**LAPORAN**

***(Pemrograman Mobile)***

**(APLIKASI MENAMPILKAN *WIDGET CHECKLIST*)**

****

|  |  |  |
| --- | --- | --- |
| **Nama** | **:** | Alvina Leony |
| **NPM** | **:** | 221510003 |
| **Dosen** | **:** | Saut Pintubipar Saragih, S.Kom., M.MSI. |

# KATA PENGANTAR

Dengan puji syukur kepada Tuhan Yang Maha Esa karena hanya dengan kasih dan karunia-Nya penulis mampu menyelesaikan proposal ini yang berjudul “**APLIKASI MENAMPILKAN *SHOPPING CHECKLIST***”. Proposal ini telah disusun untuk memenuhi salah satu syarat memperoleh kelulusan untuk mata kuliah Pemrograman Mobile pada jurusan Sistem Informasi. Selama menuliskan proposal penelitian ini, penulis telah mengalami kesulitan dan kendala, hingga penulis sadar bahwa proposal penelitian ini masih sangat jauh dari kesempurnaan. Oleh karena itu, penulis berharap akan kritik dan saran yang membangun demi kesempurnaan proposal penelitian ini.

Dengan kesempatan ini, penulis ingin menyatakan ucapan terima kasih sebesar-besarnya kepada Bapak Saut Pintubipar Saragih, S.Kom., M.MSI. selaku dosen pengajar mata kuliah ini yang telah memberikan arahan dan bimbingan kepada penulis selama proses penyelesaian proposal penelitian ini. Penulis juga ingin mengucapkan terima kasih kepada Bapak Tukino, S.Kom., M.SI. sebagai dosen pembimbing yang juga memberikan nasehat dan saran pada proses penulisan proposal penelitian ini.

Penulis sangat berharap bahwa proposal ini dapat bermanfaat dan dapat membantu para pembaca.

Batam, 9 November 2024

Alvina Leony

# **DAFTAR ISI**

[KATA PENGANTAR i](#_Toc182049961)

[DAFTAR ISI ii](#_Toc182049962)

[DAFTAR GAMBAR iii](#_Toc182049963)

[DAFTAR LAMPIRAN iv](#_Toc182049964)

[BAB I 1](#_Toc182049965)

[PENDAHULUAN 1](#_Toc182049966)

[1.1. Latar Belakang 1](#_Toc182049967)

[1.2. Rumusan Masalah 3](#_Toc182049968)

[1.3. Batasan Masalah 3](#_Toc182049969)

[1.4. Tujuan Penelitian 3](#_Toc182049970)

[1.5. Manfaat Penelitian 3](#_Toc182049971)

[BAB II 4](#_Toc182049972)

[PEMBAHASAN 4](#_Toc182049973)

[2.1. Teori Umum 4](#_Toc182049974)

[2.1.1. Aplikasi Mobile 4](#_Toc182049975)

[2.1.2. Checklist 4](#_Toc182049976)

[2.2. Teori Khusus 4](#_Toc182049977)

[2.2.1. Flutter 4](#_Toc182049978)

[2.2.2. Dart 5](#_Toc182049979)

[2.2.3. Android Studio 5](#_Toc182049980)

[2.2.4. Figma 5](#_Toc182049981)

[2.2.5. Widget 6](#_Toc182049982)

[BAB III 7](#_Toc182049983)

[PEMBAHASAN 7](#_Toc182049984)

[3.1. Desain Sistem Projek *Widget Checklist* 7](#_Toc182049985)

[3.2. User Interface (UI) Widget Checklist 8](#_Toc182049986)

[3.3. Coding Widget Checklist 11](#_Toc182049987)

[DAFTAR PUSTAKA 19](#_Toc182049988)

[LAMPIRAN 20](#_Toc182049989)

# DAFTAR GAMBAR

[Gambar 3. 1 *Flowchart* *Widget* *Checklist* 7](#_Toc182049990)

[Gambar 3. 2 Desain *UI Widget Checklist* pada Figma 8](#_Toc182049991)

[Gambar 3. 3 Teks pada *Checklist* 8](#_Toc182049992)

[Gambar 3. 4 *Checkbox* pada *Checklist* 9](#_Toc182049993)

[Gambar 3. 5 Komponen *Checklist* 9](#_Toc182049994)

[Gambar 3. 6 Desain dan Komponen *Checklist* 10](#_Toc182049995)

[Gambar 3. 7 Desain *UI Widget Checklist* setelah dijalankan pada *Figma* 10](#_Toc182049996)

[Gambar 3. 8 *Coding Widget Checklist* pada *Android Studio* 17](#_Toc182049997)

[Gambar 3. 9 *Screenshot* Aplikasi *Widget Checklist* pada *Android Studio* 18](#_Toc182049998)

[Gambar 3. 10 *Screenshot* Aplikasi *Widget Checklist Checked* pada *Android Studio* 18](#_Toc182049999)

\

# DAFTAR LAMPIRAN

[Lampiran 1. Merancang Desain Awal *Widget Checklist* 20](#_Toc182050000)

[Lampiran 2. Menjalankan Desain Awal *Widget Checklist* 20](#_Toc182050001)

[Lampiran 3. Revisi Desain *Widget Checklist* 21](#_Toc182050002)

[Lampiran 4. Menjalankan Revisi Desain *Widget Checklist* 21](#_Toc182050003)

[Lampiran 5. *Coding* *Widget Checklist* pada *Android Studio* 22](#_Toc182050004)

# BAB I

# PENDAHULUAN

## Latar Belakang

Dalam ekosistem perangkat mobile, *widget* memainkan peran penting sebagai salah satu elemen antarmuka pengguna (*UI*) yang sering digunakan dalam berbagai macam aplikasi untuk memberi pengguna akses kepada informasi atau fungsi aplikasi tertentu tanpa harus membuka aplikasi baru, diluar aplikasi yang sudah dibuka.

