BAB III

PROSES DAN HASIL BELAJAR DI DU/DI

1. Instalasi Android Studio IDE
   1. Penjelasan Pekerjaan



Gambar 3.1. Logo Android Studio

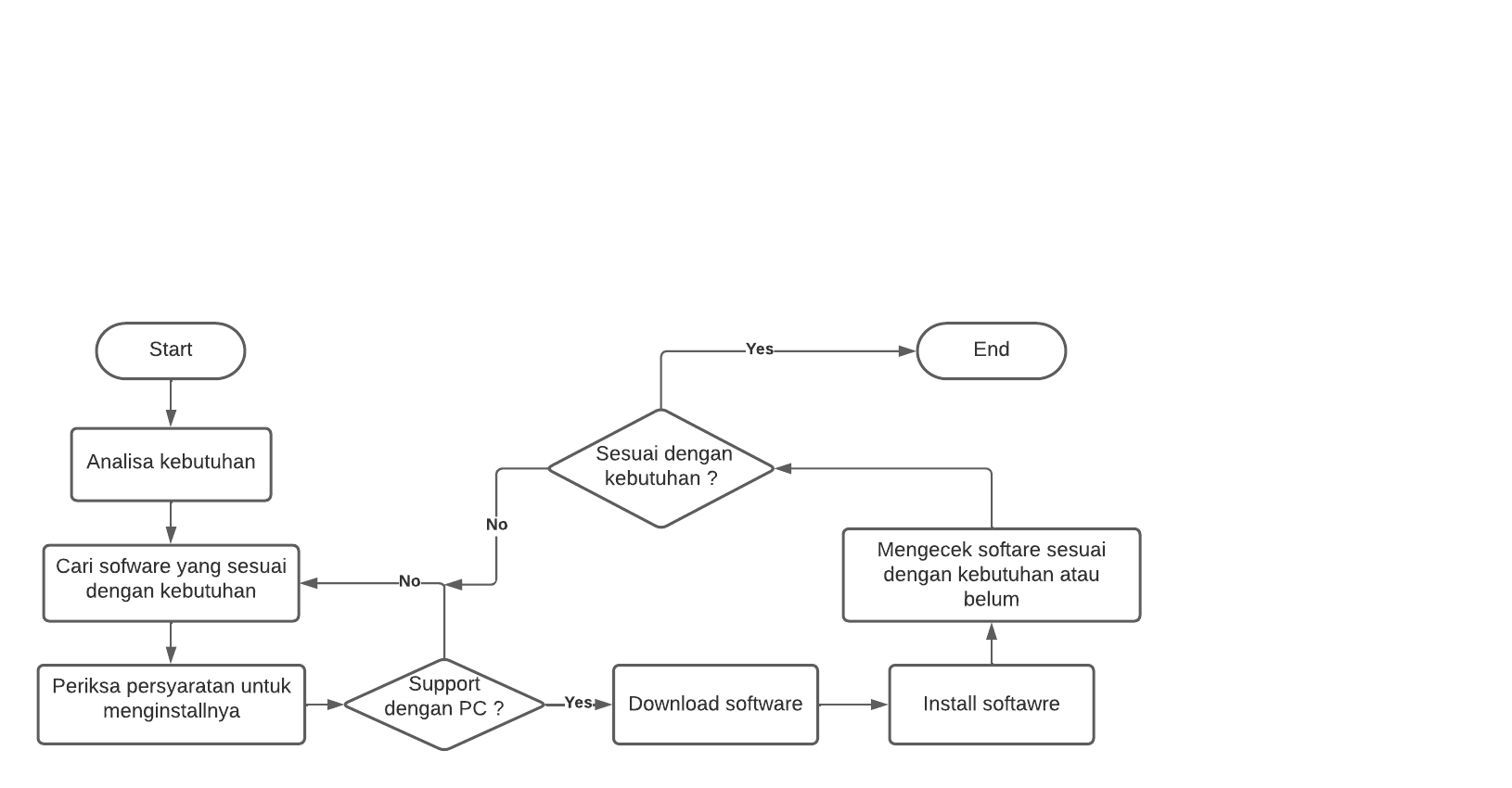
Untuk pengembangan aplikasi *mobile*, *programmer* di CV Karya Hidup Sentosa menggunakan aplikasi Android Studio sebagai IDE (*integrated development environment*) untuk pembuatan dan pengembangan aplikasi berbasis Android. Android Studio merupakan program yang paling populer dipakai untuk pengembangan Aplikasi Android.

Pertama kali Android Studio diumumkan di Google I/O Conference pada tahun 2013 dan dirilis ke publik pada tahun 2014. Sebelum lahirnya Android Studio, aplikasi pada Android dikembangkan dengan Eclipse IDE yaitu IDE Java. Setelah adanya Android Studio yang open source dapat memudahkan bagi *programmer* yang ingin membuat aplikasi dengan Android Studio.

Android Studio menyediakan antarmuka untuk *programmer* dalam membuat aplikasi serta mengelola manajemen *file* aplikasi. Untuk bahasa pemrograman yang bisa digunakan adalah Java dan juga Kotlin. Dalam Android Studio, *programmer* dapat menulis, mengedit, menyimpan dan menguji proyek beserta dan *file* lainnya yang ada dalam proyek itu hanya dengan Android Studio. Android Studio juga memberi akses ke Android *Software Development Kit* (SDK). SDK adalah sebuah ekstensi dari kode Java yang memperbolehkannya untuk berjalan dengan mulus di perangkat Android.

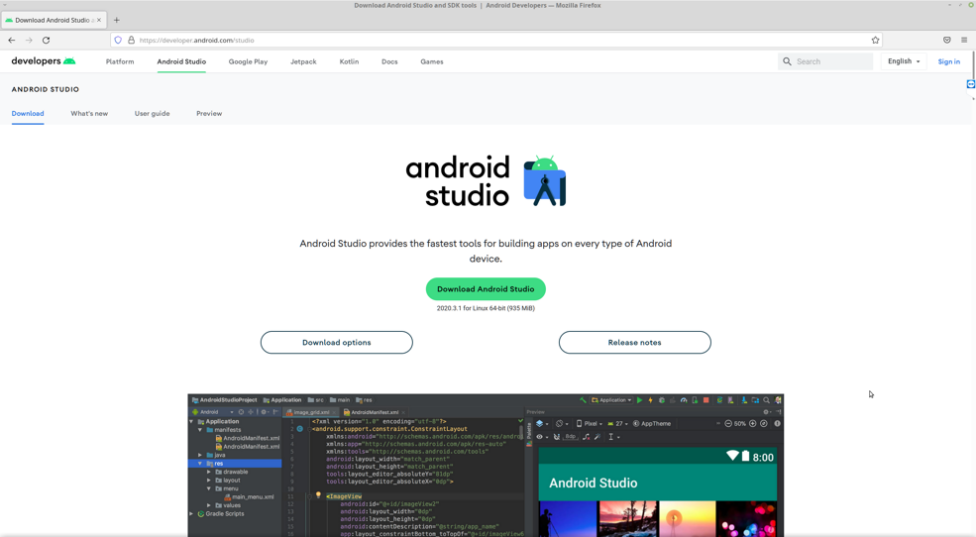
Android SDK sangat diperlukan untuk menjalankan programnya di Android. Maka dari itu dengan menggabungkan keduanya, *programmer* memerlukan Android Studio. Sehingga ketika *programmer* menemukan *bug* pada aplikasi, *programmer* bisa mengetahui *bug* tersebut dengan menggunakan Android Studio untuk memperbaikinya.

* 1. Alat dan Bahan
     1. PC
     2. Web Browser
     3. Koneksi Internet
  2. Keselamatan Kerja
     1. Berdoa sebelum mengerjakan
     2. Memastikan alat dan bahan sesuai dengan yang dibutuhkan
     3. Menggunakan alat sesuai dengan fungsinya
     4. Memperhatikan posisi duduk dan jarak pandang dengan komputer
     5. Berhati hati dengan air dan listrik
     6. Mematikan perangkat jika telah selesai digunakan
     7. Memastikan alat dan bahan berada di posisi yang tepat
  3. Gambar Kerja



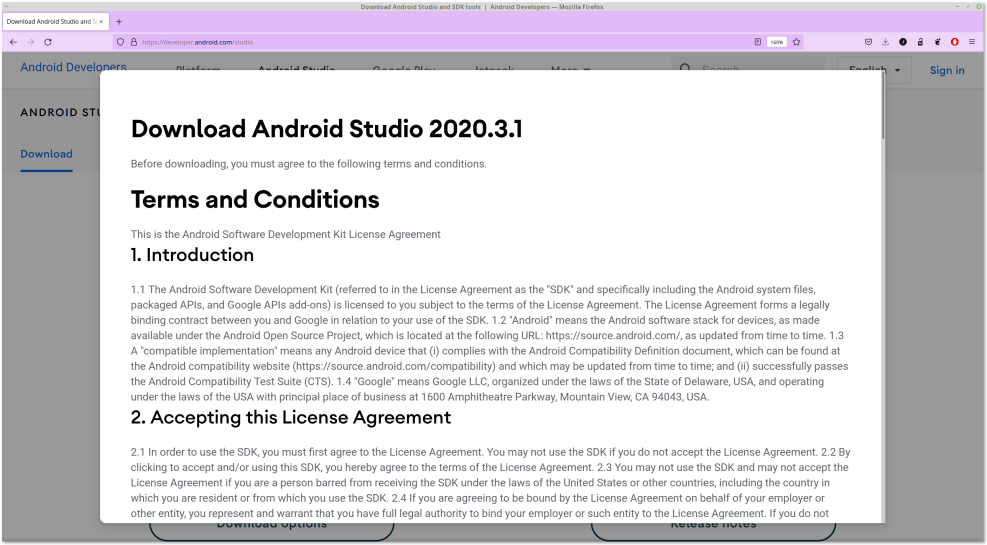
Gambar 3.2. Gambar Kerja Instalasi Android Studio

* 1. Analisa dan Langkah Kerja
     1. Pertama - tama, download file installer Android Studio sesuai *operating system* yang digunakan dari website resminya di <https://developer.android.com/studio>.



Gambar 3.3. Download Android Studio

Setelah website terbuaka, tekan tombol Download Android Studio untuk mulai mendownload, sebelum itu akan muncul dialog *Terms and Conditions*.



Gambar 3.4. *Dialog Terms and Conditions*

*Terms and Conditions (T&C)* adalah perjanjian hukum antara penyedia layanan dan orang yang menggunakan layanan tersebut, orang tersebut harus menyetujui nya untuk menggunakan layanan yang ditawarkan.

Sehingga sebelum kita menggunakan Android Studio ini, kita harus membaca dan memahami T&C ini, serta mematuhi syarat dan ketentuan nya. Setelah itu, dibagian bawah ada checkbox “*I have read and agree with the above terms and conditions*” kemudian tekan Download Android Studio untuk mulai mendownload installernya.

* + 1. File Installer Android Studio berbeda beda tiap Operating System. Untuk Windows, file installernya berektensi .exe dan untuk memulai proses instalasinya tinggal double click installernya dan ikuti petunjuk instalasinya. Untuk Mac OS, file installernya akan bereksistensi .dmg dan untuk cara instalasinya adalah double click file dmg nya dan drag Android Studionya ke folder Aplication.

Sedangkan untuk linux, file nya akan bereksistensi .tar.gz.



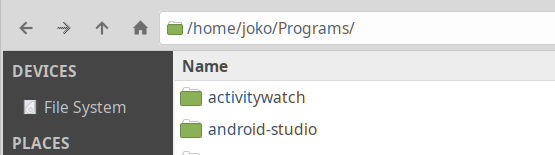
Gambar 3.5. File installer Android Studio untuk Linux

* + 1. Sebelum menjalankan Android Studio, jika komputer yang digunakan berbasis 64-bit, maka diharuskan untuk menginstall beberapa library 32-bit dengan menjalankan perintah berikut di terminal.



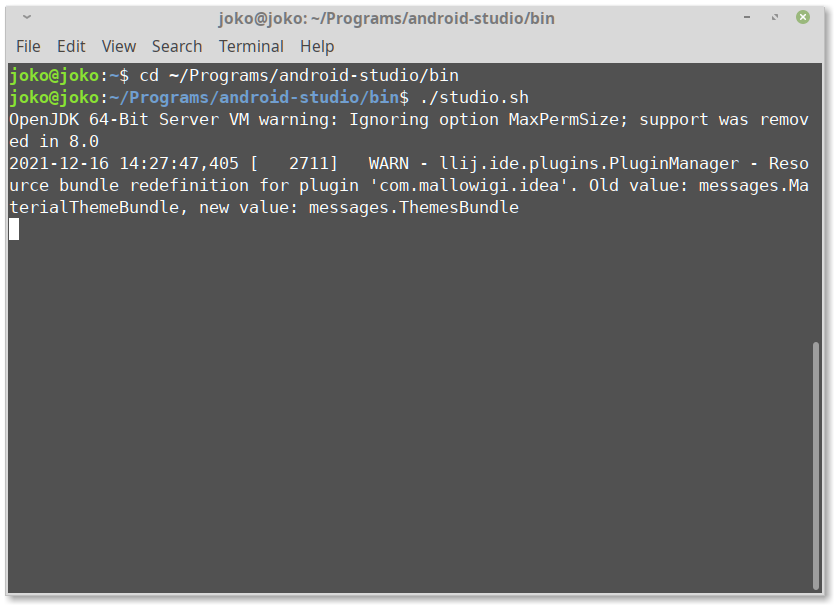
Gambar 3.6. Menginstall Library Tambahan

* + 1. Karena di PC penulis menggunakan Linux Mint, maka file installer nya berbentuk tar.gz. Setelah file installer nya selesai terdownload, unpack file .tar.gz ke lokasi yang sesuai seperti /usr/local/ untuk spesifik user atau /opt/ untuk semua user. Tapi disini penulis akan menaruhnya di /home/{username}/Programs/.



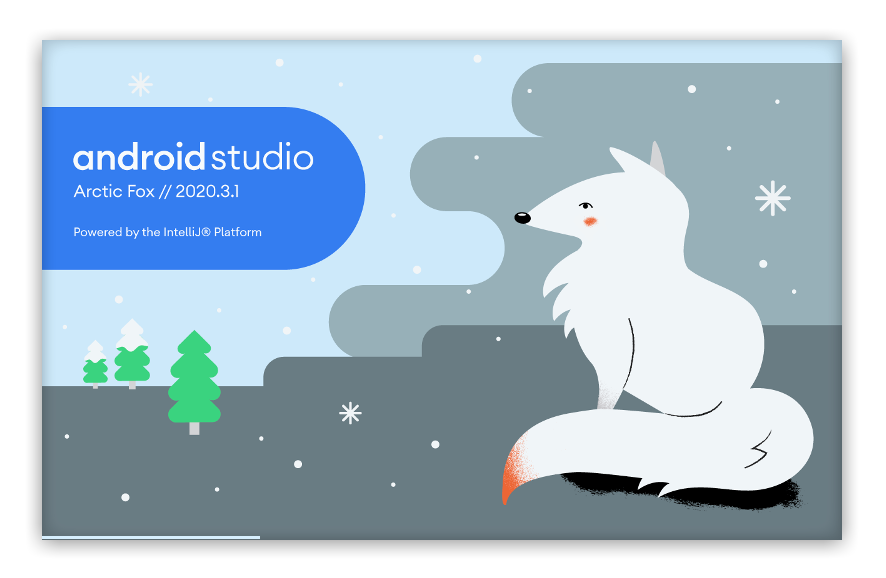
Gambar 3.7. Lokasi Android Studio

* + 1. Kemudian untuk menjalankan Android Studio, buka terminal dan pindah ke directory tempat instalasi nya dan jalankan file studio.sh yang berada di folder bin.



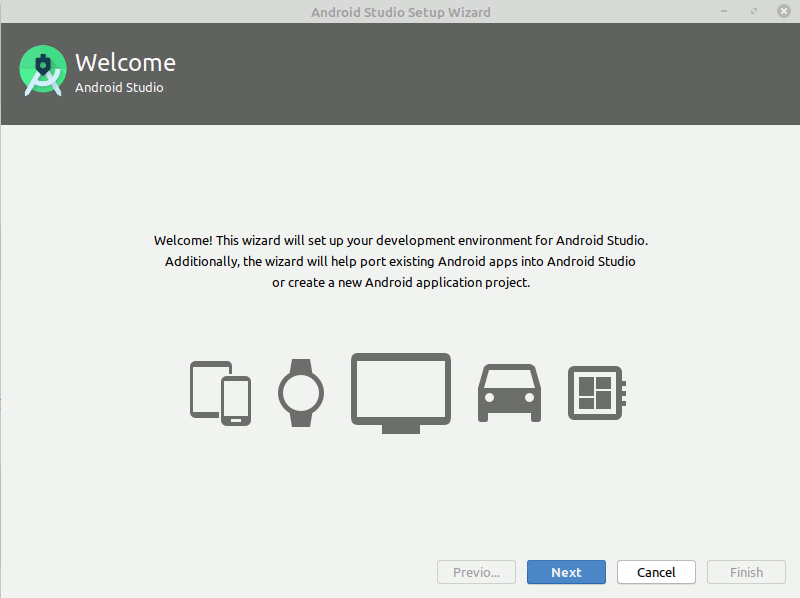
Gambar 3.8. Menjalankan Android Studio Melalui Terminal

Setelah itu Android studio akan mulai berjalan, tunggu hingga loadingnya selesai.



