**LAPORAN PRAKTIKUM PENGEMBANGAN APLIKASI BERGERAK**

**Android Fundamental 4 - WEEK 10**



Disusun oleh:

|  |  |  |
| --- | --- | --- |
| Nama | : | Alfath Roziq Widhayaka |
| Nim | : | L0122012 |
| Kelas | : | A |

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS TEKNOLOGI INFORMASI DAN SAINS DATA**

**UNIVERSITAS SEBELAS MARET**

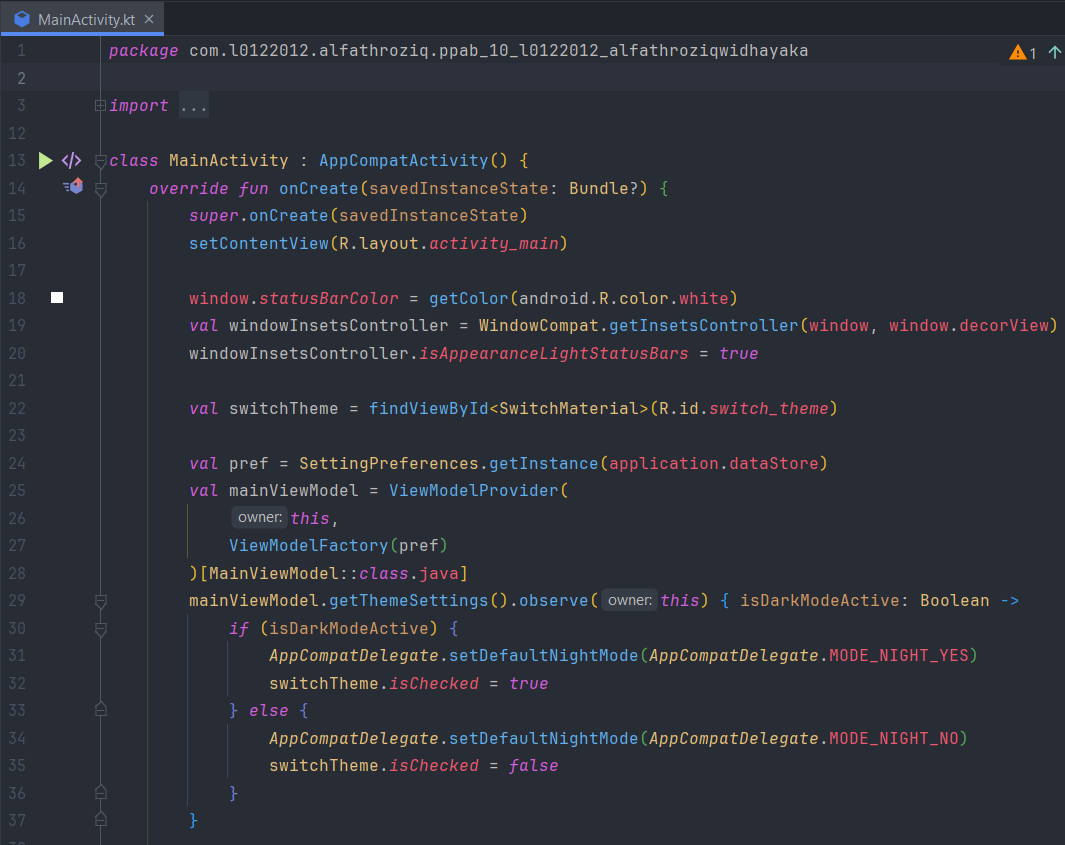
**2024**

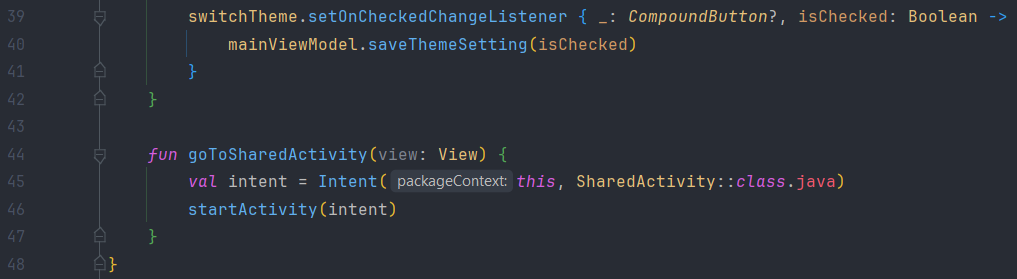
1. **Screenshot Source Code**

Berikut adalah source code yang saya gunakan pada praktikum minggu ke-10 dalam mata kuliah Praktikum Pengembangan Aplikasi Bergerak. Dalam pengerjaannya, saya menerapkan **SharedPreferences** untuk mekanisme penyimpanan foto dan data diri yang saya miliki dan **DataStore** untuk menerapkan light/dark mode pada aplikasi. Berikut penjelasan masing-masing source code-nya.

1. **Membuat Main Activity untuk membuat tampilan awal pada aplikasi dan diberi tombol light/dark mode.**

* **MainActivity.kt**

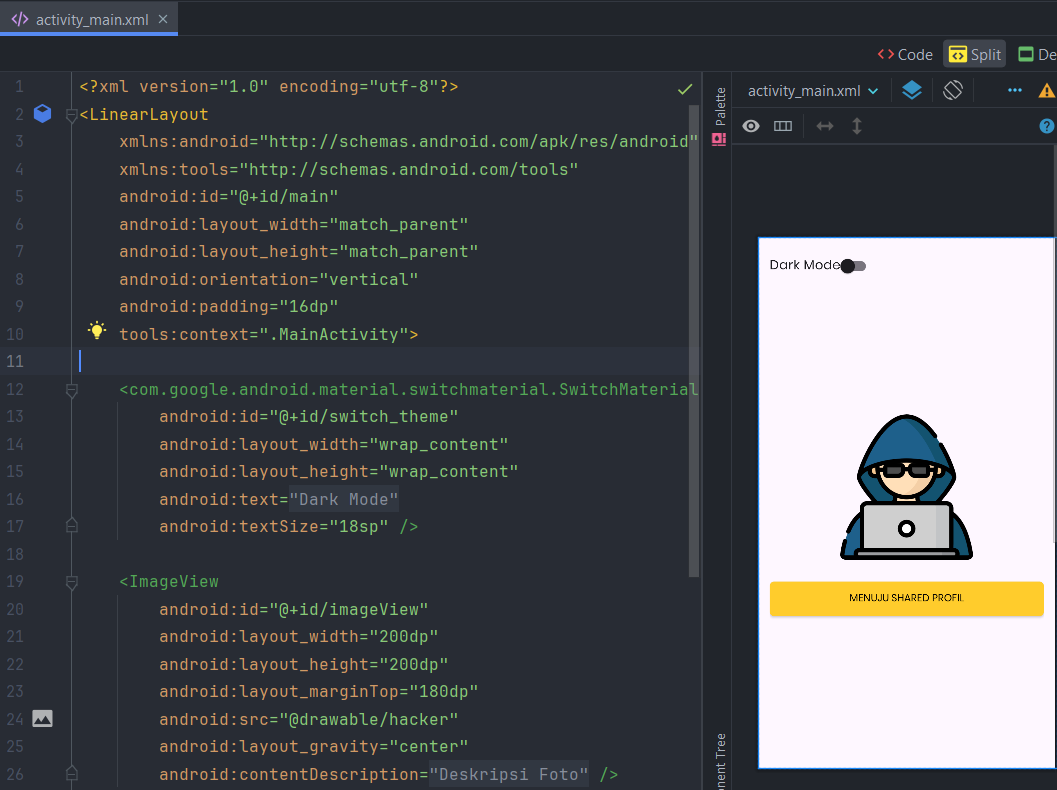
****

****

*Gambar 1 MainActivity.kt*

Source code di atas adalah MainActivity.kt untuk mengelola tema aplikasi **terang** atau **gelap** menggunakan **DataStore** untuk menyimpan preferensi pengguna. MainActivity mengatur tampilan status bar menjadi terang dan inisialisasi **SwitchMaterial** untuk mengubah tema. DataStore dikelola oleh **SettingPreferences**, sementara **MainViewModel** menangani logika pengaturan tema. Pengaturan terhadap **getThemeSettings** menentukan apakah mode gelap aktif dan menyesuaikan **AppCompatDelegate** serta status saklar. Ketika tombol switch ditekan, **saveThemeSetting** di **MainViewModel** memperbarui preferensi tema di DataStore. Metode **goToSharedActivity** mengarahkan pengguna ke **SharedActivity**.