Salah satu *widget* yang sering diterapkan oleh *developer* dan ditemukan dalam berbagai macam aplikasi merupakan *widget* *checklist*, sebuah *widget* yang memungkinkan pengguna untuk mencatat, memantau, dan menandai tugas-tugas atau benda-benda langsung kedalam *widget* itu. *Widget* *checklist* ini sangat bermanfaat bagi pengguna yang ingin mengelola aktivitas yang perlu dilakukan dengan lebih efisien dan terorganisir.

Pengembangan *widget*, terutama pada platform *Android*, dipermudah dengan hadirnya berbagai *framework* lintas platform, salah satunya merupakan framework *Flutter* dengan bahasa pemrogramnan *Dart,* sebuah bahasa pemrograman yang dibangun khusus untuk *Flutter*.

*Flutter* merupakan *UI toolkit* yang portabel dan *framework open-source*, dengan performa aplikasi sekelas *native*, yang dibangun oleh *Google* untuk mengembangkan aplikasi yang *cross-platform*, sehingga *Flutter* dapat digunakan untuk pengembangan aplikasi mobile, aplikasi berbasis *web* ataupun aplikasi desktop (B. J. M. Putra & Fu’adi, 2024).

*Framework* *Flutter* ini menggunakan *Dart*, sebuah bahasa pemrograman yang membentuk formasi *Flutter*. *Dart* menyediakan bahasa dan runtime yang medukung aplikasi *Flutter* serta melakukan banyak tugas yang mendukung pengembangan sebuah aplikasi seperti *formatting*, analisis dan pengujian kode (Dart, 2024).

Sebelum merancangkan *widget checklist,* dibutuhkan desain awal untuk tampilan *interface checklist* tersebut. Untuk memenuhi kebutuhan itu, penulis menggunakan *Figmra,* sebuah alat yang digunakan untuk mendesain produk digital seperti, *mock-up, mindmap,* dan tampilan *interface* aplikasi yang ingin dikembangkan.

*Widget* *checklist* berbasis *Flutter* memberi kemampuan kepada pengguna untuk meningkatkan produktivitas mereka dengan menambah, menghapus, atau menandai tugas yang telah selesai langsung pada layar utama suatu perangkat. Aksi ini meningkatkan efisiensi dibandingkan apabila pengguna harus membuka sebuah aplikasi terpisah untuk melakukan aktivitas serupa. Selain itu, desain yang intuitif dan pengalaman pengguna (*UX*) yang baik juga menjadi faktor penting dalam perancangan *widget* *checklist* yang fungsional dan mudah digunakan oleh pengguna dari berbagai latar belakang.

Penelitian ini bertujuan untuk merancangkan sebuah *widget* *checklist* berbasis *Flutter* di *Android* *Studio* yang dapat diakses secara cepat dan mudah dari layar perangkat *smartphone*. Penelitian ini berharap untuk menghasilkan *widget* *checklist* yang efisien, fungsional *user-friendly* serta mendukung pengalaman pengguna yang lebih baik dalam mengelola barang atau aktivitas.

## Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan sebelumnya, maka rumusan masalah pada penelitian ini berupa:

1. Bagaimana merancang sebuah *widget checklist* pada *Android Studio* menggunakan *framework Flutter?*

## Batasan Masalah

Untuk menjelaskan lebih jelas pembahasan laporan agar hasil akhir penelitian ini spesifik dan detail, maka penelitian ini dibatasi pada:

1. Proses merancang *widget checklist* pada *Android Studio* menggunakan *framework Flutter* dan bahasa pemrograman *Dart.*

## Tujuan Penelitian

Berdasarkan rumusan masalah yang telah ditetapkan pada poin diatas, berikut tujuan makalah ini:

1. Merancang sebuah *widget checklist* yang efisien, fungsional dan *user-friendly.*

## Manfaat Penelitian

Berdasarkan rumusan masalah yang telah ditetapkan pada poin diatas, berikut tujuan makalah ini:

1. Membantu meningkatkan efisiensi dan produktivitas pengguna dengan adanya *widget checklist*

# BAB II

# PEMBAHASAN

## Teori Umum

### Aplikasi Mobile

Aplikasi mobile adalah sejenis perangkat lunak yang dirancang untuk berjalan di ponsel, tablet, atau smartphone. Aplikasi ini menawarkan layanan yang mirip dengan yang dapat diakses di PC (Y. W. S. Putra et al., 2023).

### Checklist

*Checklist* merupakan sebuah metode mencatat dan merekam kumpulan data. Dengan widget *checklist*, pengguna dapat lebih mudah mengatur tugas atau benda dengan lebih efisien, sehingga memastikan bahwa semua tugas atau item yang diperlukan dapat dipantau dan ditangani dengan baik. Pada aplikasi *shopping list* yang akan dibangun dalam proposal penelitian ini, checklist digunakan untuk menandai bahan-bahan atau barang yang dibutuhkan untuk sebuah *user*. Pengguna dapat mencentang setiap bahan atau barang yang ada, sehingga meningkatkan efisiensi mereka.

