**PENERAPAN ARSITEKTUR *MODEL VIEW PRESENTER***

**MENGGUNAKAN BAHASA PEMROGRAMAN KOTLIN**

**PADA APLIKASI ANDROID TUKUTU**

Diajukan untuk memenuhi salah satu syarat mengerjakan dan menempuh ujian

KKI



Disususn oleh :

Nama : Aditya Widayanto

NIM : A11.2015.08854

Program Studi : Teknik Informatika

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS DIAN NUSWANTORO**

**SEMARANG**

**2019**

# PERSETUJUAN SKRIPSI

Nama : Aditya Widayanto

NIM : A11.2015.08854

Program Studi : Teknik Informatika

Fakultas : Ilmu Komputer

Judul Tugas Akhir : Penerapan Arsitektur Model View Presenter Menggunakan Bahasa Pemrograman Kotlin Pada Aplikasi Android Tukutu

Tugas Akhir ini telah diperiksa dan disetujui,

Semarang, 5 Maret 2019

|  |  |
| --- | --- |
| Menyetujui | Mengetahui |
| Pembimbing | Dekan Fakultas Ilmu Komputer |
| Dr. Pulung Nurtantio A., S.T, M.Kom | Dr. Drs. Abdul Syukur, MM |
| NPP. 0686.11.2006.331 | NPP.0686.11.1992.017 |

# PENGESAHAN DEWAN PENGUJI

Nama : Aditya Widayanto

NIM : A11.2015.08854

Program Studi : Teknik Informatika

Fakultas : Ilmu Komputer

Judul Tugas Akhir : Penerapan Arsitektur Model View Presenter Menggunakan Bahasa Pemrograman Kotlin Pada Aplikasi Android Tukutu

Tugas Akhir ini telah diujikan dan dipertahankan di hadapan Dewan Penguji pada Sidang tugas akhir pada 5 Maret 2019. Menurut pandangan kami, tugas akhir ini memadai dari segi kualitas maupun kuantitas untuk tujuan penganugerahan gelar Sarjana Komputer (S.Kom).

Semarang, 5 Maret 2019

Dewan Penguji:

|  |  |
| --- | --- |
| Penguji 1 | Penguji 2 |
| Eko Hari Rachmawanto, M. Kom | Egia Rosi Subhiyakto, M. Kom |
| NPP. 0686.11.2012.458 | NPP.0686.11.2014.585 |

Ketua Penguji

Wijanarto, M. Kom

NPP. 0686.11.2009.354

# PERNYATAAN KEASLIAN SKRIPSI

Sebagai mahasiswa Universitas Dian Nuswantoro, yang bertanda tangan dibawah ini, saya :

Nama : Aditya Widayanto

NIM : A11.2015.08854

Menyatakan bahwa karya ilmiah saya yang berjudul :

PENERAPAN ARSITEKTUR MODEL VIEW PRESENTER MENGGUNAKAN BAHASA PEMROGRAMAN KOTLIN

PADA APLIKASI ANDROID TUKUTU

Merupakan karya asli saya (kecuali cuplikan dan ringkasan yang masing – masing telah saya jelaskan sumbernya dan perangkat pendukung. Apabila dikemudian hari, karya saya disinyalir bukan merupakan karya asli saya, dan disertai dengan bukti – bukti yang cukup, maka saya bersedia untuk dibatalkan gelar saya beserta hak dan kewajiban yang melekat pada gelar tersebut. Demikian surat pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Semarang

Pada tanggal : 5 Maret 2019

Yang menyatakan

Aditya Widayanto

# PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Sebagai mahasiswa Universitas Dian Nuswantoro, yang bertanda tangan dibawah ini, saya :

Nama : Aditya Widayanto

NIM : A11.2015.08854

Demi mengembangkan Ilmu Pengetahuan, menyetujui untuk memberikan kepada Universitas Dian Nuswantoro Hak Bebas Royalti Non-Eksklusif (*Non-Exclusive Royalty-Free Right*) atas karya ilmiah saya yang berjudul :

**PENERAPAN ARSITEKTUR MODEL VIEW PRESENTER MENGGUNAKAN BAHASA PEMROGRAMAN KOTLIN**

**PADA APLIKASI ANDROID TUKUTU**

Berserta perangkat yang diperlukan (bila ada). Dengan Hak Bebas Royalti Non-Eksklusif ini Universitas Dian Nuswantoro berhak untuk menyimpan, meng-*copy*, ulang (memperbanyak), menggunakaan, mengelolanya dalam bentuk pangkalan data (database), mendistribusikannya dan menampilkan/ mempublikasikannya di internet atau media lain untuk kepentingan akademis tanpa perlu meminta ijin dari saya selama tetap mencantumkan nama saya sebagai penulis/ pencipta.

Saya bersedia untuk menanggung secara pribadi, tanpa melibatkan pihak Universitas Dian Nuswantoro, segala bentuk tuntutan hokum yang timbul atas pelanggaran Hak Cipta dalam karya ilmiah saya ini.

Demikian surat pernyataan saya ini saya buat dengan sebenarnya.

Dibuat di : Semarang

Pada tanggal : 5 Maret 2019

Yang menyatakan

Aditya Widayanto

# ABSTRAK

Sebagai salah satu startup, Tukutu merupakan marketplace untuk jual beli sepatu yang harus menggunakan metode agar dapat memenuhi kebutuhan penggunanya secara cepat dan tepat. Tukutu secara bertahap harus mampu menyediakan fitur-fitur baru untuk meningkatkan kenyamanan pengguna dalam bertransaksi. Setiap fitur baru tersebut harus melalui tahapan pengujian.

Project Aplikasi Android Tukutu yang dikembangkan dengan bahasa pemrograman Kotlin dapat digunakan pada perangkat dengan sistem operasi Android Oreo dan juga dapat digunakan dengan lancar pada versi android terdahulu sesuai standar minimal SDK yang ditentukan. Sedangkan arsitektur MVP yang diterapkan pada metode Agile dan *Test-Driven Development* digunakan untuk mempermudah pengujian perangkat lunak dan mempercepat proses pengembangan perangkat lunak yang berkualitas. MVP dapat mempermudah dan dapat memisahkan pengujian pada sisi tampilan pengguna (user interface) yang kompleks dan pengujian pada fungsi sistem. Dengan menggunakan arsitektur MVP, pengujian dapat dilakukan secara otomatis.

**Kata kunci:** MVP, Presenter, Agile, Kotlin, Android, Tukutu

# UCAPAN TERIMAKASIH

Dengan mengucapkan syukur kepada Tuhan Yang Maha Esa atas berkat, rahmat serta karunia-Nya, Penulis dapat menyelesaikan Tugas Akhir berjudul “Penerapan Arsitektur Model View Presenter Menggunakan Bahasa Pemrograman Kotlin Pada Aplikasi Android Tukutu”.

Tugas Akhir ini ditujukan untuk memenuhi salah satu persyaratan ujian guna memperoleh gelar Sarjana Komputer (S.Kom.) pada Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Dian Nuswantoro Semarang.

Penulis menyadari dalam penulisan tugas akhir ini masih jauh dari sempurna, dan banyak kekurangan baik dalam metode penulisan maupun dalam pembahasan materi. Hal tersebut dikarenakan keterbatasan kemampuan Penulis. Sehingga Penulis mengharapkan saran dan kritik yang bersifat membangun mudah-mudahan dikemudian hari dapat memperbaiki segala kekuranganya.

Dalam penulisan tugas akhir ini, Penulis selalu mendapatkan bimbingan, dorongan, serta semangat dari banyak pihak. Oleh karena itu Penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada pembimbing dan pihak yang membantu dalam penyusunan Tugas Akhir ini, yaitu kepada:

1. Kedua orangtua yang telah memberikan dorongan dan doa sehingga Penulis dapat menyelesaikan tugas akhir ini.
2. Prof. Dr. Ir. Edi Noersasongko, M. Kom, selaku Rektor Universitas Dian Nuswantoro.
3. Dr. Drs. Abdul Syukur, MM, selaku Dekan Fakultas Ilmu Komputer.
4. Heru Agus Santoso Ph. D, selaku Ketua Program Studi Teknik Informatika – S1.
5. Dr. Pulung Nurtantio Andono, S. T, M. Kom, selaku Dosen Pembimbing, yang telah meluangkan waktunya, tenaga dan pikirannya untuk membimbing Penulis dalam penulisan tugas akhir ini.
6. Semua pihak yang namanya tidak dapat disebutkan satu per satu.

Akhirnya, Penulis mengucapkan rasa terima kasih kepada semua pihak dan apabila ada yang tidak tersebutkan Penulis mohon maaf, dengan besar harapan semoga tugas akhir yang ditulis oleh Penulis ini dapat bermanfaat khususnya bagi Penulis sendiri dan umumnya bagi pembaca.

Semarang, Februari 2019

Penulis

# DAFTAR ISI

[**PERSETUJUAN SKRIPSI ii**](#_Toc2716758)

[**PENGESAHAN DEWAN PENGUJI iii**](#_Toc2716759)

[**PERNYATAAN KEASLIAN SKRIPSI iv**](#_Toc2716760)

[**PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS v**](#_Toc2716761)

[**ABSTRAK vi**](#_Toc2716762)

[**UCAPAN TERIMAKASIH vii**](#_Toc2716763)

[**DAFTAR ISI ix**](#_Toc2716764)

[**DAFTAR GAMBAR xiii**](#_Toc2716765)

[**DAFTAR TABEL xv**](#_Toc2716766)

[**BAB I PENDAHULUAN 1**](#_Toc2716767)

[**1.1.** **Latar Belakang Masalah** 1](#_Toc2716769)

[**1.2. Rumusan Masalah 4**](#_Toc2716770)

[**1.3. Batasan Masalah 4**](#_Toc2716771)

[**1.4. Tujuan Penelitian 5**](#_Toc2716772)

[**1.5. Manfaat Penelitian 5**](#_Toc2716773)

[**1.5.1. Bagi Mahasiswa 5**](#_Toc2716774)

[**1.5.2. Bagi Akademik 5**](#_Toc2716775)

[**1.5.3. Bagi Instansi 5**](#_Toc2716776)

[**BAB II TINJAUAN PUSTAKA DAN LANDASAN TEORI 6**](#_Toc2716777)

[**2.1. State of The Art 6**](#_Toc2716778)

[**2.2. Landasan Teori 8**](#_Toc2716779)

[**2.2.1. Sistem Operasi Android 8**](#_Toc2716781)

[**2.2.2. Android Studio 12**](#_Toc2716782)

[**2.2.2.1. Persyaratan Minimum 13**](#_Toc2716783)

[**2.2.2.2. Antarmuka Android Studio 14**](#_Toc2716784)

[**2.2.2.3. Struktur Proyek Android Studio 14**](#_Toc2716785)

[**2.2.2.4. Cara Melakukan Pengujian Project 17**](#_Toc2716786)

[**2.2.3. Bahasa Pemrograman JAVA dan KOTLIN 18**](#_Toc2716787)

[**2.2.4. Retrofit dan Reactive pada Android 22**](#_Toc2716788)

[**2.2.5. JSON (Java Script Object Notation) 24**](#_Toc2716789)

[**2.2.6. Arsitektur Model View Presenter ( MVP ) 25**](#_Toc2716790)

[**2.2.7. Firebase 27**](#_Toc2716791)

[**2.3. Deskripsi Tempat KKI 34**](#_Toc2716792)

[**2.4. Logo dan Makna Tempat KKI 34**](#_Toc2716793)

[**2.5. Struktur Organisasi Tempat KKI 35**](#_Toc2716794)

[**2.6. Visi dan Misi Tempat KKI 35**](#_Toc2716795)

[**BAB III METODE PENELITIAN 36**](#_Toc2716797)

[**3.1. Instrument Penelitian 36**](#_Toc2716798)

[**3.2. Prosedur Pengambilan Data 36**](#_Toc2716799)

[**3.2.1. Observasi 36**](#_Toc2716800)

[**3.2.2. Studi Literatur 36**](#_Toc2716801)

[**3.2.3. Forum Diskusi & Tutorial 36**](#_Toc2716802)

[**3.3. Teknik Analisis Data 37**](#_Toc2716803)

[**3.4. Metode 37**](#_Toc2716804)

[**3.4.1. Pembuatan Test 38**](#_Toc2716805)

[**3.4.2. Menjalankan Test 39**](#_Toc2716806)

[**3.4.3. Menuliskan Kode 40**](#_Toc2716807)

[**3.4.4. Menjalankan Ulang Test 40**](#_Toc2716808)

[**3.4.5. Refactoring 40**](#_Toc2716809)

[**3.5. Eksperimen dan Pengujian 41**](#_Toc2716810)

[**BAB IV PENERAPAN SISTEM DAN PEMBAHASAN 42**](#_Toc2716811)

[**4.1. Gambaran Umum Sistem 42**](#_Toc2716812)

[**4.2. Perancangan Kebutuhan Sistem 42**](#_Toc2716813)

[**4.2.1. Kebutuhan Perangkat lunak 42**](#_Toc2716814)

[**4.2.2. Identifikasi Kebutuhan Informasi 43**](#_Toc2716815)

[**4.2.3. Identifikasi Kebutuhan Hardware 43**](#_Toc2716816)

[**4.3. Analisa Perancangan Sistem 44**](#_Toc2716817)

[**4.3.1. Identifikasi Aktor 44**](#_Toc2716818)

[**4.3.2. Usecase Diagram 45**](#_Toc2716819)

[**4.3.3. Activity Diagram 46**](#_Toc2716820)

[**4.3.3.1. Activity Diagram Register 46**](#_Toc2716821)

[**4.3.3.2. Activity Diagram Login 47**](#_Toc2716822)

[**4.3.3.3. Activity Diagram Reset Password 47**](#_Toc2716823)

[**4.3.3.4. Activity Diagram Edit Password 48**](#_Toc2716824)

[**4.4. Implementasi Antarmuka 49**](#_Toc2716825)

[**4.4.1. Layout Halaman Splash 49**](#_Toc2716826)

[**4.4.2. Layout halaman intro 50**](#_Toc2716827)

[**4.4.3. Layout Halaman Login 51**](#_Toc2716828)

[**4.4.4. Layout Halaman Registrasi 53**](#_Toc2716829)

[**4.4.5. Layout halaman utama 54**](#_Toc2716830)

[**4.4.6. Layout Halaman Profil 55**](#_Toc2716831)

[**4.4.7. Layout halaman edit profil 56**](#_Toc2716832)

[**4.4.8. Layout halaman ganti password 57**](#_Toc2716833)

[**4.4.9. Layout history transaksi 58**](#_Toc2716834)

[**4.4.10. Layout pencarian 59**](#_Toc2716835)

[**4.4.11. Layout list sepatu 60**](#_Toc2716836)

[**4.4.12. Layout form pengajuan merchant 61**](#_Toc2716837)

[**4.4.13. Layout pengaturan merchant 62**](#_Toc2716838)

[**4.5. Penerapan Bahasa Pemrograman Kotlin 63**](#_Toc2716839)

[**4.5.1. Librari yang digunakan 63**](#_Toc2716840)

[**4.5.2. Struktur File Aplikasi Tukutu 64**](#_Toc2716841)

[**4.5.3. Implementasi Kotlin pada objek Class Model 65**](#_Toc2716842)

[**4.5.4. Implementasi Retrofit dan RxKotlin/RxJava 65**](#_Toc2716843)

[**4.6. Penerapan Model View Presenter 68**](#_Toc2716844)

[**4.6.1. Struktur File MVP 68**](#_Toc2716845)

[**4.6.2. Penerapan MVP Pada Halaman Edit Password 69**](#_Toc2716846)

[**4.7. Penerapan Test-Driven Development 70**](#_Toc2716847)

[**BAB V KESIMPULAN DAN SARAN 73**](#_Toc2716849)

[**5.1. Kesimpulan 73**](#_Toc2716850)

[**5.2. Saran 73**](#_Toc2716851)

[**DAFTAR PUSTAKA 74**](#_Toc2716852)

# DAFTAR GAMBAR

[Gambar 2.1 Logo Android Studio 12](#_Toc1499679)

[Gambar 2.2. Antarmuka Android Studio 14](#_Toc1499680)

[Gambar 2.3. Manifest Proyek Android 14](#_Toc1499681)

[Gambar 2.4. Bagian Java Proyek Android 15](#_Toc1499682)

[Gambar 2.5. Bagian Resource Proyek Android 16](#_Toc1499683)

[Gambar 2.6. Bagian Gradle Proyek Android 17](#_Toc1499684)

[Gambar 2.7. Tampilan Pilihan Device Untuk Pengujian Project 18](#_Toc1499685)

[Gambar 2.8. Deklarasi variable pada Java 19](#_Toc1499686)

[Gambar 2.9. Deklarasi Variable Pada Kotlin 20](#_Toc1499687)

[Gambar 2.10. Contoh null safety pada kotlin 21](#_Toc1499688)

[Gambar 2.11. Penerapan OnClick pada Java 21](#_Toc1499689)

[Gambar 2.12 Penerapan OnClick pada Kotlin 22](#_Toc1499690)

[Gambar 2.13. Gradle RxKotlin 23](#_Toc1499691)

[Gambar 2.14. Repository Maven Jitpack 23](#_Toc1499692)

[Gambar 2.15. Dependency RxKotlin 23](#_Toc1499693)

[Gambar 2.16 Contoh format JSON [13] 24](#_Toc1499694)

[Gambar 2.17 Diagram sederhana MVC (kiri) dan MVP(kanan) 25](#_Toc1499695)

[Gambar 2.18. Alur Dan Aturan Model MVP 26](#_Toc1499696)

[Gambar 2.19 Logo Firebase 27](#_Toc1499697)

[Gambar 2.20 Contoh Usecase Diagram. [19] 29](#_Toc1499698)

[Gambar 2.21. Logo PT Dian Nuswantoro Teknologi dan Informasi 34](#_Toc1499699)

[Gambar 3. 1. Alur Test Driven Development 37](#_Toc1499700)

[Gambar 3. 2. Alur Metode Agile 38](#_Toc1499701)

[Gambar 3. 3. Contoh Test Success 39](#_Toc1499702)

[Gambar 3. 4.Contoh Test Gagal 40](#_Toc1499703)

[Gambar 4. 1.Gradle Kotlin 42](#_Toc2716563)

[Gambar 4. 2. Retrofit & RX-Kotlin 43](#_Toc2716564)

[Gambar 4. 3. Diagram Usecase 45](#_Toc2716565)

[Gambar 4. 4.Diagram aktivitas Register 46](#_Toc2716566)

[Gambar 4. 5.Diagram Aktivitas Login 47](#_Toc2716567)

[Gambar 4. 6 Diagram Aktivitas Edit Password 47](#_Toc2716568)

[Gambar 4. 7. Diagram Aktivitas Edit Password 48](#_Toc2716569)

[Gambar 4. 8. Halaman Splash 49](#_Toc2716570)

[Gambar 4. 9. Tampilan Antarmuka Halaman Login 51](#_Toc2716571)

[Gambar 4. 10. Tampilan Antarmuka Login Google 52](#_Toc2716572)

[Gambar 4. 11. Tampilan Antarmuka Halaman Register 53](#_Toc2716573)