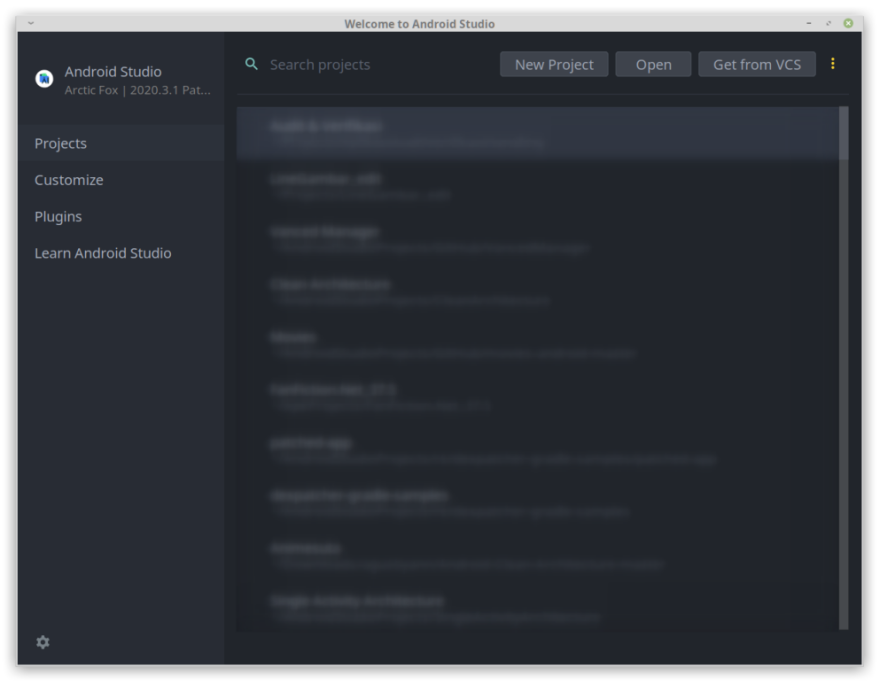
Gambar 3.9. Loading Android Studio

* + 1. Kemudian pilih apakah ingin import setting Android Studio yang sudah ada atau tidak, kemudian tekan OK.
    2. Android Studio akan menampilkan Setup Wizard yang akan memandu melalui proses instalasi yang tersisa, seperti memilih tema, download Android SDK components yang dibutuhkan untuk proses development.



Gambar 3.10. Android Studio Setup Wizard

* + 1. Setelah seluruh proses selesai, Android Studio sudah siap digunakan untuk mengembangkan aplikasi android.



Gambar 3.11. Jendela Utama Android Studio

* 1. Quality Control

Setelah instalasi Android Studio selesai, perlu dilakukan beberapa setup sebelum Android Studio benar benar dapat digunakan dengan maksimal. Diantaranya adalah pengecekan fitur yang akan digunakan. Setting Android Studio sesuai preferensi seperti ukuran font editor dan aplikasi, penggunaan memory, pengaturan tema dan lain-lain. Atau bahkan install beberapa plugin yang dibutuhkan untuk mempermudah pekerjaan di Android Studio.

* 1. Kesimpulan

Android Studio adalah Lingkungan Pengembangan Terpadu - *Integrated Development Environment* (IDE) untuk pengembangan aplikasi Android, berdasarkan IntelliJ IDEA . Selain merupakan editor code IntelliJ dan alat pengembang yang berdaya guna, Android Studio menawarkan lebih banyak fitur. Gunanya untuk meningkatkan produktivitas saat membuat aplikasi Android, dan mendapat dukungan langsung dari pihak Google.

Dalam Android Studio dapat menggunakan Java dan juga Kotlin. Dalam Android Studio, *programmer* dapat menulis, mengedit, menyimpan dan menguji proyek beserta dan *file* lainnya yang ada dalam proyek itu hanya dengan Android Studio. Android Studio juga memberi akses ke Android *Software Development Kit* (SDK). SDK adalah sebuah ekstensi dari kode Java yang memperbolehkannya untuk berjalan dengan mulus di perangkat Android.

1. Instalasi Postman
   1. Penjelasan Pekerjaan



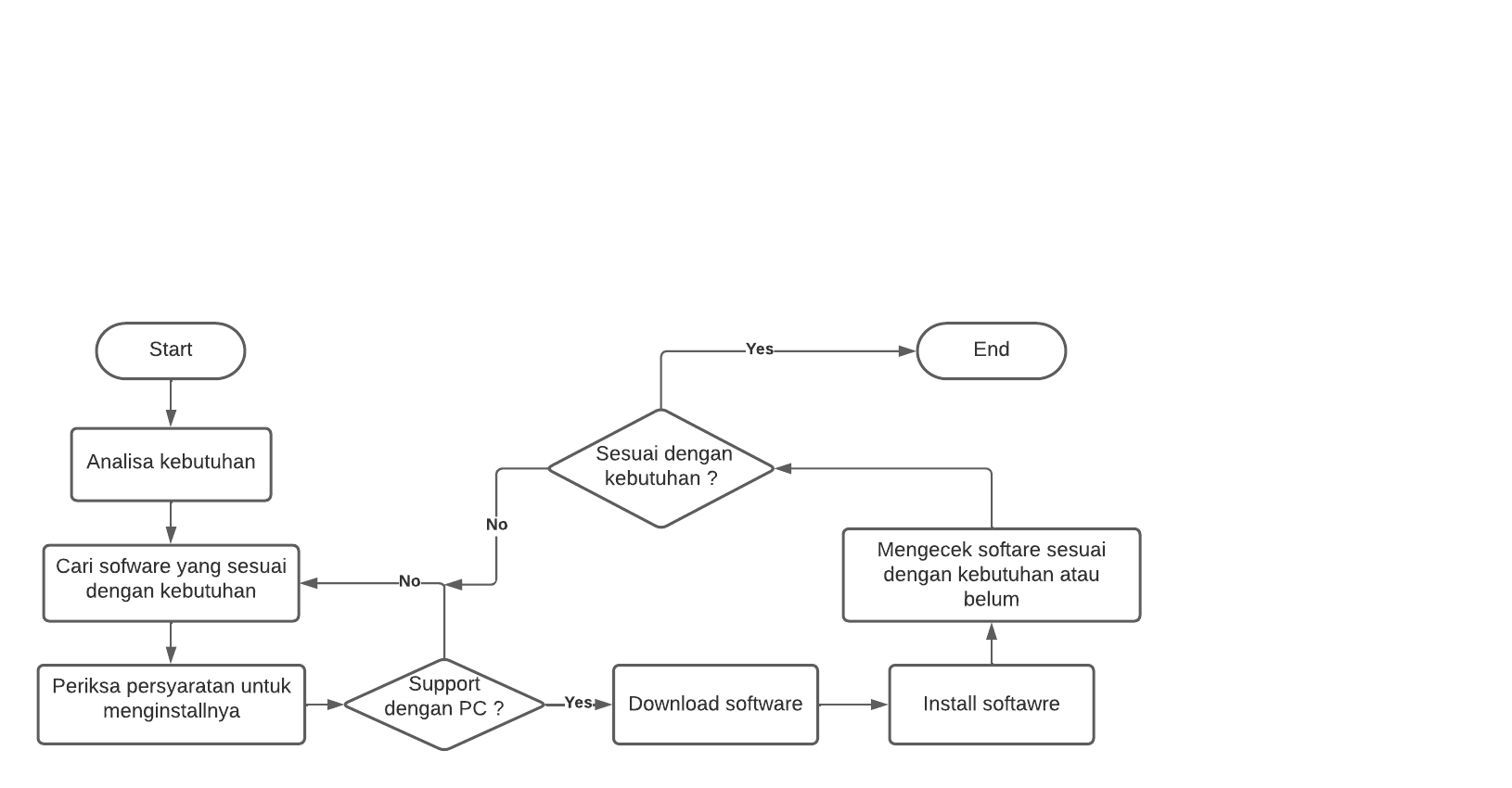
Gambar 3.12. Logo Postman

Postman adalah platform kolaborasi untuk pengembangan API (*Application Programming Interface*). Fitur – fitur postman dapat menyederhanakan pembangunan API dan mempersingkat kolaborasi sehingga anda dapat membuat API yang lebih baik dan cepat. Selain pengembangan, Postman juga sering digunakan sebagai API Testing, QA, dan Manajemen produk perangkat lunak.

API adalah sebuah *software* yang dapat menghubungkan dua atau lebih aplikasi dalam platform apapun. Cara kerja API adalah dengan menerima request dari sebuah aplikasi kemudian memberikan respon sesuai dari request tersebut. Dengan kata lain, API bisa disebut sebagai server, namun server ini tidak hanya memberikan pelayanan kepada satu platform, melainkan banyak platform.

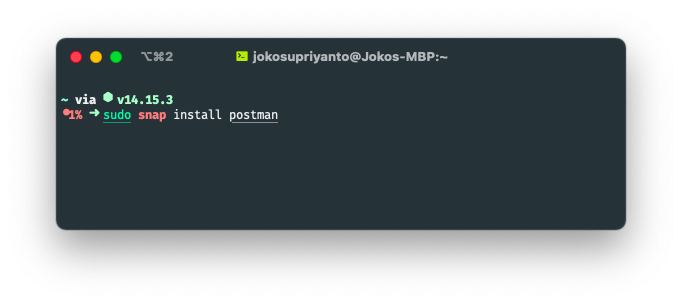
Postman dapat digunakan secara gratis dan instalasinya sangat mudah, cepat dan dapat berjalan pada sistem operasi Windows, Linux, ataupun MacOS. Di CV. Karya Hidup Sentosa aplikasi ini biasa digunakan oleh *programmer* untuk melakukan trial kepada API. Postman sendiri juga digunakan untuk melihat hasil atau result dari API ketika dijalankan baik dengan metode GET ,POST , PUT, DELETE, dan lainnya dan menampilkannya dalam bentuk JSON, XML, maupun HTML.

* 1. Alat dan Bahan
     1. PC
     2. Terminal
     3. Koneksi Internet (rekomendasi >1MB/s)
  2. Keselamatan Kerja
     1. Berdoa sebelum mengerjakan
     2. Memastikan alat dan bahan sesuai dengan yang dibutuhkan
     3. Memastikan alat dan bahan berada di posisi yang tepat
     4. Menggunakan alat sesuai dengan fungsinya
     5. Memperhatikan posisi duduk dan jarak pandang dengan komputer
     6. Berhati hati dengan air dan listrik
     7. Mematikan perangkat jika telah selesai digunakan
  3. Gambar Kerja



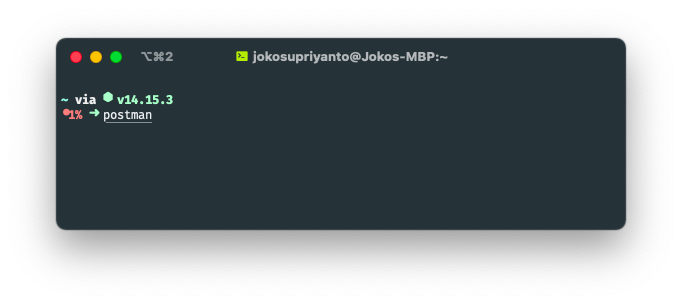
Gambar 3.13. Gambar Kerja Instalasi Postman

* 1. Analisa dan Langkah Kerja
     1. Untuk menginstall postman di Linux caranya cukup mudah, pertama buka terminal kemudian jalankan perintah berikut untuk menginstall postman menggunakan snap package manager.



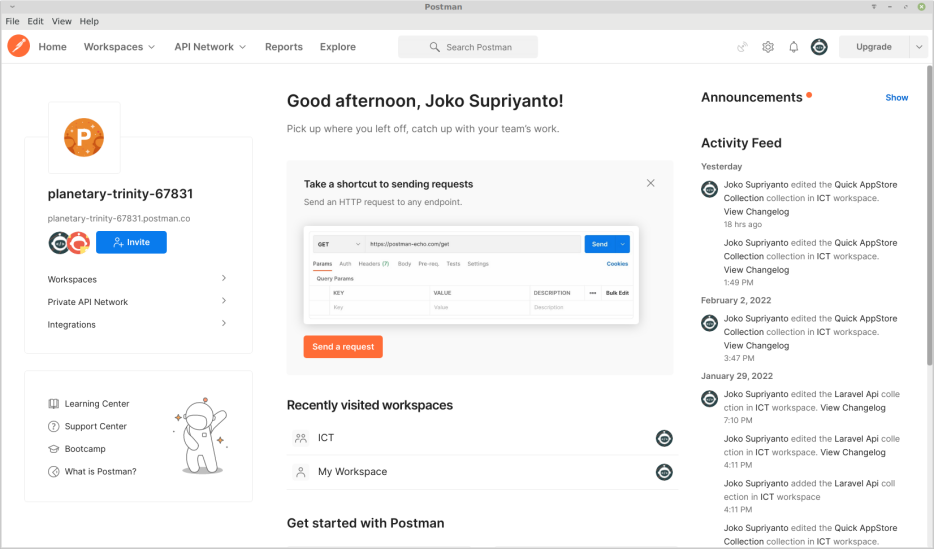
Gambar 3.14. Install Postman Dengan Terminal

* + 1. Setelah proses instalasi selesai, buka application menu dan cek apakah postman sudah berhasil terinstall atau belum, jika ada maka tinggal membukanya dengan mengeklik icon tersebut. Atau bisa juga dengan menjalankan perintah postman di terminal.



Gambar 3.15. Menjalankan Postman Melalui Terminal

* + 1. Setelah menjalankan perintah untuk menjalankan Postman di terminal, akan muncul loading dialog dari Postman. Tunggu hingga selesai dan jika muncul seperti berikut maka Postman sudah berhasil di install dan siap digunakan.

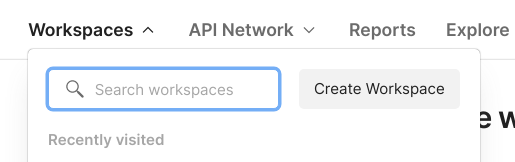


Gambar 3.16. Dashboard Postman

* 1. Quality Control

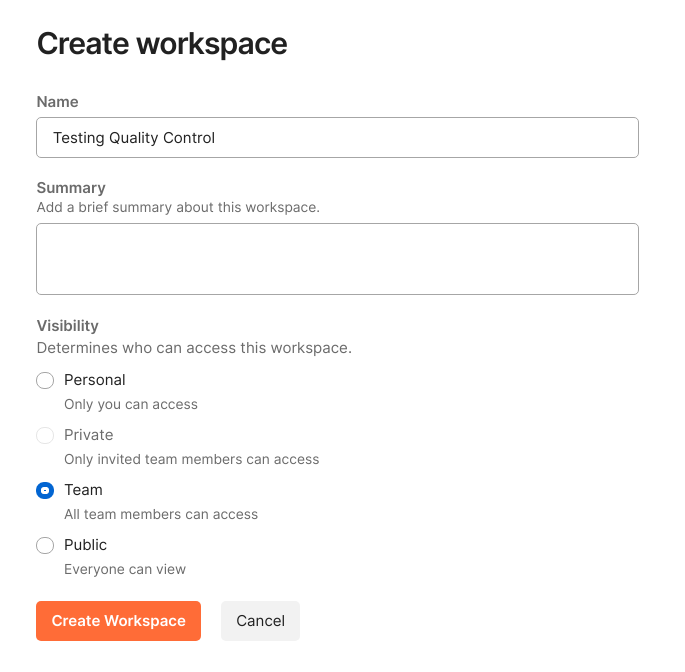
Setelah selesai menginstall Postman, perlu dilakukan pengecekan apakah fitur-fitur yang dibutuhkan pada aplikasi tersebut dapat berjalan dengan baik. Apabila ada yang belum lengkap atau corrupt dapat dilakukan instalasi ulang.

* + 1. Klik menu Workspaces di bagian atas postman, kemudian tekan tombol Create Workspace untuk membuat workspace baru.



Gambar 3.17. Proses Pembuatan Workspace Baru

* + 1. Setelah itu akan muncul tampilan untuk membuat workspace baru. Isi nama workspace dan summary sesuai keinginan, dan juga pilih visibility nya. Setelah semua terisi tekan tombol Create Workspace untuk membuat workspace.