## Teori Khusus

### Flutter

*Flutter* adalah *framework open-source* yang menawarkan performa setara dengan aplikasi *native* dan mempercepat pengembangan aplikasi lintas platform untuk *Android* dan *iOS*. Dibuat dengan bahasa pemrograman *Dart*, *Flutter* memungkinkan developer menggunakan satu basis kode tanpa perlu mempelajari dua bahasa pemrograman. (Raharjo, 2021).

*Flutter* memberi beberapa keunggulan kepada *developer* berupa: *fast* *development*, *expressive* *and* *flexible* *UI* serta *native performance*. *Fast* *development* memungkinkan proses pembuatan aplikasi yang lebih efisien bagi *developer*. *Expressive and flexible UI* memberikan tampilan yang menarik untuk *Android (Material Design)* dan *iOS (Cupertino),* dengan berbagai API, scrolling yang alami, serta kemampuan mengenali platform secara otomatis. Selain itu, *Flutter* juga memiliki fitur hot reload yang mempermudah pengembangan, memungkinkan desain antarmuka, penambahan fitur, dan *debugging* (Suryono & Hardiansah, 2020).

### Dart

*Dart* dirancang sebagai bahasa pemrograman yang efisien untuk membangun aplikasi cepat di berbagai platform. *Dart* fokus untuk meningkatkan produktivitas *developer* dalam pengembangan *multi-platform*, dengan runtime yang fleksibel untuk mendukung berbagai *framework* aplikasi. Sebagai fondasi dari Flutter, *Dart* tidak hanya menyediakan bahasa dan *runtime* untuk aplikasi *Flutter*, tetapi juga mendukung berbagai tugas penting pengembang seperti memformat kode, melakukan analisis, dan menjalankan pengujian (Dart, 2024).

### Android Studio

Android Studio adalah sebuah *Integrated Development Environment (IDE)* yang dirancang untuk pengembangan aplikasi *Android*, dibangun dari basis *IntelliJ* *IDEA*. *IDE* ini menawarkan berbagai fitur untuk meningkatkan efisiensi developer, seperti emulator yang cepat dan kaya fitur, lingkungan terintegrasi untuk pengembangan di semua perangkat *Android*, fitur *Instant Run* yang memungkinkan perubahan langsung tanpa perlu membangun *APK* baru, sistem versi fleksibel berbasis *Gradle*, dukungan terintegrasi untuk *Google Cloud Platform*, dan masih banyak fitur lainnya (Suryana, 2018).

### Figma

*Figma* adalah alat desain berbasis *web* yang memungkinkan pengguna membuat desain digital berkualitas tinggi, seperti website dan aplikasi. *Figma* mendukung kolaborasi tim secara *real*-*time*, memudahkan pemberian masukan dan pengelolaan revisi. Fitur-fiturnya meliputi elemen desain, *grid*, *library*, dan prototyping. Elemen desain membantu menambahkan bentuk, *grid* menjaga tata letak rapi, *library* menyimpan elemen untuk digunakan kembali, dan *prototyping* memungkinkan pembuatan interaksi serta animasi (Sakti Wibowo, 2023).

### Widget

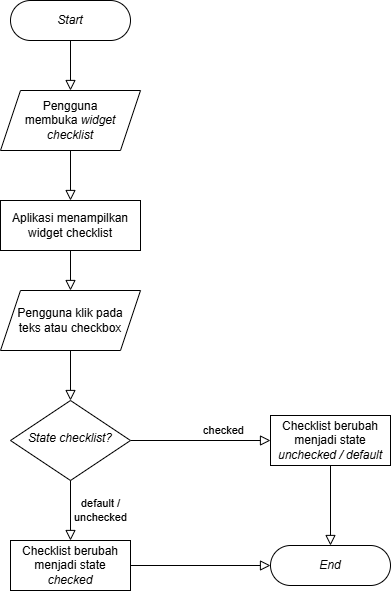
Pada *Flutter, widget* merupakan komponen-komponen penting yang membangun *user* *interface* yang akan dilihat oleh para pengguna. Beberapa contoh *widget* berupa *button, textbox, icon, appbar, image, container* dan lebih banyak lagi.

# BAB III

# PEMBAHASAN

## Desain Sistem Projek *Widget Checklist*

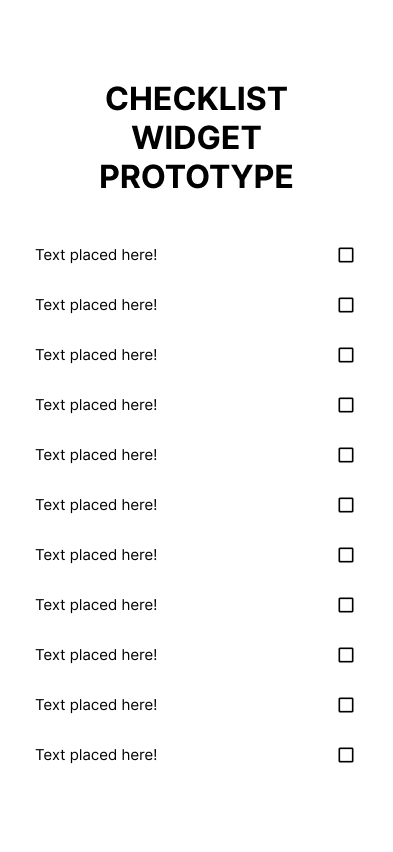
Berikut desain sistem menggunakan Flowchart yang telah dirancang untuk projek *widget checklist* ini:



Gambar 3. 1 *Flowchart* *Widget* *Checklist*

## User Interface (UI) Widget Checklist

Berikut merupakan tampilan *user interface* (antarmuka pengguna) *widget checklist* yang telah dirancang menggunakan Figma:

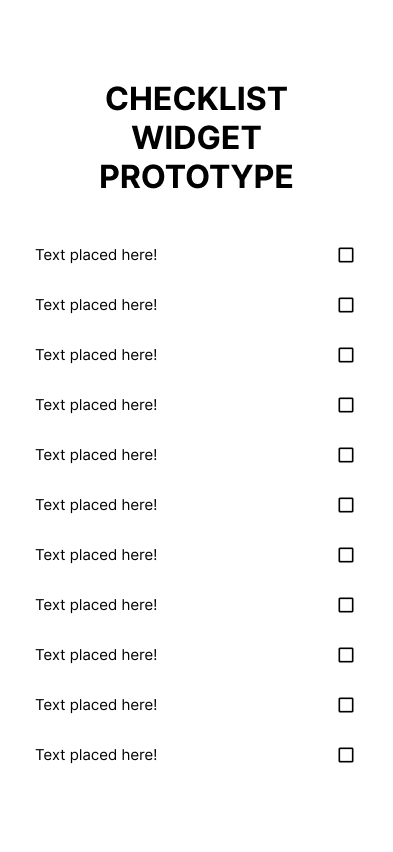


Gambar 3. 2 Desain *UI Widget Checklist* pada Figma

Ketika pengguna membuka *widget* *checklist,* aplikasi akan menampilkan judul widget teks “CHECKLIST WIDGET PROTOTYPE”.

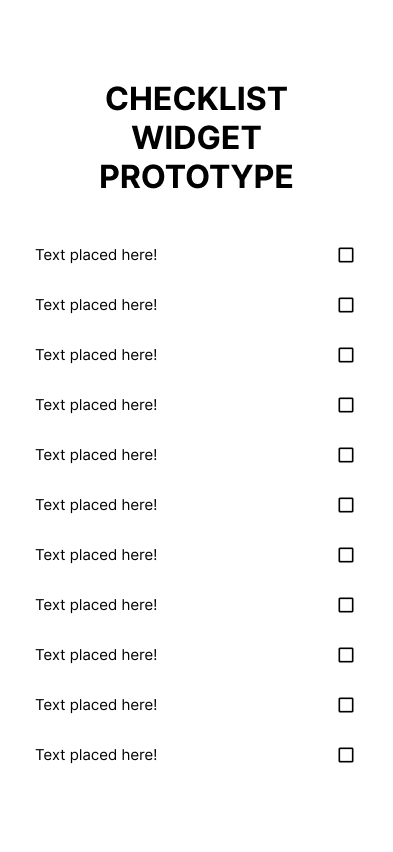
Dibawah judul itu, aplikasi menampilkan widget *checklist* yang dibagi menjadi 2 elemen utama, yaitu;

* Teks pada sisi kiri yang menuliskan teks benda (dan bisa juga suatu aktivitas yang perlu dilakukan).



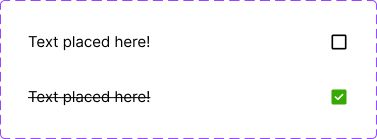
Gambar 3. 3 Teks pada *Checklist*

* Kotak *checkbox* pada ujung kanan yang akan berfungsi sebagai tanda apabila benda terpenuhi atau tidaknya benda.



Gambar 3. 4 *Checkbox* pada *Checklist*

*Prototype* widget checklist ini dibangun menggunakan beberapa elemen seperti: teks, *icon,* dan *rectangle.* Elemen teks, *icon* dan *rectangle* kemudian digunakan untuk membangun komponen utama *widget checklist* ini:

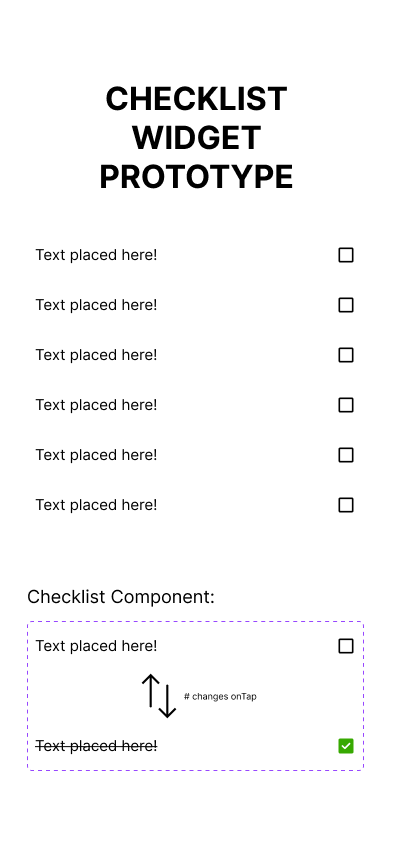


Gambar 3. 5 Komponen *Checklist*

Komponen ini menggunakan interaksi *On Tap,* dimana tampilan *checklist* berubah ketika pengguna klik pada teks atau checkbox.