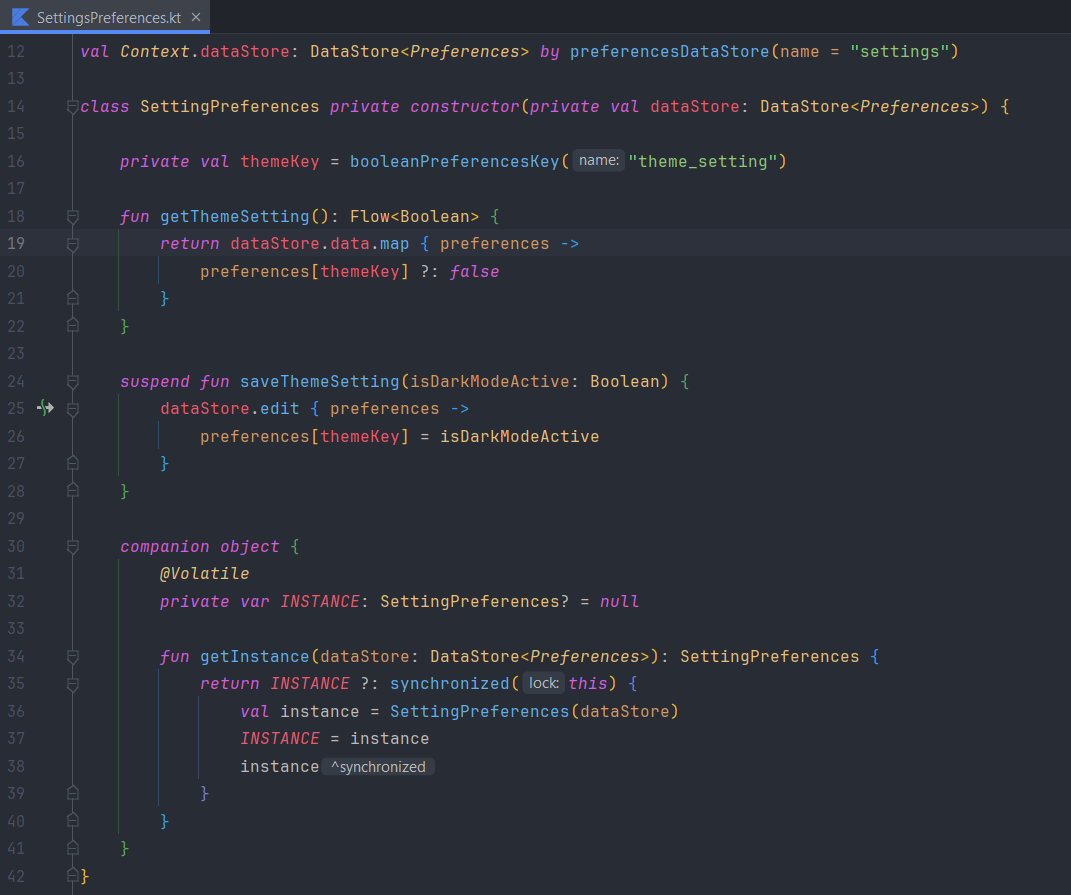
* **activity\_main.xml**

****

*Gambar 2 activity\_main.xml*

activity\_main.xml di atas merupakan layout vertikal untuk **MainActivity** menggunakan LinearLayout. Layout ini mencakup tiga elemen utama yaitu **SwitchMaterial** untuk mengubah tema aplikasi, **ImageView** yang menampilkan gambar di tengah dengan margin atas 180dp, dan **AppCompatButton** yang menggunakan drawable khusus sebagai background serta memanggil metode **goToSharedActivity** saat diklik.

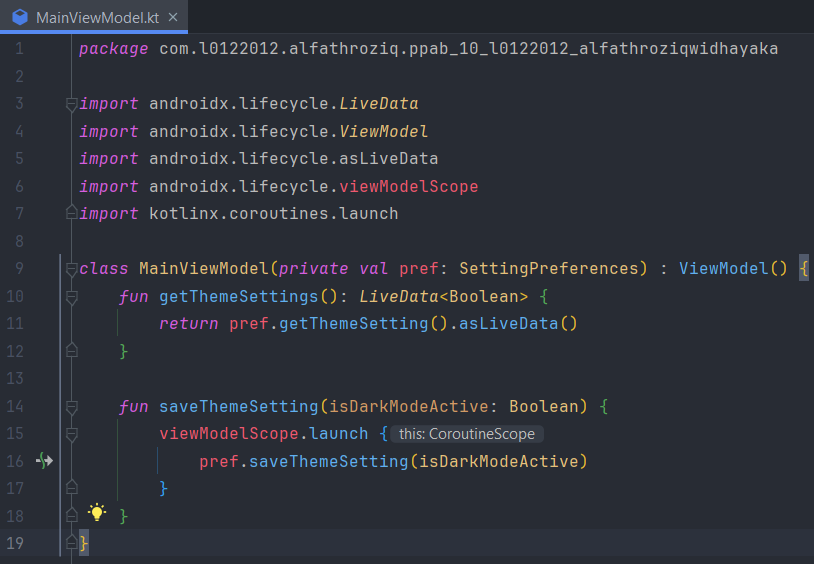
* **SettingPreferences.kt**

****

*Gambar 3 SettingPreferences.kt*

Kode di atas merupakan kelas **SettingPreferences** yang menggunakan **DataStore** untuk menyimpan preferensi tema aplikasi. Ini menyediakan metode untuk mengambil dan menyimpan preferensi tema dalam bentuk boolean. Metode **getInstance** memastikan hanya ada satu instance dari kelas ini, sementara metode **getThemeSetting** mengambil status tema saat ini dan **saveThemeSetting** menyimpan perubahan preferensi. Dengan demikian, kelas ini menyediakan antarmuka sederhana untuk mengelola preferensi tema aplikasi melalui DataStore.