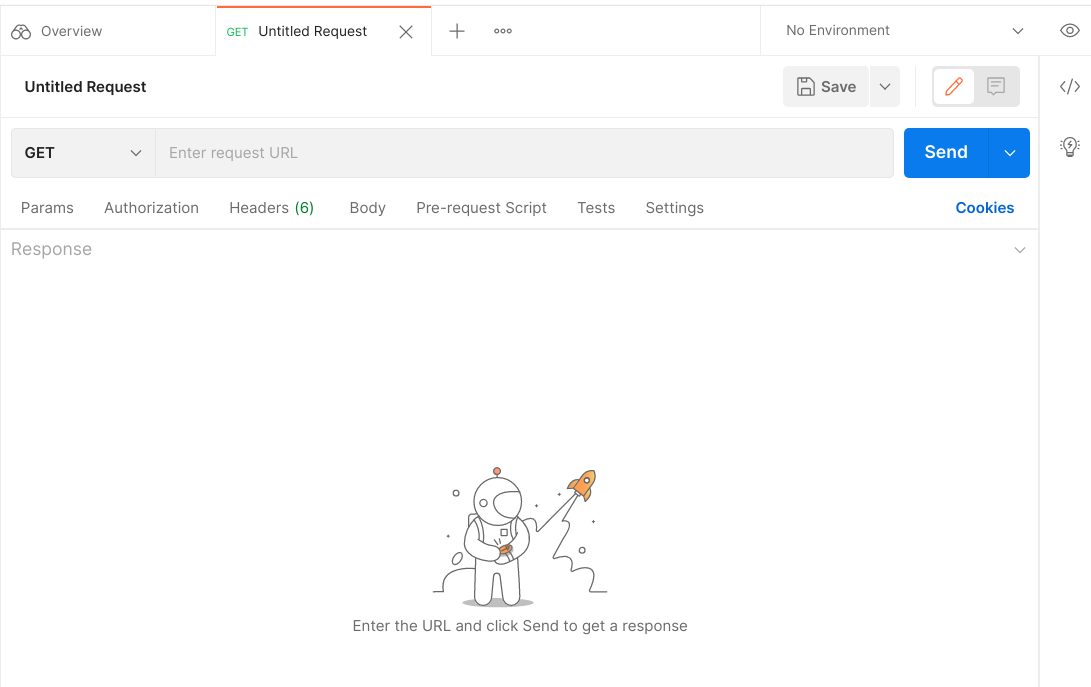
Gambar 3.18. Membuat Workspace Baru

* + 1. Kemudian akan muncul tampilan berikut ini yang tandanya workspace sudah berhasil dibuat. Kemudian tekan tombol + di bagian atas untuk membuat request baru.



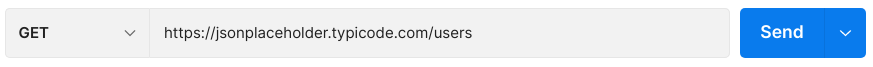
Gambar 3.19. Workspace Baru Yang Telah Dibuat

* + 1. Selanjutnya akan muncul tampilan seperti berikut.



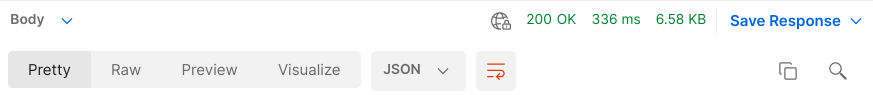
Gambar 3.20. Tampilan Request Baru

* + 1. Isi request URL dengan url api yang bisa kita gunakan untuk testing. Semisal menggunakan json placeholder dari url <https://jsonplaceholder.typicode.com/users.> Setelah URL nya terisi, klik tombol Send untuk mulai mengirim request.



Gambar 3.21. Pengisian URL Request

* + 1. Jika mendapat request 200 OK berarti postman dapat berjalan dengan baik untuk melakukan network requesting



Gambar 3.22. Hasil Request Yang Berhasil

* 1. Kesimpulan

Postman merupakan sebuah sebuah software otomatisasi untuk API testing yang banyak digunakan oleh developer untuk menguji apakah API bekerja dengan baik pada *backend* dan *frontend* pada sebuah sistem. Postman dapat digunakan secara gratis dan instalasinya sangat mudah, cepat dan dapat berjalan pada sistem operasi Windows, Linux, ataupun MacOS

1. Instalasi dan Konfigurasi Aplikasi DBeaver
   1. Penjelasan Pekerjaan



Gambar 3.23. Logo DBeaver

DBeaver adalah sebuah SQL *client software* dan *database administration tool open source* dan multi platform yang dibuat dengan Java dan berdasarkan platform Eclipse. DBeaver menggunakan JDBC *application programming interface* (API) untuk berinteraksi dengan *relational database* melalui JDBC Driver.

DBeaver menyediakan editor yang mensupport *code completion* dan *syntax highlighting*. Aplikasi ini menyediakan sebuah arsitektur plug-in yang memungkinkan user untuk memodifikasi sebagian besar dari perilaku aplikasi untuk setiap spesifik *database fungsionalitas* atau fitur yang dibutuhkan *database*.

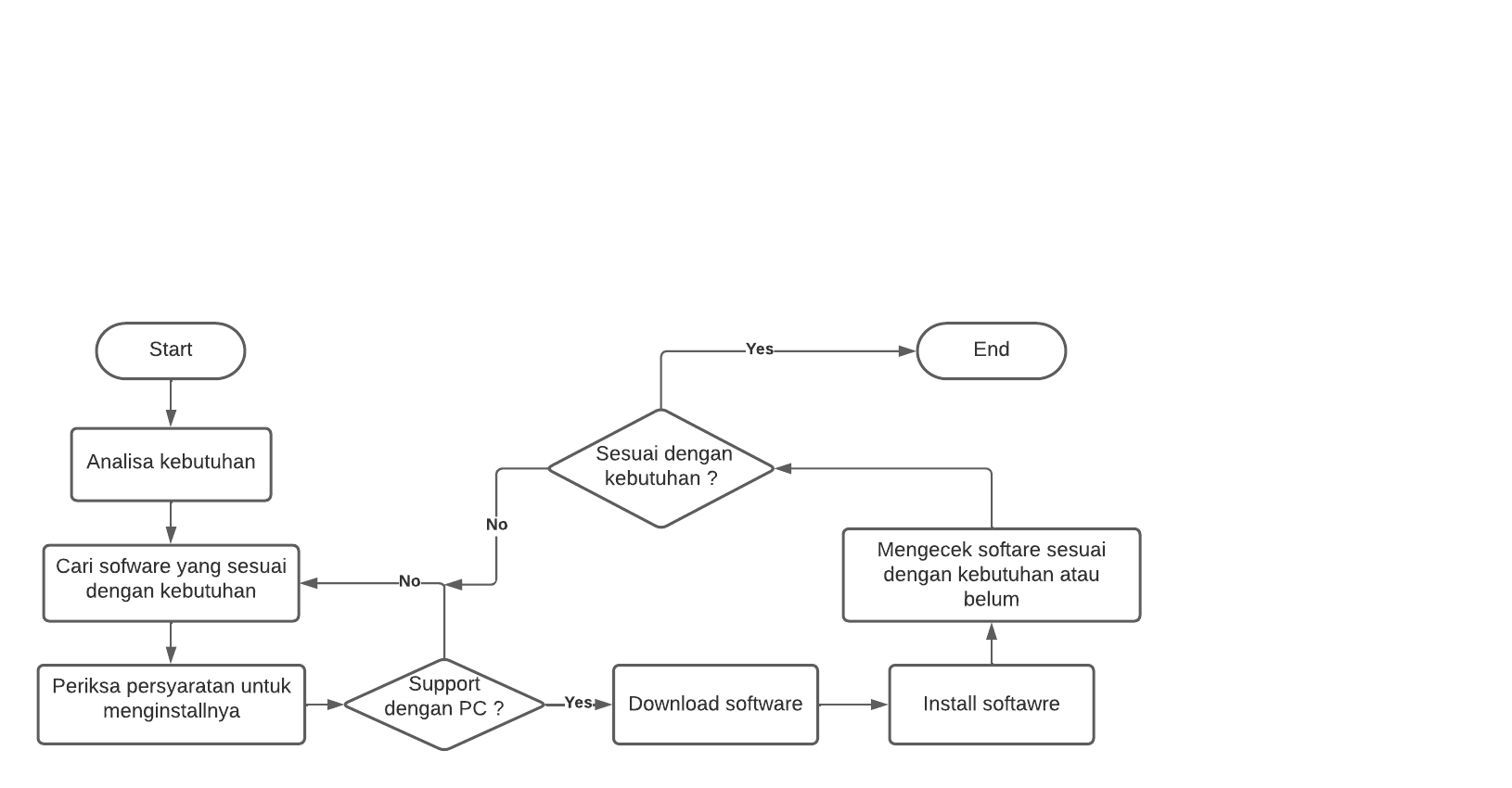
DBeaver Community Edition adalah versi pertama dari DBeaver yang di rilis di tahun 2010 dan menjadi *open source* di 2011. Versi Community Edition *includes extended support* beberapa *database* seperti MySQL and MariaDB, PostgreSQL, Greenplum, Oracle, DB2 (LUW), EXASOL, SQL Server, Sybase, Firebird, Teradata, Vertica, SAP HANA, Apache Phoenix, Netezza, Informix, Apache Derby, H2, SQLite, SnappyData, dan *database* lain yang mempunyai driver JDBC atau ODBC.

Aplikasi ini mempunyai beberapa fitur yang diantaranya :

1. [*SQL*](https://en.wikipedia.org/wiki/SQL)*queries execution*
2. *Data browser/editor with a huge number of features*
3. *Syntax highlighting and SQL auto-completion*
4. *Database structure ([metadata](https://en.wikipedia.org/wiki/Metadata" \o "Metadata)) browse and edit*
5. *SQL scripts management*
6. [*DDL*](https://en.wikipedia.org/wiki/Data_definition_language)*generation*
7. [*ERD*](https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model)*(Entity Relationship Diagrams) rendering*
8. [*SSH tunnelling*](https://en.wikipedia.org/wiki/Tunneling_protocol)
9. *SSL support (MySQL and PostgreSQL)*
10. *Data export/migration*
11. *Import, export and backup of data (MySQL and PostgreSQL)*
12. *Mock data generation for [database testing](https://en.wikipedia.org/wiki/Database_testing" \o "Database testing)*

Kami biasanya menggunakan aplikasi ini untuk melakukan operasi ke *database* sehari hari. Karena operasi *database* adalah pekerjaan yang sering kami lakukan, maka adanya aplikasi ini sangat dibutuhkan, sehingga kami harus menginstall dan mengkonfigurasinya dengan benar.

* 1. Alat dan Bahan
     1. Linux PC
     2. Koneksi Internet
     3. Firefox Web Browser
  2. Keselamatan Kerja
     1. Berdoa sebelum mengerjakan
     2. Memastikan alat dan bahan sesuai dengan yang dibutuhkan
     3. Memastikan alat dan bahan berada di posisi yang tepat
     4. Menggunakan alat sesuai dengan fungsinya
     5. Memperhatikan posisi duduk dan jarak pandang dengan komputer
     6. Berhati hati dengan air dan listrik
     7. Mematikan perangkat jika telah selesai digunakan
  3. Gambar Kerja



Gambar 3.24. Gambar Langkah Kerja Instalasi DBeaver

* 1. Analisa dan Langkah Kerja
     1. Untuk menginstall DBeaver di Linux, kita bisa menggunakan beberapa cara yaitu dengan file installer, dengan archive aplikasinya langsung, atau dengan melalui *command line*. Disini kita akan menggunakan file installernya.
     2. Untuk mendownload file installernya, buka website resmi DBeaver di <https://dbeaver.io/download/> kemudian scroll sedikit ke bawah untuk melihat bagian untuk sistem operasi Linux. Lalu download yang Linux Debian package 64 bit (installer).



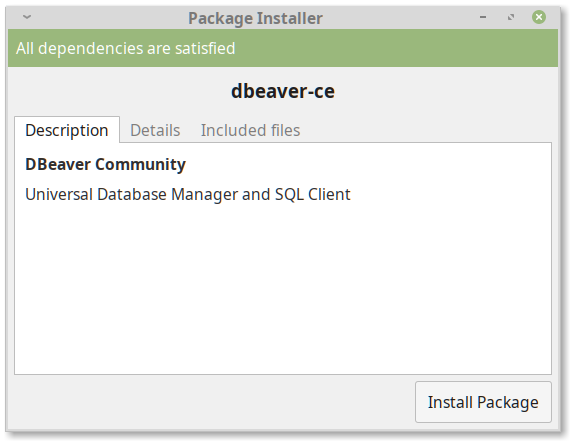
Gambar 3.25. Link Download DBeaver Untuk Linux

* + 1. Setelah installernya terdownload, double click file nya untuk memulai proses instalasinya.



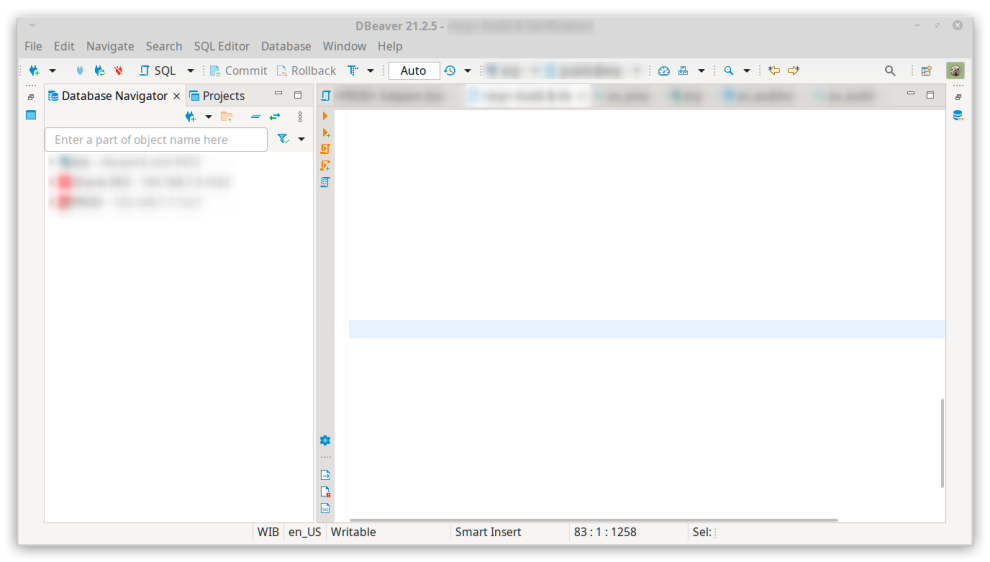
Gambar 3.26. File Installer DBeaver

* + 1. Kemudian akan muncul jendela instalasi seperti berikut, untuk memulai proses instalasi klik tombol Install Package dan juga masukan user password jika diminta.



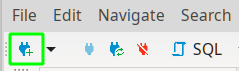
Gambar 3.27. Jendela Instalasi DBeaver

* + 1. Setelah selesai, jalankan aplikasi DBeaver dari launcher. Jika sudah muncul tampilan seperti berikut ini maka DBeaver sudah berhasil terinstall.



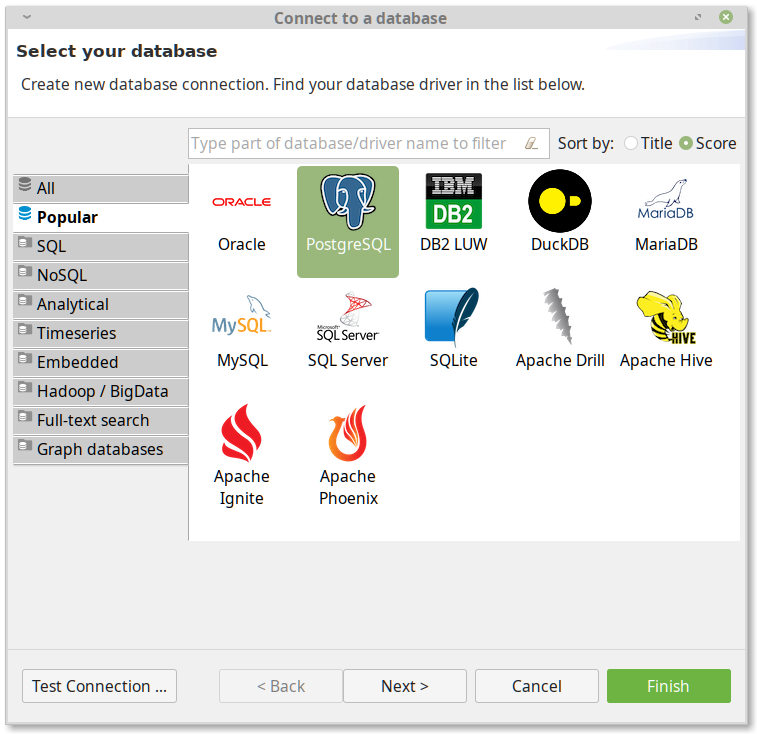
Gambar 3.28. Jendela Utama DBeaver

* + 1. Kemudian sebelum bisa digunakan, DBeaver harus disetting dulu untuk koneksi *database* servernya. Untuk menambah koneksi *database* server baru, tekan icon *Create new connection* di bagian kiri atas DBeaver.