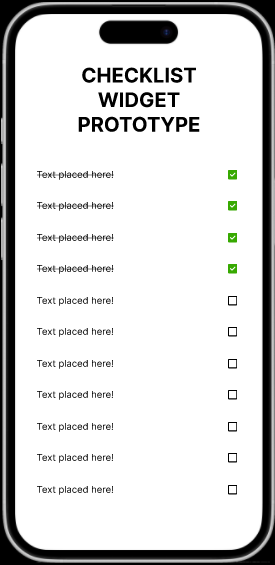
Dalam tampilan *default,* teks berwarna hitam dengan checkbox kosong yang memiliki *border* hitam.

Ketika pengguna klik teks atau checkbox tersebut, teks akan berubah dan menambahkan efek *strikethrough* dan checkbox warna hijau dengan *icon* centang.



Gambar 3. 6 Desain dan Komponen *Checklist*

*Prototype* ini dirancang mengikuti spesifikasi ukuran iPhone 16. Berikut tampilan *prototype widget checklist* setelah dijalankan:



Gambar 3. 7 Desain *UI Widget Checklist* setelah dijalankan pada *Figma*

Berikut tautan untuk *file* Figma ini:

<https://www.figma.com/design/MVze5fqU09Wp2gZfDh6Dqn/Checklist-Widget-%231?node-id=0-1&t=5snMC9LAkhoAjW3z-1>

## Coding Widget Checklist

Widget checklist ini dibangun menggunakan bahasa pemrograman Dart dan *framework* Flutter dengan emulator *device* Pixel 6a API 33Android 13.0 (“Tiramisu”) pada Android Studio.

import 'package:flutter/material.dart';  
  
void main() {  
 runApp(const MyApp());  
}  
  
class MyApp extends StatelessWidget {  
 const MyApp({super.key});  
  
 // This widget is the root of your application.  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 theme: ThemeData(  
 useMaterial3: true,  
 ), // ThemeData  
 home: const MyHomePage(title: '),  
 ); // MaterialApp  
 }  
}  
  
class MyHomePage extends StatefulWidget {  
 const MyHomePage({super.key, required this.title});  
  
  
 final String title;  
  
 @override  
 State<MyHomePage> createState() => \_MyHomePageState();  
}  
  
class \_MyHomePageState extends State<MyHomePage> {  
 List<Map> categories = [  
 {"name": "Broom", "isChecked": false},  
 {"name": "Minced Chicken", "isChecked": false},  
 {"name": "Sesame Oil", "isChecked": false},  
 {"name": "Mouse Trap", "isChecked": false},  
 {"name": "Spoons", "isChecked": false},  
 {"name": "Cabinet", "isChecked": false},  
 {"name": "Apple", "isChecked": false},  
 {"name": "Wipes", "isChecked": false},  
 {"name": "Puzzle", "isChecked": false},  
 {"name": "Paper", "isChecked": false},  
 ];  
  
@override  
Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text(widget.title),  
 ), // AppBar  
 body: SingleChildScrollView(  
 child: Padding(  
 padding: const EdgeInsets.all(20),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 const Text(  
 'CHECKLIST WIDGET PROTOTYPE',  
 textAlign: TextAlign.center,  
 style: TextStyle(  
 fontSize: 45,  
 fontWeight: FontWeight.bold,  
 ), // TextStyle  
 ), // Text  
 const SizedBox(height: 10),  
 Column(  
 children: categories.map((favorite) {  
 return CheckboxListTile(  
 activeColor: Colors.green,  
 onChanged: (val) {  
 setState(() {  
 favorite['isChecked'] = val ?? false;  
 });  
 },  
 value: favorite['isChecked'],  
 title: Text(  
 favorite['name'] ?? '',  
 style: TextStyle(  
 fontSize: 20,  
 decoration: favorite['isChecked']  
 ? TextDecoration.lineThrough  
 : TextDecoration.none,  
 ), // TextStyle  
 ), // Text  
 ); // CheckboxListTile  
 }).toList(),  
 ) // Column  
 ],  
 ),  
 ),  
 )  
 );  
 }  
}

Perintah ini mengimport paket Material Design hingga semua widget dan *property pre-built* dapat diakses dan digunakan dalam proyek ini.

import 'package:flutter/material.dart';

Function main()dibawah ini digunakan untuk memanggil dan menjalankan function global runApp()yang mengambil widget MyApp() sebagai argumen dan akan mengembangkan widget tersebut ke perangkat apa pun yang sedang dijalankan.

void main() {  
 runApp(const MyApp());  
}

Custom widget MyApp() berfungsi sebagai representasi aplikasi. Class *StatelessWidget* merupakan sebuah class yang dibangun kedalam Flutter yang digunakan untuk membangun elemen UI yang statis dan tidak berubah sama sekali.

class MyApp extends StatelessWidget {  
 const MyApp({super.key});

Metode build berikut ini akan mengembalikan sebuah widget dan akan selalu dipanggil oleh Flutter apabila UI dijalankan lagi (*reload*). Widget yang dikembalikan atau di*return* adalah MaterialApp() yang digunakan sebagai akar dari seluruh aplikasi dan memberi pengguna kemampuan untuk mengkonfigurasi tema dan rute aplikasi.

Widget build ini menentukan bahwa aplikasi ini menggunakan tema *Material13* serta memanggil *class* MyHomePage()pada halaman utama (*home*).