[Gambar 4. 12. Tampilan Antarmuka Halaman Utama 54](#_Toc2716574)

[Gambar 4. 13. Tampilan Halaman Profil 55](#_Toc2716575)

[Gambar 4. 14. Tampilan Edit Profil 56](#_Toc2716576)

[Gambar 4. 15. Tampilan Edit Password 57](#_Toc2716577)

[Gambar 4. 16. Tampilan History Transaksi 58](#_Toc2716578)

[Gambar 4. 17. Tampilan Halaman Pencarian 59](#_Toc2716579)

[Gambar 4. 18Tampilan Halaman List Sepatu 60](#_Toc2716580)

[Gambar 4. 19. Tampilan Form Pengajuan Merchant 61](#_Toc2716581)

[Gambar 4. 20. Halaman Pengaturan Merchant 62](#_Toc2716582)

[Gambar 4. 21. Librari yang pada Aplikasi Tukutu 63](#_Toc2716583)

[Gambar 4. 22. Struktur File Aplikasi 64](#_Toc2716584)

[Gambar 4. 23. Class AuthUser Kotlin 65](#_Toc2716585)

[Gambar 4. 24. Konfigurasi Retrofit dan RxKotlin 65](#_Toc2716586)

[Gambar 4. 25. Class Interface method changePassword 66](#_Toc2716587)

[Gambar 4. 26. ImplementasiRetrofit pada Object Class EditPasswordRepository 67](#_Toc2716588)

[Gambar 4. 27. Struktur File MVP 68](#_Toc2716589)

[Gambar 4. 28. MVP pada Halaman Edit Password 69](#_Toc2716590)

[Gambar 4. 29.Test Case Ganti Password 71](#_Toc2716591)

[Gambar 4. 30 Hasil Test Case Ganti Password 71](#_Toc2716592)

[Gambar 4. 31.Test Case Ganti Password Gagal 72](#_Toc2716593)

[Gambar 4. 32 Hasil Test Case Ganti Password Jika Gagal 72](#_Toc2716594)

# DAFTAR TABEL

[Tabel 2.1 State of The Art 6](#_Toc1497102)

[Tabel 2.2. Perkembangan Sistem Operasi Android 10](#_Toc1497103)

[Tabel 2.3. Simbol-simbol Use Case Diagram 30](#_Toc1497104)

[Tabel 2.4. Simbol-simbol Class Diagram 31](#_Toc1497105)

[Tabel 2.5.Simbol-simbol *Activity Diagram* 33](#_Toc1497106)

[Tabel 4.1. Identifikasi Aktor 44](#_Toc1497337)

[Tabel 4.2. Tutorial Intro 50](#_Toc1497338)

# BAB I

# PENDAHULUAN

* 1. **Latar Belakang Masalah**

Di era modern saat ini, seluruh aktivitas manusia bergantung pada teknologi. Semua sektor dilakukan secara digital tak terkecuali pada sektor komersil dengan adanya *e-commerce* berupa *marketplace*. Di Indonesia banyak sekali *marketplace* yang berkembang dan memprioritaskan target pengguna mereka adalah para pengguna perangkat bergerak.

Perangkat bergerak atau yang lebih sering dikenal sebagai perangkat *mobile* ini memiliki beberapa sistem operasi yang populer , salah satu diantaranya adalah sistem operasi Android. Sistem operasi Android banyak digunakan pada perangkat *mobile smartphone*. Android merupakan sistem operasi *open source* yang diakuisisi dan dikembangkan oleh Google. Perkembangan sistem operasi ini sangatlah cepat, oleh sebab itu para pengembang aplikasi diharuskan untuk selalu mengikuti perkembangan sistem operasi populer ini agar aplikasi yang mereka kembangkan dapat digunakan dengan baik oleh para pengguna.

Pada Google I/O tahun 2017 Google telah merilis versi terbaru android dengan nama Android Oreo dan juga secara resmi telah mengumumkan bahasa pemrograman baru dengan nama Kotlin sebagai bahasa resmi yang didukung oleh Google untuk pengembangan aplikasi Android. Berdasarkan pengalaman dari beberapa pengembang aplikasi Android, pada Android Oreo sering terjadi kegagalan sistem jika aplikasi yang dikembangkan menggunakan bahasa pemrograman Java. Sejak saat itu, bahasa Kotlin menjadi salah satu alternatif untuk menyelesaikan masalah tersebut.

Berikut ini merupakan 4 alasan penggunaan Bahasa pemrograman kotlin:

1. Concise / Ringkas

Ringkas artinya penulisan Bahasa pemrograman kotlin dapat lebih ringkas dibandingkan pemrograman java.

1. Safe / Aman

Yang dimaksud dengan aman adalah Bahasa pemrograman kotlin sudah menyediakan *nullsafety* yang dapat digunakan untuk mengantisipasi data null/kosong bahkan sebelum di compile.

1. Interoperable / Bersilangan

Dalam penerapannya Bahasa pemrograman kotlin dapat di gunakan bersamaan dengan Bahasa pemrograman java.

1. Tool-friendly

Dapat digunakan pada tools IDE apapun yang mendukung Bahasa pemrograman java.

Beberapa aplikasi terlaris juga telah menggunakan Bahasa pemrograman Kotlin untuk mengembangkan aplikasinya, diantaranya adalah :

1. Pinterest

Pinterest yang merupakan aplikasi social media berbagi gambar ini telah menggunakan kotlin pada aplikasi mereka.

1. Coursera

Coursera hanya menerapkan kotlin pada beberapa bagian dari aplikasi android mereka.

1. Evernote

Evernote telah menerapkan kotlin pada aplikasi android yang mereka pasarkan.

1. Uber

Tim pengembangan aplikasi Uber telah menggunakan kotlin untuk menunjang kebutuhan internal mereka.

1. Corda

Corda merupakan salah satu *open source* yang telah menggunakan Bahasa pemrograman kotlin pada keseluruhan proyek  *open-source* .

1. Atlassian

Atlassian menggunakan Bahasa pemrograman kotlin untuk mengembangkan aplikasi android Trello.

Berdasarkan jurnal penelitian [1] menemukan bahwa dalam 10 tahun metode Scrum Agile lebih popular dibandingkan metode tradisional. Metode ini diterapkan pada industri karena dengan metode ini mengutamakan proses pengujian untuk mendapatkan kualitas perangkat lunak yang baik. Proses pengujian sangatlah penting untuk dilakukan, namun pada penerapannya proses pengujian ini sering diabaikan dan dianggap sebagai hal yang sulit dilakukan.

Berdasarkan penelitian yang dilakukan oleh Bo Wu dan Xiaohu Yang dari Fakultas Ilmu Komputer dan Teknologi Universitas Zhejiang disimpulkan bahwa desain arsitektur Model View Controller dapat lebih optimal digunakan pada *platform* berbasis web dimana terdapat beberapa tampilan yang bisa digunakan secara berulang – ulang. Sedangkan untuk Model View Presenter baik digunakan untuk “Fat Client” dengan tampilan yang kompleks dan logika yang simple, lebih tepatnya baik untuk sistem seperti *financial system* [2].

Menurut Paul Graham [3], “*Startup is a company designed to grow fast,*” sedangkan menurut Eric Ries [4], “*Startup is a human institution design that create something new under condition extream and serenity*”.

Sebagai salah satu usaha rintisan atau yang sering disebut dengan *startup*, Tukutu harus menggunakan metode yang tepat agar dapat memenuhi kebutuhan penggunanya dengan cepat dan tepat. Tukutu merupakan *marketplace* yang dikhususkan untuk jual beli sepatu. Sebagai *platform* yang mempertemukan penjual dan pembeli secara *online*, secara bertahap Tukutu harus menyediakan fitur-fitur baru yang dapat meningkatkan kenyamanan dalam bertransaksi. Setiap fitur baru tersebut harus melalui tahapan pengujian.

Berdasarkan masalah tersebut maka penelitian ini akan membahas tentang penggunaan Bahasa pemrograman Kotlin dengan arsitektur MVP yang diterapkan diterapkan pada metode *Agile testing* untuk mempermudah pengujian perangkat lunak dan mempercepat proses pengembangan perangkat lunak yang berkualitas. Arsitektur *pattern* Model View Presenter (MVP) digunakan dalam pembuatan aplikasi dengan *platform* Android karena arsitektur ini dapat mempermudah dan dapat memisahkan pengujian pada sisi tampilan pengguna (*user interface*) yang kompleks dan pengujian pada fungsi sistem. Dengan menggunakan arsitektur MVP, pengujian dapat dilakukan secara otomatis.

* 1. **Rumusan Masalah**

Kurangnya pemahaman tentang metode pengembangan aplikasi dan arsitektur *pattern* pada pengembangan perangkat lunak menimbulkan pemikiran bahwa pengujian perangkat lunak sangat sulit dan membutuhkan waktu yang lama sehingga para pengembang aplikasi cenderung melewatkan tahapan pengujian. Penggunaan arsitektur MVP dalam pembuatan aplikasi dengan *platform* Android diharapkan dapat mempermudah tahapan ini karena pengujian dapat dilakukan secara otomatis.

* 1. **Batasan Masalah**

Penelitian ini akan membahas Aplikasi Android Tukutu dengan batasan sebagai berikut:

* + 1. Aplikasi Tukutu yang berkembang pada sistem operasi Android.
    2. Aplikasi Tukutu dikembangkan oleh PT. Dinustek.
    3. Aplikasi Tukutu dikembangkan dengan bahasa pemrograman Kotlin yang menjadi bahasa official Android.
    4. Aplikasi Tukutu dikembangkan dengan menggunakan metode Agile yang dikombinasikan dengan proses *Test Driven Development* dan arsitektur pattern MVP.
  1. **Tujuan Penelitian**

Penulisan laporan ini bertujuan untuk menghasilkan aplikasi Tukutu dengan menggunakan bahasa pemrograman Kotlin dan menerapkan arsitektur pattern Model View Presenter sehingga dapat menghasilkan aplikasi yang berkualitas dan memenuhi standart pengujian serta dapat mengikuti perkembangan kebutuhan yang bergerak cepat.

* 1. **Manfaat Penelitian**
     1. **Bagi Mahasiswa**
        1. Memperdalam ilmu tentang pemrograman perangkat bergerak khususnya pada pemrograman Android.
        2. Mengerti bagaimana proses merancang sebuah aplikasi berbasis Android.
        3. Mengerti bagaimana implementasi librari dan komunikasi aplikasi mobile dengan webservis.
        4. Memahami penerapan bahasa pemrograman Kotlin dan *reactive* *programming* pada pemrograman Android.
     2. **Bagi Akademik**
        1. Mengetahui seberapa jauh perkembangan kompetensi mahasiswa.
        2. Mampu mengetahui target akademik dalam membuat SDM yang mampu bersaing di dunia kerja
     3. **Bagi Instansi**

Aplikasi Tukutu dapat memenuhi kebutuhan penguna yang menginginkan aplikasi yang berkualitas untuk mempermudah transaksi yang ada di dalam aplikasi Tukutu

# BAB II TINJAUAN PUSTAKA DAN LANDASAN TEORI

* 1. **State of The Art**

Berikut ini adalah tabel *state of the art* jurnal penelitian sebelumnya yang menjadi dasar penelitian:

Tabel 2.1 State of The Art

| **No** | **Peneliti** | **Tahun** | **Judul** | **Masalah** | **Metode** | **Hasil** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Bo Wu & Yang Xiaohu | 2009 | Application of MVP Architecture in Reengineering of  Legacy Financial System | Beberapa logika dideklarasikan secara berulang di beberapa file | MVP (Model View Presenter) dengan SubPresenter | Peningkatan efisiensi pada proses test aplikasi |
| 2 | Vijayasarathy & Butler | 2015 | Choice of Software Development Methodologies - Do Project, Team and Organizational Characteristics Matter? | Pengujian seringkali diabaikan | Scrum Agile | Metode Scrum Agile lebih popular diterapkan pada industry karena dengan metode ini mengutamakan proses pengujian untuk mendapatkan kualitas perangkat lunak yang baik |
| 3 | Yang Zhang & Yanjing Luo | 2010 | An Architecture and Implement Model for Model-View-Presenter Pattern | Testing pada View dan Bussiness Logic | MVP (Model View Presenter) | Efisiensi dan mempermudah testing untuk View dan Bussiness Logic |
| 4 | Artem Syromiatnikov & Danny Weyns | 2014 | A Journey Through the Land of Model-View-Design Patterns | Integrasi antara UI dengan domain aplikasi atau *model* | MVP (Model View Presenter), MVC (Model View Controller), MVVM (Model View View Model) | Tiap model memiliki ciri khas tersendiri |
| 5 | Rijwan, Srivastava, & Pandey | 2016 | Agile Approach for Software Testing Process | Pengujian Aplikasi sangat penting bagi industri sehingga harus diperhatikan dengan baik | Metode Agile Testing | Seluruh tim harus bekerjasama pada saat implementasi koding dan juga pada saat pengujian untuk dapat menghasilkan perangkat lunak yang berkualitas |

Arsitektur MVP (Model View Presenter) telah diterapkan dalam beberapa bisnis logic dan juga pada beberapa industri, yang membedakan penelitian ini dengan penelitian lain yang pernah dilakukan sebelumnya adalah pada penerapan dalam penggunaan arsitektur MVP tersebut. Pada penelitian ini arsitektur MVP akan digunakan dengan menggunakan Bahasa pemrograman kotlin pada aplikasi Tukutu yang merupakan aplikasi berbasi Android yang digunakan untuk mempertemukan penjual dan pembeli sepatu secara online dengan jaminan bahwa sepatu yang sampai ke tangan pembeli adalah sepatu asli.

* 1. **Landasan Teori**

Berikut ini adalah landasan teori yang digunakan dalam penelitian ini, landasan teori yang digunakan berasal dari literatur jurnal, buku dan juga dari dokumentasi google android.

* + 1. **Sistem Operasi Android**

Sistem operasi Android merupakan sistem operasi berbasis *opensource* Linux yang dirancang dan dikembangkan oleh *Android Inc* untuk perangkat mobile dengan spesifikasi layar sentuh. Dalam perkembangannya sistem operasi Android ini tidak hanya diterapkan pada telepon pintar tetapi juga diterapkan pada televisi, jam pintar, dan komputer tablet. Android juga mempunyai *store*dengan lebih dari 2 miliar pengguna aktif, per Januari 2018. [5]

Aplikasi yang berbasis Android dapat didistribusikan atau dipasarkan melalui *store* khusus yang diberi nama Google Play. Google Play ini merupakan store resmi Android yang dikelola oleh Google. [5]

Setelah sistem operasi ini dibeli oleh Google, perkembangan sistem operasi ini sangat cepat. Saat ini versi terbaru yang stabil dan telah dirilis secara umum sudah mencapai API 26 yang lebih dikenal dengan nama Android Oreo. Banyak fitur baru yang diperkenalkan oleh Google pada versi Android Oreo ini demi memenuhi kebutuhan para pengguna. *Source code* dari Android bersifat open source. Hal ini sangat menarik bagi komunitas pengembang aplikasi, karena lisensi *open source* dianggap sangat mendukung untuk mengembangkan produk dengan aman.

Untuk memfasilitasi para pengembang aplikasi Android, Google telah menyediakan Android Studio yang menjadi IDE kerjasama antara Google dan JetBrain. Bahasa pemrograman yang didukung oleh Android Studio adalah pemrograman Java dan Kotlin. Namun ada bahasa pemrograman lain yang dapat digunakan untuk mengembangkan aplikasi Android yaitu bahasa C# dengan Xamarin, dan Javascript dengan framework ReactNative.

Tabel 2.2. Perkembangan Sistem Operasi Android [5]

| **Version** | **Code name** | **Release date** | **API level** | **New features** |
| --- | --- | --- | --- | --- |
| **9** | Android P | - | - | - |
| **8.1** | Oreo | 25-Oct-17 | 26 - 27 | Picture-in-Picture |
| **7.1 - 7.1.2** | Nougat | 22-Aug-16 | 24 - 25 | Multi window |
| GIF Keyboard |
| **6.0 - 6.0.1** | Marshmallow | 5-Oct-15 | 23 | Now On Tap |
| Permissions |
| Battery (Doze & App Standy) |
| **5.1 - 5.1.1** | Lollipop | 12-Nov-14 | 21 - 22 | Material Design |
| Multiscreen |
| Notifications |
| **4.4 - 4.4.4** | KitKat | 31-Oct-13 | 19 - 20 | Voice : Ok Google |
| Immersive Design |
| Smart Dialer |
| **4.1 - 4.3.1** | Jelly Bean | 9-Jul-12 | 18 | Google Now |
| Actionable Notifications |
| Account Switching |
| **4.0 - 4.6** | Ice Cream Sandwich | 19-Oct-11 | 15 | Custom Home Screen |
| Data Usage Control |
| Android Beam |
| **3.0 - 3.2.6** | HoneyComb | 22-Feb-11 | 13-Nov | Tablet-Friendly Design |
| System Bar |
| Quick Settings |
| **2.3 - 2.3.7** | Gingerbread | 9-Feb-11 | 10-Sep | Gaming APIs |
| NFC |
| Battery Management |
| **2.2 - 2.23** | Froyo | 20-May-10 | 8 | Voice Action |
| Portable Hotspot |
| Dalvik JIT |
| **2.0 - 2.1** | Eclair | 26-Oct-09 | 5 | Google Maps Navigation |
| Home Screen Customization |
| Speech-to-Text |
| **1.6** | Donut | 15-Sep-09 | 4 | Quick Search Box |
| Screen Size Diversity |
| Android Market |
| **1.5** | Cupcake | April 27, 2009 | 3 | - |

Berdasarkan tabel diatas dapat diketahui bahwa versi terbaru dari Sistem Operasi Android adalah Android P. Google hanya memperkenalkan nama Android P pada Google I/O 2018 , namun belum memberikan secara rinci kepanjangan dari “P” dan fitur-fitur apa saja yang terdapat pada Android P. Android P belum secara resmi dirilis dan belum dapat diinstal pada perangkat oleh sebab itu para pengembang aplikasi masih memfokuskan pengembangan aplikasi mereka pada perangkat dengan sistem operasi Android Oreo.

