Kuis, Brilian, Tes, Android

I. Pendahuluan

1.1 Latar Belakang

Perangkat bergerak atau *mobile device* merupakan sesuatu yang tidak bisa dilepaskan dari kehidupan manusia zaman sekarang. Melalui perangkat bergerak, proses kehidupan dapat dilakukan dengan mudah seperti dengan adanya *m-banking*, aplikasi *chatting*, aplikasi hiburan, dan sebagainya. Selain itu, saat ini permainan tidak hanya berfungsi sebagai permainan, namun juga dapat berfungsi sebagai sarana edukasi yang interaktif kepada masyarakat. Melalui aplikasi Android Kuis Brilian ini, pengguna dapat menguji seberapa "brilian" kemampuan pengguna.

1.2 Maksud dan Tujuan

Maksud dan tujuan dari pembuatan aplikasi Kuis Brilian ini adalah untuk memenuhi Tugas Akhir dari Praktikum Pemrograman Perangkat Bergerak yang mana merupakan salah satu syarat untuk lulus mata kuliah. Selain itu, melalui pembuatan aplikasi ini agar praktikan mampu mengimplementasikan ilmu yang telah didapatkan selama praktikum terutama pemrograman menggunakan Android Studio.

1.3 Pembatasan Masalah

Dalam pembuatan makalah ini batasan-batasan membahas tentang Aplikasi berbasis Android dan dibangun menggunakan Android Studio yang mana total soal yang tersedia hanya 20 soal dan ditampilkan lima soal tiap sesi kuis. Selain itu soal dan jawaban

tidak dapat diubah setelah program dijalankan.

II. DASAR TEORI

2.1 Android

Android adalah sistem operasi untuk telepon seluler yang berbasiskan Linux. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri sehingga dapat digunakan oleh bermacam peranti penggerak.

Android tersedia secara *open source* bagi manufaktur perangkat keras untuk memodifikasi sesuai kebutuhan. Meskipun konfigurasi perangkat Android tidak sama antara satu dengan yang lain, namun Android sendiri mendukung beberapa fitur seperti penyimpanan menggunakan SQLite (basis data relasional yang ringan untuk menyimpan data), koneksi, *web browser*, dan sebagainya.

2.2 Pengenalan Android Studio

Android Studio adalah sebuah IDE untuk *Android Development* yang dikenalkan pihak google pada acara Google I/O di tahun 2013. Android Studio merupakan suatu pengembangan dari Eclipse IDE, dan dibuat berdasarkan IDE Java populer, yaitu IntelliJ IDEA. Android Studio merupakan IDE resmi untuk pengembangan aplikasi Android. Berbeda dengan Eclipse yang menggunakan Ant, Android Studio menggunakan Gradle sebagai *build environment*.

Fitur-fitur lainnya yang sudah disematkan pada Android Studio adalah sebagai berikut :

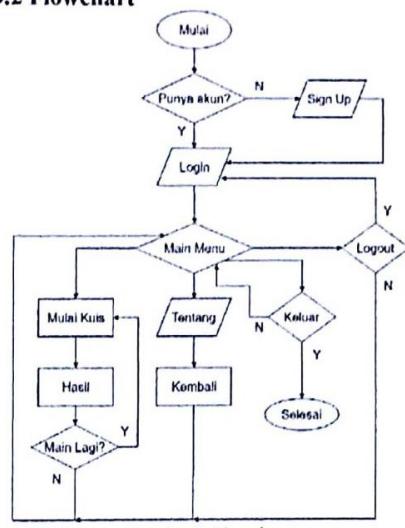
1. Menggunakan *Gradle-based build system* yang fleksibel.
2. Bisa mem-build *multiple APK*.
3. *Template support* untuk Google Services dan berbagai macam tipe perangkat.
4. *Layout editor* yang lebih bagus.
5. *Built-in support* untuk Google Cloud Platform, sehingga mudah untuk integrasi dengan Google Cloud Messaging dan App Engine.
6. *Import library* langsung dari Maven repository dan sebagainya.

III. PERANCANGAN APLIKASI

3.1 Algoritma Program

1. Mulai
2. Login
3. Menu
4. Pilihan
5. Selesai

3.2 Flowchart



Gambar 1 Flowchart

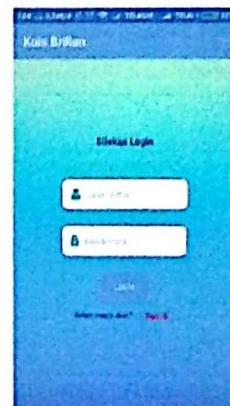
3.3 Pengujian dan Hasil

Berikut adalah tampilan program aplikasi:



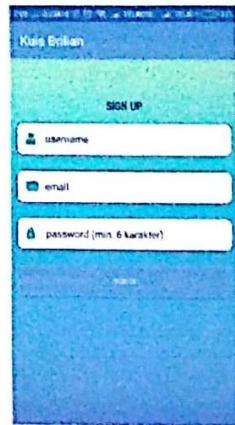
Gambar 2 Splashscreen

Gambar di atas ialah *Splashscreen* saat program dibuka pertama kali.

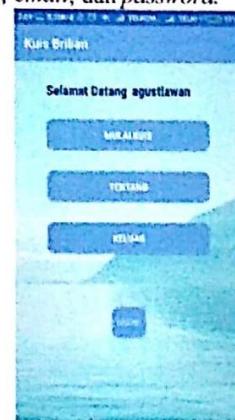


Gambar 3 Tampilan Awal Program

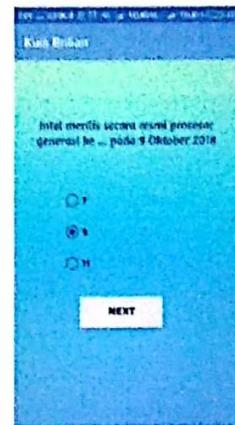
Gambar di atas merupakan tampilan awal program setelah *splashscreen*. Pengguna diminta untuk memasukkan *username* dan *password*. Apabila belum mempunyai akun, pengguna bisa mendaftar melalui tombol *Sign Up* dibawah tombol *Login*.

