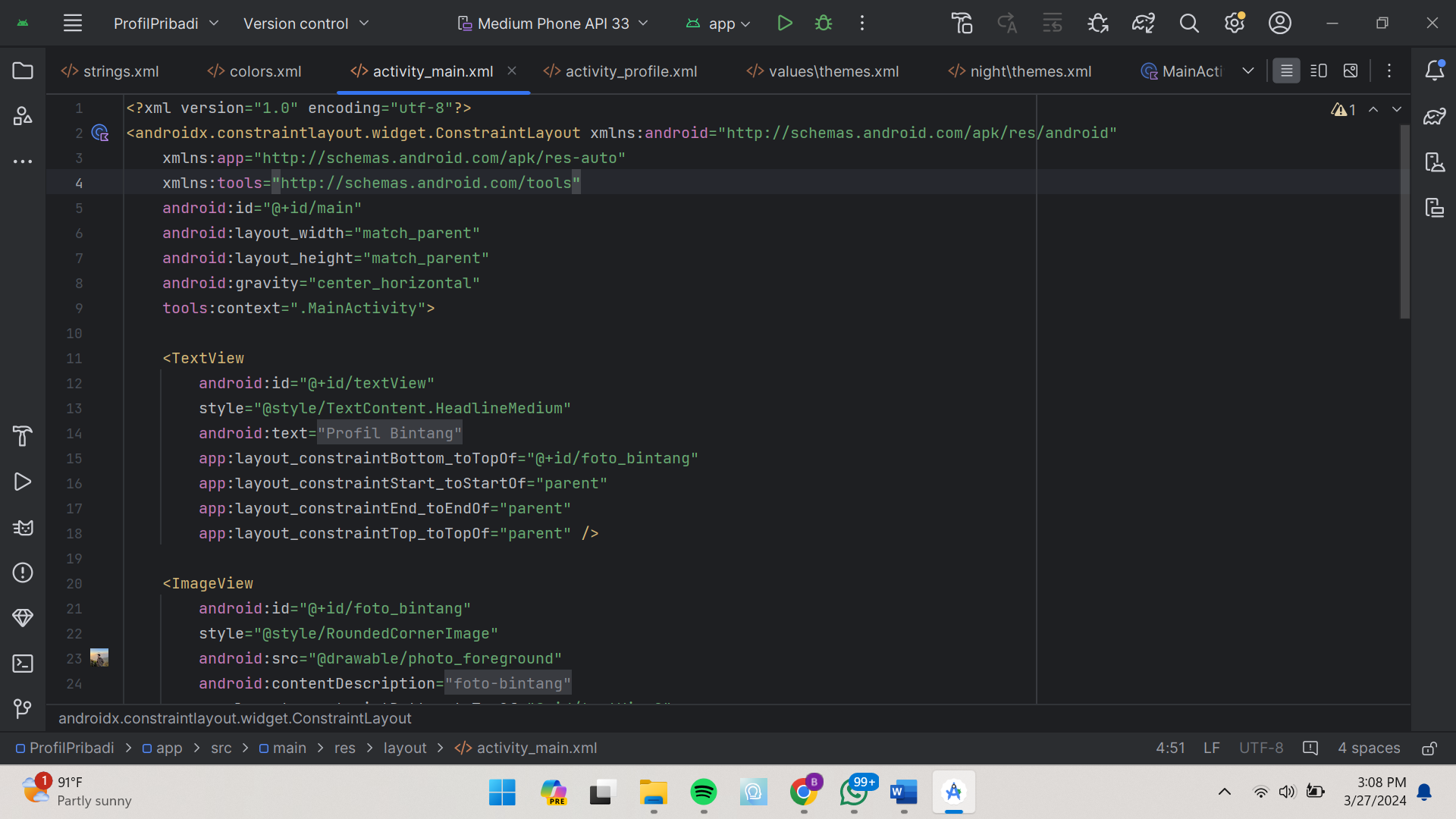
|  |
| --- |
| Nama : Bintang Harida Ramadhan  NIM : L0122034  Kelas : A |

**TUGAS PRAKTIKUM WEEK 4**

1. Screenshot Sourcecode
2. File “activity\_main.xml”



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. File “MainActivity.kt”

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

1. File “activity\_profile.xml

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated

1. File “ProfileActivity.kt”

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated

1. Penjelasan Implementasi Materi
2. Constraint Layout

Untuk menentukan posisi tampilan pada Constraint Layout, *view* harus memiliki minimal satu *constraint* (batasan) horizontal dan satu *constraint* vertikal untuk tampilan tersebut. Setiap *constraint* mewakili koneksi atau penyesuaian ke tampilan lain, *parent layout*, atau *invisible guideline*. Setiap *constraint* menentukan posisi tampilan sepanjang sumbu vertikal atau horizontal.