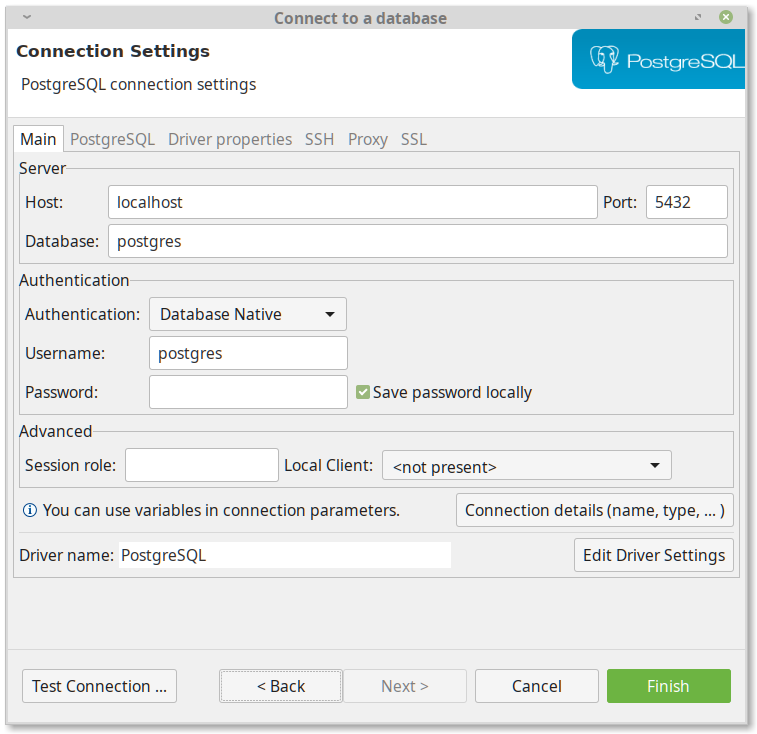
Gambar 3.29. Membuat Koneksi Baru

* + 1. Kemudian akan muncul jendela baru seperti berikut ini, di sini pilih jenis *database* yang ingin ditambahkan, sebagai contoh kita akan memilih PostgreSQL. Setelah itu tekan tombol Next.



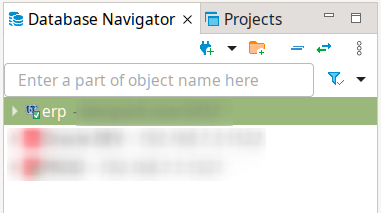
Gambar 3.30. Jendela Pemilihan Jenis Koneksi Baru

* + 1. Kemudian akan muncul jendela konfigurasi server *database*nya seperti berikut ini. Di sini isi semua pengaturan *database*nya sesuai dengan benar. Setelah selesai tekan tombol *Finish*.



Gambar 3.31. Jendela Pembuatan Koneksi Baru

* + 1. Setelah selesai, koneksi yang baru dibuat akan muncul di *Database Navigator DBeaver.*



Gambar 3.32. Koneksi Baru Yang Berhasil Dibuat

* 1. Quality Control

Setelah DBeaver terinstall, Langkah selanjutnya adalah menguji fitur – fitur utama pada aplikasi ini. Jika terdapat masalah pada aplikasi ini, dapat dilakukan instalasi ulang.

* 1. Kesimpulan

DBeaver Community Edition adalah sebuah SQL *client software* dan *database administration tool* *open source* dan multi platform yang dibuat dengan Java dan berdasarkan platform Eclipse.

1. Pembuatan Database Aplikasi Audit & Verification
   1. Penjelasan Pekerjaan



Gambar 3.33. Ilustrasi Database

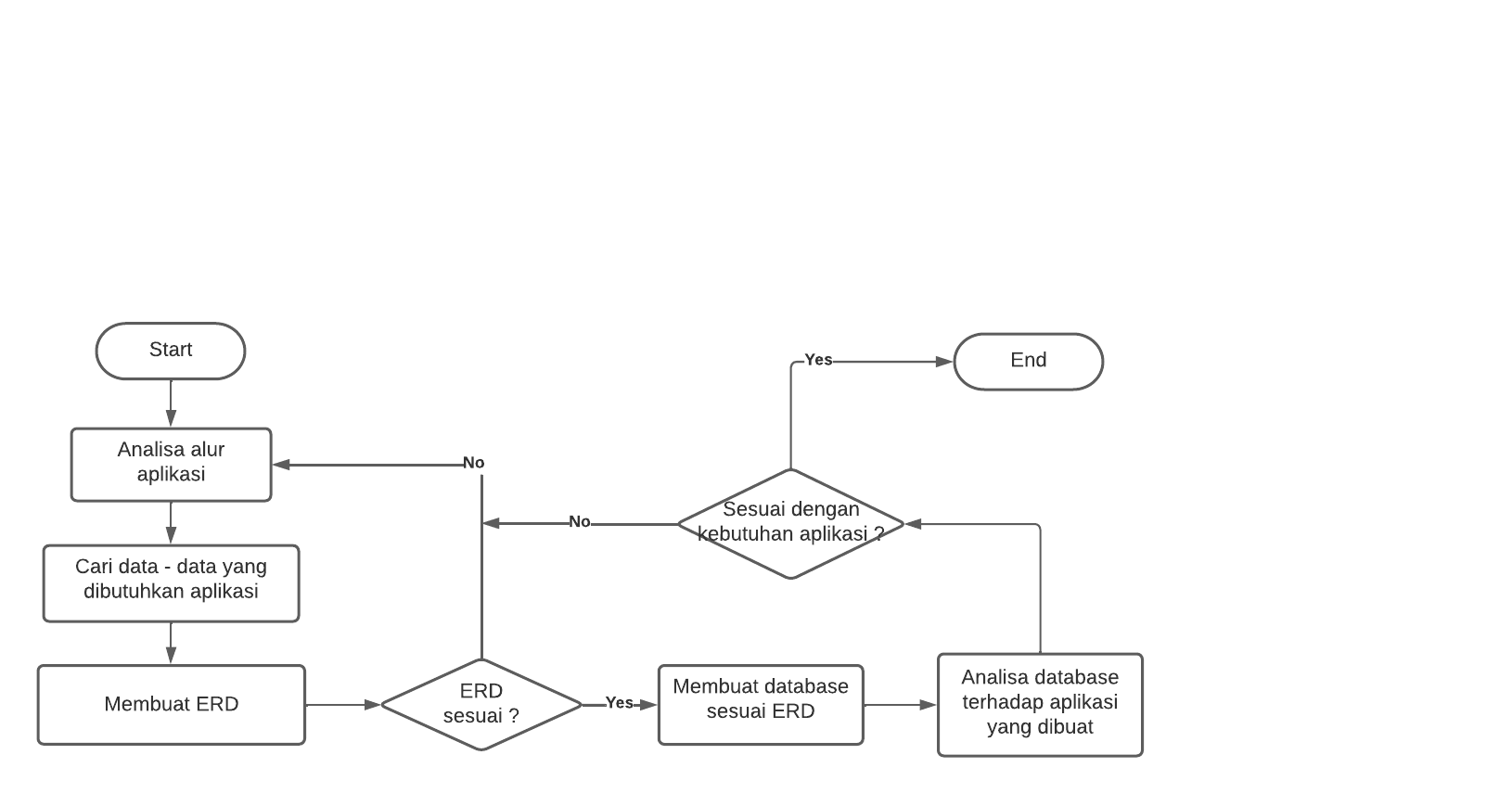
*Database* (Pangkalan Data) adalah kumpulan informasi yang terorganisir, atau data yang biasanya disimpan secara elektronik di dalam sistem komputer. Sebuah *database* biasanya dikontrol dengan sebuah *Database Management System* (DBMS).

Data dalam tipe *database* yang paling umum saat ini biasanya dimodelkan dalam bentuk baris dan kolom dalam serangkaian tabel untuk membuat pemrosesan dan query data menjadi efisien. Data kemudian dapat dengan mudah diakses, dikelola, dimodifikasi, diperbaharui, dikendalikan dan diatur. Sebagian besar besar *database* menggunakan bahasa *Structured Query Language* (SQL) untuk menulis dan mendapatkan data.

SQL adalah bahasa pemrograman yang digunakan hampir semua *relaional database* untuk *query*, memanipulasi, dan mendefinisikan data, dan untuk menyediakan akses kontrol. SQL pertama kali dikembangkan di IBM pada tahun 1940-an dengan Oracle sebagai kontributor utama, yang menyebabkan penerapan standar SQL ANSI, SQL telah memacu banyak ekstensi dari perusahaan seperti IBM, Oracle, dan Microsoft. Meskipun SQL masih banyak digunakan saat ini, bahasa pemrograman lain juga mulai muncul.

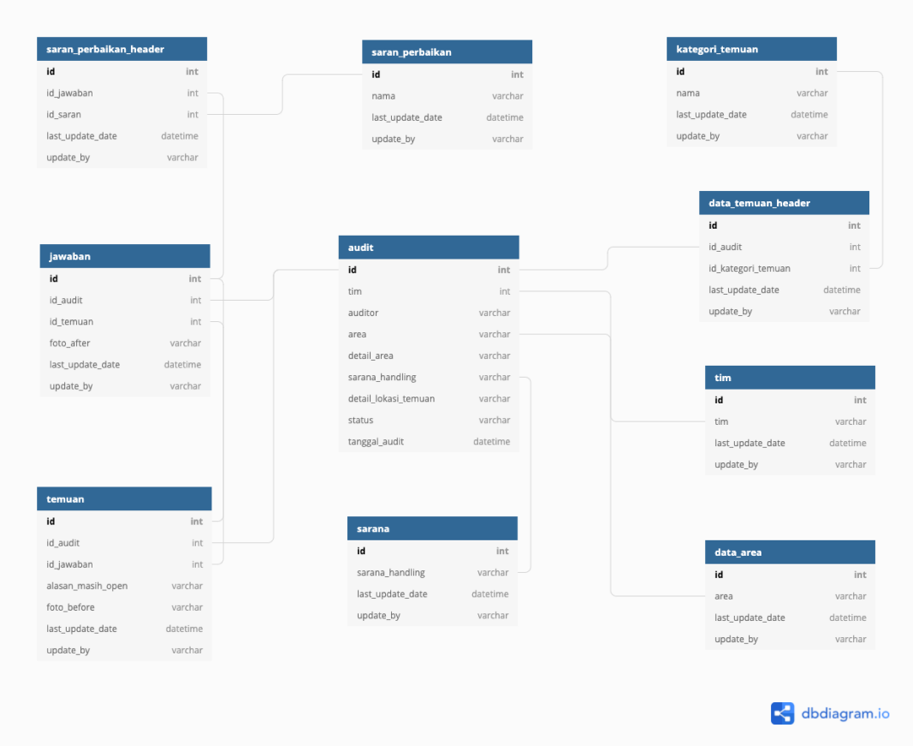
Hampir setiap aplikasi android yang kami buat saat di CV. KHS pasti membutuhkan sebuah *database* atau mungkin beberapa *database* untuk menyimpan data yang diproses oleh aplikasi tersebut. *Database* yang kami gunakan di aplikasi Audit & Verification ini adalah PostgreSQL.

* 1. Alat dan Bahan
     1. Komputer
     2. DBeaver
     3. https://dbdiagram.io/
     4. Koneksi Internet
  2. Keselamatan Kerja
     1. Berdoa sebelum mengerjakan
     2. Memastikan alat dan bahan sesuai dengan yang dibutuhkan
     3. Memastikan alat dan bahan berada di posisi yang tepat
     4. Menggunakan alat sesuai dengan fungsinya
     5. Memperhatikan posisi duduk dan jarak pandang dengan komputer
     6. Berhati hati dengan air dan listrik
     7. Mematikan perangkat jika telah selesai digunakan
  3. Gambar Kerja



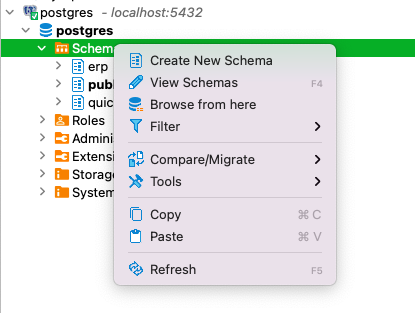
Gambar 3.34. Gambar Kerja Pembuatan Database Aplikasi Audit & Verification

* 1. Analisa dan Langkah Kerja
     1. Langkah pertama adalah menganalisa alur aplikasi yang dibutuhkan user. analisa data - data apa yang dibutuhkan, yang diolah dan yang di gunakan.
     2. Setelah itu, buat ERD sesuai hasil analisa yang kami lakukan. Untuk membuat nya bisa menggunakan service dbdiagram.io.



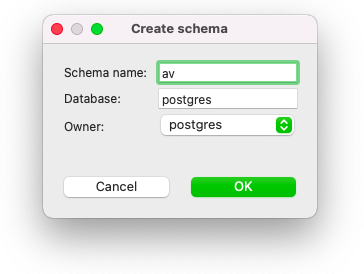
Gambar 3.35. ERD Database

* + 1. Setelah ERD sudah dibuat, kemudian buat *database* nya di PostgreSQL dengan menggunakan Sql Tools DBeaver.
    2. Untuk membuat *database* / skema di DBeaver, buka DBeaver kemudian hubungkan dengan server. Setelah terhubung dengan server, buat skema baru dengan nama av. Caranya dengan klik kanan di Schemas pada salah satu Database dan pilh Create New Schema.



Gambar 3.36. Membuat Skema Baru

Setelah itu akan muncul dialog Create Schema, isi Schema name dan tekan tombol OK untuk membuat schema nya.



Gambar 3.37. Jendela Untuk Membuat Skema Baru

* + 1. Kemudian buat script file baru dan tulis query untuk membuat tabel tabel nya. Berikut query untuk membuat tabel tabel yang dibutuhkan aplikasi Audit & Verification :

CREATE TABLE "audit" (

"id" SERIAL PRIMARY KEY,

"tim" int,

"auditor" varchar,

"area" varchar,

"detail\_area" varchar,

"sarana\_handling" varchar,

"detail\_lokasi\_temuan" varchar,

"status" varchar,

"tanggal\_audit" datetime

);

CREATE TABLE "data\_temuan\_header" (

"id" SERIAL PRIMARY KEY,

"id\_audit" int,

"id\_kategori\_temuan" int,

"last\_update\_date" datetime,

"update\_by" varchar

);

CREATE TABLE "data\_area" (

"id" SERIAL PRIMARY KEY,

"area" varchar,

"last\_update\_date" datetime,

"update\_by" varchar

);

CREATE TABLE "tim" (

"id" SERIAL PRIMARY KEY,

"tim" varchar,

"last\_update\_date" datetime,

"update\_by" varchar

);

CREATE TABLE "kategori\_temuan" (

"id" SERIAL PRIMARY KEY,

"nama" varchar,

"last\_update\_date" datetime,

"update\_by" varchar

);

CREATE TABLE "saran\_perbaikan\_header" (

"id" SERIAL PRIMARY KEY,

"id\_jawaban" int,

"id\_saran" int,

"last\_update\_date" datetime,

"update\_by" varchar

);

CREATE TABLE "saran\_perbaikan" (

"id" SERIAL PRIMARY KEY,

"nama" varchar,

"last\_update\_date" datetime,

"update\_by" varchar

);

CREATE TABLE "sarana" (

"id" SERIAL PRIMARY KEY,

"sarana\_handling" varchar,

"last\_update\_date" datetime,

"update\_by" varchar

);

CREATE TABLE "jawaban" (

"id" SERIAL PRIMARY KEY,

"id\_audit" int,

"id\_temuan" int,

"foto\_after" varchar,

"last\_update\_date" datetime,

"update\_by" varchar

);

CREATE TABLE "temuan" (

"id" SERIAL PRIMARY KEY,

"id\_audit" int,

"id\_jawaban" int,

"alasan\_masih\_open" varchar,

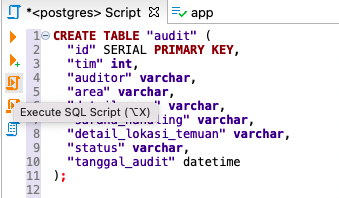
"foto\_before" varchar,

"last\_update\_date" datetime,

"update\_by" varchar

);

* + 1. Setelah ditulis semua, klik tombol *execute* untuk menjalankan *query* nya dan membuat tabel tabel nya.



Gambar 3.38. Menjalankan Query Di DBeaver

* 1. Quality Control

Setelah query untuk membuat semua table nya dijalankan, periksa table - table nya di DBeaver. Caranya dengan mengklik skema av yang sudah dibuat dan periksa apakah tabel - tabel nya sudah benar - benar terbuat atau belum.

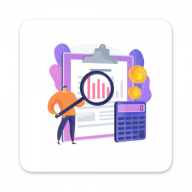


Gambar 3.39. Daftar Tabel Yang Berhasil Dibuat

* 1. Kesimpulan

Database adalah kumpulan informasi yang terorganisir, atau data yang biasanya disimpan secara elektronik di dalam sistem komputer. Untuk mengelola *database* biasanya menggunakan bahasa pemrograman SQL.