* + 1. **Android Studio**



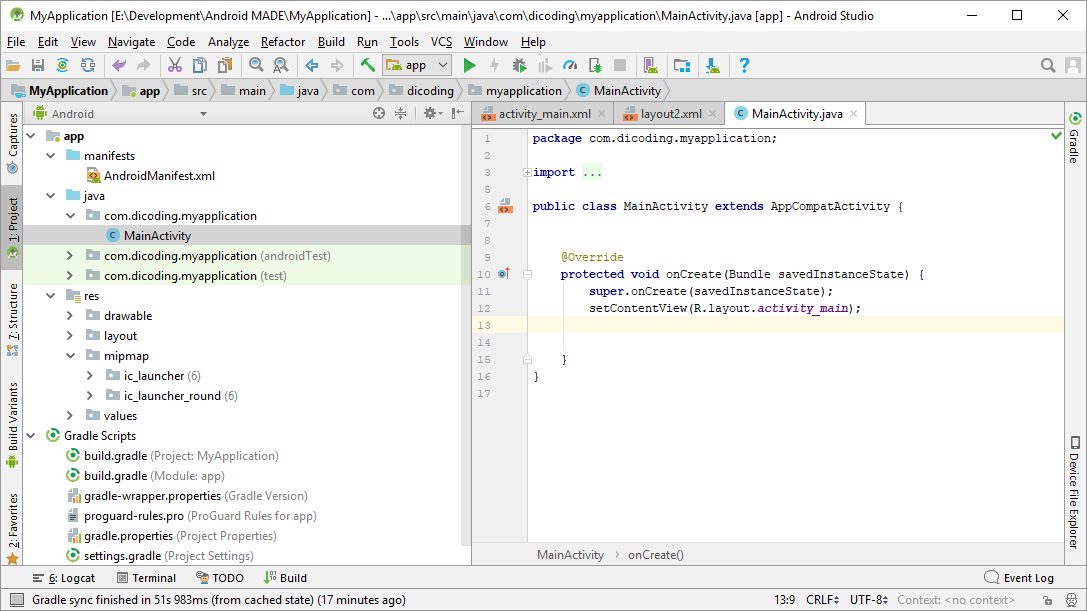
Gambar 2.1 Logo Android Studio [5]

Android Studio adalah Integrated Development Environment (IDE) resmi untuk pengembangan aplikasi Android, berdasarkan IntelliJ IDEA. [6] Selain sebagai alat editor dan pengembang yang kuat dari IntelliJ, Android Studio menawarkan lebih banyak fitur yang meningkatkan produktivitas Anda saat membuat aplikasi Android, seperti:

1. Sistem bangun berbasis Gradle yang fleksibel.
2. Emulator yang cepat dan kaya fitur.
3. Lingkungan terpadu tempat Anda dapat mengembangkan untuk semua perangkat Android.
4. Instant Run untuk mendorong perubahan pada aplikasi yang sedang berjalan tanpa membangun APK baru.
5. Template kode dan integrasi GitHub untuk membantu Anda membuat fitur aplikasi umum dan mengimpor *sample code*.
6. Alat dan kerangka pengujian ekstensif.
7. Alat serat untuk menangkap kinerja, kegunaan, kompatibilitas versi, dan masalah lainnya.
8. Dukungan C ++ dan NDK.
9. Dukungan bawaan untuk Google Cloud Platform, sehingga mudah untuk mengintegrasikan Google Cloud Messaging dan App Engine.
   * + 1. **Persyaratan Minimum**

Agar dapat menjalankan Android Studio dengan baik, perangkat komputer harus memenuhi persyaratan minimum berikut ini : [7]

* Windows 2003 (32-bit atau 64-bit)
* Windows Vista (32-bit atau 64-bit)
* Windows 7 (32-bit atau 64-bit)
* Windows 8 / Windows 8.1
* Mac OS X 10.5.8 atau yang lebih baru (sistem berbasis Intel saja)
* Sistem Linux dengan versi 2.11 atau yang lebih baru dari GNU C Library (glibc)
* Minimal 2GB RAM (4GB lebih diutamakan)
* 1.5 Gb ruang disk yang tersedia
  + - 1. **Antarmuka Android Studio**

****

Gambar 2.2. Antarmuka Android Studio [5]

Gambar di atas merupakan tampilan penuh dari IDE Android studio yang berbasis IntelliJ IDEA. Tampilan layar kerja dibagi menjadi 2 (dua) sisi kanan dan sisi kiri. Sisi kanan merupakan Project Editor sedangkan sisi kiri merupakan Project Explorer. [8]

Pada Project Explorer berisi daftar file dan package pada project. Pengguna dapat memilih file yang akan mereka kelola dan mereka edit. Sedangkan pada Project Editor , pengguna dapat menambahkan baris kode pada file yang telah dipilih.

* + - 1. **Struktur Proyek Android Studio**

Secara garis besar, struktur proyek Android yang terdapat pada IDE Android Studio dibagi menjadi 4 bagian utama, yaitu Manifest, Java , Res , Gradle. Berikut ini adalah penjelasan 4 bagian tersebut :

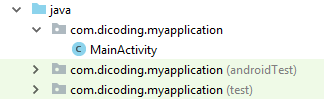
1. Manifest

Y:\Desktop\image8.png

Gambar 2.3. Manifest Proyek Android [5]

Manifest merupakan salah satu berkas dengan format \*.XML yang harus ada dalam sebuah proyek Android. Pada satu file ini , terdapat beragam informasi penting yang berhubungan dengan sistem Android. Sistem operasi Android harus mengetahui informasi tersebut sebelum suatu aplikasi dapat dijalankan.

1. Java

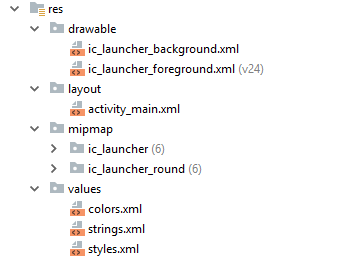


Gambar 2.4. Bagian Java Proyek Android [5]

Bagian ini berisi berkas *source code* algoritma yang ditulis dalam bahasa pemrograman Java maupun Kotlin. Bagian Java ini memiliki 3 package utama yaitu Main , AndroidTest dan juga Test (Unit Test ) .

Main package berisikan berkas java yang digunakan untuk sistem utama dari proyek aplikasi android. AndroidTest berisikan berkas – berkas yang akan digunakan untuk pengujian instrumentasi , sedangkan Unit Test berisikan berkas yang akan digunakan untuk pengujian unit aplikasi.

1. Res



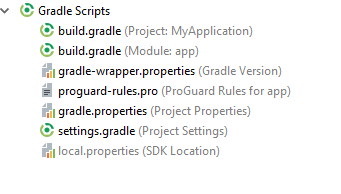
Gambar 2.5. Bagian Resource Proyek Android [5]

Bagian Res berisi berkas – berkas yang mengatur sejumlah resource atau sumber daya untuk proyek aplikasi. Resource yang terdapat pada bagian ini bukan berupa kode algoritma, melainkan layout aplikasi, sumber gambar, ikon, string teks hingga style.

Di bagian Res ini terdapat 4 folder utama dengan fungsi yang berbeda, antara lain :

* Drawable : folder yang berisikan gambar, custom button.
* Layout : folder yang berisikan layout / tata letak tampilan aplikasi
* Mipmap : folder yang berisikan ikon.
* Values : folder yang berisikan berkas dasar yang digunakan untuk mengkustomisasi tampilan aplikasi.

1. Gradle

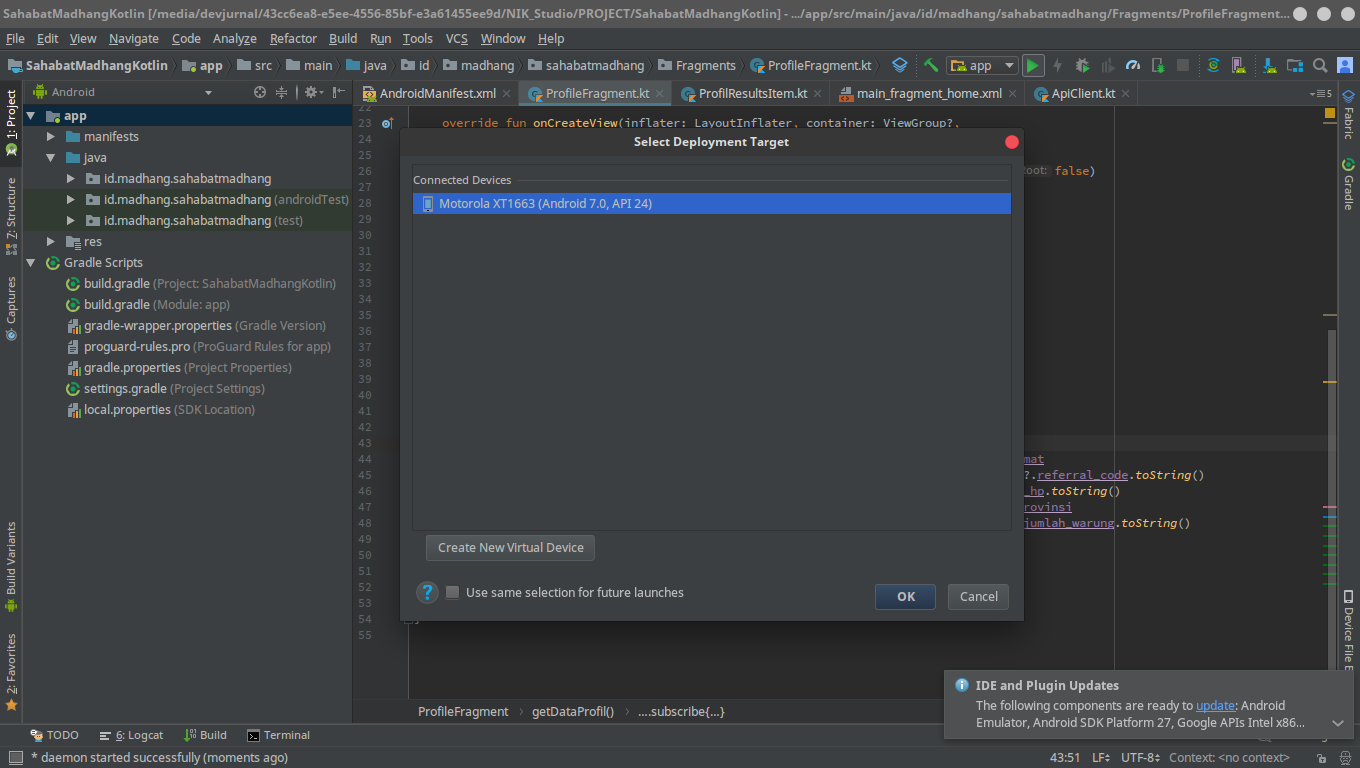


Gambar 2.6. Bagian Gradle Proyek Android [5]

Secara default Android merupakan build tools yang digunakan oleh Android Studio. Fungsinya adalah untuk membantu kita mengkompilasi dan menjalankan source code aplikasi yang kita kembangkan berdasarkan konfigurasi di Gradle. Gradle sendiri juga mendukung manajemen proyek dalam hal penambahan library di luar framework Android.

* + - 1. **Cara Melakukan Pengujian Project**

Untuk dapat melakukan pengujian project aplikasi yang dikembangkan menggunakan IDE Android Studio ada 2 macam metode yaitu menjalankan pengujian pada real device android atau pada emulator android.



Gambar 2.7. Tampilan Pilihan Device Untuk Pengujian Project

Gambar di atas menampilkan real device android yang dapat digunakan untuk melakukan pengujian aplikasi. Tekan tombol OK, untuk dapat memulai proses instalasi dan pengujian aplikasi.

* + 1. **Bahasa Pemrograman JAVA dan KOTLIN**

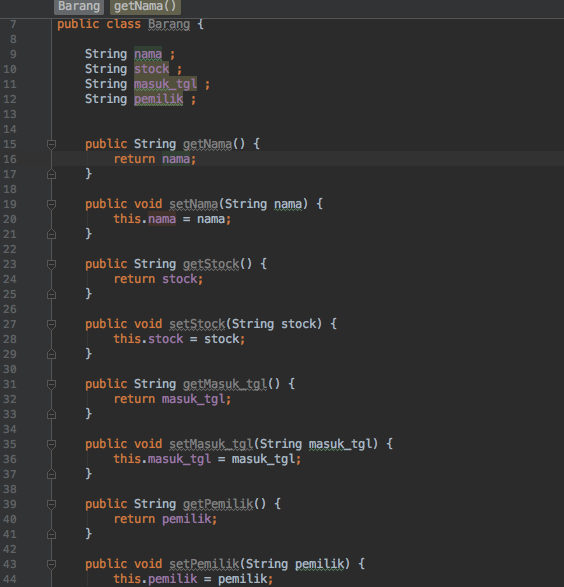
Java dan Kotlin merupakan 2 ( dua ) bahasa pemrograman yang sering digunakan dan direkomendasikan oleh Google untuk mengembangkan aplikasi Android. Java sendiri merupakan bahasa pemrograman berorientasi objek yang merupakan pengembangan dari bahasa C . Java memiliki keunggulan karena lebih sederhana dan lebih mudah digunakan dibandingkan bahasa C. Selain itu, bahasa java dapat dijalankan pada berbagai platform dan pada berbagai sistem operasi karena Java berjalan di atas JVM ( Java Virtual Machine).

Bahasa pemrograman Kotlin merupakan bahasa yang dikembangkan oleh jetBrain. Kotlin resmi menjadi bahasa pemrograman official IDE Android Studio setelah diperkenalkan pada Google I/O 2017. Bagi para pengembang aplikasi Android, Kotlin memiliki fitur yang menarik karena bahasa pemrograman ini dikembangkan oleh JetBrain yang mengembangkan IDE Android Studio dan secara resmi di dukung oleh Google Inc sebagai pengembang sistem operasi Android [9].

Kotlin merupakan turunan dari bahasa pemrograman Java, oleh sebab itu dalam hal sintax tidak jauh berbeda dengan Java. Beberapa hal yang membedakan Kotlin dan Java antara lain :

* + - 1. **Deklarasi Variabel dan Class**

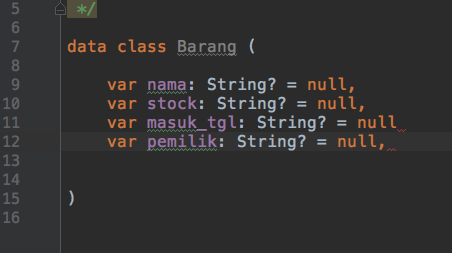
Pada Java , jika mendeklarasikan variable baru ditulis dengan : [10]



Gambar 2.8. Deklarasi variable pada Java

Pada gambar diatas untuk dapat membuat variable pada java , terlebih dahulu harus dituliskan tipe data variable yang akan dibuat diikuti dengan nama variable. Jika ingin memberikan nilai atau mendapatkan nilai yang dimiliki oleh variable , harus dibuat method *Set* dan method *Get.*

Sedangkan pada Kotlin, dapat dituliskan dengan cara :

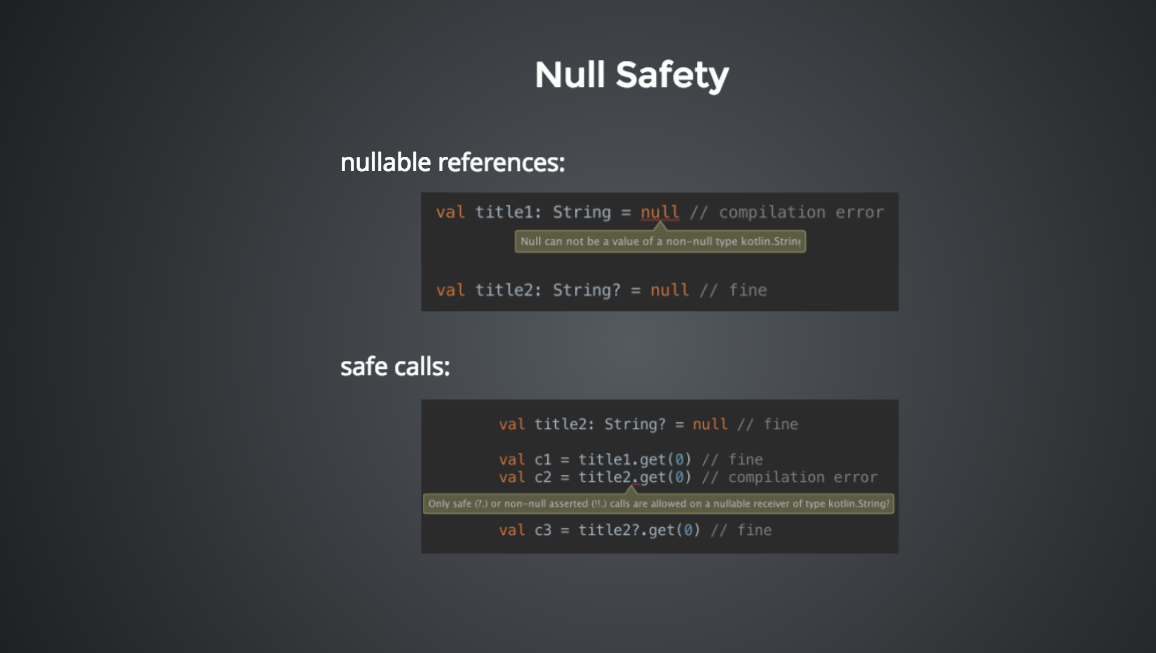
****

Gambar 2.9. Deklarasi Variable Pada Kotlin

Pada gambar diatas untuk dapat membuat variable pada kotlin diawali dengan *var*  atau *val.* Var digunakan untuk variable yang nilainya dapat berubah dinamis, sedangkan val digunakan untuk variable yang memiliki nilai final statis. Setelah menuliskan var, diikuti dengan nama variable yang akan dibuat. Untuk tipe data tidak harus dideklarasikan, namun jika ingin memberikan nilai awal harus memberikan tipe data agar compiler mengetahui tipe data nilai awal.

Jika ingin memberikan nilai atau mendapatkan nilai yang dimiliki oleh variable , tidak diperlukan method *Set* dan method *Get.* Pada bahasa pemrograman kotlin, method *Set and Get* sudah diakomodasi langsung oleh compiler. Hal ini lah yang menyebabkan sintaks untuk bahasa pemrograman kotlin lebih sederhana dibandingkan dengan bahasa pemrograman java.

* + - 1. **Null Safety**

****

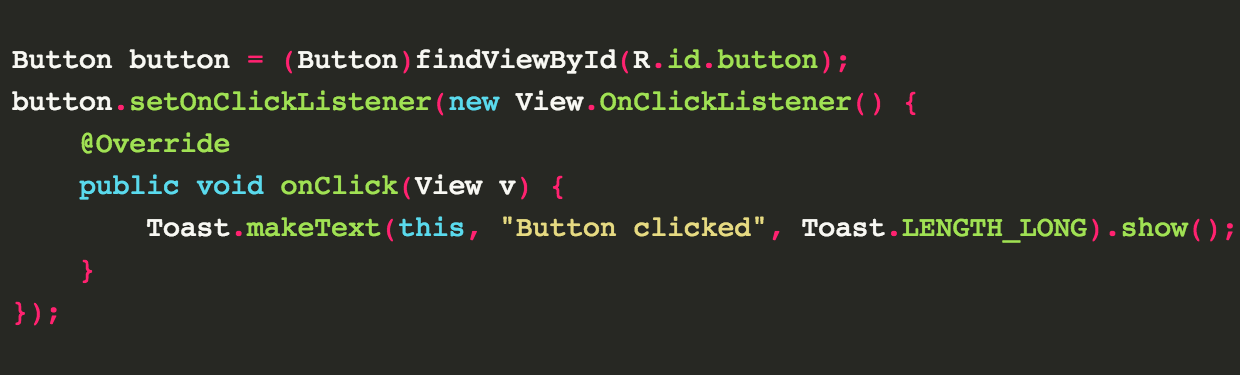
Gambar 2.10. Contoh null safety pada kotlin

Pada gambar diatas dapat dilihat bahwa terdapat peringatan pada variable c2 dikarenakan variable tersebut dapat bernilai null / kosong. Agar tidak terjadi eror saat aplikasi dijalankan, Kotlin telah memberikan peringatan terlebih dahulu untuk menambahkan *null safe* berupa “?” sebelum “.get” .