Pada bagian *Main Activity* dari aplikasi yang saya buat, *Headline* yang bertuliskan “Profil Bintang” memiliki *constraint bottom* dengan id “foto-bintang” yang mana itu adalah *ImageView* di bawahnya. Ini berarti bahwa posisi *TextView* tersebut memiliki batasan dengan *ImageView,* posisi *TextView* terhubung dan menyesuaikan dengan *ImageView* dibawahnya. Selain itu, *TextView* ini juga memiliki *constraint top, constraint start,* dan *constraint end* ke parent. Ini berarti bahwa *TextView* tersebut memiliki koneksi dengan parent-nya, yaitu constraint layout itu sendiri.

Untuk komponen *view* yang lain kurang lebih sama seperti *Headline* yang dijelaskan sebelumnya. Namun, berbeda di *constriaint top* dan *constraint bottom-*nya yang harus disesuaikan dengan id dari komponen *view* yang diinginkan susunannya (berada di atas atau berada di bawah dari komponen tersebut). Untuk *constraint start* dan *constraint end* dari seluruh komponen disini dihubungkan dengan parent, karena saya menginginkan setiap komponen berjajar secara vertikal dan berada di tengah tengah.

1. Style

Pada pengembangan aplikasi android, Style dan Theme digunakan untuk mengubah tampilan dari sebuah aplikasi sesuai dengan keinginan programmer. Dengan deklarasi satu style saja, style tersebut dapat digunakan di beberapa *view* yang membutuhkan sekaligus. Karena itu, *Style* dan *Theme* dapat mengurangi duplikasi kode yang tidak perlu dan membuat aplikasi kita lebih ringan dan responsif untuk pengguna.

Pada tugas kali ini, saya membuat beberapa style yang akan digunakan untuk headline, title, dan body dari *TextView*. Pada style headline, saya menggunakan *textAppearance* dari *Material3.HeadlineMedium* yang artinya pada style ini akan mewarisi sifat design yang ada pada *Material3.HeadlineMedium.* Untuk text title menggunakan *textApperance* dari *Material3.TitleMedium,* sedangkanuntuk text body menggunakan *textApperance* dari *Material3.BodyMedium.* Material Design sendiri adalah panduan desain dan sistem desain yang dikembangkan oleh Google untuk menyediakan panduan dan alat bagi para desainer dan pengembang untuk menciptakan pengalaman pengguna yang konsisten, intuitif, dan indah di berbagai platform dan perangkat.

Masing-masing dari *style* tersebut akan diberikan atribut dan value yang sesuai dengan kegunaannya. Sebagai atribut dasar yang harus dimiliki oleh setiap *view*, maka seluruh *style* dini menerapkan atribut *la*yout\_*width* dan *layout\_height.* Selain itu, setiap view juga diberikan atribut untuk memberikan *margin top* dan *margin bottom.* Untuk *text style* diberi tambahan atribut seperti *textStyle bold* pada *Headline* dan *Title,* serta *justificationMode center* dan *gravity center* pada *Text Body.* Pada style button, disini juga diterapkan atribut *textColor* dan *backgroundTint* untuk memberikan warna pada text dan background dari tombol. Selain itu, juga terdapat *style* untuk *image* yang isinya atirbut *adjustViewBounds* dan *scaleType.*