// This widget is the root of your application.  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 theme: ThemeData(  
 useMaterial3: true,  
 ), // ThemeData  
 home: const MyHomePage(title: ''),  
 ); // MaterialApp  
 } // build  
} // MyApp

MyHomePage() dibawah ini merupakan *subclass* dari *StatefulWidget,* sebuah *class* yang digunakan apabila *widget* yang dirancang dinamis atau bisa mengubah UI aplikasi sesuai dengan aksi yang dilakukan pengguna.

Dalam *subclass* MyHomePage() terdapat parameter title yang diwajibkan dan diambil dari widget build sebelumnya. Title ini dikeluarkan sebagai output tipe data *string.*

class MyHomePage extends StatefulWidget {  
 const MyHomePage({super.key, required this.title});  
  
  
 final String title;

Metode createState() dibawah ini mengembalikan atau *return instance* \_MyHomePageState, yang akan menangani *state* widget sebenarnya.

@override  
 State<MyHomePage> createState() => \_MyHomePageState();  
} // StatefulWidget

*Class* \_MyHomePageState ini merupakan sebuah *private class* (ditandai oleh simbol \_ pada awal nama *class*), yang memperluas *State<MyHomePage>*. *Class* ini akan menangani state, interaksi dan pembaruan dari widget MyHomePage.

class \_MyHomePageState extends State<MyHomePage> {

List<Map> ini diberi nama categories dan mengandung kumpulan item, tiap kumpulan item ini direpresentasikan sebagai sebuah Map dengan dua *key-value pairs,* yaitu:

* “name”, yaitu nama item pada catergory, seperti: “Broom”, “Minced Chicken”, “Sesame Oil” dan lainnya.
* “isChecked”, merupakan nilai boolean yang secara *default* mulai dalam state *false* untuk tiap item, hinggan menandakan bahwa setiap item masih tidak mencentang*.*

List ini digunakan untuk menampilkan semua item pada *widget checklist* dalam bentuk daftar list dengan kotak centang di UI. Ketika pengguna mencentang atau tidak mencentang sebuah item, nilai isChecked akan berubah sesuai dengan interaksi pengguna.

List<Map> categories = [  
 {"name": "Broom", "isChecked": false},  
 {"name": "Minced Chicken", "isChecked": false},  
 {"name": "Sesame Oil", "isChecked": false},  
 {"name": "Mouse Trap", "isChecked": false},  
 {"name": "Spoons", "isChecked": false},  
 {"name": "Cabinet", "isChecked": false},  
 {"name": "Apple", "isChecked": false},  
 {"name": "Wipes", "isChecked": false},  
 {"name": "Puzzle", "isChecked": false},  
 {"name": "Paper", "isChecked": false},  
 ]; // categories

Scaffold() merupakan sebuah struktur layout pada Flutter yang berfungsi sebagai dasar dari layar aplikasi. Scaffold() juga menyediakan elemen-elemen dasar UI seperti appBar dan body.

Pada kode dibawah ini, build() mengembalikan Scaffold() untuk memanggil appBar dengan judul atau *title* yang diambil dari *StatefulWidget* MyHomePage() sebelumnya.

@override  
Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text(widget.title),  
 ), // AppBar

SingleChildScrollView() merupakan sebuah *class* yang menyediakan sebuah kotak tidak terlihat dimana sebuah widget dapat scroll. Pada kode dibawah ini, SingleChildScrollView() digunakan untuk membungkus Column() ke area yang dapat discroll oleh pengguna, hingga konten didalam kotak tersebut dapat discroll secara vertikal apabila konten lebih dari kotak tersebut.

Padding() menambahkan 20 pixel padding kepada seluruh sisi kolom untuk memberi ruang antara konten dan ujung layar.

Column() digunakan sebagai container utama untuk *widget* dan mengatur widget tersebut dalam layout vertikal.

body: SingleChildScrollView(  
 child: Padding(  
 padding: const EdgeInsets.all(20),  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [

Konstanta text dibawah ini menampilkan header “CHECKLIST WIDGET PROTOTYPE” dengan font berukuran 45 dan efek *bold* atau tebal.

const Text(  
 'CHECKLIST WIDGET PROTOTYPE',  
 textAlign: TextAlign.center,  
 style: TextStyle(  
 fontSize: 45,  
 fontWeight: FontWeight.*bold*,  
 ), // TextStyle  
 ), // Text

SizedBox() menambahkan ruang vertikal dengan ukuran 10pixel dibawah teks header sebelumnnya.