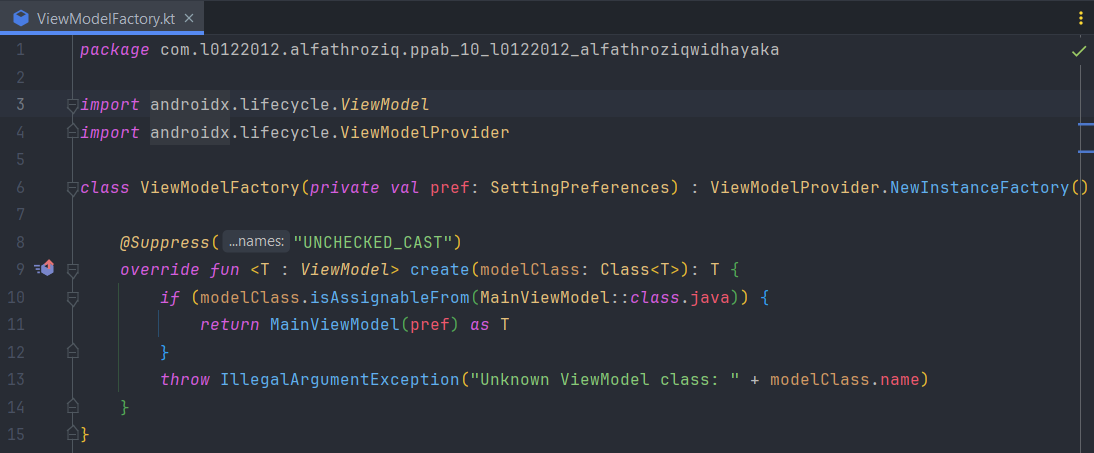
* **MainViewModel.kt**

****

*Gambar 4 MainViewModel.kt*

Kode MainViewModel.kt di atas adalah kelas **MainViewModel** yang menghubungkan antarmuka pengguna dengan logika pengaturan tema menggunakan **DataStore**. Metode **getThemeSettings** mengambil status tema sebagai **LiveData**, sedangkan **saveThemeSetting** memungkinkan UI untuk memperbarui preferensi tema secara asinkron melalui **SettingPreferences**. Dengan demikian, **MainViewModel** menyederhanakan interaksi antara UI dan penyimpanan preferensi tema ketika aplikasi ditutup lalu dibuka lagi maka tema akan tetap disimpan.

* **ViewModelFactory.kt**

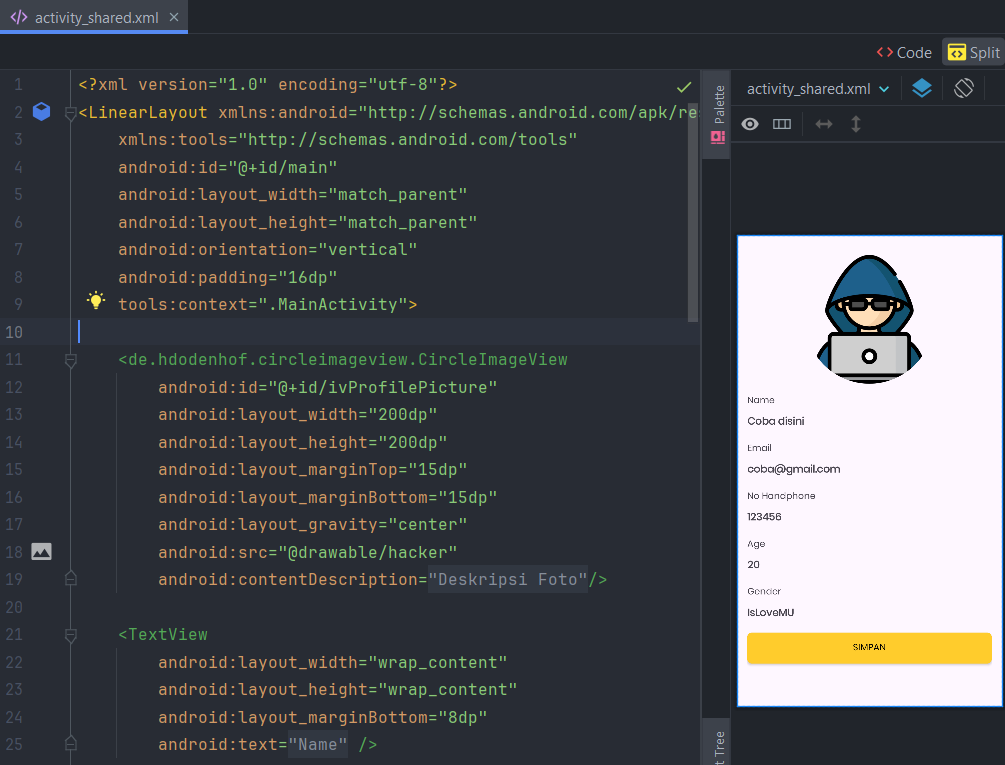
****

*Gambar 5 ViewModelFactory.kt*

Kode di atas merupakan kelas **ViewModelFactory** yang bertanggung jawab untuk membuat instance dari **MainViewModel** dengan menyediakan dependensi yang diperlukan. Metode create mengimplementasikan logika untuk membuat instance MainViewModel dengan memeriksa tipe model yang diminta. Jika tipe model yang diminta adalah MainViewModel, maka instance baru dari MainViewModel akan dibuat dengan menyediakan SettingPreferences yang diperlukan. Jika tipe model yang diminta tidak dikenali, akan dilemparkan **IllegalArgumentException**. Dengan demikian, ViewModelFactory memastikan bahwa dependensi diberikan dengan benar saat membuat instance MainViewModel.

1. **Membuat SharedActivity dan FormUserPreferenceActivity untuk membuat tampilan menggunakan mekanisme SharedPreferences dengan formulir.**

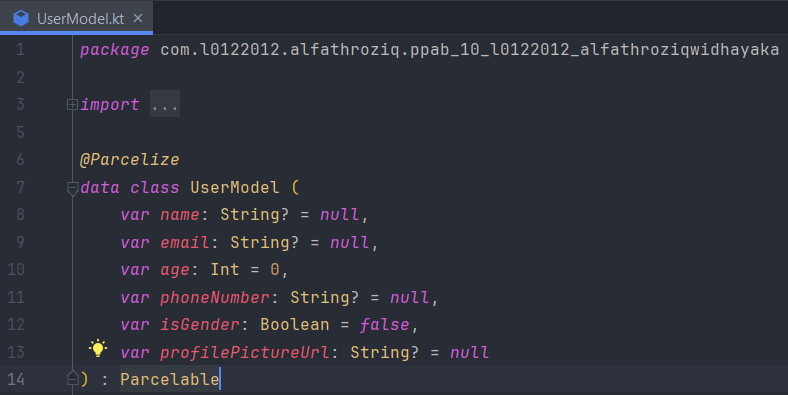
* **activity\_shared.xml**

****

*Gambar 6 activity\_shared.xml*

Kode XML diatas merupakan antarmuka pada **SharedActivity.kt**. Kode ini menggunakan LinearLayout sebagai wadah utama dengan beberapa elemen UI seperti ImageView untuk gambar profil, TextView untuk menampilkan informasi pengguna, dan sebuah tombol untuk menyimpan data. **SharedPreferences** disini digunakan untuk menyimpan data pengguna saat tombol "Save" ditekan. Ini memungkinkan pengguna untuk melanjutkan sesi mereka dengan data yang disimpan sebelumnya setelah aplikasi ditutup dan dibuka kembali.