Hal ini tidak terdapat pada bahasa Java, walaupun sudah dilengkapi dengan sintaks *try … catch* … sintaks tersebut tidak dapat menyelesaikan masalah pada null variable.

* + - 1. Tidak diperlukan semicolon atau titik koma di bagian akhir sintaks.
      2. Integrasi Lambda pada Kotlin untuk menyederhanakan sintaks.

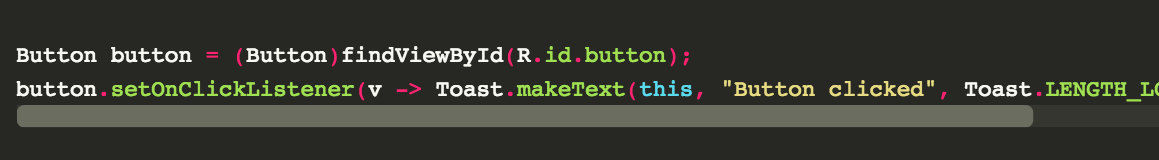
Pemanggilan fungsi OnClick pada Java:

****

Gambar 2.11. Penerapan OnClick pada Java

Pada bahasa pemrograman java Android, untuk dapat memanggil fungsi *onClick*, harus dibuat object baru berupa View OnClickListener.

Pemanggilan fungsi OnClick pada Kotlin yang sudah menerapkan Lambda:



Gambar 2.12 Penerapan OnClick pada Kotlin

Pada gambar di atas, untuk memanggil fungsi onClick terlihat lebih sederhana karena bantuan dari sintaks Lambda.

* + 1. **Retrofit dan Reactive pada Android**

Retrofit merupakan rest client library yang menggunakan komponen OkHttp untuk android yang di kembangkan oleh Squere. [11] Library ini bekerja dengan baik untuk otentikasi dan berinteraksi dengan API. Retrofit adalah kelas di mana antarmuka API diubah menjadi objek yang dapat dipanggil. Secara default, retrofit akan memberi default memungkinkan untuk platform, namun memungkinkan penyesuaian.

Secara default respon yang diterima oleh retrofit berupa *responseBodyOkHttp*, namun converter tambahan dapat digunakan untuk meningkatkan kinerja library retrofit ini. Berikut ini adalah 6 modul yang dapat ditambahkan untuk mengkonversi respon yang diterima oleh retrofit: [12]

* Gson: mengkonversi JSON yang diterima menjadi format Gson
* Jackson: mengkonversi JSON yang diterima menjadi format Jackson
* Moshi: mengkonversi JSON yang diterima menjadi format Moshi
* Protobuf : mengkonversi JSON yang diterima menjadi format Protobu
* Wire: mengkonversi JSON yang diterima menjadi format Wire
* Simple XML: mengkonversi JSON yang diterima menjadi formatSimple XML

Reactive Programming merupakan sebuah paradigm pemrograman yang berorientasi pada aliran data (*data flows*) dan penyebaran dari perubahan data. RectiveX merupakan sebuah kamus / pustaka / *library* untuk menggabungkan *asynchronus* dan *event-base* program dengan menggunakan *observable sequences*. Jadi Rx java dan Rx Kotlin merupakan sebuah *dependency* atau *library* yang digunakan untuk mengimplementasikan reactive programming (*asyncrhronus and event-base*). Untuk dapat menggunakan library ini ada beberapa langkah yang harus dilakukan , yaitu :

* + - 1. Menambahkan Gradle App

Tambahkan baris kode berikut pada bagian library gradle :

**

Gambar 2.13. Gradle RxKotlin [12]

Silakan mengganti “x.y.z “ dengan versi terbaru dari library rxKotlin.

* + - 1. Menambahkan repository pada Gradle project

Tambahkan baris kode berikut pada bagian repository :

**

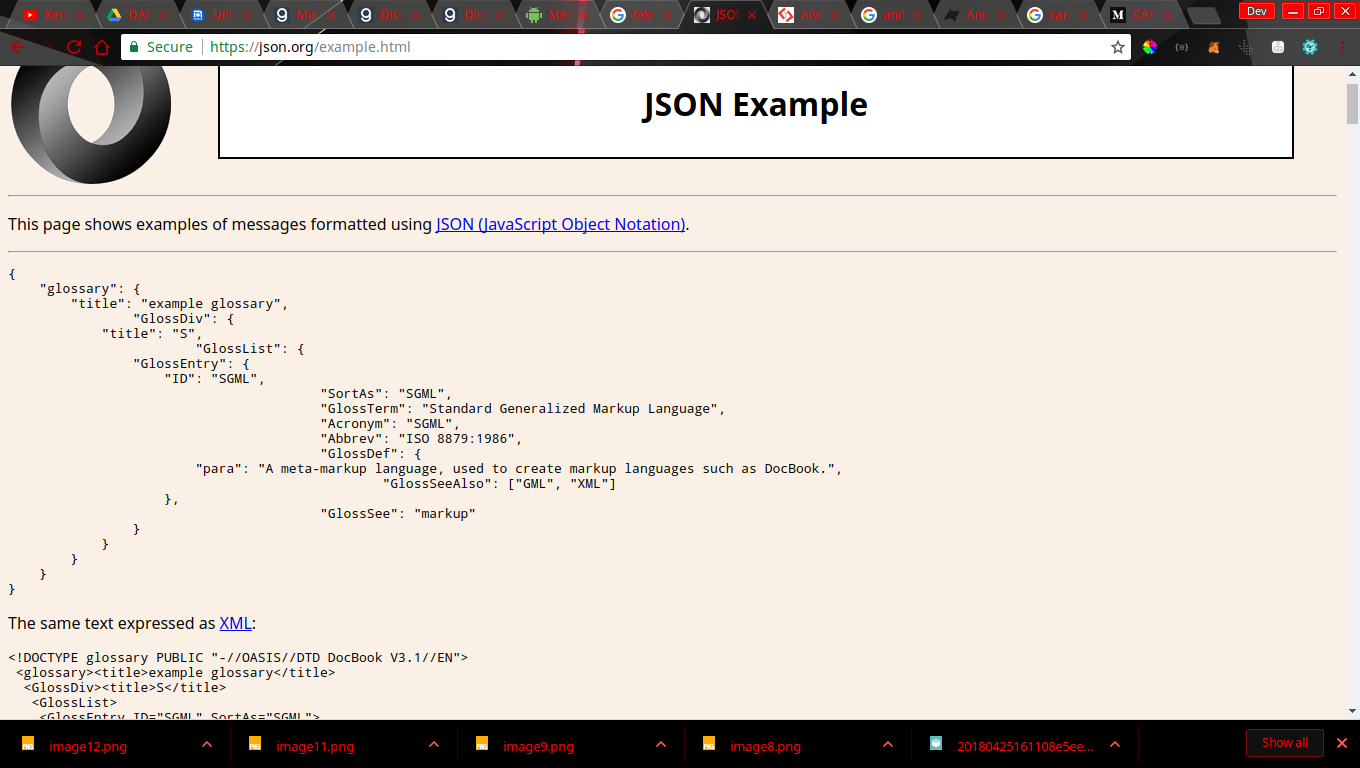
Gambar 2.14. Repository Maven Jitpack [12]

dan tambahkan kode berikut ini pada bagian dependencies :

**

Gambar 2.15. Dependency RxKotlin [12]

* + 1. **JSON (Java Script Object Notation)**

****

Gambar 2.16 Contoh format JSON [13]

JSON atau JavaScript Object Notation merupakan format struktur data yang mudah dibaca dan digunakan untuk mengirimkan data antara server dan aplikasi klien. JSON membaca data dari webserver dan memberikan data tersebut untuk dapat ditampilkan pada halaman aplikasi. [14]

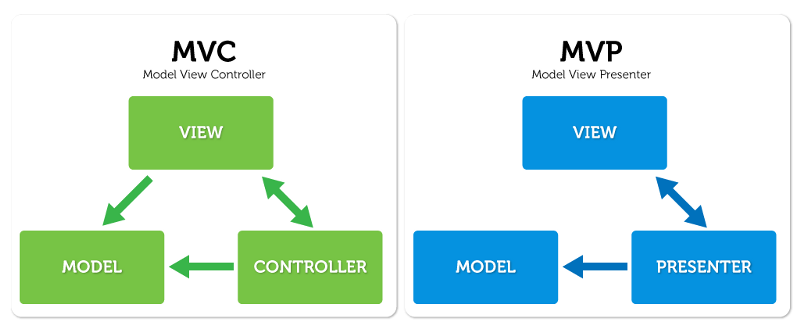
JSON berbasis teks/tulisan dengan format yang dapat dibaca dan dikenali oleh manusia untuk merepresentasikan struktur data sederhana dan array asosiatif. JSON merupakan bahasa independen yang lengkap dan menggunakan konvensi yang familiar bagi para programmer bahasa C, antara lain bahasa pemrograman C, C++, C#, Java, JavaScript, Perl, Python, dan lainnya.

Secara garis besar proses komunikasi antara klien dan server yang memberikan respon berupa JSON adalah klien meminta ke server sebuah data , setelah menerima permintaan tersebut server akan memproses database dan menciptakan sebuah file Json berisi data yang dibutuhkan oleh klien. [15]

JSON memiliki beberapa kelebihan diantaranya sebagai berikut :

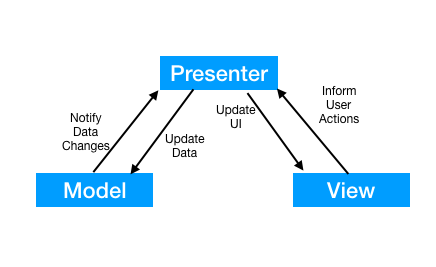
1. Mempunyai banyak tipe data.
2. Tipe Data JSON diantaranya : *number, string, boolean, array*.
3. Dapat diakses dengan mudah sebagai *object* JSON.
4. *Value* dapat diambil dengan mudah.
5. Semua browser telah *support* JSON.
6. API yang simple.
7. Di *JavaScript*, *object* cepat direalisasi.
8. *AJAX Toolkits support* dengan JSON.
9. Sepenuhnya otomatis *serialize object JavaScript*.
   * 1. **Arsitektur Model View Presenter ( MVP )**

Arsitektur Model View Presenter atau yang lebih dikenal dengan istilah MVP termasuk dalam pattern arsitektur yang digunakan untuk mengembangkan suatu sistem perangkat lunak. MVP merupakan pengembangan dari arsitektur lainnya yaitu MVC atau *Model View Controller* dan paling sering digunakan untuk membangun aplikasi yang mengancalkan user interface (*frontend*). MVP (*Model View* Presenter) pertama kali muncul di IBM dan lebih terlihat di Taligent the 1990's [16]. Arsitektur ini berfungsi sama seperti arsitektur lainnya yang digunakan dengan tujuan agar mempermudah beberapa pengembang perangkat lunak dalam bekerja bersamaan.



Gambar 2.17 Diagram sederhana MVC (kiri) dan MVP (kanan) [7]

Perbedaan dari MVC dan MVP dapat dilihat bahwa pada MVP *View* tidak memiliki komunikasi langsung dengan model melainkan harus melalui *Presenter*. Sedangkan pada arsitektur MVC *Controller* dapat dimanfaatkan dalam proses pengaturan logika bisnis dari *Model* dan dari *Model* dapat langsung melakukan komunikasi data ke *View*.



Gambar 2.18. Alur Dan Aturan Model MVP [2]

Agar aplikasi mudah dikembangkan dan pengembang yang mudah dipelihara perlu mendefinisikan lapisan yang terpisah dengan baik. Jadi dengan menggunakan MVP :

1. Tugas latar belakang (*logic, business model*) dipisahkan dari *activity* / *view* / *fragment* agar tidak terlepas dari sebagian dari *life-cycle*.
2. Tugas kompleks dipecah menjadi yang lebih sederhana dan mudah dipecahkan bila ada masalah (mudah untuk di*maintenance*).
3. Membuat *View* (*Activity, Fragment, View*) menjadi independen dari sumber.
4. Memfasilitasi pengujian unit / *unit test* secara otomatis.
   * 1. **Firebase**

****

Gambar 2.19 Logo Firebase

Firebase adalah sebuah layanan infrastruktur *backend-as-a-service* (BaaS) yang diakuisisi oleh Google pada bulan oktober tahun 2014. [17]

Beberapa fitur yang dimiliki oleh Firebase adalah sebagai berikut :

1. Firebase Analytics.
2. Firebase Cloud Messaging dan Notifications.
3. Firebase Authentication.
4. Firebase Remote Config.
5. Firebase Real Time Database.
6. Firebase Crash Reporting.

Semua fitur tersebut telah dikemas dalam SDK ( *Software Development Kit)* sehingga dapat dengan mudah digunakan oleh para pengembang aplikasi untuk focus dalam menyelesaikan masalah klien dan tidak terlalu memikirkan managemen database. Sebagai infrastruktur BaaS, firebase dapat digunakan diberbagai platform diantaranya adalah pada Website, Mobile Android dan juga platform IOS.

* + - 1. **Firebase Authentication**

Firebase User Authentification merupakan sebuah fitur dari Firebase yang memungkinkan aplikasi untuk melakukan *user authentification* menggunakan berbagai macam metode autentifikasi menggunakan Facebook, Google , Twitter, maupun Github, *email and password*, *custom authentification system integration*, hingga *anonymouse* *authentification*.

* + - 1. **Firebase Realtime Database**

Firebase Realtime Database adalah database yang di-*host* di *cloud*. Data disimpan sebagai JSON (JavaScript Object Notation) dan disingkronkan secara realtime ke setiap klien yang terhubung, apabila ada perubahan pada data yang tersimpan, maka setiap user yang terhubung akan menerima *update* data secara otomati

* + - 1. **Firebase Crash Reporting**

Firebase Crash Reporting adalah fitur dari Firebase yang akan memberikan pemberitahuan jika terjadi kesalahan saat aplikasi dijalankan. Data- data yang ditampilkan pada Crash Report ini cukuplah lengkap, diantaranya adalah waktu terjadinya kesalahan, penyebab kesalahan serta jenis perangkat yang mengalami kesalahan. Laporan ini sangatlah bermanfaat bagi pengembang aplikasi untuk dapat memperbaiki kesalahan pada aplikasinya.

* + 1. **Unified Modeling Language ( UML )**

*Unified Modeling* *Language* (UML) adalah sebuah bahasa pemodelan untuk sistem atau perngkat lunak yang berparadigma beriontasi objek (Nugroho, 2010). Pemodelan sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kompleks sedemikian rupa sehingga lebih mudah dipelajari dan dipahami.

UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML dapat membuat model untuk semua jenis aplikasi perangkat lunak, dimana aplikasi tersebut dapat berjalan pada perangkat keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi UML juga menggunakan class dan operation dalam konsep dasarnya, maka lebih cocok untuk penulisan perangkat lunak dalam bahasa-bahasa berorientasi objek.

* + - 1. **Use Case Diagram**

*Use case diagram* merupakan titik awal yang baik dalam memahami dan menganalisis kebutuhan sistem pada saat perancangan [18](Indrajani, 2015). *Use case diagram* dapat digunakan untuk kebutuhan apa saja yang diperlukan dalam suatu sistem, sehingga sistem dapat digambarkan dengan jelas bagaimana proses dari sistem tersebut, bagaimana cara *actor* menggunakan sistem, serta apa saja yang dapat dilakukan pada suatu sistem



Gambar 2.20 Contoh Usecase Diagram. [19]

Gambar diatas merupakan contoh usecase diagram pada sebuah sistem peminjaman ruang. Di dalam use case tersebut terdapat 2 actor yang terlibat. Untuk dapat mengetahui bagian – bagian dari usecase diagram , dapat dilihat pada table berikut ini :

Tabel 2.3. Simbol-simbol Use Case Diagram

|  |  |  |
| --- | --- | --- |
| Nama | Simbol | Keterangan |
| *Use Case* | D:\KP\1 AA Laporan KP\Diagram\usecase.png | Merupakan perilaku sistem. Terdiri dari predikat dan keterangan |
| *Actor* | D:\KP\1 AA Laporan KP\Diagram\actor.png | Merupakan pengguna yang terlibat dengan sistem |
| *Relationship* | D:\KP\1 AA Laporan KP\Diagram\relationship uc.png | Menghubungkan actor dengan use case |
| *Boundary* | D:\KP\1 AA Laporan KP\Diagram\boundary.png | Batasan yang dari sistem terhadap lingkungan |

Tabel diatas merupakan simbol – simbol yang sering digunakan dalam pembuatan *usecase diagram*. Actor merupakan pengguna yang terlibat didalam sistem. Actor tidak hanya mewakili manusia / user, tetapi juga dapat mewakili sistem lain yang terlibat.

* + - 1. **Class Diagram**

*Class diagram* biasanya digunakan untuk menggambarkan perbedaan yang mendasar antara *class-class*, hubungan antara *class*, dan dimana sub-sistem *class* tersebut [18]. Adapun simbol yang digunakan dalam *class diagram* adalah sebagai berikut:

Tabel 2.4. Simbol-simbol Class Diagram

| Nama | Simbol | Keterangan |
| --- | --- | --- |
| *Class* | D:\KP\1 AA Laporan KP\Diagram\class.png | Digunakan untuk mendefinisikan property/atribut class (bagian tengah). Sedangkan bagian akhir untuk mendefinisikan method – method dari sebuah class. |
| *Asosiasi* | D:\KP\1 AA Laporan KP\Diagram\Asosiasi.png | Digunakan untuk menghubungkan antara 2 class. |
| *Composition* | D:\KP\1 AA Laporan KP\Diagram\composition.png | Digunakan apabila class tidak merupakan bagian dari suatu class. |
| *Dependency* |  | Digunakan untuk menunjukkan operasi pada suatu class yang menggunakan class yang lain. |
| *Agregation* | D:\KP\1 AA Laporan KP\Diagram\agregasi.png | Digunakan untuk mengindikasikan keseluruhan bagian relationship dan biasanya disebut sebagai relasi. |

Tabel diatas berisikan daftar simbol – simbol yang sering digunakan dalam pembuatan class diagram. Tidak semua simbol tersebut digunakan dalam satu class diagram, hal itu tergantung dari hubungan antar entitas yang terdapat pada class diagram.

* + - 1. **Activity Diagram**

*Activity Diagram* atau Diagram aktivitas adalah bentuk visual dari alur kerja yang berisi aktivitas dan tindakan, yang juga dapat berisi pilihan, atau pengulangan [20]. Dalam *Unified Modeling Language* (UML), diagram aktivitas dibuat untuk menjelaskan aktivitas komputer maupun alur aktivitas dalam organisasi. Selain itu diagram aktivitas juga menggambarkan alur kontrol secara garis besar.

Diagram aktivitas memiliki komponen dengan bentuk tertentu, dihubungan dengan tanda panah. Panah tersebut mengarahkan urutan aktivitas yang terjadi, dari awal sampai akhir. Yang perlu diperhatikan yaitu diagram aktivitas bukan menggambarkan aktivitas sistem yang dilakukan aktor, tetapi menggambarkan aktivitas yang dapat dilakukan oleh sistem. .