const SizedBox(height: 10),

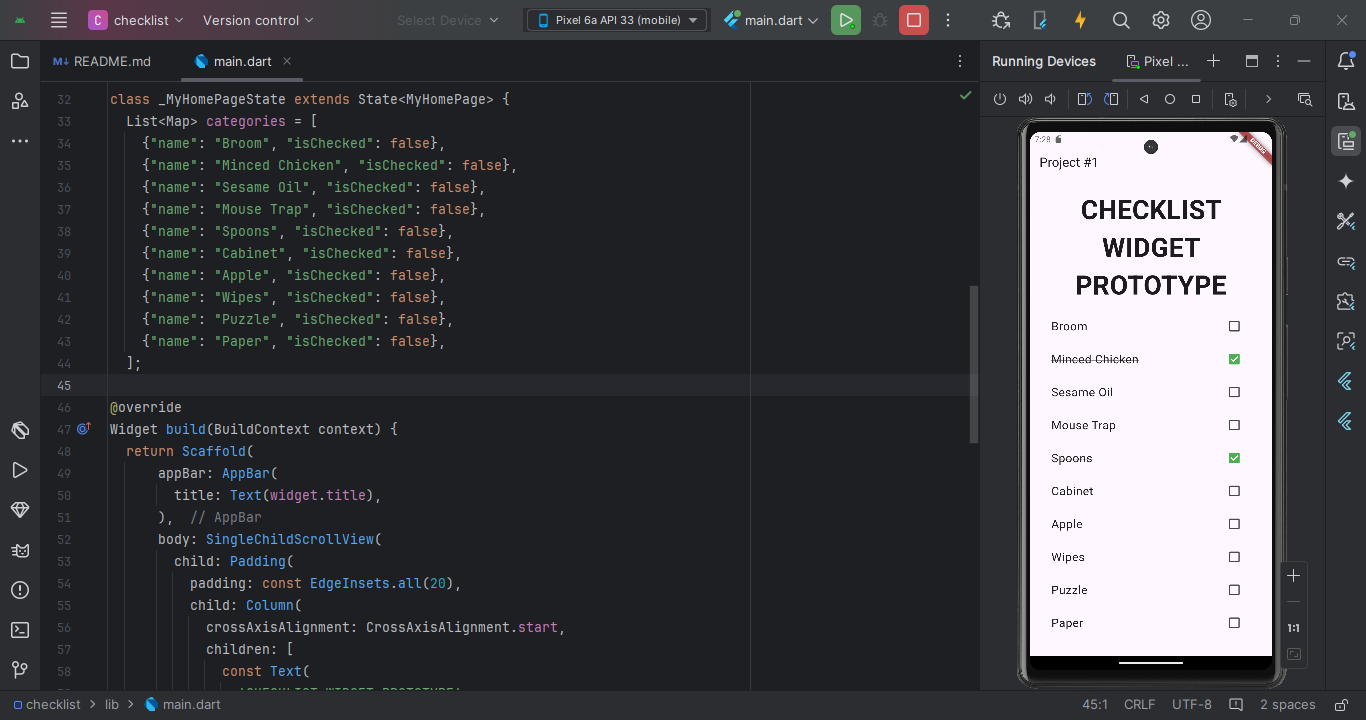
Column() dibawah ini mengandung daftar widget yang telah dibangun dari *list categories* sebelumnya dengan menggunakan .map.

CheckboxListTile() akan menampilkan checkbox dengan label yang diambil dari (favorite[‘name’]). Atribut-atribut penting dalam CheckboxListTile ini berupa:

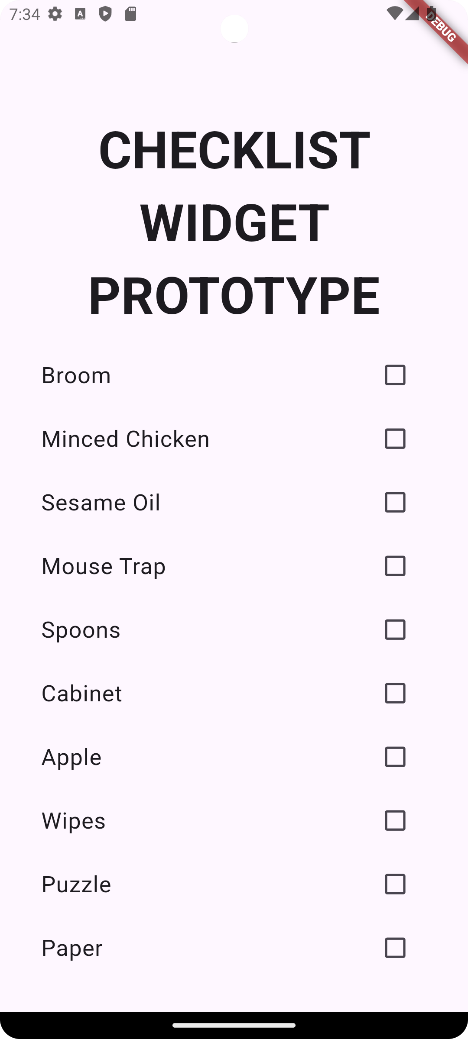
* activeColor, yang mengatur warna kotak *checkbox* menjadi hijau apabila aktif.
* onChanged, yaitu fungsi callback yang mengubah status isChecked saat kotak centang di-toggle. Metode setState() dipanggil untuk memicu pembaruan ulang (rebuild) widget agar UI mengupdate nilai terbaru.
* value, yaitu status apakah checkbox tercentang atau tidak, diambil dari favorite['isChecked'].
* title, yaitu nama item yang diambil dari favorite[‘name’] dari List<map>
* style, yaitu memberi garis pada teks apabila isChecked bernilai true; jika tidak, teks akan tampil seperti biasa, tanpa ada garis.