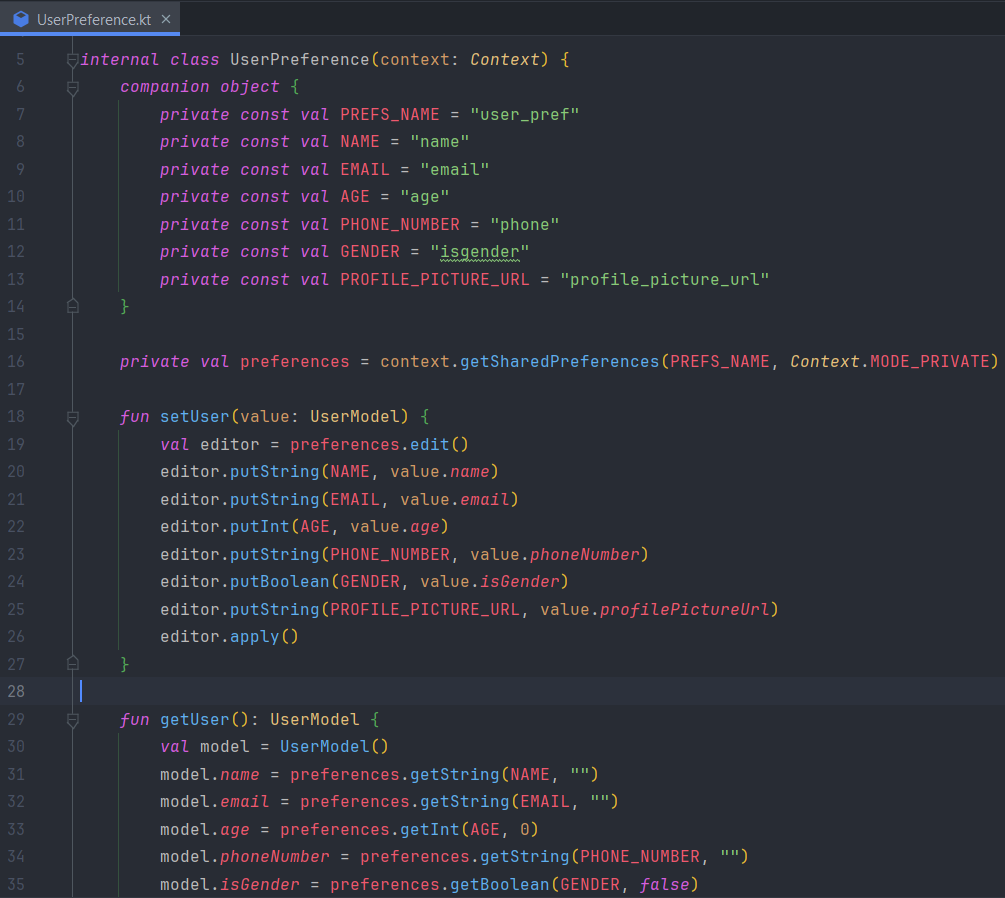
* **UserModel.kt**

****

*Gambar 7 UserModel.kt*

Kode di atas adalah definisi kelas data Kotlin **UserModel** yang menyimpan informasi pengguna seperti nama, email, usia, nomor telepon, gender, dan URL gambar profil. Dengan mengimplementasikan Parcelable, kelas ini dapat digunakan untuk mentransfer objek antar komponen aplikasi. Meskipun tidak langsung terkait dengan **SharedPreferences**, UserModel dapat bekerja dengan SharedPreferences dengan mengonversinya menjadi string JSON untuk disimpan dan dipulihkan dari SharedPreferences, memungkinkan pengelolaan data pengguna yang efisien dan pemeliharaan keadaan antara sesi aplikasi.