*Activity diagram* pada beberapa waktu dianggap sama seperti *flowchart* (diagram alur), namun meskipun diagram terlihat seperti sebuah diagram alur, tetapi sebenarnya berbeda. *Activity diagram* menunjukkan aliran yang berbeda seperti paralel, bercabang, bersamaan dan tunggal.

Tujuan utama dari *Activity Diagram* adalah menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses agar lebih mudah dipahami. *Activity diagram* juga digunakan untuk menunjukkan aliran pesan dari satu aktivitas ke aktivitas lainnya. Adapun tujuan pembuatan *Activity Diagram* lainnya yaitu :

1. Menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses.
2. Dipakai pada business modeling untuk memperlihatkan urutan aktifitas proses bisnis.
3. Membantu memahami proses secara keseluruhan.
4. Menggambarkan aliran paralel, bercabang dan bersamaan dari sistem.

Komponen yang ada pada *Activity Diagram* antara lain:

Tabel 2.5.Simbol-simbol *Activity Diagram*

| **Nama** | **Simbol** | **Keterangan** |
| --- | --- | --- |
| *Start Point/Initial State* (Titik Mulai) |  | *Start Point* merupakan lingkaran hitam kecil, yang menandakan tindakan awal atau titik awal aktivitas untuk setiap diagram aktivitas. |
| *Activity* (Aktivitas) |  | *Activity* menunjukan aktivitas yang dilakukan atau yang sedang terjadi dalam *activity* *diagram*. |
| *Action Flow* (Arah) |  | *Action Flow* digunakan untuk transisi dari suatu tindakan ke tindakan yang lain atau menunjukan aktivitas selanjutnya setelah aktivitas sebelumnya. |
| *Decision* (Keputusan) |  | *Decision* adalah suatu titik atau *point* pada *activity* *diagram* yang mengindikasikan suatu kondisi dimana ada kemungkinan perbedaan transisi. |

* 1. **Deskripsi Tempat KKI**

PT. Dian Nuswantoro Teknologi dan Informasi merupakan perusahaan di bidang teknologi komputerisasi yang berdiri sejak tahun 2006. PT Dian Nuswantoro Teknologi dan Informasi ini lebih dikelan dengan nama PT Dinustek berfokus pada pelayanan tentang informasi dan teknologi (IT) , Layanan jaringan internet (Internet Service Provider) dan layanan pembuatan aplikasi (Software Development).

Informasi tentang PT Dinustek :

Alamat : Jl. Arjuna No 36 ,Kota Semarang, Jawa Tengah, Indonesia.

Telepon : (024) 3568492

Email : marketing@dinustek.com

Website : www.dinustek.com

* 1. **Logo dan Makna Tempat KKI**

****

Gambar 2.21. Logo PT Dian Nuswantoro Teknologi dan Informasi [21]

Logo PT Dinustek terbagi menjadi 3 (tiga) bagian yaitu :

* + - 1. Simbol Api

Simbol api berwarna oranye melambangkan solusi untuk klien yang bermaksud memberikan penyelesaian masalah terbaik dari sisi teknologi dan informasi bagi bisnis klien.

* + - 1. Warna Biru

Warna biru melambangkan solusi teknologi informasi yang kami tawarkan dapat dipercaya kehandalan bagi klien.

* + - 1. Bentuk Huruf

Huruf-huruf yang tersusun berbentuk small caps melambangkan keramah-tamahan perusahaan kepada klien, dan nuansa kekeluargaan di perusahaan. Bentuk huruf yang dinamis namun kuat menggambarkan tim perusahaan yang kreatif dan solid.

* 1. **Struktur Organisasi Tempat KKI**

Struktur organisasi PT Dinustek:

* + 1. **Direktur Utama**

Mohamad Sidiq, S.Si., M. Kom

* + 1. **Direktur**

Dr. Pulung Nurtantio Andono, ST, M. Kom

* + 1. **Manager of Software Department**

Abu Salam, M. Kom

* + 1. **Kepala Divisi Startup Software**

Moh. Tofa Nurzaki, S. Kom

* 1. **Visi dan Misi Tempat KKI**

Visi : Menjadi perusahaan teknologi informasi dan komunikasi kelas dunia.

Misi : Memberikan solusi keseluruhan teknologi informasi dan komunikasi untuk klien.

# BAB III

# METODE PENELITIAN

* 1. **Instrument Penelitian**

Penyusunan suatu proyek penelitian, sesuai dengan data-data permasalahan pokok yang dihadapi. Data dapat dikatakan baik apabila data tersebut dapat mewakili objek yang sedang diteliti dan untuk mendapatkan data yang baik diperlukan metode atau serangkaian cara yang sesuai dengan kebutuhan penelitian.

* 1. **Prosedur Pengambilan Data**

Untuk mendapatkan data yang benar, akurat dan relevan serta sesuai dengan sumber data dan tujuan penyusunan Laporan Kuliah Kerja Industri ini, maka penulis dalam pengumpulan data menggunakan beberapa teknik antara lain adalah sebagai berikut :

* + 1. **Observasi**

Teknik pengumpulan data dengan cara melakukan pengamatan dan terlibat secara langsung di dalam sistem pengembangan yang telah ada di dalam Tim Pengembang Aplikasi Tukutu yang berada di bawah pengawasan PT Dinustek.

* + 1. **Studi Literatur**

Melakukan berbagai pencarian terhadap teori-teori dan literatur yang dapat digunakan sebagai dasar melakukan penelitian. Serta mempelajari penelitian yang pernah dilakukan sebelumnya sehingga dapat lebih memahami teori yang ada.

* + 1. **Forum Diskusi & Tutorial**

Teknik pengumpulan data dengan cara membuat sebuah forum diskusi, yang berisikan para pengembang aplikasi. Peneliti membuka diskusi pada group chat tim pengembang aplikasi Tukutu, guna mengetahui masalah yang sebenarnya dialami oleh

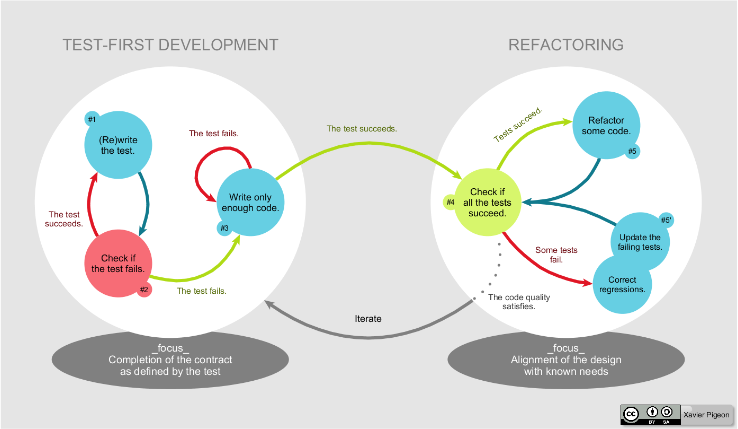
tim pengembang. Secara rutin melakukan pertemuan dengan pengembang aplikas Tukutu untuk berdiskusi.

* 1. **Teknik Analisis Data**

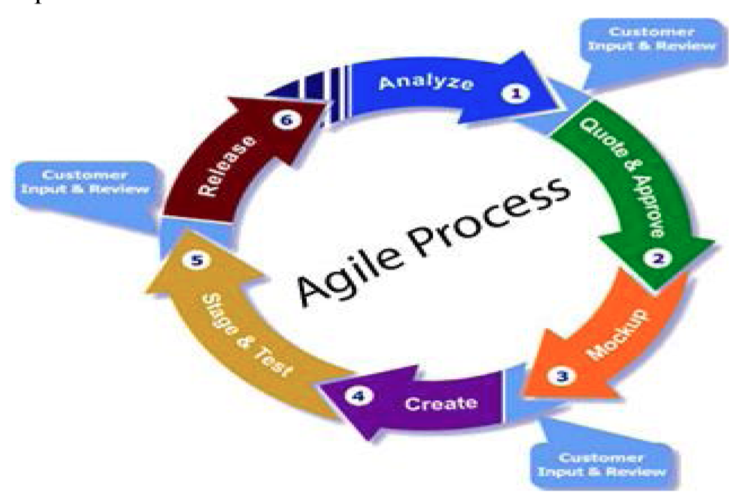
Data yang diperoleh dari literatur, forum diskusi dan Observasi akan diolah sesuai dengan bagian – bagian dari sistem aplikasi Tukutu. Dari hasil analisa yang didapatkan akan dilakukan analisa masalah dengan pengembangan fitur yang cocok sebagai solusi dari masalah tersebut, sehingga diharapkan fitur baru yang dibuat akan mampu menyelesaikan masalah yang ada dan berfungsi dengan baik.

* 1. **Metode**

Dalam proses mengembangkan aplikasi Tukutu ini tim pengembang menggunakan proses pengembangan *Extreme Programming* yaitu *Test-Driven Development* yang diterapkan di dalam metode *Agile* dengan yaitu dengan mempersiapkan skenario test case sebelum membuat fungsional code. Berikut ini adalah tahapan – tahapan yang dilakukan dalam menerapkan metode *Test driven Development* :



Gambar 3. 1. Alur Test Driven Development [4]



Gambar 3. 2. Alur Metode Agile [22]

Penjelasan dari masing – masing proses *test driven development* adalah sebagai berikut:

* + 1. **Pembuatan Test**

Membuat *unit test* merupakan langkah pertama yang harus dilakukan selama proses TDD ketika mengerjakan sebuat fitur. Spesifikasi, deskripsi dan detail kebutuhan serta hasil yang diinginkan harus jelas dalam pembuatan setiap *test*.

Terdapat beberapa fitur utama yang wajib tersedia didalam aplikasi ini, diantaranya adalah sebagai berikut :

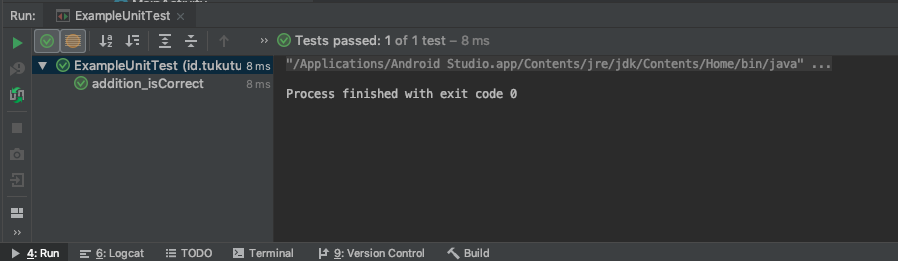
* Fitur autentikasi yang meliputi sub-fitur *login – register, change password, forgot password* dan *logout*.
* Fitur pembelian yang meliputi sub-fitur pencarian dan menampilkan brand sepatu dan produk sepatu.
* Fitur merchant yang meliputi sub-fitur menambah sepatu, menghapus dan menampilkan sepatu, memantau pesanan dan pembayaran sepatu.
* Fitur diskusi yang meliputi sub-fitur membuat pertanyaan, dan menjawab pertanyaan.
* Fitur diskon/promo yang meliputi sub-fitur diskon umum dan diskon khusus.

Berdasarkan data fitur-fitur tersebut akan dibuat suatu skenario pengujian dengan ketentuan bahwa setiap sub-fitur akan dibuatkan setidaknya dua kondisi *pengujian* yaitu kondisi pengujian positif dan negatif. Dua kondisi ini diperlukan agar dapat dipastikan bahwa fungsi yang dibuat dapan menangani ketika seluruh parameter yang diperlukan benar dan dapat mengantisipasi jika parameter yang dikirim ada yang salah.

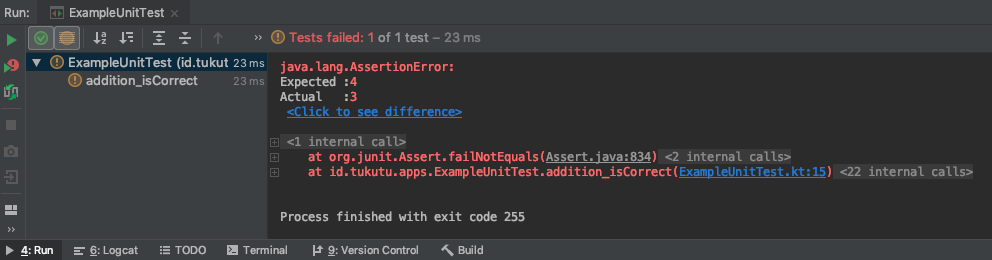
* + 1. **Menjalankan Test**

Pastikan seluruh *test* yang dilakukan telah lolos, jika belum lolos, maka perlu dilakukan iterasi atau pengulangan pengujian pada kode program hingga semua lolos *test*. Dengan menggunakan iterasi secara terus – menerus, *programmer* dapat dengan mudah memantau kinerja dari aplikasi sekaligus meminimalisir kesalahan kode setelah penambahan fitur baru.

Setiap kali selesai pembuatan *test case* untuk sebuah fitur, *test* runner akan berjalan sesuai dengan *file* yang akan dicek dengan menjalankan file test yang diinginkan dan menekan icon *runtest* yang tersedia pada android studio.



Gambar 3. 3. Contoh Test Success



Gambar 3. 4.Contoh Test Gagal

* + 1. **Menuliskan Kode**

Penulisan kode program pada fase ini memiliki tujuan utama yaitu membuat program lulus dari *test*. Untuk bagian ini kode mungkin masih kurang rapi, kurang efisien atau berantakan, hal tersebut dikarenakan untuk focus utama mengarah pada kelulusan *test* dan tidak memerlukan pembuatan fungsi tambahan diluar meluluskan *test*. Hal tersebut dikarenakan akan dilakukan *refactoring* pada tahapan kelima. Setelah kode yang dibuat dirasa akan dapat memenuuhi *test case* maka barulah dilanutkan pada tahap selanjutnya.

* + 1. **Menjalankan Ulang Test**

Pada tahapan ini dilakukan dengan tujuan agar dapat memastikan bahwa kode yang ditulis pada tahapan sebelumnya dapat memenuhi persyaratan *assertion* pada *test case* yang diinginkan. Namun bila pada saat dijalankan nantinya masih belum lulu *test case*, maka perlu dilakukan penulisan ulang kode dan akan dilakukan iterasi terus pada tahapan ini hingga seluruhnya lulus *test case*.

* + 1. **Refactoring**

Pada tahapan ini kode yang berantakan atau kurang rapi, duplikasi, dan tidak efisien, dengan adanya iterasi *testing* juga sekaligus untuk memastikan perubahan yang dilakukan menghasilkan hal yang sama sebelum di *refactoring*. Seiring dengan perkembangan aplikasi kode yang banyak, Test Driven Development (TDD) membantu membuat kode menjadi lebih mudah dipelihara dan dibaca kembali oleh programmer, baik programmer tersebut maupun programmer lain. Karena setiap fungsi yang dibuat telah diberi deskripsi, contoh input dan ouput, sehingga kode tersebut dapat menjelaskan dengan sendirinya.

Setiap kali *refactoring* selesai dilakukan, maka akan dijalankan *test* ulang guna memastikan bahwa proses *refactoring* tidak merusak *function* yang telah lolos pengujian sebelumnya.

Jika pada tahapan ini membuat aplikasi yang sesuai *test case*  menjadi tidak sesuai, maka diperlukan iterasi ulang hingga seluruhnya memenuhi *test case* atau tahapan *refactoring* ini dapat dibatalkan atau tidak dilakukan.

* 1. **Eksperimen dan Pengujian**

Proses pengembangan sistem yang digunakan di dalam penelitian ini adalah *Test Driven Developmen* dimana skenario pengujian atau *test case* harus dibuat terlebih dahulu dan dilanjutkan dengan penulisan kode aplikasi. Penulisan kode didasarkan pada skenario pengujian sehingga aplikasi yang dihasilkan telah teruji setelah selesai dikerjakan. Skenario pengujian yang dibuat harus memenuhi dua kondisi yaitu positif test dan negatif test atau skenario berhasil dan skenario ketika terjadi kesalahan pada aplikasi.

**BAB IV**

# PENERAPAN SISTEM DAN PEMBAHASAN

* 1. **Gambaran Umum Sistem**

PT Dinustek merupakan perusahaan yang mengembangkan aplikasi android yang diberi nama aplikasi TUKUTU. Aplikasi Tukutu ini digunakan untuk jual beli sepatu secara online dengan target para pecinta sepatu yang ingin membeli atau bahkan menjual koleksi mereka ke masyarakat umum.

Aplikasi Tukutu ini lebih dikembangkan untuk membantu para pemilik merk sepatu lokal untuk dapat memasarkan produk mereka sehingga dapat bersaing dengan merk internasional yang sudah lebih dahulu dikenal oleh masyarakat. Platform android dipilih karena sebagian besar target pengguna merupakan pengguna android.

* 1. **Perancangan Kebutuhan Sistem**

Untuk membangun aplikasi ini, dibutuhkan beberapa komponen, diantaranya yang terdiri dari perangkat keras (*hardware*) ,perangkat lunak (*software*), dan informasi.

* + 1. **Kebutuhan Perangkat lunak**

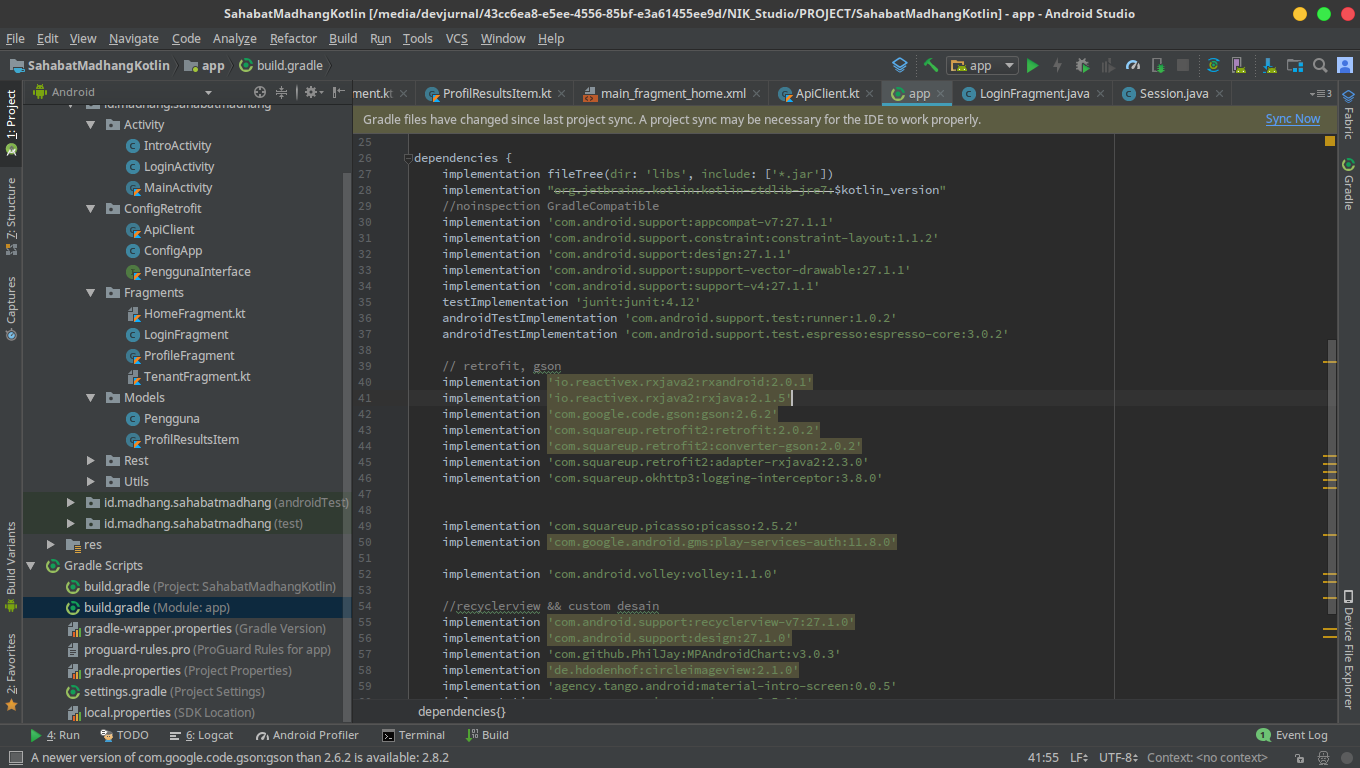
Perangkat lunak yang dibutuhkan dalam pembuatan aplikasi ini dibagi menjadi 2 kategori, yaitu:

* + - 1. Development
         1. Sistem operasi MacOS Mojave
         2. Android Studio minimal versi 3.2.0 dengan rincian:
* Minimal SDK versi 16 yaitu SDK untuk Android Jelly Bean
* Kotlin Extension



Gambar 4. 1.Gradle Kotlin

* Dependency tambahan untuk httprequest komunikasi dengan webservis



Gambar 4. 2. Retrofit & RX-Kotlin

* + - * 1. PHP (framework laravel)
        2. Postman
      1. *Production*
         1. Sistem operasi CentOS
         2. Docker
         3. PHP
    1. **Identifikasi Kebutuhan Informasi**

Terdapat kebutuhan informasi dan data dalam pembuatan aplikasi Tukutu.