1. Screenshot User Interface

* Light Mode

A screenshot of a phone

Description automatically generated A screenshot of a phone

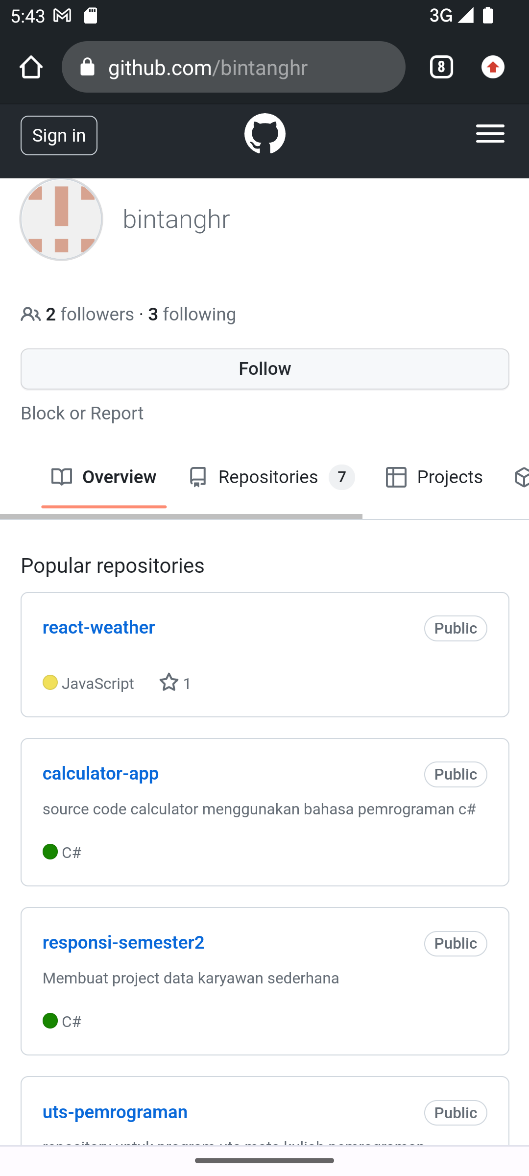
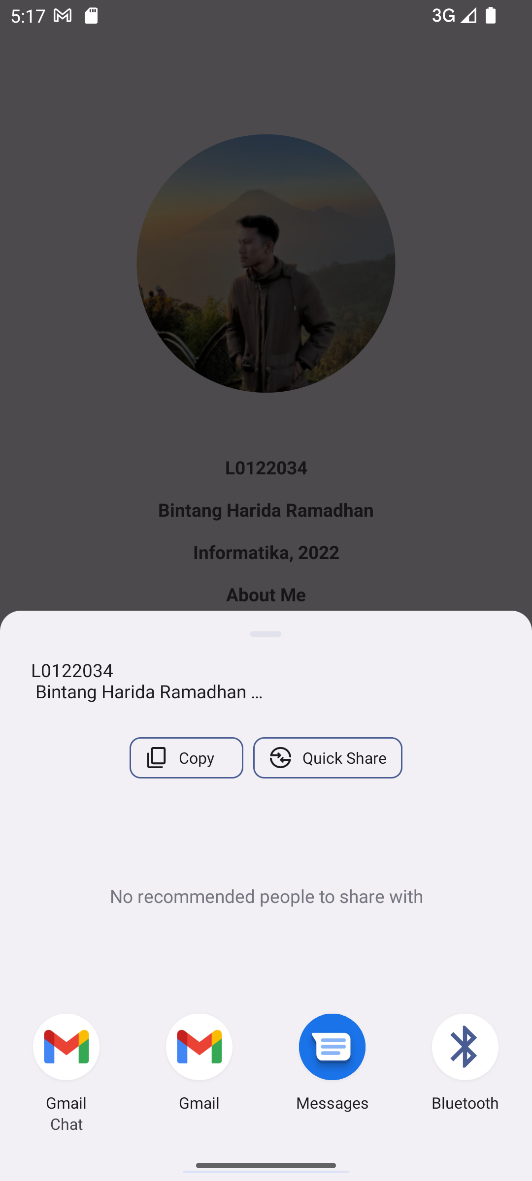
Description automatically generated