1. Pembuatan Aplikasi Audit & Verification
   1. Penjelasan Pekerjaan



Gambar 3.40. Logo Aplikasi Audit & Verification

Aplikasi Audit & Verification adalah aplikasi yang digunakan untuk melakukan audit di kawasan CV. Karya Hidup Sensosa cabang Tusono. Aplikasi ini digunakan untuk menginput audit yang dilakukan oleh auditor jika ada audit yang ditemukan saat patroli. Jika ditemukan audit di suatu seksi, maka dengan menggunakan aplikasi ini, membuat audit baru sesuai keadaan di lokasi.

Kemudian auditee menggunakan aplikasi ini untuk melihat audit yang dibuat auditor diseksinya. Dan menggunakan data yang telah dininputkan oleh auditor untuk melakukan tindak lanjut / follow up terhadap audit tersebut dan kemudian menginputkan follow up nya di aplikasi ini.

Setelah di follow up, maka auditor akan memverifikasi follow up yang telah dilakukan oleh auditee dan memutuskan apakah audit tersebut telah selesai (close) atau perlu dilakukan tindakan lebih lanjut lagi (open). Dan kemudian jika di open lagi maka auditee harus melakukan follow up lagi sesuai dengan saran dari auditor. Dan proses ini berulang terus sampai auditor menutup (close) audit tersebut.

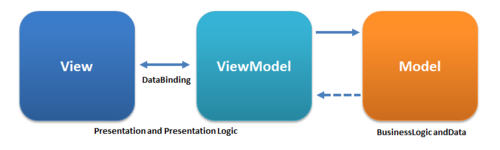
Di aplikasi ini juga ada menu monitoring untuk memonitoring semua data audit yang ada di aplikasi ini.

Dalam project ini penulis menggunakan Android Studio sebagai IDE nya, Kotlin sebagai bahasa pemrogramannya, dan MVVM sebagai software architecture patternnya.



Gambar 3.41. Logo Kotlin

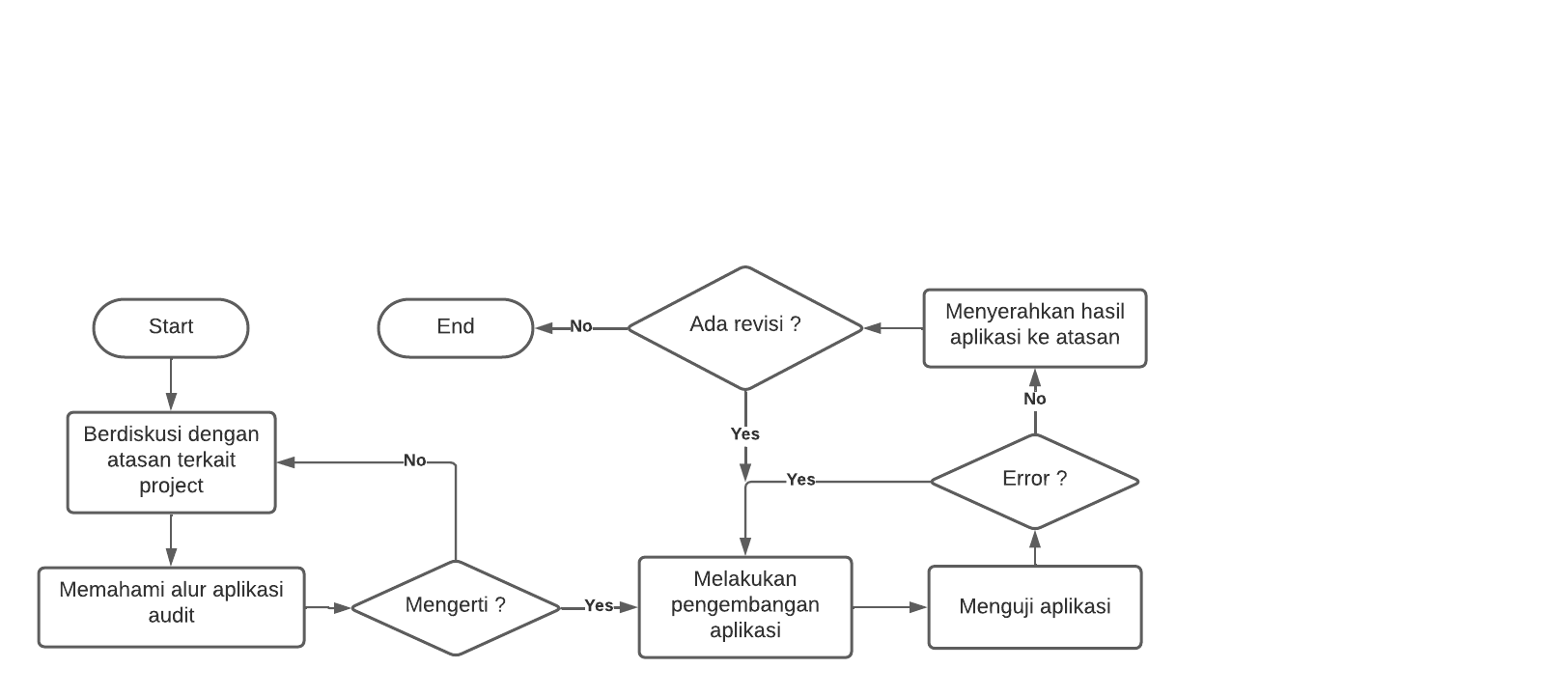
Kotlin adalah sebuah bahasa pemrograman dengan pengetikan statis yang berjalan pada Mesin Virtual Java (JVM) ataupun menggunakan kompiler LLVM. Pengembang utama Kotlin adalah tim programmer dari JetBrains yang bermarkas di Rusia yang pertama dirilis tahun 2011. Setelah Google mengumumkan bahwa Kotlin menjadi bahasa utama atau yang direkomendasikan ketika membuat aplikasi Android bersama dengan Java dan C++.



Gambar 3.42. Diagram Arsiteksur MVVM

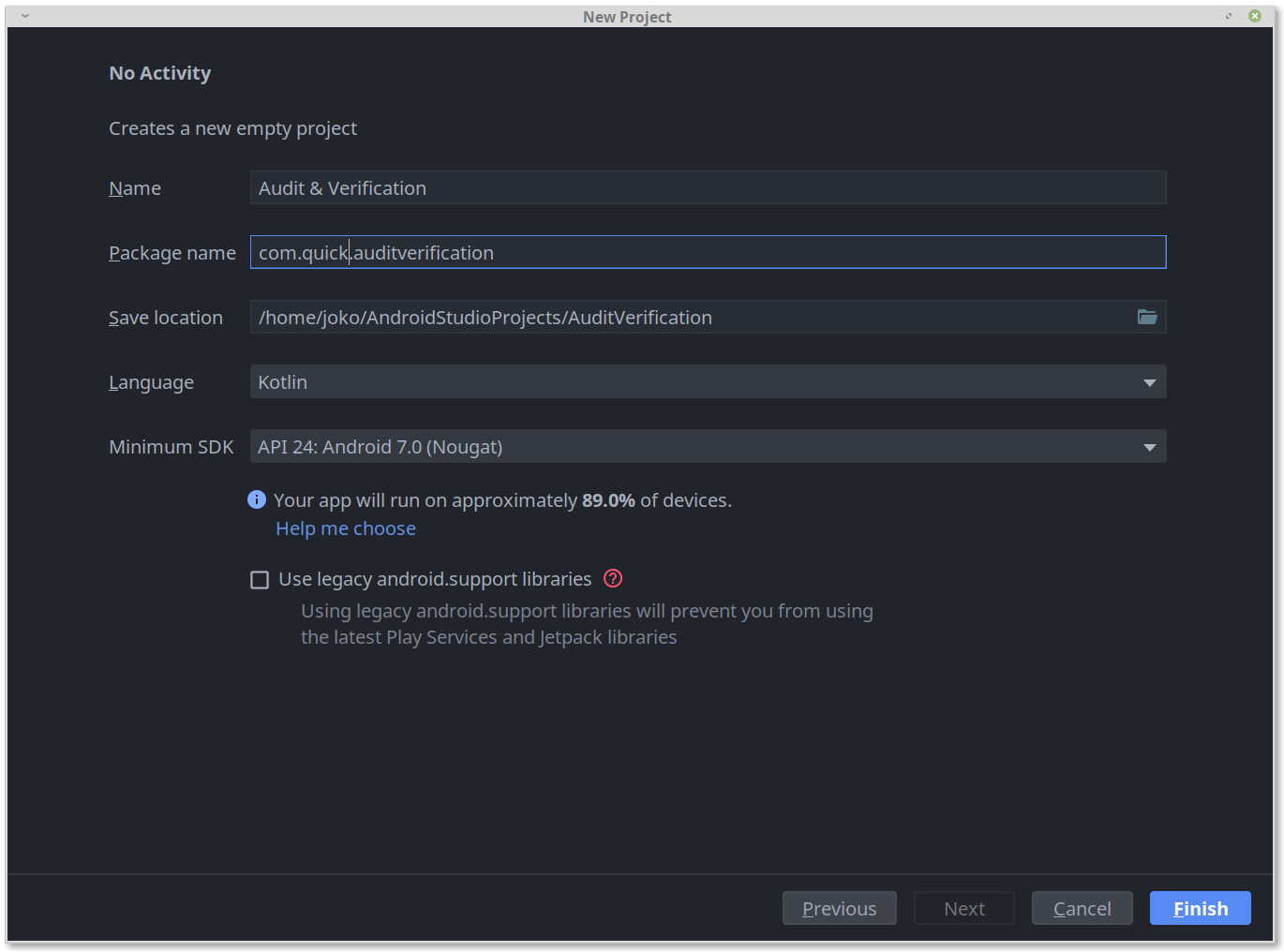
MVVM (Model View ViewModel) adalah salah satu Architectural Patterns yamg membagi tanggung jawab dalam pembuatan Graphical User Interface (View) kepada tiga komponen utama, yaitu Model, View dan ViewModel. View bertanggung jawab untuk semua hal yang berhubungan dengan UI. Model merupakan komponen yang bertanggung jawab untuk menyediakan data yang dibutuhkan. ViewModel adalah komponen inti dari Architectural Pattern ini, yang tugasnya menyimpan dan mengambil data dari Model untuk nantinya ditampilkan oleh View.

* 1. Alat dan Bahan
     1. PC
     2. Android Studio
     3. Visual Studio Code
     4. Insomnia
     5. DBeaver
     6. Web Browser
     7. Internet
  2. Keselamatan Kerja
     1. Berdoa sebelum mengerjakan
     2. Memastikan alat dan bahan sesuai dengan yang dibutuhkan
     3. Memastikan alat dan bahan berada di posisi yang tepat
     4. Menggunakan alat sesuai dengan fungsinya
     5. Memperhatikan posisi duduk dan jarak pandang dengan komputer
     6. Berhati hati dengan air dan listrik
     7. Mematikan perangkat jika telah selesai digunakan
  3. Gambar Kerja



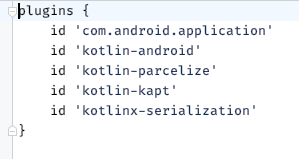
Gambar 3.43. Gambar Kerja Pembuatan Aplikasi Audit & Verification

* 1. Analisa dan Langkah Kerja
     1. Langkah pertama adalah membuka Android Studio IDE dan membuat project baru dengan nama Audit & Verification, nama package com.quick.auditverification, bahasa pemrograman Kotlin, dan minimum versi SDK 24 (Android versi Nougat).



Gambar 3.44. Jendela Pembuatan Project Baru

* + 1. Setelah project selesai dibuild, kemudian buka file build.gradle dan tambahkan beberapa *dependencies* yang dibutuhkan.



Gambar 3.45. Plugin Di Gradle



Gambar 3.46. Daftar Dependencies



Gambar 3.47. Daftar Dependencies



Gambar 3.48. Daftar Dependencies

* + 1. Setelah semua dependencies terdownload, kemudian buat class App yang mewarisi class Application.



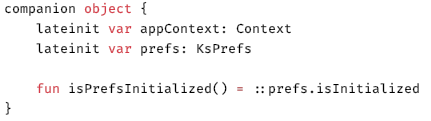
Gambar 3.49. Kelas App Untuk Aplikasi Audit

Di function onCreate yang dioverride, tambah baris berikut.



Gambar 3.50. Code Function onCreate() App

Di class App ini juga ada companion object yang berisi appContext dan prefs yang dapat diakses di seluruh aplikasi.



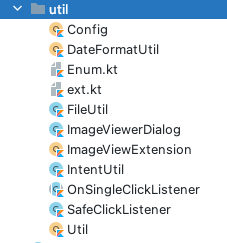
Gambar 3.51. Companion Object di Kelas App

Kemudian agar class App ini dijalankan Ketika aplikasi ini dibuka, tambahkan App ini di manifest pada bagian android:name.



Gambar 3.52. Penggunaan Kelas App Di Manifest

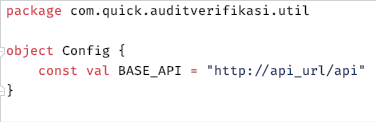
* + 1. Kemudian buat package baru dengan nama util. Dalam package ini akan berisi class class dan fungsi fungsi yang berguna untuk aplikasi nya nanti.



Gambar 3.53. Package Util

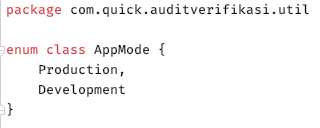
* + 1. Kemudian didalam package util ini, buat beberapa file berikut:

File Config.kt berisi code untuk menyimpan base url dari api yang digunakan aplikasi ini.

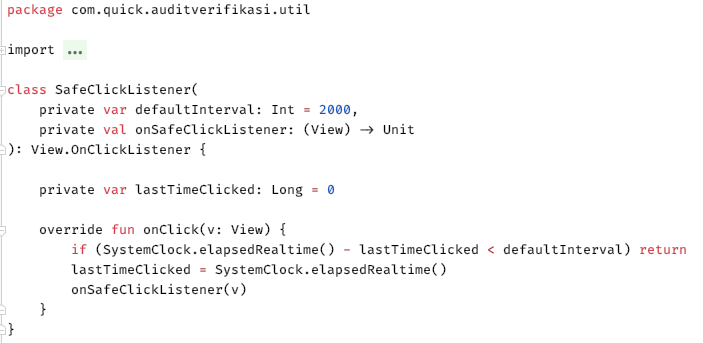


Gambar 3.54. Code Di File Config.kt

File AppMode.kt yang berfungsi untuk mendefinisikan mode yang ada di aplikasi ini.

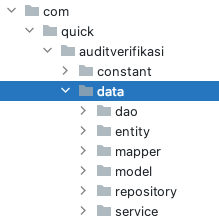


Gambar 3.55. Code Di File AppModel.kt

Class yang berfungsi mencegah user mengeklik tombol berulang kali. 

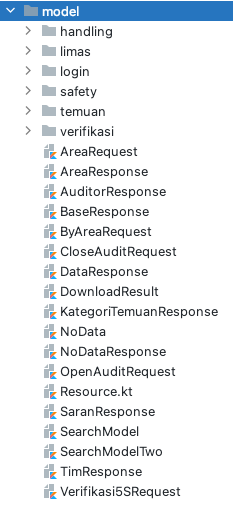
Gambar 3.56. Code Safe Button Click Listener

* + 1. Kemudian buat package data. Package data ini berisi class class yang bertanggung jawab sebagai layer data untuk aplikasi ini.



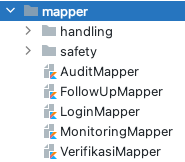
Gambar 3.57. Package Data

* + 1. Kemudian di dalam package data, buat package model. Package model ini berisi data – data class yang bergungsi sebagai data transaction object untuk semua yang berhubungan dengan network.



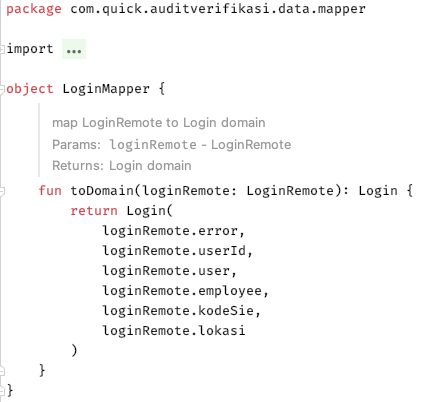
Gambar 3.58. Package Model

* + 1. Setelah itu, buat package lain dengan nama mapper. Dalam package ini berisi object object yang berfungsi untuk mengubah data transaction object menjadi domain object.



Gambar 3.59. Package Mapper

Berikut adalah contoh salah satu mapper. Mapper ini berfungsi untuk mengubah dto login menjadi domain object login.



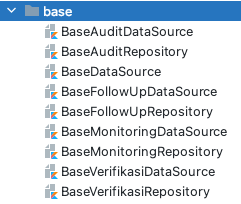
Gambar 3.60. Code Object Mapper

* + 1. Kemudian buat package lain dengan nama repository. Package ini berisi class – class datastore dan repository untuk aplikasi ini.