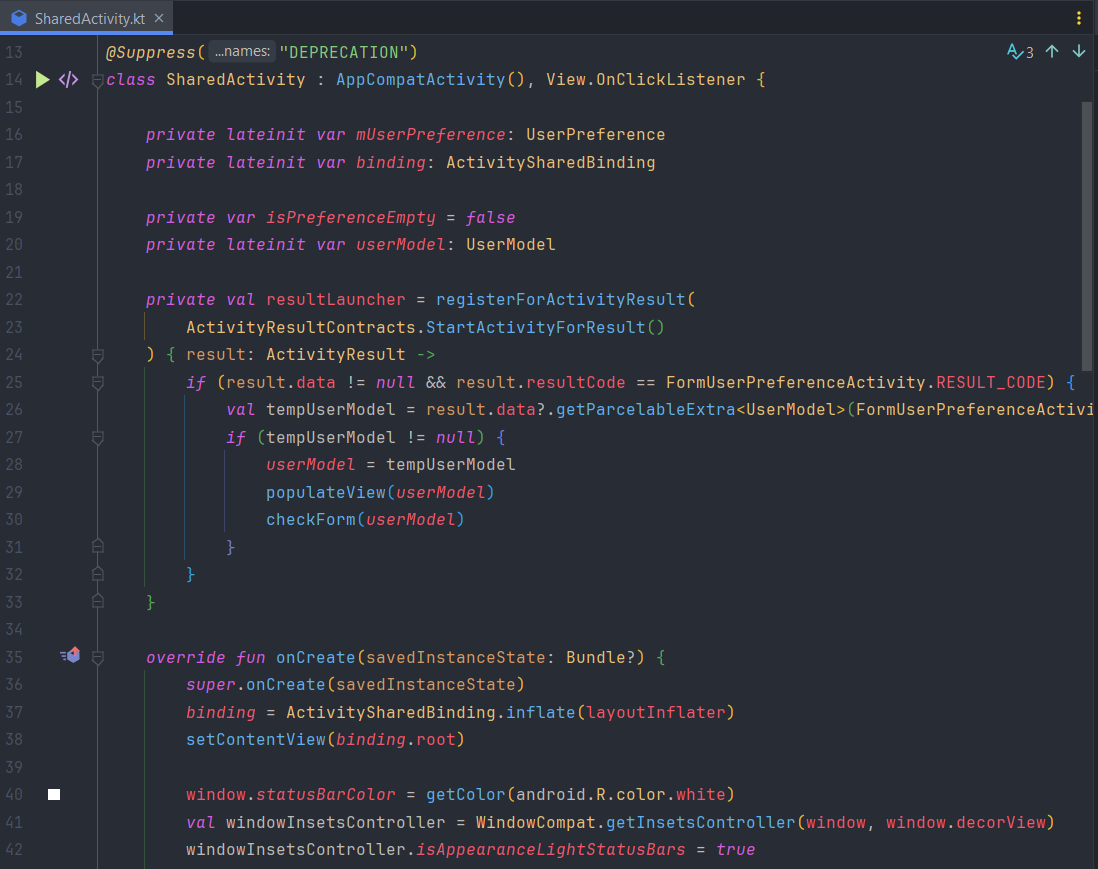
* **UserPreference.kt**

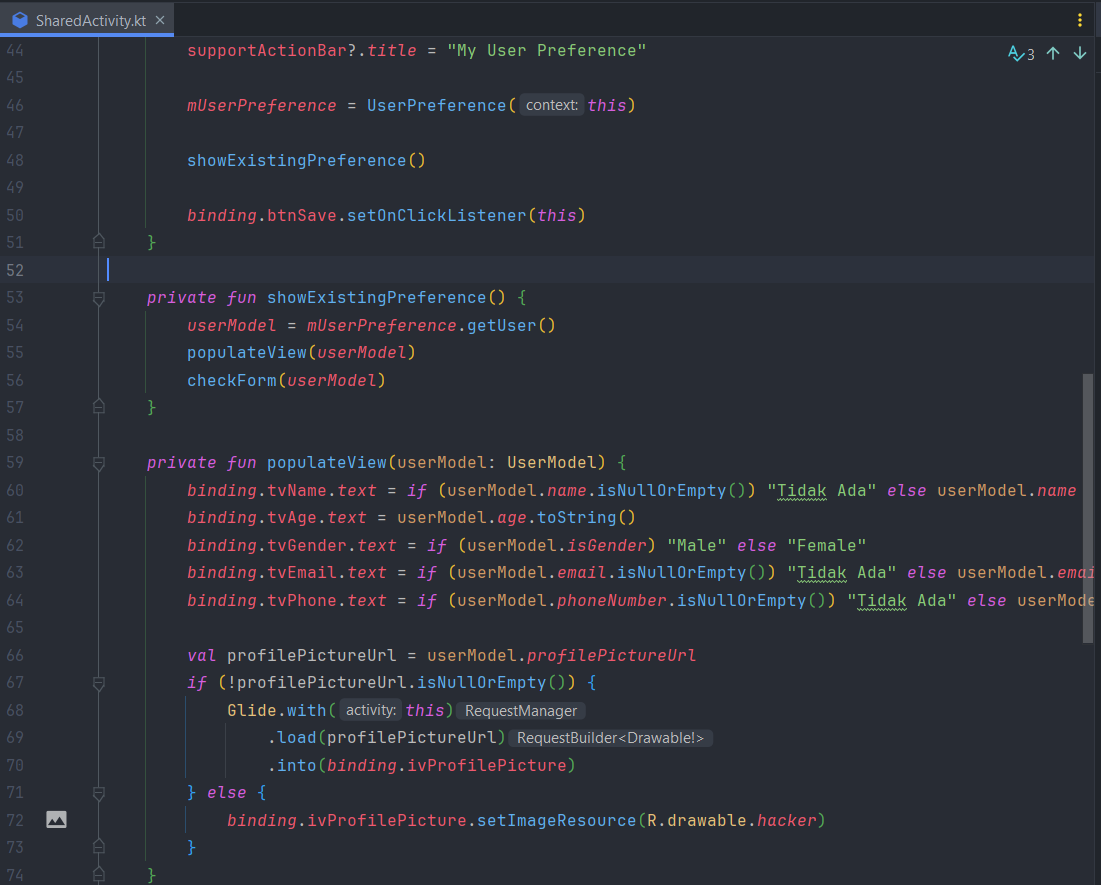
****

*Gambar 8 UserPreference.kt*

Kode di atas adalah kelas **UserPreference** yang mengelola **SharedPreferences** untuk data pengguna. Metode **setUser()** digunakan untuk menyimpan data pengguna ke SharedPreferences, sedangkan metode **getUser()** digunakan untuk mengambilnya kembali. Properti yang disimpan dan diambil termasuk nama, email, usia, nomor telepon, gender, dan URL gambar profil. Dengan demikian, kelas ini menyediakan mekanisme yang efisien untuk menyimpan dan mengambil data pengguna dalam aplikasi menggunakan SharedPreferences.