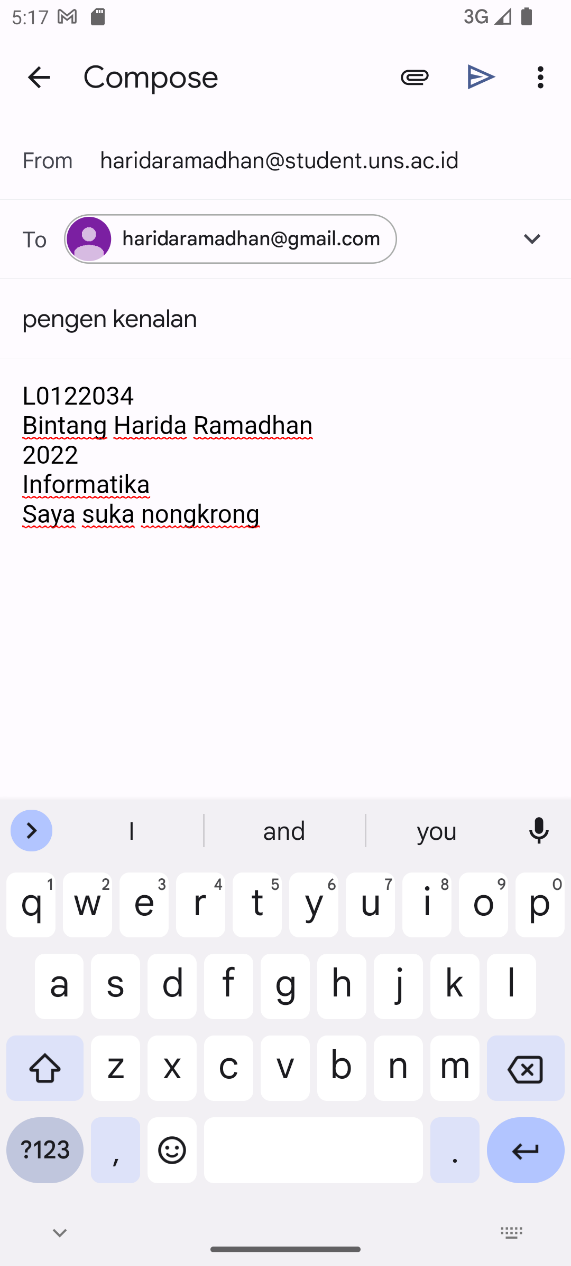
* Dark Mode

A screenshot of a phone

Description automatically generated A screenshot of a phone

Description automatically generated



1. Penjelasan User Interface

Aplikasi ini merupakan aplikasi pribadi milik saya, Bintang Harida Ramadhan. Aplikasi ini dibuat untuk menampilkan informasi pribadi saya seperti nama, nim, jurusan, angkatan, dan deskripsi diri. Pada bagian awal (Main Activity) ditampilkan judul, gambar, deskripsi singkat, tombol “Go To Profile”, dan tombol “Go To My Github”. Jika tombol “Go To Profile” ditekan maka aplikasi akan menuju ke Profile Activity. Jika tombol “Go To My Github” ditekan maka akan menelusuri link github saya melalui default browser pada perangkat.

Pada Profile Activity ditampilkan data diri saya seperti nim, nama, jurusan, tahun angkatan, deskripsi diri, foto, dan tombol “Share”. Jika tombol “Share” ditekan maka aplikasi akan menampilkan beberapa aplikasi yang dipilih untuk membagikan. Ketika user memilih menggunakan gmail, maka otomatis terisi email tujuan (email saya) beserta pesan default data diri saya.

1. Kesimpulan

*ConstraintLayout* adalah layout yang mengharuskan komponen *view* memiliki minimai satu *constraint* (batasan) secara horizontal dan satu *constraint* secara vertikal untuk dapat meletakkan komponen. Setiap *constraint* mewakili koneksi atau penyesuaian ke *view* lain atau *parent.* Dengan *ConstraintLayout*, developer dapat dengan mudah membuat tata letak komponen yang besar dan kompleks dengan hierarki tampilan yang datar (sejajar) tanpa view group yang bertumpuk.

Style adalah kombinasi beberapa atribut atau komponen yang dibutuhkan untuk mendefinisikan sebuah komponen *view*. Seperti font color, font style, padding, margin, height, width, dan background\_color. Sederhananya, ketika menerapkan *style* maka akan memindahkan atribut-atribut dari komponen untuk dituliskan pada file “themes.xml”. Dengan menggunakan *style,* developer akan lebih mudah untuk menuliskan kode, karena tidak perlu menulis kode berulang kali di beberapa kompnen yang memungkinkan untuk memiliki kesamaan atribut, cukup dengan mengubah di satu tempat saja. Untuk mengubahnya, Style berada pada file themes.xml.