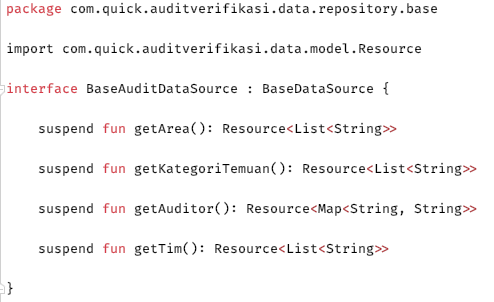
Gambar 3.61. Package Repository

Dalam package ini ada package utama dengan nama base. Jadi semua datastore dan repository mewarisi interface base ini.



Gambar 3.62. Package Base Repository

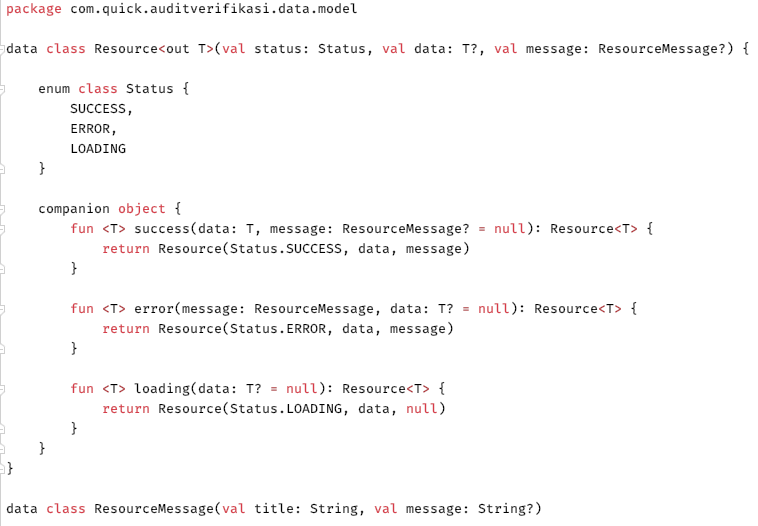
Ini adalah contoh base data source audit.



Gambar 3.63. Interface Base Audit Data Source

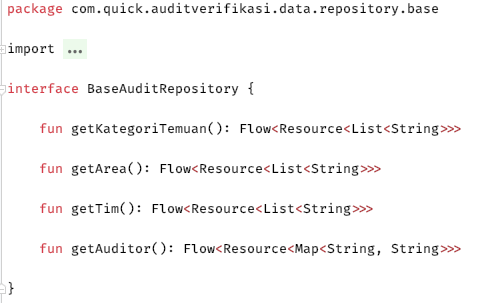
Pada interface base audit tersebut, interface ini mewarisi BaseDataSource, sehingga BaseAuditDataSource ini menjadi memiliki semua behaviour yang ada di interface BaseDataSource. Kemudian function function yang ada didalamnya menggunakan operator suspend fun yang berarti function ini dapat ditunda pemanggilannya dan dapat dijalankan kembali kapanpun secara asyncronus melalui suspend function yang lain atau melalui coroutine scope.

Kemudian function funcion ini juga mereturn object yang bertipe Resource<T>, dimana Resource ini adalah sebuah generic wrapper class yang berfungsi untuk wrapping data yang didalam tipe generic nya dan layer UI kita untuk mengetahui semua kemungkinan network request yang dibuat, seperti ketika sukses, error ataupun ketika loading. Sehingga proses nya dapat dengan mudah dideteksi oleh layer UI.



Gambar 3.64. Data Class Resource

Ini adalah contoh base repository audit.



Gambar 3.65. Interface Base Audit Repository

Interface ini digunakan sebagai kontrak yang harus dipenuhi oleh implementasi sesungguhnya dari repository ini. Function function nya adalah function biasa yang mengembalikan tipe data Kotlon Flow (Flow<Resource<T>>). Kotlin Flow sendiri adalah tipe yang dapat memancarkan beberapa nilai secara berurutan, berlawanan dengan fungsi suspend yang mengembalikan hanya satu nilai.

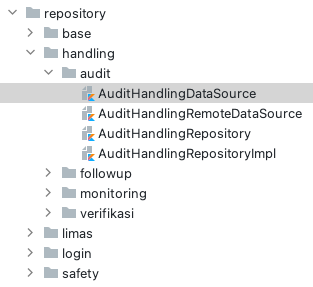
Repository ini adalah class yang dipanggil secara langsung oleh viewmodel untuk melakukan network request, sehingga function – function nya menggunakan return type flow, karena repository dapat memancarkan / emit banyak nilai secara berurutan. Hal ini dilakukan karena network request adalah sebuah operasi panjang yang tidak pasti berhasil, sehingga dengan menggunakan flow ini, viewmodel dapat mengeahui state network request yang sedang terjadi, seperti sedang loading, result nya error, atau sukses.

* + 1. Kemudian buat package lain yang merupakan implementasi dari base repository dan dataaseource yang dibuat di package base, seperti handling, limas, login dan safety.



Gambar 3.66. Package Repository Handling

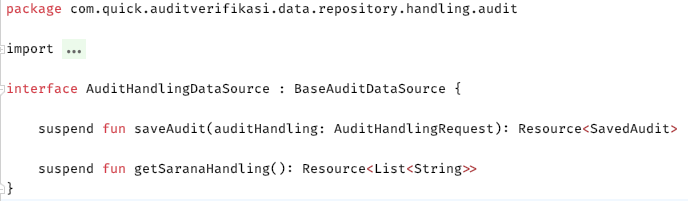
Setiap package repository handling, limas, dan safety memiliki struktur berikut didalamnya.



Gambar 3.67. Package Repository Handling

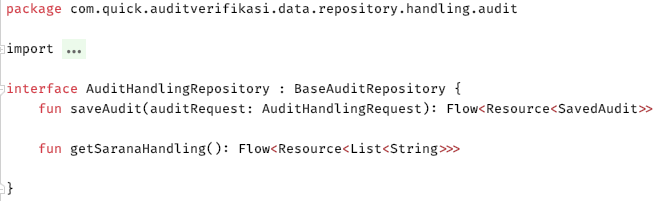
Package audit digunakan untuk membuat audit baru. Package follow up berisi code yang digunakan untuk follow up audit yang telah dibuat. Package verifikasi digunakan untuk memverifikasi follow up yang telah dibuat terhadap audit tersebut. Dan yang terakhir package monitoring berfungsi untuk memonitoring semua audit yang ada di aplikasi ini.

Berikut adalah salah satu datasource audit handling. Interface ini juga mewarisi interface base audit datasource yang ada di package base.



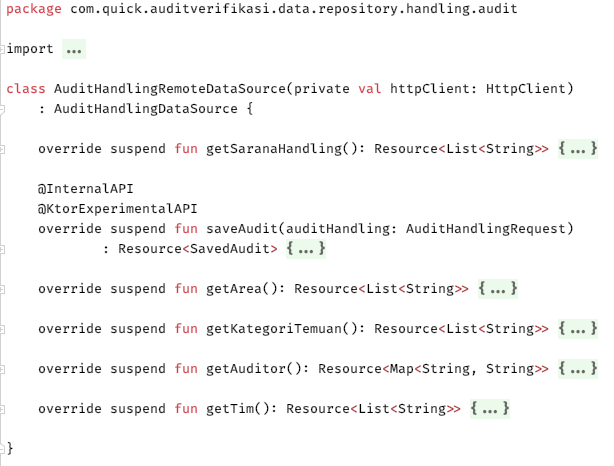
Gambar 3.68. Interface Audit Handling Data Source

Berikut adalah salah satu repository audit handling. Interface ini juga mewarisi interface base audit repository yang berada di package base.



Gambar 3.69. Interface Audit Handling Repository

Kemudian berikut adalah salah satu datasource audit di bagian handling.



Gambar 3.70. Kelas Audit Handling Data Source

Class datasource ini adalah bentuk implemeentasi dari interface audit datasource. Class ini memiliki constructor yang berisi http client, yang nantinya akan di inject oleh dependency injector. http client ini digunakan sebagai client yang digunakan untuk melakukan network request.

Berkut adalah salah satu function di class audit handling remote datasource. Function ini digunakan untuk mendapatkan semua sarana handling dari api menggunakan http request.



Gambar 3.71. Function Di Kelas Data Source

Semua network request yang dilakukan di datasource, dilakukan didalam extension function dataStoreRunCatching. Function ini berfungsi untuk error handling dari network request yang dilakukan, sehingga code ini tidak perlu ditulis di semua network request yang dilakukan di semua datasource.



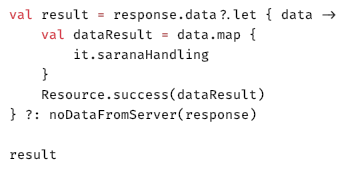
Gambar 3.72. Extension Function Untuk Semua Data Source

Kemudian untuk melakukan network request, menggunakan object httpClient. Caranya dengan menggunakan method get<T>(url: String), dengan T adalah jenis tipe data yang diminta dari nework request yang dilakukan, dan url sebagai endpoint dari api nya. Kemudian addDefaultHeader() digunakan untuk menambah default header untuks setiap request, default header ini berisi jenis connection yang digunakan, apakah development atau production berdasarkan pilihan mode yang dipilih user ketika login di aplikasi ini.



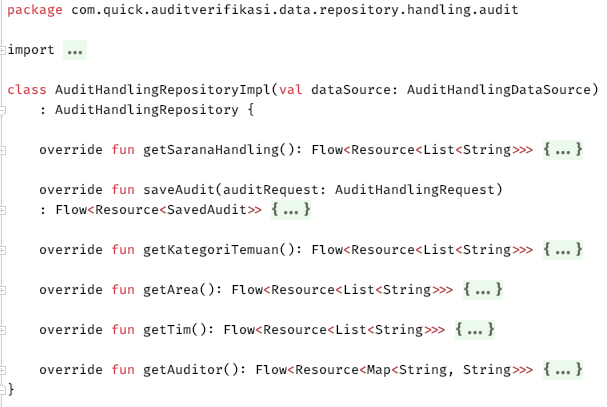
Gambar 3.73. Code Untuk Mendapatkan Response Dari Server

Kemudian setelah data diperoleh dari api, ubah data dari api tersebut sesuai kebutuhan dan kambalikan hasil yang didapatkan. Dalam kasus ini, jika hasil dari network request tidak null maka akan di map kemudian diubah menjadi list of sarana handling dan kemudian disimpan dalam object Resource.success() yang kemudian nanti akan dikembalikan. Tetapi jika response nya null, maka akan mengembalikan error tidak ada data dari server.



Gambar 3.74. Code Untuk Mengubah Response Dari Server Ke Bentuk Yang Dibutuhkan

Kemudian berikut adalah salah satu contoh repository dari audit handling.



Gambar 3.75. Kelas Implementasi Interface Repository

Class ini adalah implementais dari interface audit handling repository.Class ini berfungsi untuk menghubungkan viewmodel dan datasource. Dibagian cunstructor class ini membutuhkan audit handling datasource yang nantinya akan di inject oleh dependency injector.

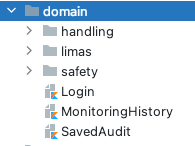
Berikut ini adalah salah satu function yang ada di class audit handling repository. Function ini adalah function yang mengembalikan flow yang berisi resource dari sebuah list of string. Karena ini adalah flow, jadi function ini dapat memancarkan beberapa nilai secara berurutan. Sehingga ketika function ini dipanggil, pertama dia akan memancarkan state loading, kemudian delay selama 400 untuk menunjukan proses loading, kemudian memancarkan lagi nilai yang didapatkan dari datasource.



Gambar 3.76. Function Di Kelas Implementasi Repository

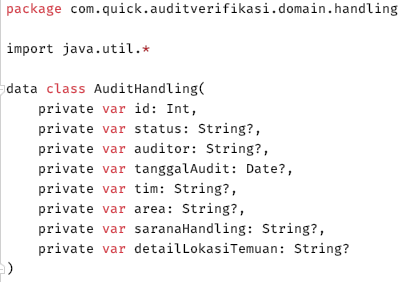
Semua datasource dan repository di aplikasi ini memiliki dasar code seperti yang sudah penulis jelaskan diatas.

* + 1. Kemudian buat package domain, package ini adalah package yang berisi bussines model dari aplikasi ini. Dalam package ini berisi model - model class yang dipakai di seluluh aplikasi ini.



Gambar 3.77. Package Domain

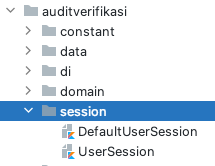
Berikut adalah contoh salah satu domain model audit handling.



Gambar 3.78. Data Class Audit Handling

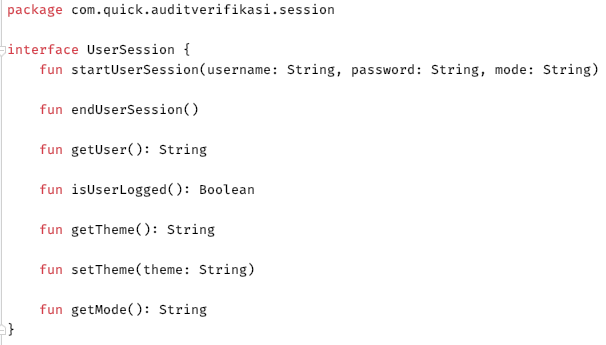
Semua domain model adalah sebuah data class yang berisi berbagai property dari sebuah entity tersebut.

* + 1. Kemudian buat package baru lagi dengan nama session. Package ini berisi code yang berfungsi untuk mengatur user session didalam aplikasi ini. Dalam package ini berisi interface UserSession dan implementasi dari UserSession yang bernama DefaultUserSession.



Gambar 3.79. Package Session

Berikut adalah code dari interface UserSession. Function startUserSesion berfungsi untuk memulai user session, function ini dipanggil setelah user berhasil login dengan username dan password yang benar. Kemudian endUserSession berfungsi untuk mengakhiri user session, function ini dipanggil ketika user melakukan logout dari keadaan sudah login.



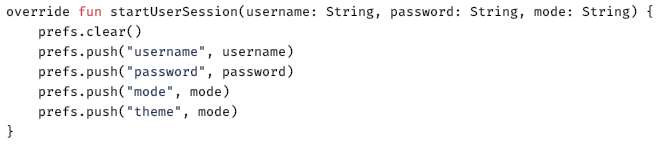
Gambar 3.80. Interface Session

Berikut adalah isi dari class DefaultUserSession. Class ini adalah implemtasi dari interface UserSession.



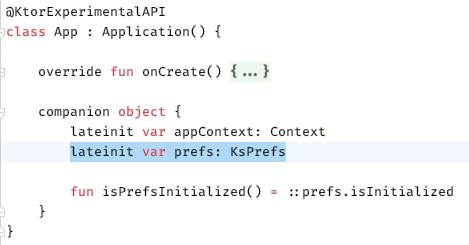
Gambar 3.81. Kelas Implemtasi Dari Interface Session

Dalam function startUserSession, berisi menyimpan pengaturan ketika login berhasil login, seperti username, password, dan mode applikasinya, yang kemudian disimpan didalam shared preferences.



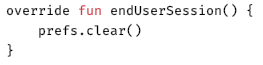
Gambar 3.82. Function Untuk Memulai Session Baru

Object prefs adalah object yang dibuat di class Application sehingga dapat dipanggil di seluruh project. Object ini digunakan untuk melakukan melakukan operasi dengan shared preferences. Shared preferences sendiri adalah salah satu teknologi yang digunakan untuk menyimpan data primitive berbentuk pasangan key value yang kemudian disimpan di storage android.



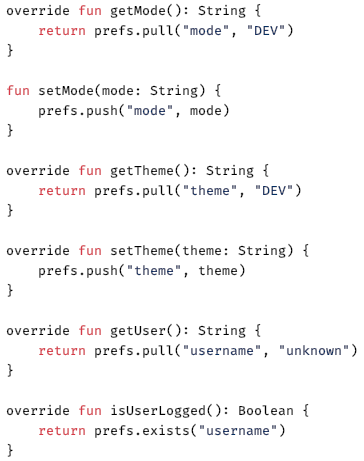
Gambar 3.83. Variabel prefs Di Class App

Kemudian function endUserSession berisi menghapus semua data yang tersimpan di shared preferences untuk aplikasi ini.



Gambar 3.84. Function Untuk Menghentikan Session

Kemudian untuk function function lainnya hanya mengambil data dari shared preferences sesuai dengan namanya.



Gambar 3.85 Code Lain di Kelas Session

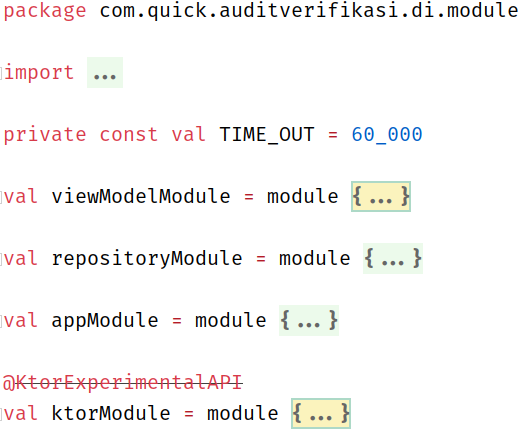
* + 1. Kemudian buat package baru dengan nama di.modules. Di dalam package inilah dependency injection dibuat. Dependency Injection adalah sebuah teknik untuk mengatur cara bagaimana suatu objek dibentuk ketika terdapat objek lain yang membutuhkan. Sehingga ketika sebuah code membuhkan instance dari sebuah object yang object tersebut dibutuhkan dibanyak tempat, kita dapat menggunakan dependency injection ini untuk mendapatkan instance tersebut.