* **SharedActivity.kt**

****

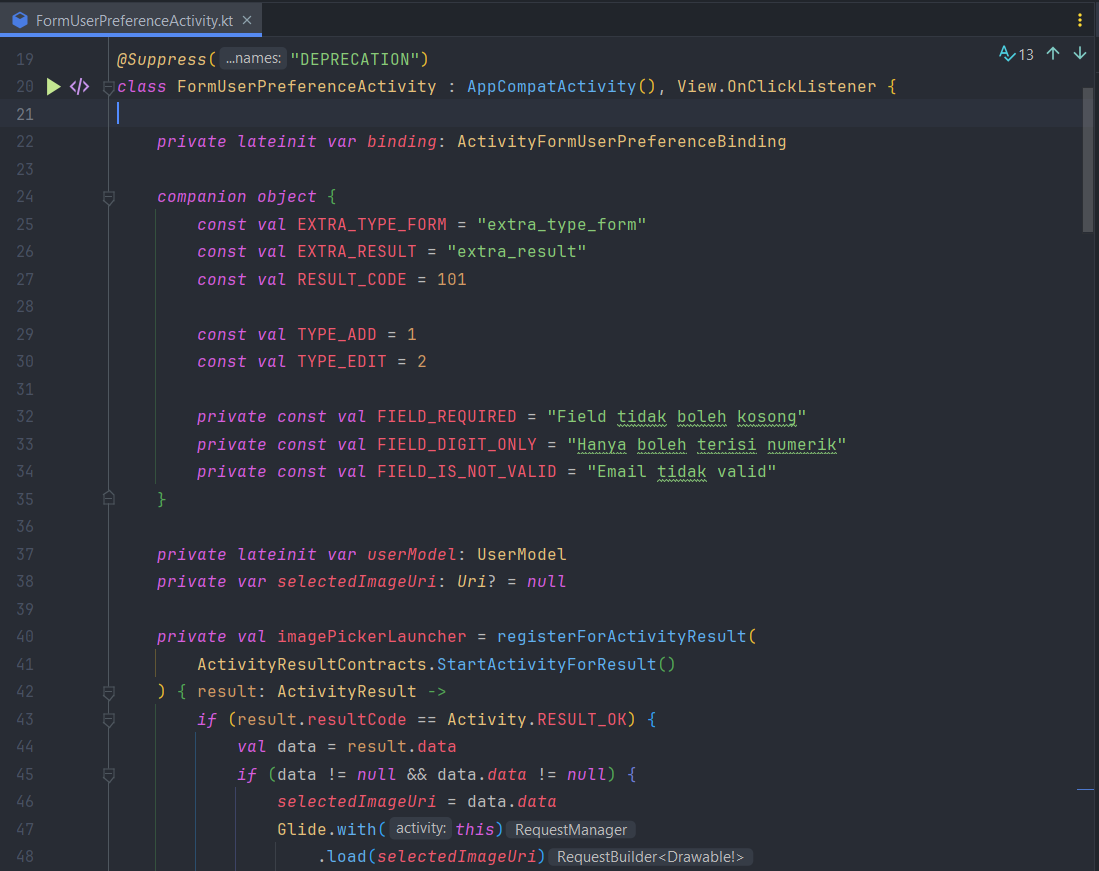
****

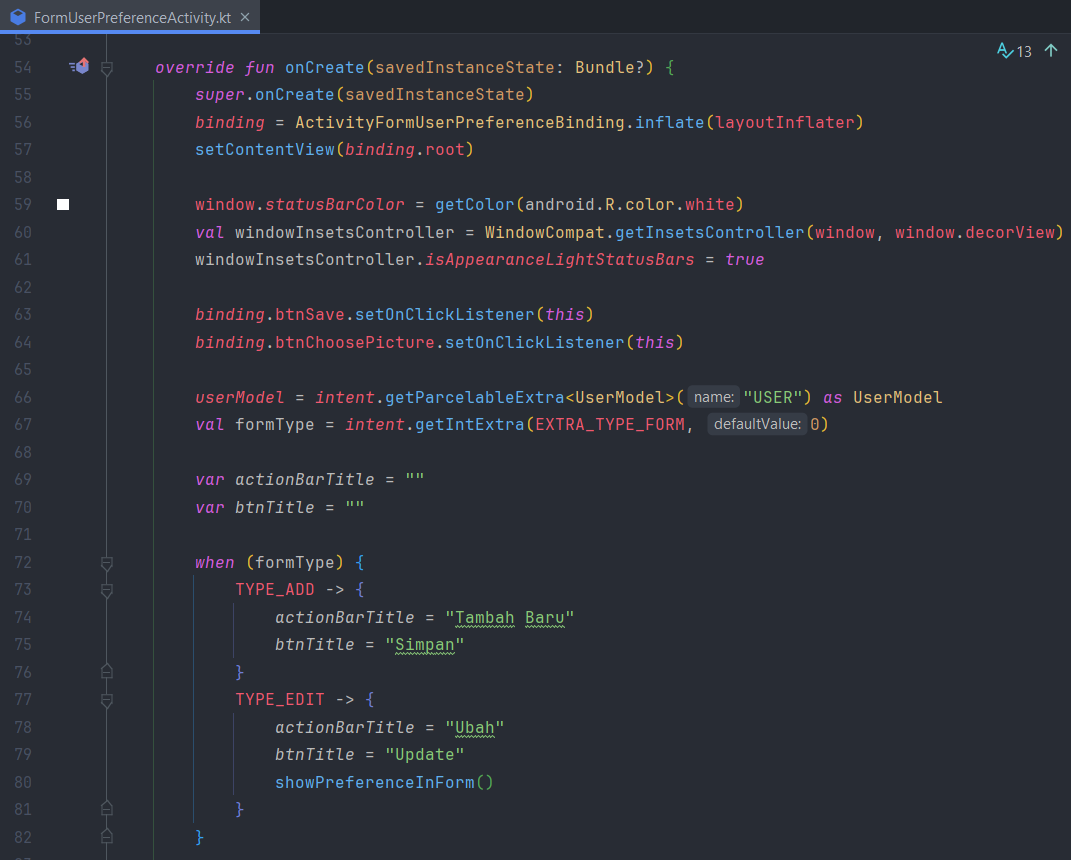
****

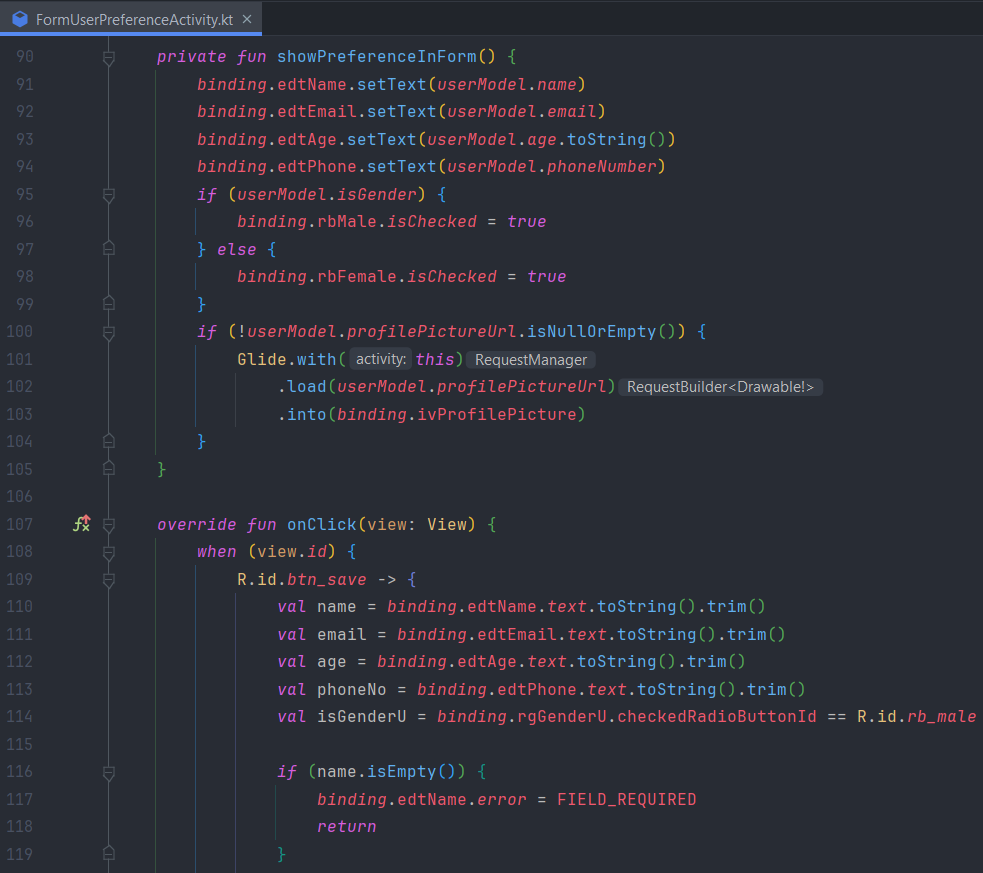
*Gambar 9 SharedActivity.kt*

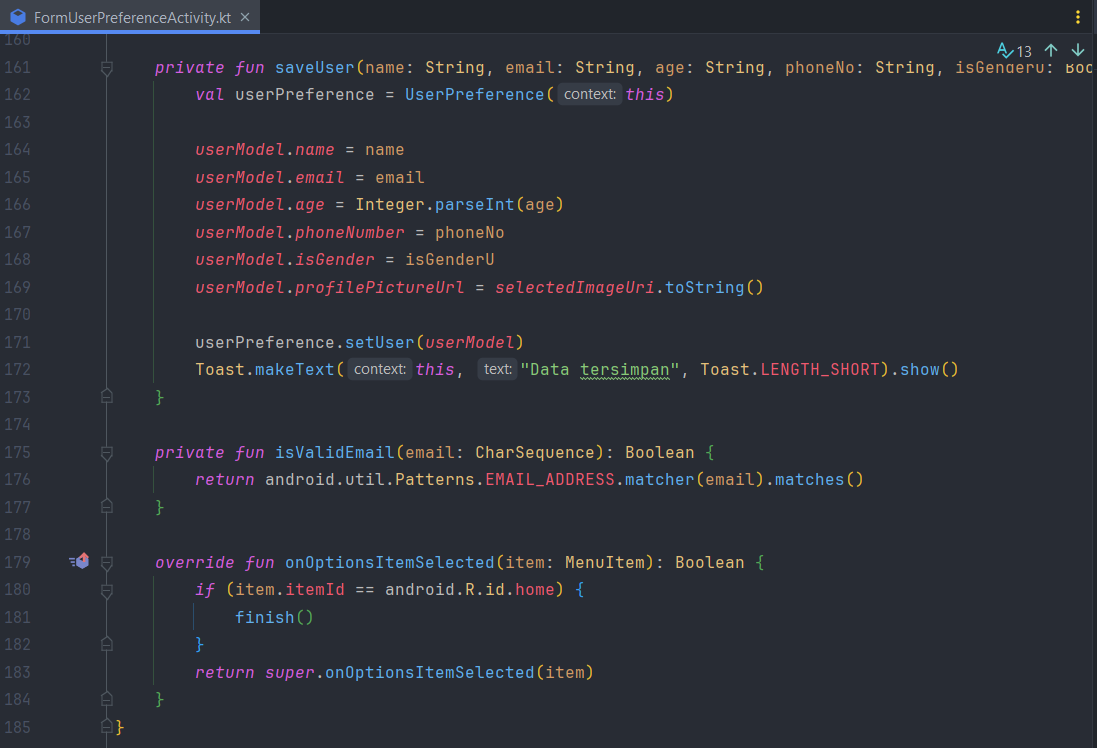
Kode di atas adalah sebuah aktivitas **SharedActivity** yang bertanggung jawab untuk menampilkan data pengguna yang disimpan dalam **SharedPreferences**, memungkinkan pengguna untuk mengubah atau mengisi data tersebut, dan kemudian menampilkan kembali hasilnya. Ketika aktivitas dibuat, metode **onCreate()** dipanggil untuk menyiapkan tampilan dan memuat data pengguna dari SharedPreferences menggunakan kelas **UserPreference**. Selain itu, aktivitas ini juga mengimplementasikan fungsi untuk menangani hasil dari aktivitas tambahan untuk mengubah data pengguna. Data pengguna ditampilkan dalam tampilan, dan pengguna dapat mengubahnya dengan menekan tombol "**Save**", yang akan membuka aktivitas tambahan yaitu **FormUserPreferenceActivity**. Setelah pengguna selesai mengubah atau mengisi data, hasilnya akan diperbarui dalam SharedPreferences dan ditampilkan kembali dalam SharedActivity. Dengan cara ini, aktivitas ini memberikan antarmuka pengguna untuk mengelola data pengguna dengan cara mengubah atau mengisi data.