Gambar 4 Tampilan Halaman *Sign Up*

Gambar diatas merupakan tampilan halaman *Sign Up*. Halaman ini ditujukan bagi mereka yang ingin membuat akun. Terdapat tiga (3) data yang harus dimasukkan yaitu *username*, *email*, dan *password*.

Gambar 5 Tampilan *Main Menu*

Setelah melakukan *login*, maka akan muncul tampilan seperti pada gambar di atas. Terdapat empat tombol yaitu Mulai Kuis untuk memulai kuis, Tentang untuk melihat info pembuat aplikasi, Keluar untuk keluar dari aplikasi, dan *logout* untuk keluar dari *user* saat ini.



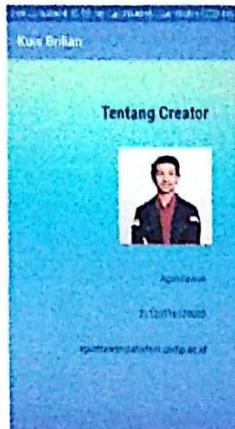
Gambar 6 Contoh Tampilan Pertanyaan dan Jawaban Setelah Klik Tombol Mulai Kuis

Ketika pengguna melakukan klik pada tombol Mulai Kuis, maka akan ditampilkan pertanyaan beserta tiga opsi jawaban. Setelah mengisi jawaban pengguna bisa melakukan klik pada tombol *Next*. Pada sesi ini, pengguna tidak bisa keluar sebelum menyelesaikan kuis.

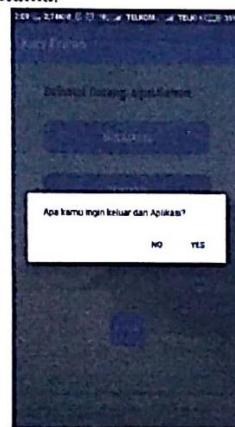


Gambar 7 Tampilan Hasil Kuis

Setelah selesai menjawab lima pertanyaan, maka akan muncul halaman Hasil dimana jumlah soal yang terjawab dengan benar akan ditampilkan sebagai skor. Pada sesi ini, terdapat dua opsi bagi pengguna, ingin main lagi atau kembali ke beranda dengan menekan tombol yang tersedia.

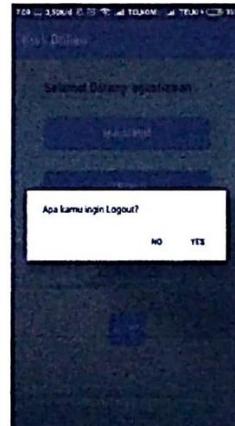


Gambar 8 Tampilan Halaman Tentang
Halaman di atas akan muncul ketika pengguna menekan tombol Tentang pada halaman utama.



Gambar 9 Tampilan Opsi Ketika Pengguna Menekan Tombol Keluar

Ketika pengguna menekan tombol Keluar terdapat dua opsi yaitu *yes* untuk keluar dari aplikasi dan *no* jika tidak ingin keluar dari aplikasi.



Gambar 10 Tampilan Opsi Ketika Pengguna Menekan Tombol *Logout*

Ketika pengguna menekan tombol *Logout* terdapat dua opsi yaitu *yes* untuk *Logout* dari *user* saat ini dan *no* jika tidak ingin keluar dari *user* saat ini. Ketika pengguna memilih *yes*, maka halaman akan berpindah ke halaman *login* lagi.

IV. PENUTUP

4.1 Kesimpulan

1. Android adalah sistem operasi perangkat bergerak yang dikembangkan oleh Google.
2. Untuk membuat aplikasi berbasis Android dapat menggunakan Android Studio sebagai IDE nya.
3. Pengembangan aplikasi berbasis Android dapat menggunakan bahasa Java atau bahasa Kotlin.
4. Untuk membuat *background* pada tampilan aplikasi dapat menggunakan warna ataupun gambar, namun lebih bagus menggunakan warna karena tidak bergantung resolusi layar *smartphone*.
5. *Database* yang digunakan adalah *database SQLite*.

4.2 Saran

Melalui pembuatan aplikasi untuk *mobile device* ini, diharapkan praktikan atau pengembang mampu memahami teknologi serta fitur-fitur yang digunakan dalam proses pengembangan aplikasi. Selain itu, pengembang seharusnya sudah memahami konsep dasar bahasa pemrograman yang dipakai yaitu bahasa pemrograman Java.

DAFTAR PUSTAKA

- [1] Modul Praktikum Pemrograman Perangkat Bergerak 2018
- [2] <http://www.jadibaru.com/android/pengenalan-android-studio-2/>

BIOGRAFI PENULIS



Penulis bernama Agustiawan lahir di Kendal, 26 Juli 1998, merupakan anak kedua dari 3 bersaudara. Penulis merupakan alumni dari SD Negeri 2 Wonodadi, SMP Negeri 1 Plantungan, dan SMA Negeri 1 Sukorejo, Kendal. Penulis saat ini sedang menempuh studi S1 di Departemen Teknik Komputer Universitas Diponegoro.

Semarang, 22 November 2018

Pembimbing Tugas
Akhir Praktikum

A handwritten signature in black ink, appearing to read "Bima".

Bima Fajar Setiawan
21120115140088

Penulis

A handwritten signature in black ink, appearing to read "Agustiawan".

Agustiawan
21120116120008