Untuk membuat dependency injection di appliaksi ini, buat kotlin file baru dengan nama AppModule.kt.



Gambar 3.86. Package DI Module

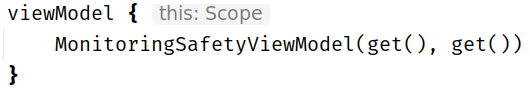
Kemudian buat code untuk file tersebut seperti berikut.



Gambar 3.87. Code FIle Modules.kt

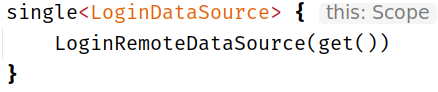
Setiap module berisi instruksi untuk memberitahui dependency injector bagaimana cara membuat sebuah instance ketika ada yang membutuhkannya.

Berikut adalah salah satu dari pendeklarasian untuk module view model.



Gambar 3.88. Pendeklarasian View Model

Berikut adalah salah satu pendeklarasian module repository.



Gambar 3.89. Pendeklarasian Data Source

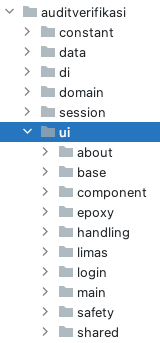
Semua module di file ini memiliki struktur code yang sama seperti diatas.

Untuk membuat agar module module ini dapat digunakan, module tersebut harus diload di function onCreate di class Application.



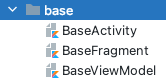
Gambar 3.90. Code Untuk Meload Module

* + 1. Kemudian membuat package ui. Package ui ini adalah package yang berisi code untuk membuat tampilan yang dilihat secara langsung oleh user selama menggunakan aplikasi.



Gambar 3.91. Package UI

Untuk package ui.base, berisi base / dasar class untuk semua class di package ui ini.



Gambar 3.92. Package Base UI

Berikut adalah code untuk class base activity. Class ini adalah sebuah abstract class yang mewarisi class app compat activity, karena class ini tidak adalah abstract class sehingga class ini tidak bisa diinisialisasi secara langsung.



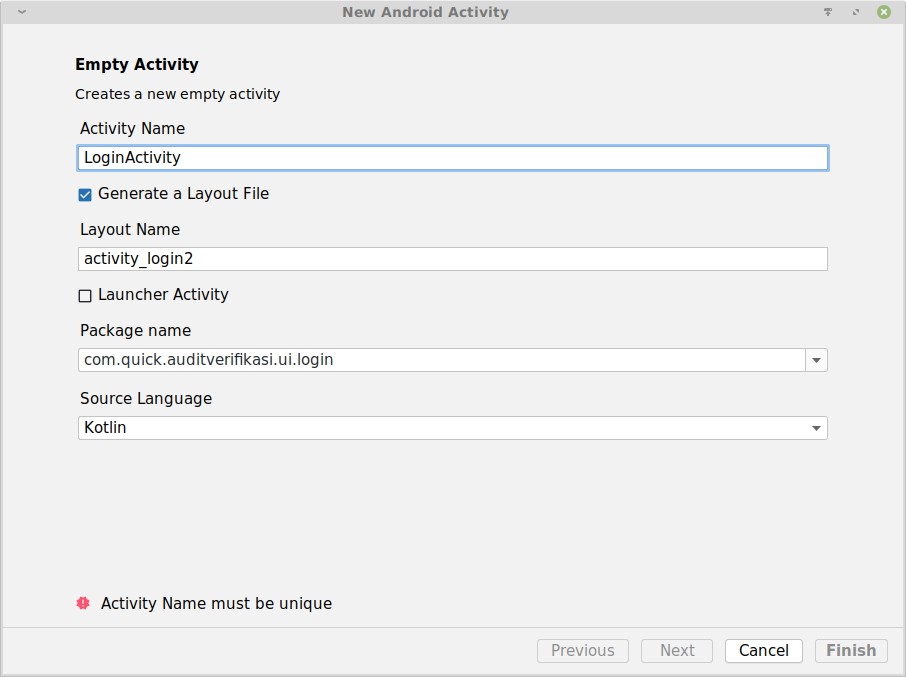
Gambar 3.93. Kelas Base Activity

Kemudian berikut adalah code di class base fragment.



Gambar 3.94. Kelas Base Fragment

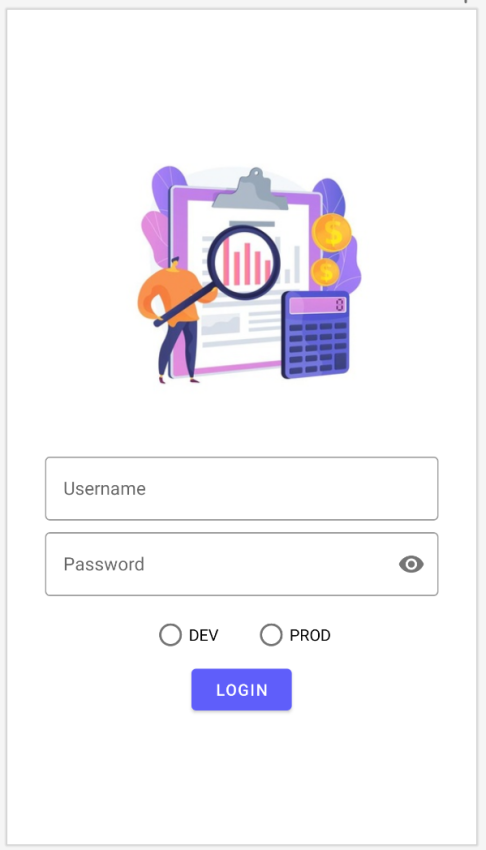
* + 1. Kemudian buat activity login untuk menampilkan halaman login. Untuk membuat activity baru, klik kanan di package tempat activity tersebut ingin dibuat, kemudian pilih New > Activity > Empty Activity.



Gambar 3.95. Jendela Untuk Membuat Activity Baru

Kemudian akan muncul dialog untuk membuat activity baru seperti diatas, beri nama ActivityLogin dan pastikan Generate a Layout File dipilih, dan juga pilih bahasa pemrograman untuk activity ini. Stelah selesai tekan finish untuk membuat activity nya.

* + 1. Kemudian buka file xml dari activity login ini dan buat ui nya menjadi seperti berikut ini.



Gambar 3.96. UI Activity Login

* + 1. Setelah ui dari activity login dibuat, kemudian buat code untuk activity login ini. Buka file ActivityLogin dan buat



Gambar 3.97. Code Activity Login

Kemudian di function onCreate, buat code berikut untuk mengecek apakah user sebelumnya sudah login atau belum, jika sudah maka tutup activity ini dan buka activity main.



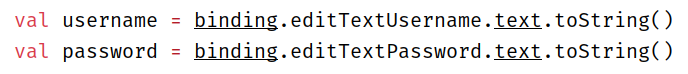
Gambar 3.98. Pengecekan Apakah User Sudah Login Atau Belum

Kemudian buat code untuk menghandle event user menekan tombol login.



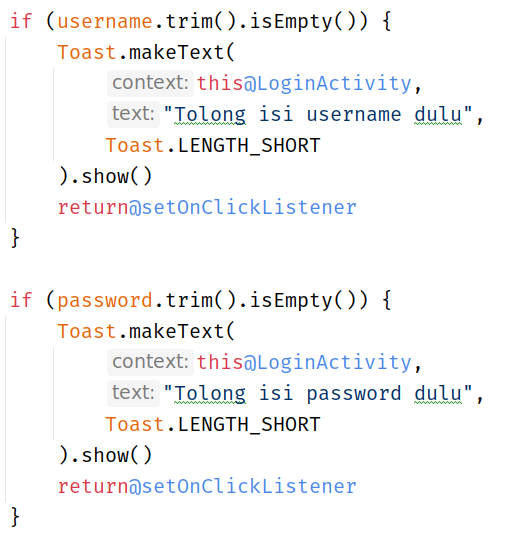
Gambar 3.99. Code Untuk Mengatur Click Listener

Ketika user menekan tombol login, yang pertama kali dilakukan adalah mengambil username dan password yang user masukan.



Gambar 3.100. Code Untuk Mendapatkan Username Dan Password Dari User

Kemudian melakukan validasi untuk mengecek apakah username dan password yang user masukan valid atau tidak. Jika username dan password yang dimasukan valid, maka melanjutkan ke proses selanjutnya, tetapi jika tidak, berhenti dan menampilkan pesan error sesuai dengan kondisinya.



Gambar 3.101. Code Untuk Validasi Username dan Password

Kemudian setelah proses validasi berhasil, melakukan proses login.



Gambar 3.102. Code Untuk Melakukan Proses Login

Jika username dan password yang dimasukan ada di database, maka user akan diarahkan ke activity main dan menyimpan session bahwa user telah login di shared preferences.



Gambar 3.103. Code Untuk Proses Login Jika Berhasil

Tetapi jika login gagal, maka akan menampilkan pesan error nya.



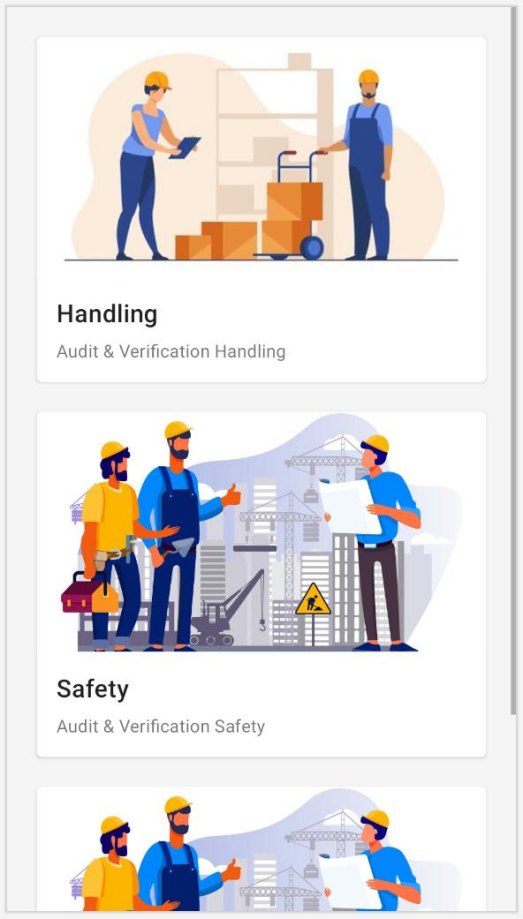
Gambar 3.104. Code Yang Dijalankan Ketika Login Gagal

* + 1. Kemudian buat activity main untuk menampilkan halaman menu utama. Activity ini dibuat dengan nama MainActivity dan berada di package ui.main.



Gambar 3.105 .Package Main

Lalu buat ui di file xml activity main.



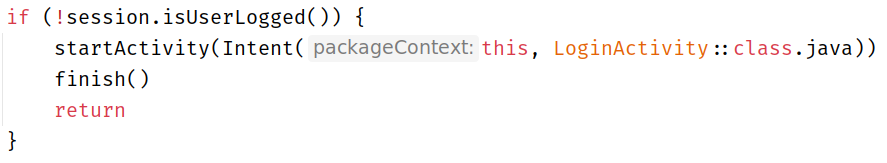
Gambar 3.106. UI Activity Main

* + 1. Setelah itu buat code di activity mainnya.



Gambar 3.107. Code Main Activity

Function onPrefReady() dijalankan ketika shared preferences sudah siap untuk digunakan. Di dalam onPrefRaedy() perama - tama dilakukan pengecekan apakah user sudah benar benar login atau belum, jika belum maka akan diarahkan kembali kehamalam login, jika sudah melanjutkan ke proses selanjutnya,



Gambar 3.108. Pengecekan User Session

Kemudian dilakukan proses pengaturan tema sesuai dengan jenis mode yang dipiih user ketika login, yaitu mode development atau production.



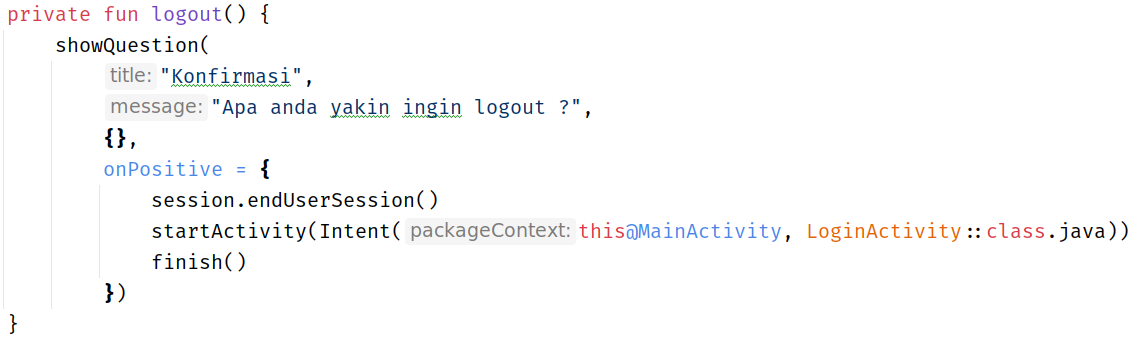
Gambar 3.109. Mengatur Tema Di Activity

Didalam activity ini terdapat 3 menu utama, yaitu menu handling, safety dan 5S, setiap menu memiliki daftar kegiatan yang sama yaitu membuat audit, follow up audit, memverifikasi audit, dan memonitoring audit. Sehingga buat code untuk handle event ketika user memilih salah satu menu yang ada.



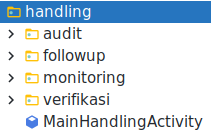
Gambar 3.110. Listener Ketika User Menekan Salah Satu Menu

Kemudian buat code untuk event ketika user ingin logout dari aplikasi ini. Code berikut dipanggil ketika user menekan tombol logout di activity main, dan meminta konfirmasi dari user apakah yakin ingin melakukan logout dari aplikasi ini atau tidak.



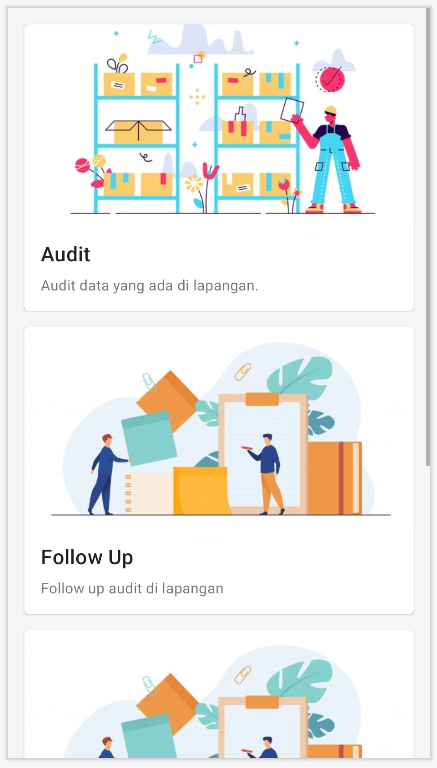
Gambar 3.111. Code Konfirmasi Sebelum Logout

* + 1. Kemudian buat package baru untuk sub menu pertama yaitu menu handling. package ini berada di ui.handling.



Gambar 3.112. Package Handling

Setelah itu buat activity main handling dan dengan ui seperti berikut.



Gambar 3.113. UI Main Activity

Setelah ui dari activity ini dibuat, kemudian buat code untuk activity ini.



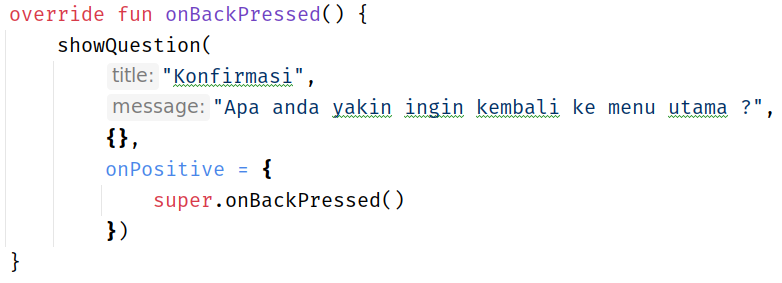
Gambar 3.114. Activity Main Handling

Hal utama yang dilakukan activity ini adalah menampilkan semua menu dari sum menu handling ini, dan mengarahkan ke menu yang dipilih user. Sehingga buat code untuk handle event user memilih menu tertentu dan mengarahkan ke menu yang dituju.