* **FormUserPreferenceActivity.kt**

****

****

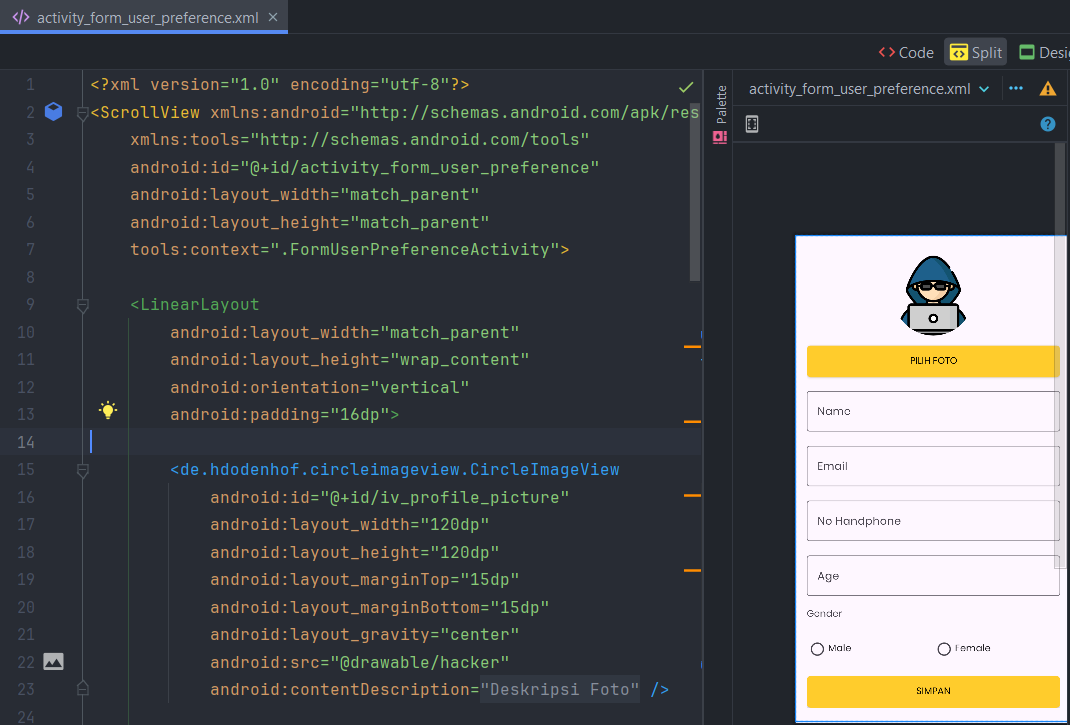
****

****

*Gambar 10 FormUserPreferenceActivity.kt*

Kode di atas adalah aktivitas **FormUserPreferenceActivity** yang bertanggung jawab untuk menampilkan formulir mengisi atau ubah data pengguna. Ketika aktivitas dibuat, formulir disesuaikan berdasarkan tipe yang diterima dari intent. Data pengguna dimuat jika aktivitas bertujuan untuk mengubah data yang ada. Pengguna dapat memilih gambar apapun menggunakan galeri, dan formulir akan memvalidasi input pengguna sebelum menyimpannya. Setelah disimpan, data pengguna diperbarui dalam **SharedPreferences** menggunakan kelas **UserPreference**. Aktivitas ini berfungsi sebagai antarmuka pengguna untuk mengelola data pengguna, dengan **SharedPreferences** digunakan sebagai media penyimpanannya.

* **activity\_form\_user\_preference.xml**

****

*Gambar 11 activity\_form\_user\_preference.xml*

Kode XML di atas adalah tata letak untuk tampilan FormUserPreferenceActivity.kt dalam aplikasi Android. Dengan ScrollView untuk memungkinkan pengguna menggulir konten jika diperlukan, elemen-elemen seperti foto profil, input teks untuk nama, email, nomor telepon, dan usia, serta pilihan "male" atau "female" untuk pertanyaan gender, semuanya disusun dalam LinearLayout. Implementasi penyimpanan preferensi menggunakan SharedPreferences akan memungkinkan aplikasi menyimpan dan mengambil nilai-nilai ini secara persisten di perangkat pengguna, sehingga data dapat diakses kembali setiap kali aplikasi dibuka.

1. **Screenshot Emulator**

Berikut adalah hasil pada emulator ketika aplikasi dijalankan atau dieksekusi. Aplikasi ini menerapkan materi penyimpanan dengan mekanisme **SharedPreferences** untuk mekanisme penyimpanan foto dan data diri yang saya miliki dan **DataStore** untuk menerapkan light/dark mode pada aplikasi. Berikut merupakan hasil screenshoot pada aplikasi.

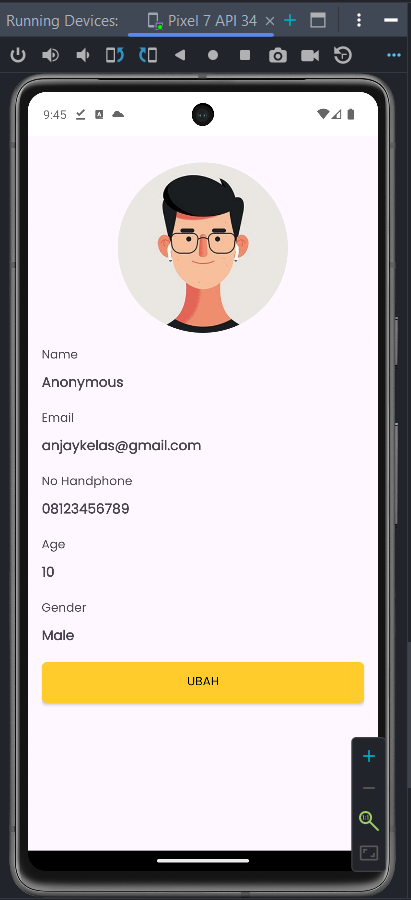
1. **Tampilan Awal**

****

*Gambar 12 tampilan awal*

Gambar diatas menampilkan hasil dari emulator setelah aplikasi dijalankan. Tampilan ini menampilkan tampilan awal aplikasi yang didalamnya terdapat tombol switch untuk mengubah tema light/dark mode dan terdapat tombol ”Menuju Shared Profil”. Pada tampilan ini menerapkan materi penyimpanan **DataStore** yang diterapkan pada switch tema light/dark mode. DataStore ini berguna untuk mengubah warna tema yang diterapkan dan akan terus menyimpan tema nya ketika user keluar dan masuk kembali aplikasi tersebut.

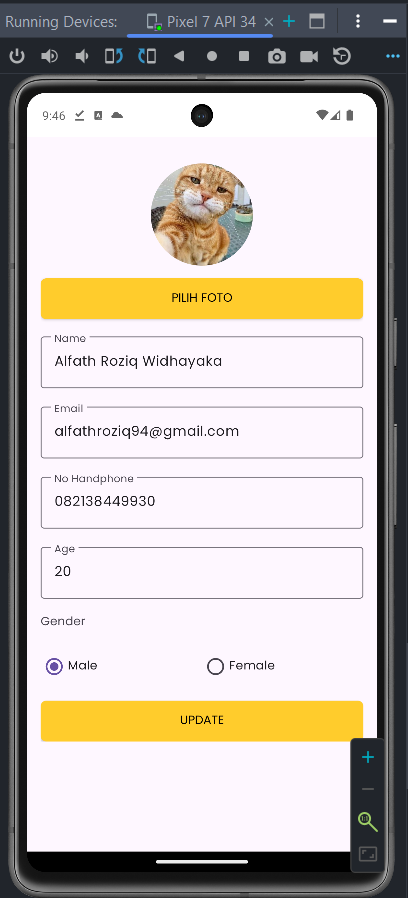
1. **Tampilan Shared Activity (Tampilan sebelum data diisi atau diubah)**

****

*Gambar 13 tampilan shared activity sebelum diubah*

Gambar diatas merupakan gambar emulator ketika user telah menekan tombol ”Menuju Shared Profil”. Pada tampilan ini (SharedActivity) merupakan tampilan untuk menampilkan data pengguna yang belum diubah/diisi oleh user.