1. Identifikasi Data
   1. Data Pengguna
   2. Data Merchant
   3. Data Sepatu
2. Identifikasi Informasi
   1. History Transaksi
   2. Detail Pembelian
   3. Detail Penawaran
      1. **Identifikasi Kebutuhan Hardware**

Untuk spesifikasi kebutuhan *hardware* dibagi menjadi tiga kategori yaitu *development , Testing* dan *product release* :

1. Development
2. Intel Core i5
3. Ram 8 GB
4. SSD 256 GB
5. Sistem Operasi MacOS
6. Testing
7. Lenovo Moto M
8. RAM 4GB
9. Prosesor Octacore 1.95 GHz
10. GPU Mali-T860MP2
11. Sistem Operasi Android 7.0 (Android Nougat)
12. Product Release
13. RAM 1 GB
14. Sistem Operasi Android 4.1 (Android Jelly Bean)
    1. **Analisa Perancangan Sistem**

Pada tahap ini dilakukan identifikasi dan analisis aplikasi yang meliputi identifikasi aktor yang terlibat, use case dan activity diagram :

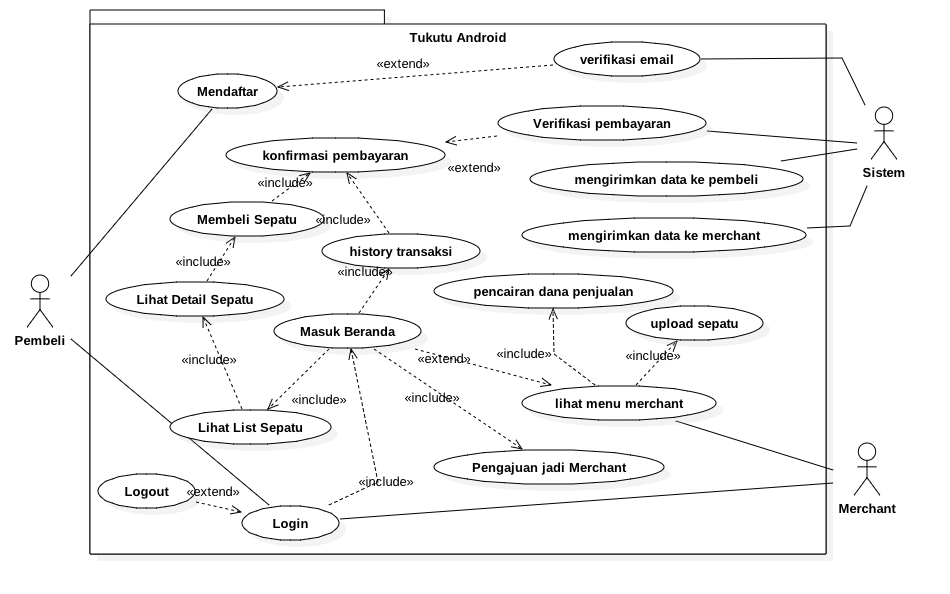
* + 1. **Identifikasi Aktor**

Aktor yang terlibat di dalam aplikasi ini terdiri atas dua actor yaitu User dan juga Sistem. Berikut ini merupakan penjelasan dari tiap actor yang terlibat

Tabel 4. 1 Identifikasi Aktor

| **No** | **Aktor** | **Deskripsi** |
| --- | --- | --- |
| 1 | Pembeli (pengguna aplikasi 1) | Merupakan aktor yang dapat melakukan transaksi pembelian sepatu melalui aplikasi Tukutu. |
| 2 | Merchant (pengguna aplikasi 2) | Merupakan aktor yang dapat melakukan transaksi pembelian serta dapat menjual sepatu melalui aplikasi Tukutu |
| 3 | Sistem (Webservis) | Merupakan aktor yang menjadi perantara komunikasi antara pengguna aplikasi yang menggunakan platform android dan server database. Sistem menjadi aktor tersendiri karena sistem ini terdapat logika dan algoritma yang terpisah dari algoritma pada platform android. |

* + 1. **Usecase Diagram**



Gambar 4. 3. Diagram Usecase

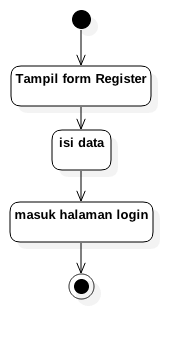
Berdasarkan gambar usecase di atas , dapat diketahui fitur – fitur utama yang dapat digunakan oleh pengguna aplikasi. Untuk dapat menggunakan fitur-fitur yang terdapat di dalam aplikasi, terlebih dahulu harus memiliki akun di aplikasi Tukutu. Jika ingin menjual barang, pengguna aplikasi harus melakukan pengajuan terlebih dahulu melalui menu pengajuan merchant yang ada di halaman profil pengguna ketika dalam mode sebagai pembeli.

Melalui menu merchant, merchant dapat mengunggah data sepatu yang akan dijual dan juga dapat mencairkan dana hasil penjualan sepatu. Selain dapat menjual sepatu, merchant juga dapat menggunakan aplikasi untuk membeli sepatu yang dijual di Tukutu kecuali sepatu yang dia jual sendiri.

* + 1. **Activity Diagram**

Activity Diagram dirancang sebagai pemodelan urutan aktivitas dan prosedur logika dari sistem yang akan dibangun. Perancangan aktivitas ini dilakukan berdasarkan usecase yang ada.

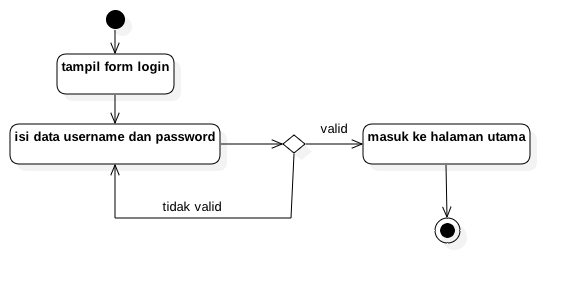
* + - 1. **Activity Diagram Register**

****

Gambar 4. 4.Diagram aktivitas Register

Ketika user menekan tombol Register / Daftar maka akan tampil form yang harus diisi oleh user. Setelah semua form berhasil diisi, maka user diarahkan ke halaman login.

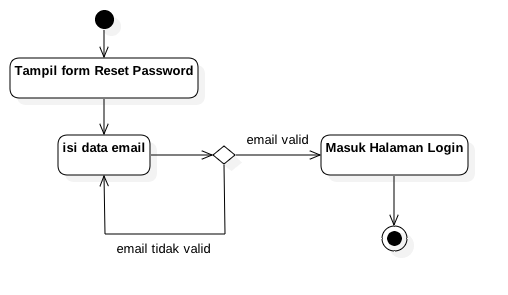
* + - 1. **Activity Diagram Login**

****

Gambar 4. 5.Diagram Aktivitas Login

Halaman login akan tampil jika user belum melakukan login, user diwajibkan mengisi username / email dan password yang akan diverifikasi oleh server. Jika username dan password yang dimasukkan valid maka user akan diarahkan masuk ke halaman utama. Namun jika tidak valid maka user harus mengisi kembali form username dan password.

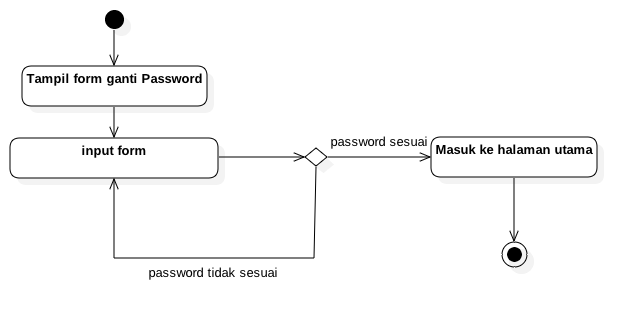
* + - 1. **Activity Diagram Reset Password**

****

Gambar 4. 6 Diagram Aktivitas Edit Password

Halaman reset password akan tampil jika user memilih menu reset password pada halaman login, user diwajibkan mengisi email yang digunakan pada saat mendaftar. Email tersebut akan diverifikasi oleh server. Jika data yang dimasukkan sesuai maka user akan diarahkan masuk ke halaman login. Namun jika tidak sesuai maka user harus mengisi kembali form email yang tersedia.

* + - 1. **Activity Diagram Edit Password**

****

Gambar 4. 7. Diagram Aktivitas Edit Password

Halaman ganti password akan tampil jika user memilih menu edit password pada halaman profil user, user diwajibkan mengisi password lama, password baru dan konfirmasi password baru yang akan diverifikasi oleh server. Jika data yang dimasukkan sesuai maka user akan diarahkan masuk ke halaman utama. Namun jika tidak sesuai maka user harus mengisi kembali form ganti password.

* 1. **Implementasi Antarmuka**
     1. **Layout Halaman Splash**

****

Gambar 4. 8. Halaman Splash

Halaman splash merupakan halaman yang pertama kali akan tampil saat membuka aplikasi Tukutu. Saat pertama kali menginstall aplikasi Tukutu, aplikasi akan mengunduh data yang akan sering digunakan dan menyimpannya ke dalam memori lokal HP pengguna. Hal ini diperlukan agar pengguna tidak perlu mengunduh data yang sama berulang kali dan juga bermanfaat untuk mengurangi kerja dari server.

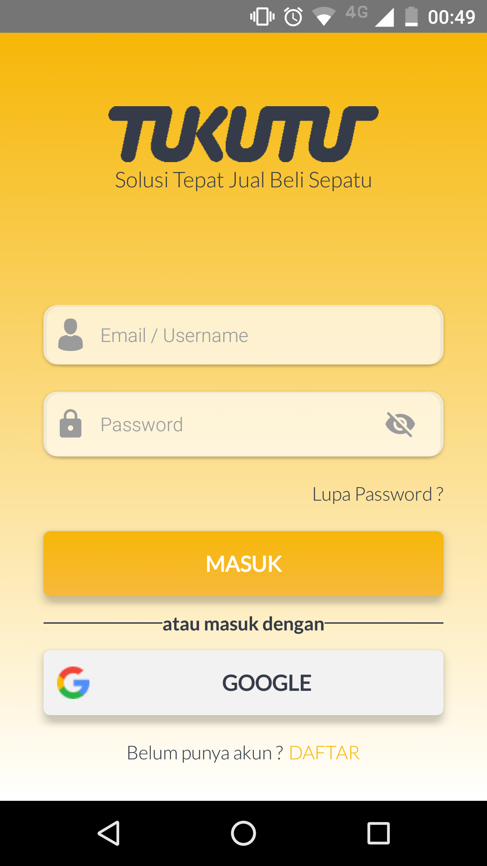
* + 1. **Layout halaman intro**

Tabel 4. 2 Tutorial Intro

|  |  |
| --- | --- |
|  |  |
|  |  |

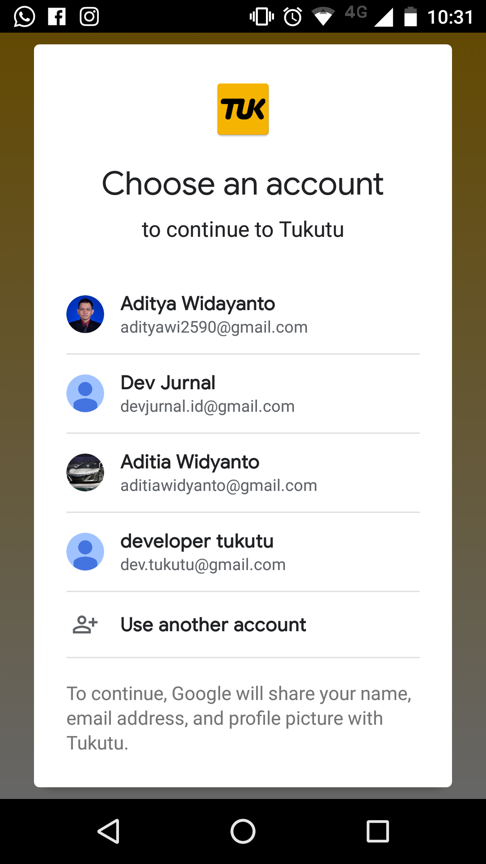
Halaman antar muka intro berisi informasi dan halaman pengenalan aplikasi Tukutu . Halaman ini hanya tampil satu kali pada saat pertama kali menjalankan aplikasi.

* + 1. **Layout Halaman Login**



Gambar 4. 9. Tampilan Antarmuka Halaman Login

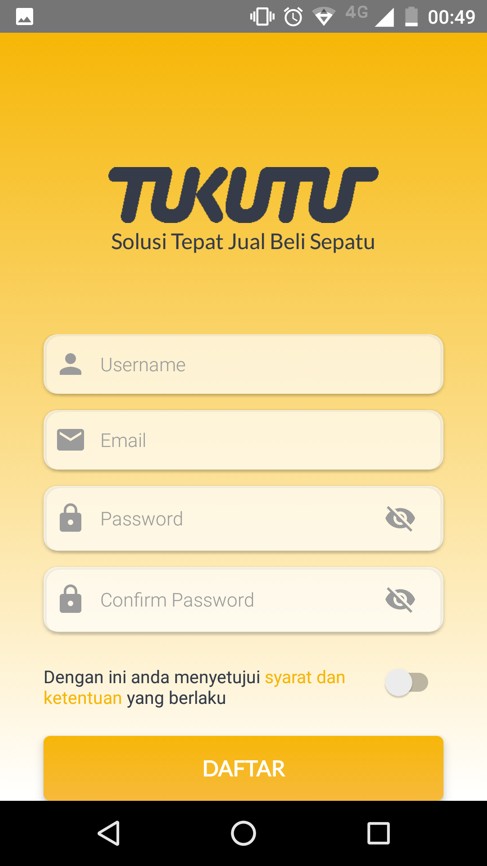
Aplikasi Tukutu ini memiliki 2 jenis login , yaitu login dengan email password, dan login dengan akun Google*.* Jika menggunakan metode login dengan akun google maka tampilan nya berubah menjadi seperti gambar di bawah ini.



Gambar 4. 10. Tampilan Antarmuka Login Google

Gambar di atas merupakan tampilan dari antarmuka Firebase Autentikasi yang digunakan sebagai halaman login melalui akun google. Dengan menginstal SDK dari firebase dan melakukan pengaturan pada halaman autentikasi aplikasi yang ada di firebase, aplikasi Tukutu dapat menggunakan fasilitas Firebase Autentikasi.

* + 1. **Layout Halaman Registrasi**

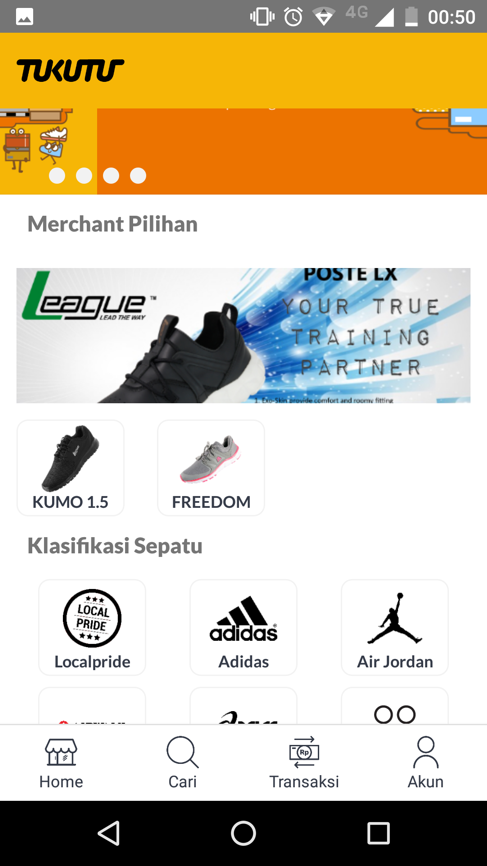


Gambar 4. 11. Tampilan Antarmuka Halaman Register

Sebelum menggunakan fitur – fitur yang terdapat pada aplikasi tukutu, pengguna aplikasi harus melakukan pendaftaran terlebih dahulu. Data – data yang diperlukan adalah username, email, password. Berdasarkan data email yang diisikan oleh pengguna, sistem akan mengirimkan pesan yang berisi link yang digunakan untuk verifikasi. Jika pendaftaran berhasil, maka pengguna akan diarahkan ke halaman login.

* + 1. **Layout halaman utama**

Berikut merupakan tampilan halaman utama dengan beberapa fitur yang ada pada *bottom tab navigation*.



Gambar 4. 12. Tampilan Antarmuka Halaman Utama

Pada halaman utama terdapat 4 bagian yaitu :

Banner

Banner yang berupa tampilan slide gambar akan digunakan untuk menampilkan promo – promo terbaru yang ada di aplikasi Tukutu.

Merchant Pilihan

Merchant pilihan merupakan fitur yang disediakan bagi brand yang bekerja sama secara khusus dengan Tukutu.

Klasifikasi Sepatu

Klasifikasi sepatu berisi brand-brand sepatu yang dijual di aplikasi Tukutu. Secara khusus Tukutu menyediakan kategori *LocalPride* yang berisikan bermacam-macam brand lokal yang ada di Indonesia. Kategori ini merupakan perwujudan dari tujuan utama pembuatan aplikasi Tukutu untuk membantu pemasaran brand sepatu lokal Indonesia.