Gambar 3.115. Button Listener Untuk Setiap Menu

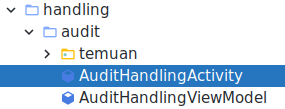
Di activity ini juga ada permintaan konfirmasi ketika user menekan tombol kembali.



Gambar 3.116. Konfirmasi Sebelum Kembali Ke Menu Utama

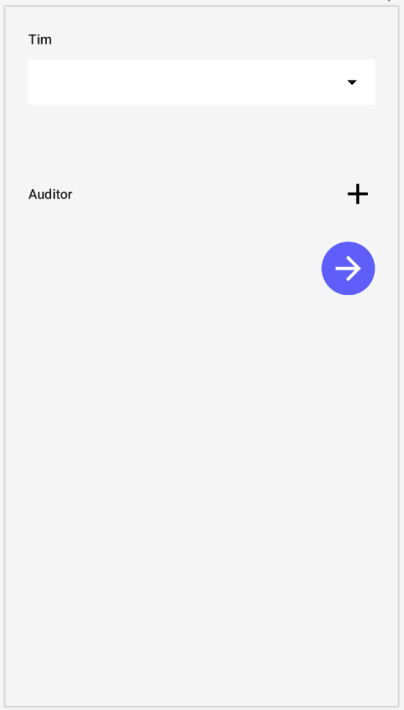
Jika user mengkonfirmasi maka akan kembali ke menu utama, tetapi jika tidak maka tidak akan melakukan apa - apa.

* + 1. Kemudian buat activity audit handling untuk membuat audit baru di sub menu handling ini.



Gambar 3.117. Package Audit Handling

Kemudian buat ui untuk activity audit handling seperti berikut.



Gambar 3.118. UI Activity Audit Handling

Setelah UI nya jadi, buat code untuk activity audit handling nya.



Gambar 3.119. Code Activity Audit Handling

Kemudian di function onCreate() buat code berikut yang akan dijalankan ketika user menambah auditor baru. Jika user yang dipilih sudah ada, maka akan menampilkan pesan peringatan bahwa user yang dipilih sudah ditambahkan sebelumnya.



Gambar 3.120. Code Untuk Menambah Data Auditor

Kemudian buat code untuk menghandle ketika user ingin menghapus data auditor yang sudah dimasukan. Sebelum data auditor benar - benar akan dihapus, lakukan konfirmasi dari user.



Gambar 3.121. Code Untuk Menghapus Data Auditor

Kemudian buat code untuk menghandle ketika user menekan tombol next ke activity selanjutnya. Sebelum ke activity selanjutnya untuk menambah audit baru, dilakukan pengecekan apakah semua field yang ada sudah dimasukan dengan benar oleh user atau belum, jika belum akan dimunculkan pesan error. Tetapi jika sudah benar semua, maka akan di arahkan ke activity data temuan audit handling.



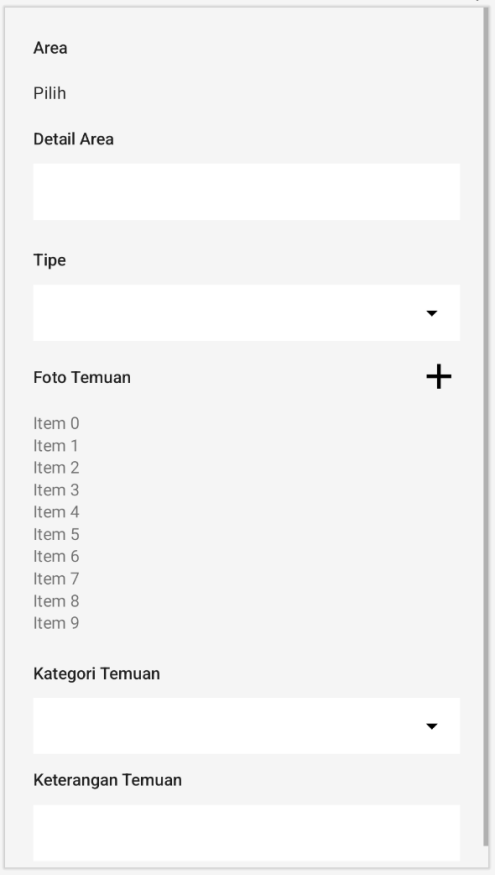
Gambar 3.122. Code Untuk Event User Menekan Tombol Next

Kemudian yang terakhir buat code di function observeViewModel(), function ini dijalankan untuk mendapatkan data tim dan auditor dari rest api dan menampilkan nya ke UI.



Gambar 3.123. Code Untuk Mendapatkan Data Time Dan Auditor

* + 1. Kemudian buat activity data temuan audit handling. Activity ini dibuat di package ui.handling.temuan dengan nama AuditDataTemuanHandling. Setelah activity dibuat, buat code untuk UI nya seperti berikut.



Gambar 3.124. UI Untuk Activity Membuat Audit Baru

Kemudian buat code untuk activitynya seperti berikut.



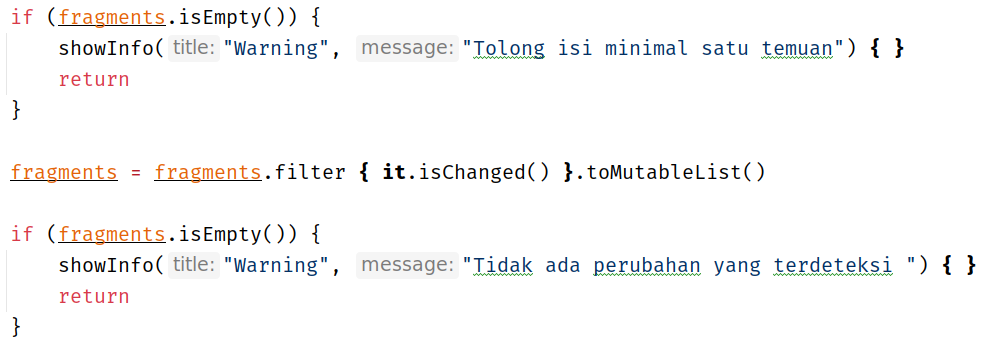
Gambar 3.125. Code Activity Audit Data Temuan Handling

Kemudian buat code di function saveTemuan(), function ini dijalankan ketika user menyimpan data temuan yang sudah dimasukan.



Gambar 3.126. Code Untuk Menyimpan Temuan Audit

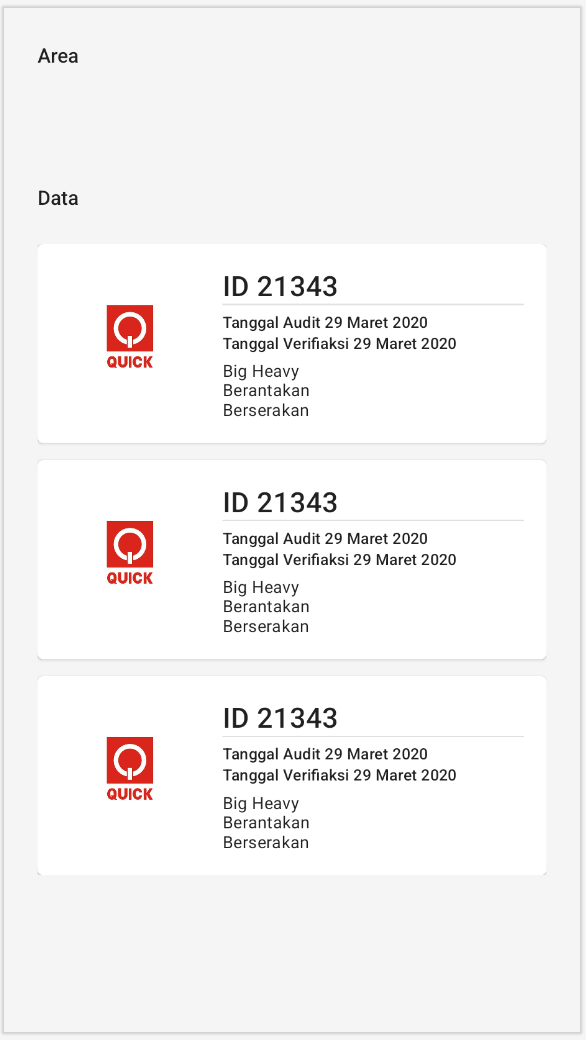
Didalam function ini juga dilakukan pengecekan data temuan yang telah dimasukan, jika data yang dimasukan tidak benar maka proses akan berhenti dan menampilkan pesan error nya.



Gambar 3.127. Validasi Sebelum Menyimpan Data Audit

* + 1. Kemudian buat activity follow up audit handling. Activity ini digunakan untuk mem follow up audit yang telah tibuat oleh audior, proses follow up ini dilakukan oleh auditee. Buat activity baru dengan nama FollowUpHandlingActivity di package ui.handling.followup.

Setelah activity terbuat, buat code untuk UI dengan nama activity\_select\_area.xml seperti berikut. File xml ini akan digunakan sebagai UI untuk semua menu kecuali audit, seperti di menu follow up, verifikasi dan monitoring.



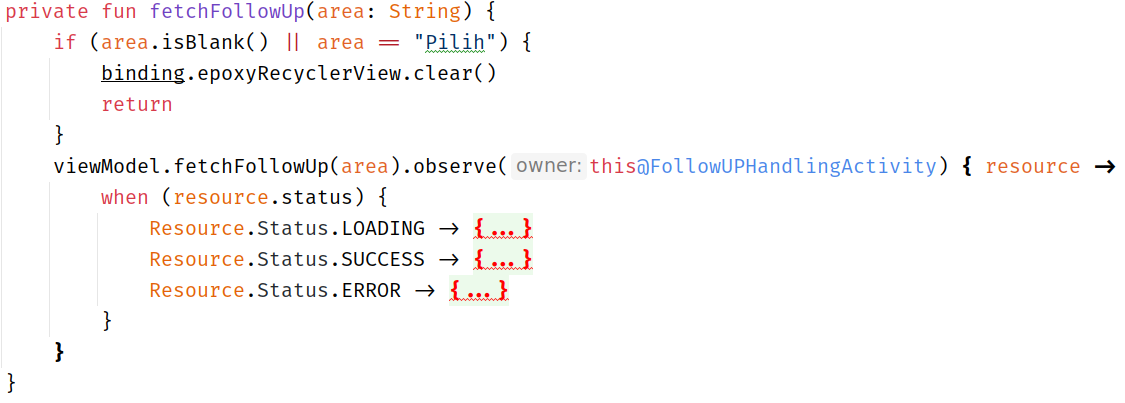
Gambar 3.128. UI Untuk Menampilkan Audit Untuk Area Tertentu

Setelah itu buat code untuk activity nya. seperti berikut.



Gambar 3.129. Code Activity Follow Up Handling

Kemudian buat code di function fetchFollowUp(), function ini dipanggil untuk mendapatkan semua data audit handling yang belum difolow up dan menampilkan nya ke UI.



Gambar 3.130. Code Untuk Mendapatkan Semua Audit Yang Bisa Difollow Up

Setelah itu buat code di function followUp(), function ini dipanggil ketika user menyimpan data follow up sesuai audit yang telah dipilih. Setelah proses penyimpanan dijalankan, akan ditampilkan pesan sesuai dengan hasil proses penyimpanannya.



Gambar 3.131. Code Untuk Follow Up Audit

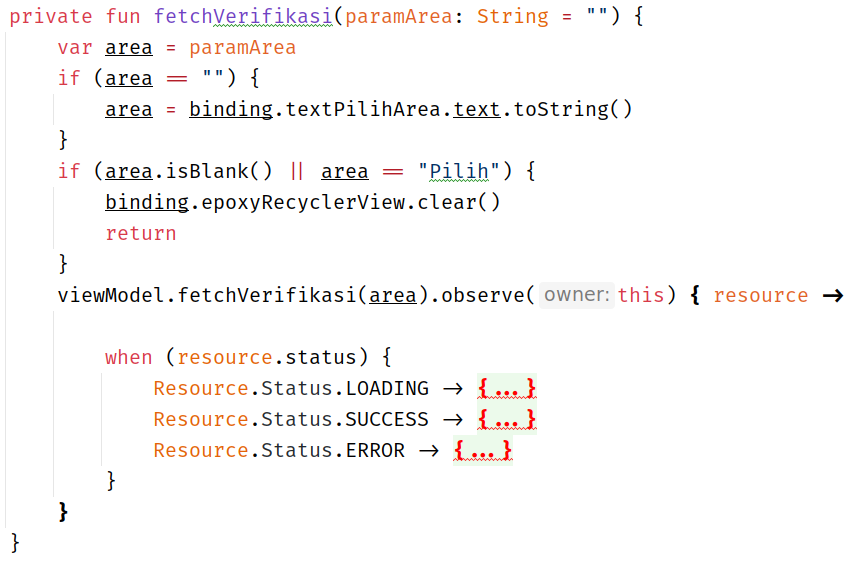
* + 1. Kemudian buat activity baru dengan nama VerifikasiHandlingActivity di package ui.handling.verifikasi. Activity ini digunakan untuk memverifikasi data audit yang telah difollow up oleh auditee, proses verifikasi ini dilakukan oleh auditor yang telah membuat audit ini.

Activity ini tetap menggunakan UI yang sama dengan activity follow up. Untuk code activity nya buat seperti berikut.



Gambar 3.132. Code Activity Verifikasi Handling

Kemudian buat code di function fetchVerifikasi(), function ini akan dipanggil untuk mendapatkan semua data audit yang bisa diverifikasi sesuai area tertentu dan menampilkannya ke UI.



Gambar 3.133. Code Untuk Mendapatkan Data Verifikasi

Kemudian buat cod di function closeAudit(), function ini dipanggil untuk menyimpan data verifikasi yang telah diputuskan oleh user apakah audit tersebut sudah bisa di close atau masih open. Sudah di close artinya data temuan audit tersebut sudah diperbaiki, namun sebaliknya jika open berarti data temuan audit tersebut belum diperbaiki dengan benar.



Gambar 3.134. Code Untuk Menutup Audit

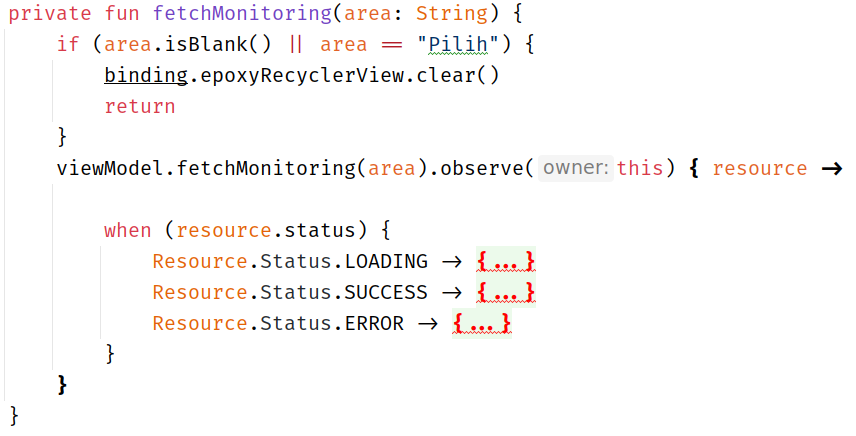
* + 1. Kemudian buat activity baru dengan nama MonitoringHandlingActivity di pacakge ui.handling.monitoring. Activity ini digunakan untuk menampilkan semua data audit yang ada di sub menu handling ini, baik yang belum difollow up sama sekali, sudah difollow up tetapi belum diverifikasi ataupun yang sudah diverifikasi dan selesai proses audit nya. Activiy ini juga masih menggunakan UI yang sama dengan activity follow up dan verifikasi.

Setelah activity nya terbuat, buat code untuk activity tersebut seperti berikut.



Gambar 3.135. Code Activity Monitoring Handling

Kemudian buat code untuk function fetchMonitoring(), function ini dipanggil untuk mendapatkan semua data audit untuk sub menu handling dan menampilkan datanya ke UI.



Gambar 3.136. Code Untuk Mendapatkan Data Monitoring Dari Server

* + 1. Setelah semua activity untuk sub menu handling sudah dibuat, ulangi langkah tersebut untuk membuat activity di submenu yang lain. Karena code untuk semua sub menu hampir mirip dan yang membedakan adalah jenis audit nya saja.
  1. Quality Control

Setelah aplikasi selesai dibuat, perlu dilakukan pengecekan pada semua fitur di aplikasi ini apakah ada bug atau pun error. Jika ada bug atau error perlu diperbaiki sampai masalah tersebut terselesaikan.

Setelah selesai laporkan ke atasan bahwa aplikasi ini sudah selesai dan kemudian atasan dan user akan mengetes aplikasi ini apakah sesuai dengan kebutuhan atau belum.

* 1. Kesimpulan

Aplikasi Audit & Verification adalah aplikasi yang digunakan untuk melakukan audit di kawasan CV. Karya Hidup Sensosa cabang Tusono. Aplikasi ini digunakan untuk menginput audit yang dilakukan oleh auditor jika ada audit yang ditemukan saat patroli. Jika ditemukan audit di suatu seksi, maka dengan menggunakan aplikasi ini untuk membuat audit baru.