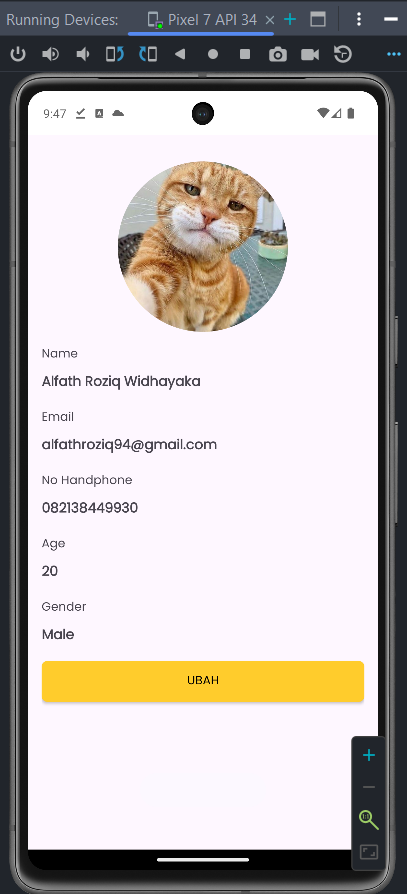
1. **Tampilan FormUserPreference (Pengisian Formulir untuk mengubah atau mengisi data)**

****

*Gambar 14 tampilan mengisi atau mengubah data pada form*

Gambar diatas merupakan gambar ketika user menekan tombol ”Ubah” pada tampilan sebelumnya. Pada tampilan ini, user dapat mengubah atau mengisi data yang diinginkan user melalui form yang harus diisi. User juga dapat mengubah foto yang dipilih melalui galeri user. Jika sudah selesai tekan tombol ”Update” untuk kembali ke tampilan sebelumnya. Materi yang diterapkan pada tampilan ini adalah mekanisme penyimpanan **SharedPreferences** dalam menjalankan formulir aplikasi yang dapat menyimpan data tersebut.

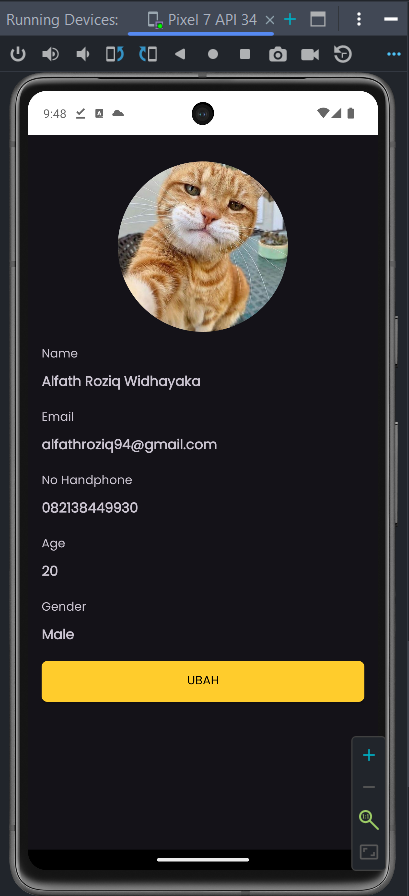
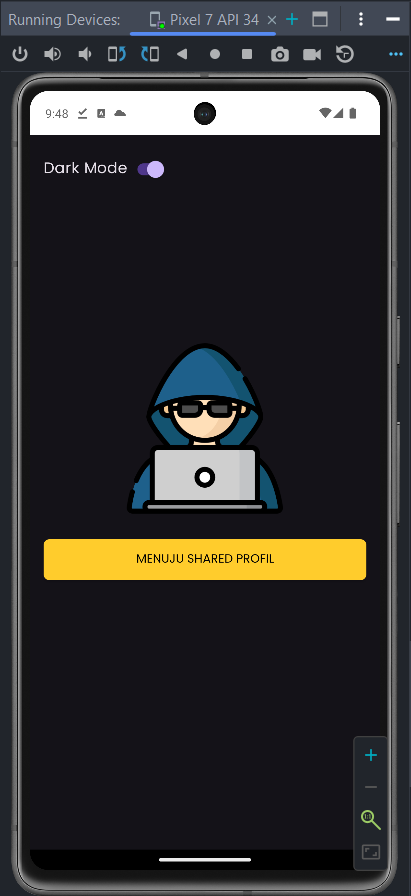
1. **Tampilan Shared Activity (Tampilan setelah data diubah)**

****

*Gambar 15 tampilan shared activity ketika tampilan data sudah diubah*

Gambar diatas merupakan tampilan emulator ketika data user telah diubah atau diisi oleh user sendiri. Disini user telah mengganti gambar dan mengganti data diri.

1. **Tampilan setelah aplikasi diubah ke mode dark mode menggunakan switch**

****

*Gambar 15 tampilan aplikasi ketika diubah menjadi dark mode menggunakan switch*

Gambar diatas merupakan tampilan ketika user kembali ke tampilan awal aplikasi dan user ingin mengubah nya menjadi tema dark mode. Ketika aplikasi diubah menjadi dark mode melalui switch yang ada di tampilan awal. Maka keseluruhan aplikasi di dalamnya ini akan berubah bertema gelap semua. Disini Aplikasi menerapkan **DataStore** dan **SharedPreferences** agar aplikasi ini menyimpan data user tema gelap ini dan data-data diri didalamnya, yaitu ketika user keluar dari aplikasi dan kembali masuk aplikasinya maka aplikasi akan tetap bertema gelap dan menyimpan data-data sebelumnya.

1. **Kesimpulan**

Pada praktikum ini, aplikasi yang saya buat mengimplementasikan mekanisme penyimpanan data menggunakan **SharedPreferences** dan **DataStore** dalam aplikasi Android. Penggunaan SharedPreferences memungkinkan penyimpanan data pengguna seperti nama, email, usia, nomor telepon, gender, dan URL gambar profil. Mekanisme ini memungkinkan data pengguna untuk tetap tersimpan dan dapat diakses kembali setelah aplikasi ditutup dan dibuka kembali. Dengan menggunakan kelas **UserPreference** dan **UserModel**, saya bisa mengelola data pengguna secara efisien dan menampilkan informasi tersebut pada antarmuka aplikasi melalui **SharedActivity** dan **FormUserPreferenceActivity**.

Selain itu, saya juga menerapkan **DataStore** untuk mengelola preferensi tema aplikasi (light/dark mode). Dengan menggunakan kelas **SettingPreferences dan MainViewModel**, aplikasi dapat menyimpan preferensi tema yang dipilih pengguna, sehingga ketika aplikasi ditutup dan dibuka kembali, tema yang dipilih akan tetap tersimpan dan diterapkan. Implementasi ini menunjukkan penggunaan arsitektur **ViewModel** untuk memisahkan logika bisnis dari antarmuka pengguna, serta memastikan bahwa preferensi pengguna disimpan dengan baik. Praktikum ini menunjukkan pentingnya pengelolaan data dan preferensi pengguna untuk meningkatkan pengalaman pengguna dalam aplikasi Android.