Trivia

Trivia merupakan info grafis yang bertujuan untuk mengedukasi pengguna Tukutu. Info grafis yang disampaikan berisi tentang sepatu, pemasaran sepatu, sejarah dan perkembangan sepatu dunia.

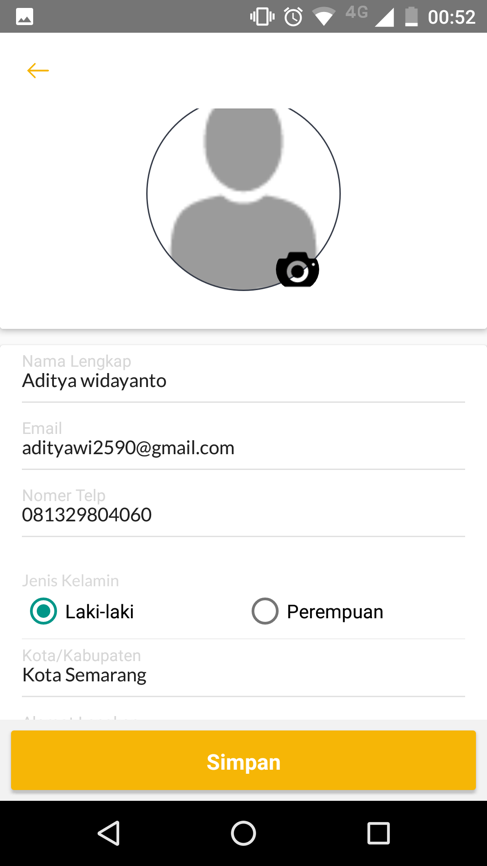
* + 1. **Layout Halaman Profil**



Gambar 4. 13. Tampilan Halaman Profil

Halaman profil dapat diakses melalui *bottom navigation* akun yang berada di sudut kanan bawah. Halaman ini berisi informasi tentang profil umum pengguna aplikasi, menu – menu bantuan serta menu untuk mendaftarkan diri menjadi merchant.

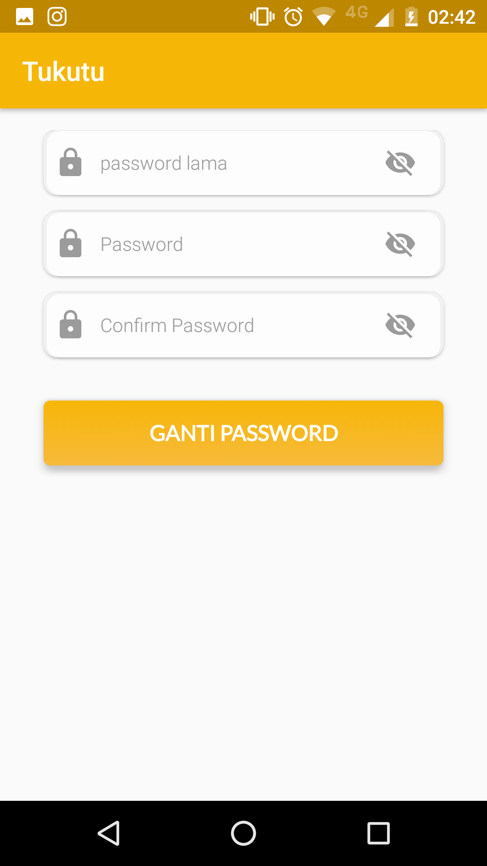
* + 1. **Layout halaman edit profil**



Gambar 4. 14. Tampilan Edit Profil

Pada halaman ini, pengguna dapat mengubah data profil yang dibutuhkan untuk transaksi di dalam aplikasi tukutu. Pada halaman ini, pengguna tidak dapat mengubah alamat email yang terhubung dengan akun tersebut.

* + 1. **Layout halaman ganti password**



Gambar 4. 15. Tampilan Edit Password

Halaman edit password dapat diakses melalui menu pengaturan akun pada halaman profil. Untuk dapat mengedit atau mengganti password, pengguna diharuskan memasukkan password lama, password baru dan konfirmasi pasword baru. Jika tidak ada kesesuaian password lama dengan password yang tersimpan di basis data server, maka password tidak dapat diganti

* + 1. **Layout history transaksi**

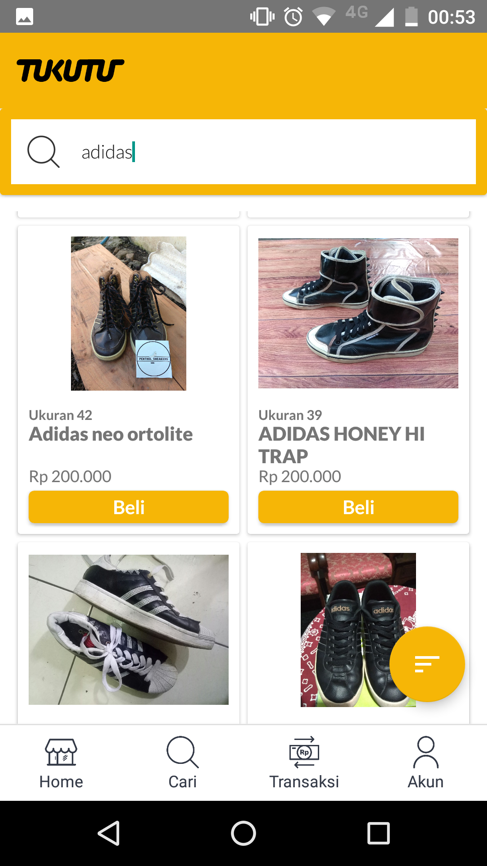


Gambar 4. 16. Tampilan History Transaksi

Halaman history transaksi dapat diakses melalui *bottomnavigation* transaksi. Halaman ini akan menampilkan history pembelian, status pembayaran, serta status pengiriman barang yang dibeli oleh pengguna.

Halaman ini memiliki 3-tab yaitu Status, Tawar dan Riwayat. Tab Status akan menampilkan history pembelian mulai dari status pembayaran sampai dengan status pengiriman. Tab Tawar akan menampilkan history penawaran yang dilakukan oleh pengguna. Jika penawaran disetujui makan pengguna harus segera melakukan pemesanan dan data dari tab tawar akan dipindahkan ke dalam tab status. Yang ke tiga adalah tab Riwayat, pada tab ini history pembelian yang berhasil dilakukan serta pembelian yang gagal atau dibatalkan akan ditampilkan.

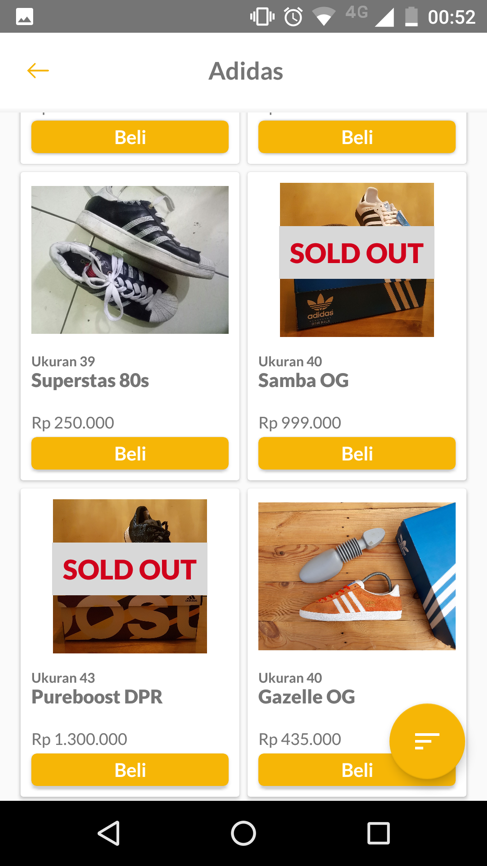
* + 1. **Layout pencarian**



Gambar 4. 17. Tampilan Halaman Pencarian

Pada halaman ini terdapat form inputan dimana pengguna dapat memasukkan kata kunci yang akan dicari. Setelah memasukkan kata kunci aplikasi Tukutu akan menampilkan data sepatu sesuai dengan kata kunci pencarian.

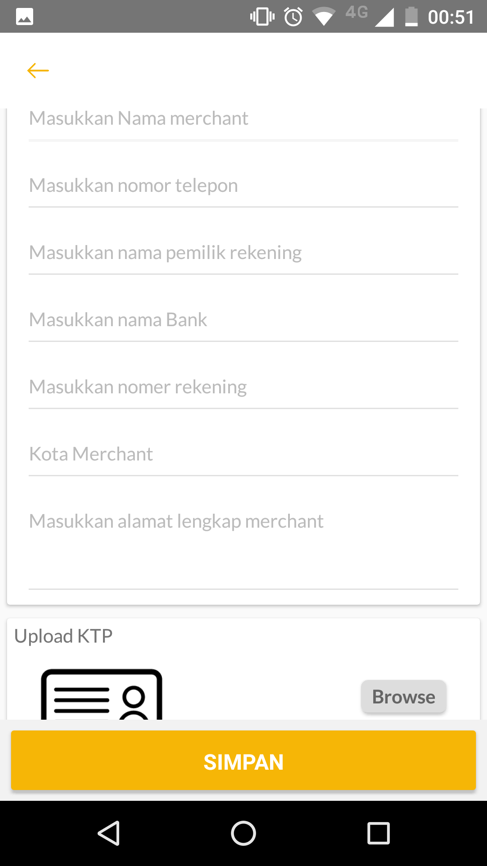
* + 1. **Layout list sepatu**



Gambar 4. 18Tampilan Halaman List Sepatu

List sepatu akan ditampilkan setelah pengguna memilih brand yang diinginkan melalui klasifikasi sepatu yang ada di halaman utama. Sepatu yang masih tersedia akan ditampilkan, sedangkan sepatu yang telah terjual habis akan diberi label *SoldOut*. Pada halaman ini pengguna dapat melakukan filter berdasarkan urutan ukuran maupun berdasarkan harga.

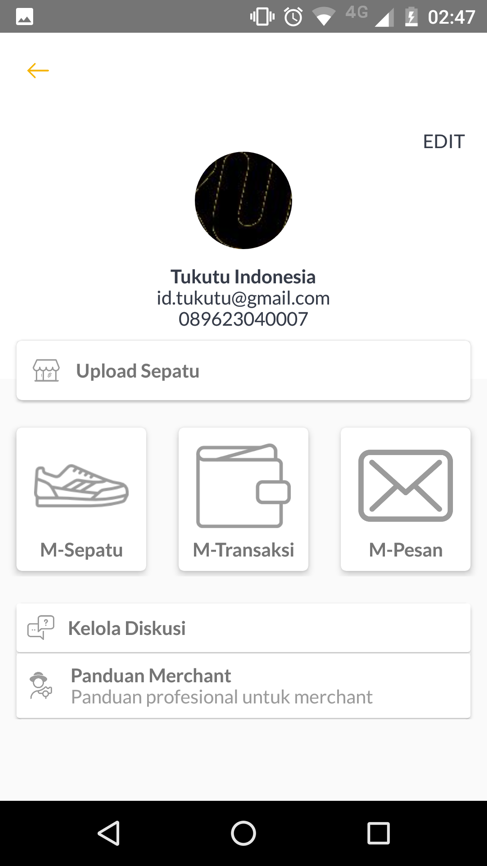
* + 1. **Layout form pengajuan merchant**



Gambar 4. 19. Tampilan Form Pengajuan Merchant

Halaman ini disediakan bagi pengguna aplikasi yang ingin berjualan di aplikasi Tukutu. Pengguna diwajibkan melengkapi data-data yang dibutuhkan serta wajib mengunggah foto KTP. Setelah mengajukan permohonan menjadi merchant, admin Tukutu akan melakukan validasi manual berdasarkan data yang tersimpan dan juga berdasarkan data KTP pengguna. Jika data yang disimpan tidak sesuai atau terindikasi palsu , admin Tukutu akan menolak pengajuan tersebut dan pengguna wajib melakukan pengajuan ulang.

* + 1. **Layout pengaturan merchant**



Gambar 4. 20. Halaman Pengaturan Merchant

Pengaturan merchant merupakan fitur yang disediakan khusus untuk merchant yang ingin menjual sepatu. Pada halaman ini terdapat 4 menu utama , yaitu :

Upload Sepatu

Merchant dapat mengunggah sepatu yang akan dijual melalui aplikasi tukutu.

M-Sepatu

Menu ini digunakan untuk mengedit detail sepatu yang pernah diunggah.

M-Transaksi

Setiap transaksi penjualan akan dapat dilihat pada menu ini. Merchant juga dapat melakukan request pencairan dana jika sepatunya telah terjual.

M-Pesan

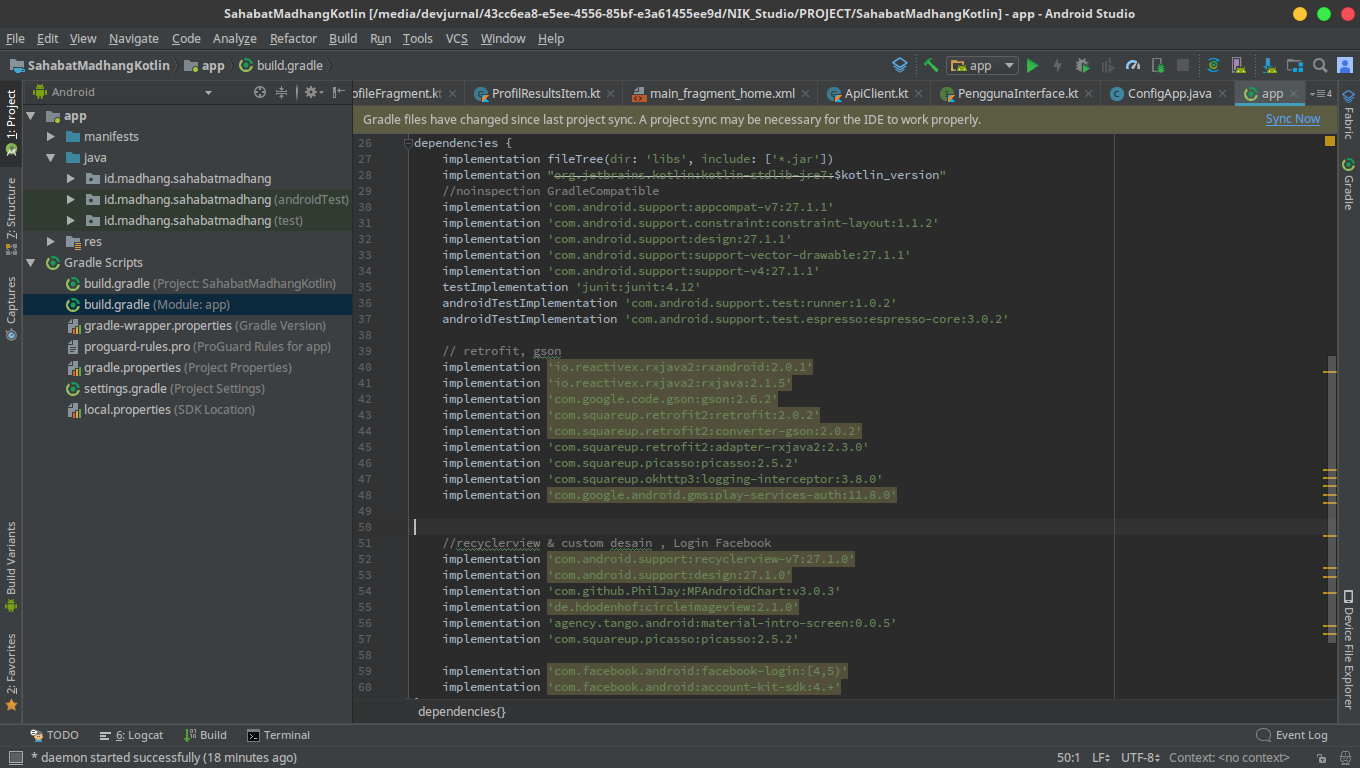
Pesan yang dikirimkan oleh pembeli akan dapat dilihat melalui menu M-Pesan.

* 1. **Penerapan Bahasa Pemrograman Kotlin**

Penerapan bahasa pemrograman Kotlin pada aplikasi Tukutu ini dilakukan pada sebagian *sourcecode* yang ada. Hal ini disebabkan terdapat beberapa library yang belum mendukung bahasa pemrograman Kotlin. Kotlin merupakan pengembangan dari Java, maka dapat dilakukan penggabungan dua jenis bahasa pemrograman antara Kotlin dan Java pada satu proyek aplikasi.

Dalam pembahasan ini , penulis hanya dapat menampilkan potongan *source code* dan struktur file. Pada laporan penelitian ini tidak dapat menampilkan seluruh *source code* dikarenakan aplikasi ini masih memiliki hubungan erat dengan aplikasi Tukutu yang merupakan aplikasi komersil yang bersifat *close source*.

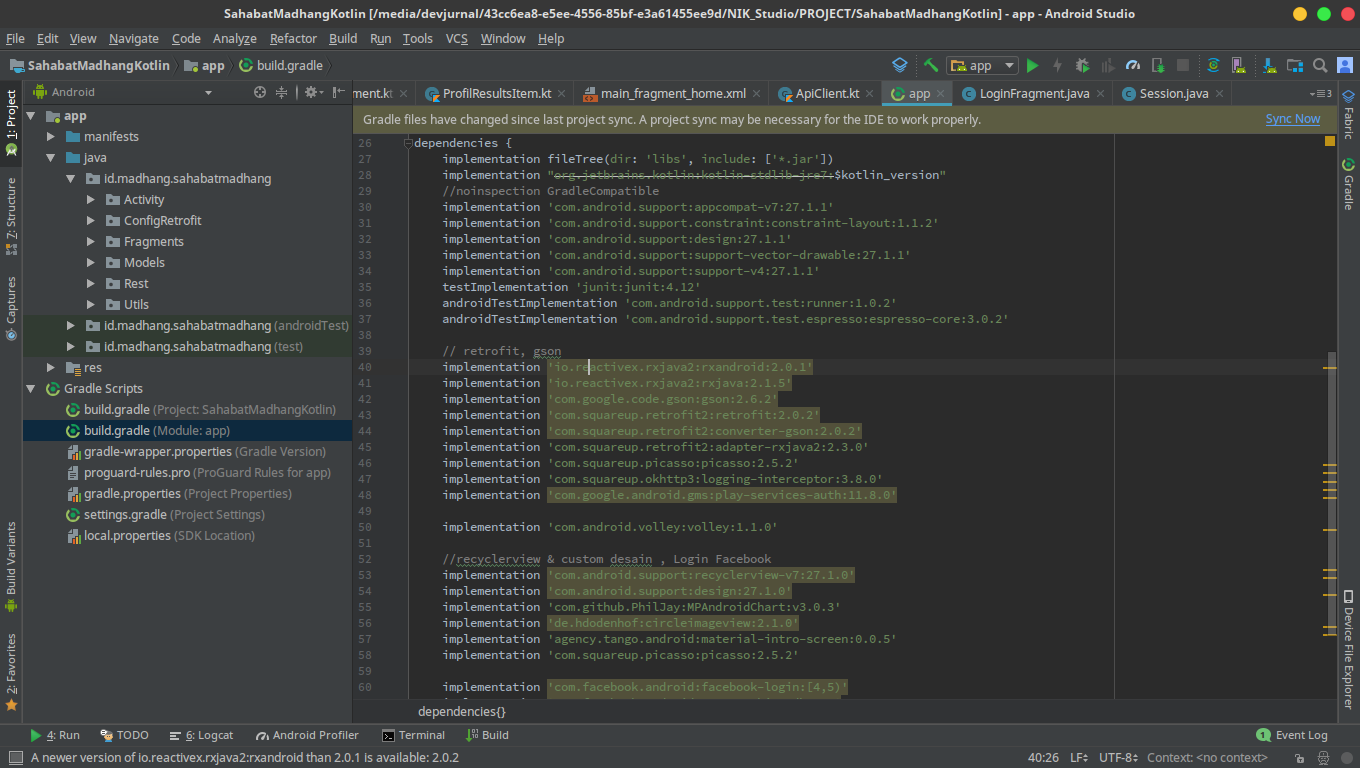
* + 1. **Librari yang digunakan**



Gambar 4. 21. Librari yang pada Aplikasi Tukutu

Pada gambar di atas dapat diketahui beberapa library yang digunakan untuk pembuatan aplikasi Tukutu. Librari yang digunakan pada dasarnya hanya digunakan untuk 3 hal yaitu untuk komunikasi data, untuk kebutuhan desain tampilan antarmuka, dan yang ketiga adalah library untuk keamanan(autentikasi).

* + 1. **Struktur File Aplikasi Tukutu**

****

Gambar 4. 22. Struktur File Aplikasi

Gambar diatas merupakan struktur *package file* *project*. Setiap package memiliki peran yang berbeda, berikut ini peran masing – masing package :

1. *Activity Package*

Package ini memiliki peran untuk mengelola activity yang terdapat pada aplikasi.

1. *ConfigRetrofit Package* dan *Rest Package*

Pada aplikasi ini menggunakan library retrofit , sehingga semua file dasar yang berhubungan dengan konfigurasi retrofit diletakkan pada package ini.

1. *Fragments Package*

Package ini berisi fragment – fragment yang terdapat pada aplikasi.

1. *Models Package*

Models Package digunakan untuk mengelola semua objek model pada aplikasi.

1. *Utils Package*

Utils Package mengelola class tambahan yang digunakan untuk mempermudah manajemen data dan konfigurasi umum.

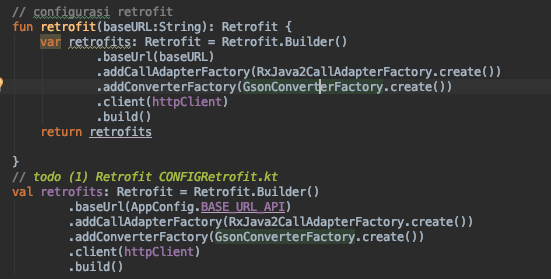
* + 1. **Implementasi Kotlin pada objek Class Model**

****

Gambar 4. 23. Class AuthUser Kotlin

Gambar diatas merupakan class *AuthUser* yang merupakan variable – variable yang akan digunakan untuk menampung data yang diterima pada struktur data JSON. Kotlin memiliki sintaks yang lebih sederhana dibandingkan dengan Java. Berbeda dengan Java yang harus menambahkan method *Set and Get* pada class object, pada Kotlin tidak diperlukan lagi karena method tersebut telah diimplementasikan secara langsung.

* + 1. **Implementasi Retrofit dan RxKotlin/RxJava**

****

Gambar 4. 24. Konfigurasi Retrofit dan RxKotlin

Retrofit merupakan library yang digunakan untuk mengambil data dari api webservis. Gambar diatas merupakan konfigurasi dasar untuk retrofit yang mengandung bagian – bagian :

BaseUrl

BaseUrl berisi url atau alamat yang digunakan untuk mengakses webservis. Baseurl ini dapat berupa alamat IP maupun nama domain yang diikuti dengan lokasi dari webservis.

AdapterRxJava/ RxKotlin

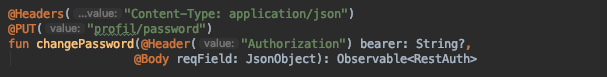
Adapter ini digunakan untuk komunikasi data secara *reactive*. Pada aplikasi ini menggunakan Rx Java/Rx Kotlin. Reactive programming digunakan karena pada aplikasi ini perubahan data terjadi dengan sangat cepat sehingga dibutuhkan metode agar data yang diterima oleh pengguna merupakan data terbaru.

Client Connection

Retrofit merupakan pengembangan dari library OkHttp oleh sebab itu koneksi yang digunakan pada retrofit menggunakan OkHttp.

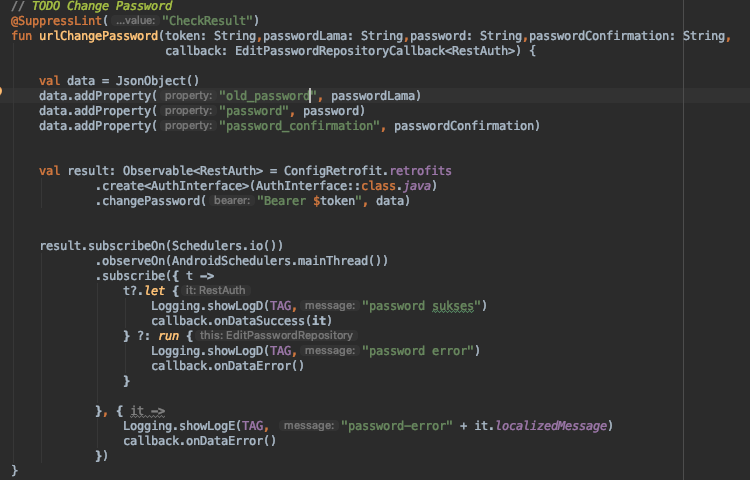
Convert to Gson

Format data yang digunakan pada library retrofit adalah Gson. Data yang dikirimkan oleh webservis berupa data JSON oleh sebab itu dibutuhkan converter untuk dapat mengubah format Json menjadi format Gson.



Gambar 4. 25. Class Interface method changePassword

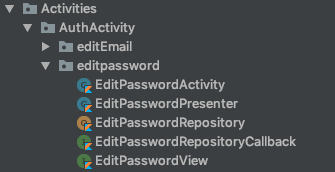
Gambar diatas merupakan potongan source code yang berisikan endpoint dari api webservis yang digunakan pada aplikasi Tukutu. Source code diatas merupakan fungsi changePassword yang memiliki metode PUT restAPI dengan parameter berupa *header* dan *jsonObject*. Observable merupakan bagian dari RxJava/RxKotlin yang memiliki fungsi untuk menampung respon yang diberikan oleh server.

****

Gambar 4. 26. ImplementasiRetrofit pada Object Class EditPasswordRepository

Gambar diatas merupakan *source code method* ***urlChangePassword*** yang terdapat pada *Object Class* ***EditPasswordRepository*.** Metode ini melakukan permintaan kepada webservis menggunakan library retrofit. Jika data berhasil diterima data tersebut akan memberikan callback sesuai data yang diterima. Namun jika data yang diterima kosong , maka akan memberikan callback eror. Callback tersebut nantinya akan diterapkan dibagian class presenter.

* 1. **Penerapan Model View Presenter**
     1. **Struktur File MVP**



Gambar 4. 27. Struktur File MVP

Gambar diatas merupakan struktur MVP (*Model View Presenter*) yang digunakan pada aplikasi Tukutu. Model diatas memiliki struktur atau pola yang terdiri dari *package* yang dipisahkan sesuai dengan tampilannya dimana masing – masing tampilan / package terdiri dari beberapa bagian file yang merupakan struktur arsitektur MVP.

Tiap *file* memiliki peran dan fungsi yang berbeda, berikut merupakan peran dari tiap *file* :

1. Activity

Activity merupakan tampilan yang terhubung dengan layout aplikasi.

1. Presenter

Presenter memiliki peran untuk mengelola *logic* dan alur data dari aplikasi yang di kembangkan. Seluruh komunikasi baik antara *view* dengan *model* (komponen penunjang kelompok data) . Dengan kata lain *presenter*  merupakan pusat seluruh kegiatan logika pemrosesan aplikasi.

1. Repository

Repository memiliki peran untuk mengelola *logic* untuk pengambilan objek data yang berasal dari httprequest maupun pengambilan data dari memori lokal . Data tersebut digunakan oleh presenter untuk diproses lebih lanjut.

1. RepositoryCallback

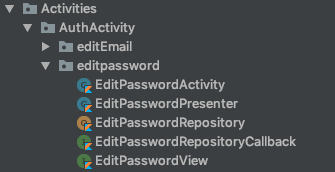
*RepositoryCallback* memiliki peran sebagai *method abstrak* pengaksesan data pada repository yang akan memecah respon data yang diterima menjadi 2 kategori yaitu data eror dan data sukses.

1. View

*View* memiliki peran sebagai *method abstrak* penampil data yang dihasilkan dari pengelolaan di proses file *presenter* yang ada di ke sebuah *activity*, maupun *fragment*.

* + 1. **Penerapan MVP Pada Halaman Edit Password**

Pada halaman edit password, penerapan MVP (*Model View Presenter*) dilakukan hanya pada *activity edit password*, hal tersebut dilakukan karena pada halaman ini layout form langsung tertuju pada file *activity* dan tidak memerlukan fragment.



Gambar 4. 28. MVP pada Halaman Edit Password

MVP pada halaman edit password meliputi beberapa file pendukung diantaranya seperti :

*Editpasswordactivity* merupakan tampilan aktivitas edit password.

*EditPasswordPresenter* berisikan logika – logika yang dibutuhkan saat mengedit password.

*EditPasswordRepository* berisikan httprequest menggunakan retrofit yang berfungsi melakukan request ke server

*EditPasswordRepositoryCallback* berisikan abstrak yang dibutuhkan di Editpassword repository.

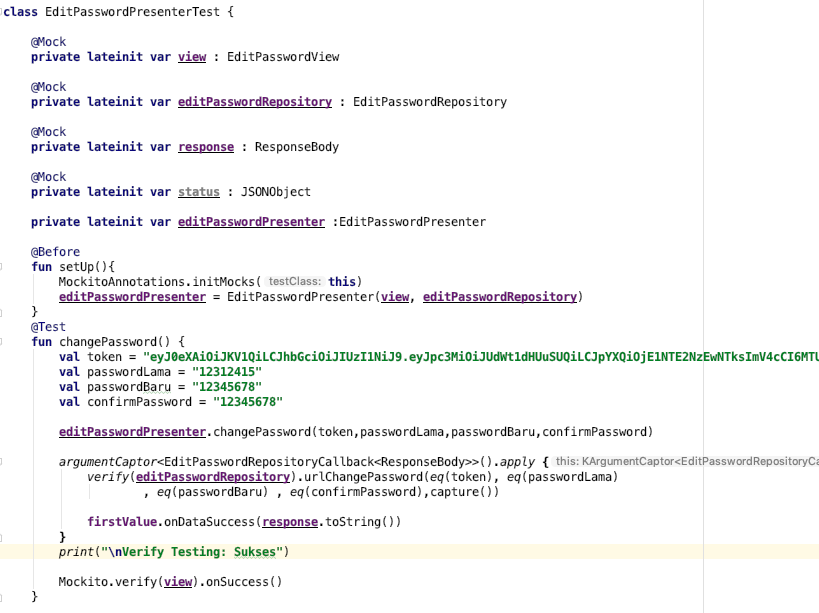
*EditPasswordView* berisikan *abstract of view* dari activity form.

* 1. **Penerapan Test-Driven Development**

Penelitian ini menerapkan proses pengembangan *Test Driven Development* pada metode Agile sehingga dalam setiap pembuatan fitur proses pengujian harus dilakukan secara berulang ulang. Proses pengujian tersebut dilakukan secara terus menerus selama proses pengembangan hingga diperoleh hasil pengujian yang sesuai dengan harapan.

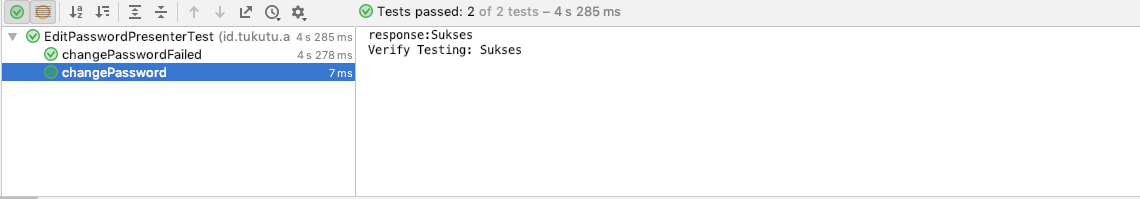
Proses *Test-Driven Development* dimulai sejak tahap analisis programmer mulai merancang skenario pengujian terlebih dahulu sehingga pada tahap *create code* programmer dapat langsung menulis kode yang sesuai dengan skenario pengujian yang telah direncanakan sebelumnya.

Berikut merupakan salah satu contoh *test case* yang dibuat sebelum proses pembuatan atau penulisan *code* aplikasi, *test case* yang dilakukan meliputi pengujian fitur dan *unit test* dengan menggunakan *library* Mockito sebagai *test runner* yang merupakan librari tambahan pada IDE Android Studio.



Gambar 4. 29.Test Case Ganti Password

Dari *test case* tersebut ketika dijalankan akan menghasilkan output sukses sebagai berikut:



Gambar 4. 30 Hasil Test Case Ganti Password

Hari hasil tersebut didapatkan bahwa *source code* yang ditulis dalam *test case* dapat berjalan dengan baik tanpa *error*. Dan runtime yang didapatkan untuk mengeksekusi test case tersebut yaitu sekitar 285ms.



Gambar 4. 31.Test Case Ganti Password Gagal

Dari *test case* tersebut ketika dijalankan akan menghasilkan output sukses sebagai berikut:



Gambar 4. 32 Hasil Test Case Ganti Password Jika Gagal

Hari hasil tersebut didapatkan bahwa *source code* yang ditulis dalam *test case* terjadi kesalahan karena ada kesalahan password. Jika terjadi kesalahan maka akan menampilkan response kesalahan. Dan runtime yang didapatkan untuk mengeksekusi test case tersebut yaitu sekitar 264ms

# BAB V

# KESIMPULAN DAN SARAN

* 1. **Kesimpulan**

Berdasarkan pembahasan pada bab sebelumnya dapat diambil kesimpulan sebagai berikut:

* + - 1. Project Aplikasi Android yang dikembangkan dengan bahasa pemrograman Kotlin dapat digunakan pada perangkat dengan sistem operasi Android Oreo dan juga dapat digunakan dengan lancar pada versi android terdahulu sesuai standar minimal SDK yang ditentukan.
      2. Bahasa pemrograman Kotlin merupakan pengembangan dari Java oleh sebab itu dalam satu file project dapat dikombinasikan bahasa Java dan Kotlin secara langsung.
      3. Penggunaan arsitektur MVP dapat mempermudah pengujian *unit testing* pada aplikasi.
  1. **Saran**

Penulis menyadari bahwa terdapat beberapa kekurangan dalam pembuatan aplikasi Tukutu dengan menggunakan bahasa pemrograman Kotlin, oleh sebab itu penulis memberikan beberapa saran:

Sebelum menggunakan bahasa pemrograman Kotlin, sebaiknya pengembang aplikasi memahami dengan baik bahasa Java.

Tidak semua library yang digunakan pada Java dapat digunakan langsung pada Kotlin, sehingga membutuhkan uji coba dan penyesuaian terlebih dahulu.

Cek versi terbaru pada setiap library yang akan digunakan, karena tidak semua library yang ada pada aplikasi ini dapat digunakan lagi.

Buatlah dokumentasi tentang arsitektur pattern yang digunakan di dalam pembuatan aplikasi, agar dapat dipahami oleh anggota tim yang baru bergabung.

# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | L. Vijayasarathy and C. Butler, "Choice of Software Development Methodologies - Do Project, Team and Organizational Characteristics Matter?," *IEEE Software,* 2015. |
| [2] | Y. X. Wu Bo, "Application of MVP Architect in Reengineering of Legacy Financial System," in *Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on*, Wuhan, 2009. |
| [3] | P. Graham, Paul Graham: The Art of Funding a Startup (A Mixergy Interview), Hyperink and Mixergy, 2011. |
| [4] | E. Ries, The Lean Startup, Amerika Serikat: Crown Publishing Group, 2011. |
| [5] | A. Imaduddin and S. Permana, Menjadi Android Developer Expert, Bandung: PT Presentologics, 2017. |
| [6] | Google, Juli 2018. [Online]. Available: https://developer.android.com/. |
| [7] | K. A. Noerwicaksono, Juli 2018. [Online]. Available: https://medium.com/@kalifardy56/. |
| [8] | A. Imaduddin and S. Permana, Januari 2018. [Online]. Available: https://www.dicoding.com/academies/14. |
| [9] | A. Leiva, Kotlin for Android Developers, Leanpub, 2017. |
| [10] | N. SEPTIAN HUSNI, "Kotlin Day Semarang Part 1," 2017. |
| [11] | R. Maulana, Februari 2016. [Online]. Available: https://teknorial.com. |
| [12] | Square, "Retrofit," Juli 2018. [Online]. Available: http://square.github.io/retrofit/. |
| [13] | The JSON Data Interchange Standard, Juli 2018. [Online]. Available: http://www.json.org/. |
| [14] | F. Al Abid and A. N. M. R. Karim, "Cross-platform development for an online food delivery application," in *2017 International Conference on Computing Networking and Informatics (ICCNI)*, Lagos, 2017. |
| [15] | Sandec, Bermain API dengan Retrofit, Semarang: Sandec, 2017. |
| [16] | A. H. B. A. P. K. Ibrahim Yunus Muhammad Fiqhan, "Pengembangan Aplikasi Pengingat Salat Dengan Konsep," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* vol. 2, no. 10, pp. 3802-3809, 2018. |
| [17] | Google, "Firebase," Juni 2018. [Online]. Available: http://firebase.google.com. |
| [18] | Indrajani, Database Design, Jakarta: Elex Media Komputindo, 2015. |
| [19] | Mulyani, A. Adila and A. Widayanto, "SKPL Sistem Peminjaman Ruang Pemkot Semarang," Semarang, 2017. |
| [20] | R. Bagaskara, "Apa yang dimaksud dengan Diagram aktivitas atau Activity Diagram," 2018. [Online]. Available: dictio.id. |
| [21] | P. Dinustek, "dinustek," PT Dinustek, Desember 2018. [Online]. Available: http://dinustek.com. |
| [22] | K. Rijwan, A. K. Srivastava and D. Pandey, "Agile Approach for Software Testing Process," in *5th International Conference on System Modeling & Advancement in Research Trends*, 2016. |
| [23] | M. J. J. S. Peter Brunn, "e-Marketplaces : Crafting A Winning Strategy," *European Management,* p. 286–298, 2002. |
| [24] | D. W. Artem Syromiatnikov, "A Journey Through the Land of," in *Software Architecture (WICSA), 2014 IEEE/IFIP Conference on*, Sydney, 2014. |