Column(  
 children: categories.map((favorite) {  
 return CheckboxListTile(  
 activeColor: Colors.*green*,  
 onChanged: (val) {  
 setState(() {  
 favorite['isChecked'] = val ?? false;  
 });  
 },  
 value: favorite['isChecked'],  
 title: Text(  
 favorite['name'] ?? '',  
 style: TextStyle(  
 fontSize: 20,  
 decoration: favorite['isChecked']  
 ? TextDecoration.*lineThrough* : TextDecoration.*none*,  
 ), // TextStyle  
 ), // Text  
 ); // CheckboxListTitle  
 }).toList(),  
 ) // Column  
 ],  
 ), // Column  
 ), // Padding  
 ) // SingleChildScrollView  
 ); // Scaffold  
 }  
}

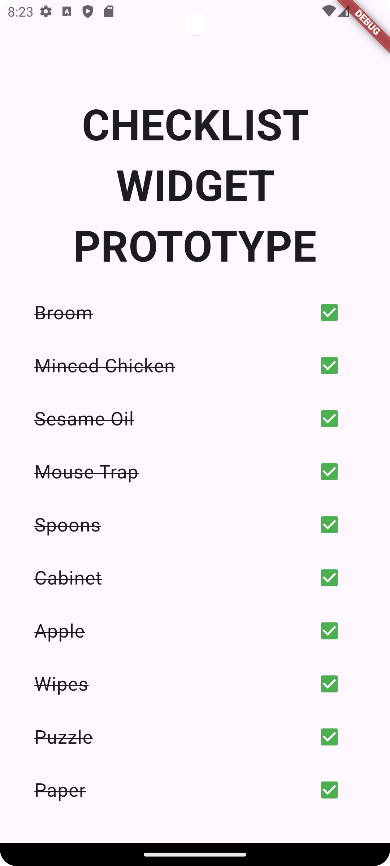
Berikut adalah hasil akhir aplikasi *widget checklist* yang telah dirancang:



Gambar 3. 8 *Coding Widget Checklist* pada *Android Studio*



Gambar 3. 9 *Screenshot* Aplikasi *Widget Checklist* pada *Android Studio*



Gambar 3. 10 *Screenshot* Aplikasi *Widget Checklist Checked* pada *Android Studio*

# DAFTAR PUSTAKA

Dart. (2024, May 30). *Dart Overview*. Dart. https://dart.dev/overview

Putra, B. J. M., & Fu’adi, A. (2024). *Belajar Pemrograman Mobile dengan Flutter* (Kurniawadi, Ed.; 1st ed., Vol. 1). Wawasan Ilmu.

Putra, Y. W. S., Dawis, A. M., Novi, N., Natsir, F., Fitria, F., Widhiyanti, A. A. S., Hasan, F. N., Somantri, S., & Maniah, M. (2023). *PENGANTAR APLIKASI MOBILE* (W. Andriyani & Erlangga, Eds.; 1st ed.).

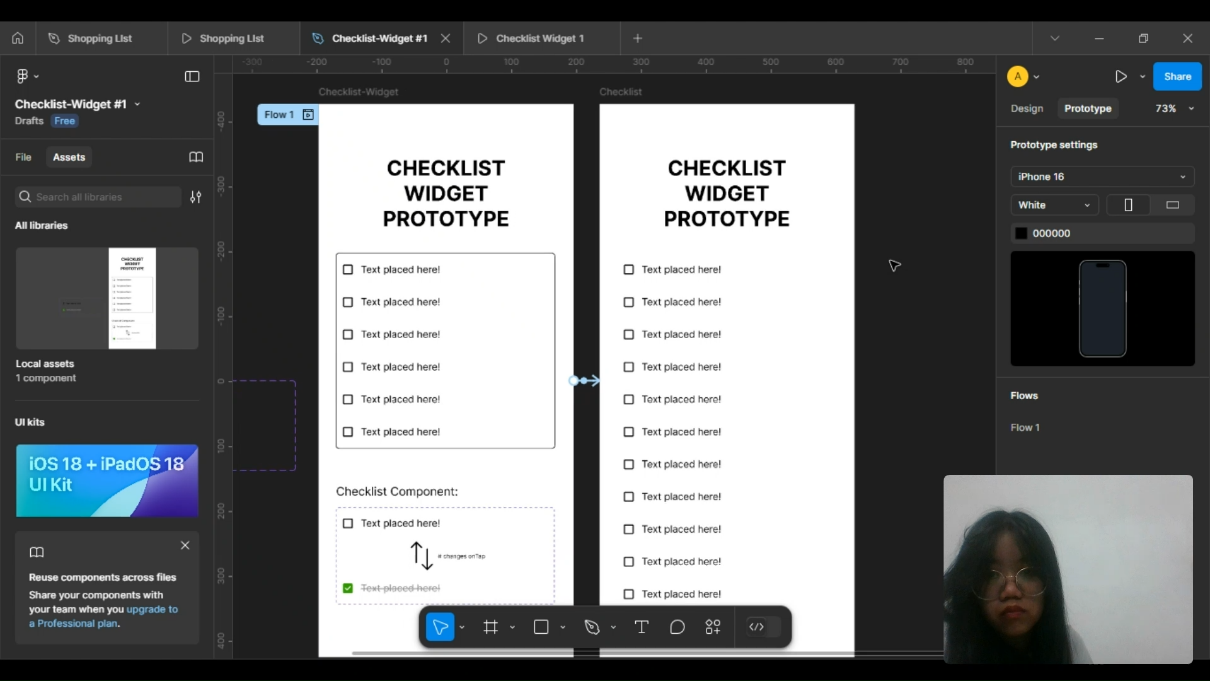
Raharjo, B. (2021). *Pemrograman Android Dengan Flutter*. Informatika Bandung.

Sakti Wibowo, H. (2023). *Desain Interaktif dengan Figma Panduan Praktis untuk Pemula dan Profesional* (W. Anita, Ed.; First Printing). Tiram Media.

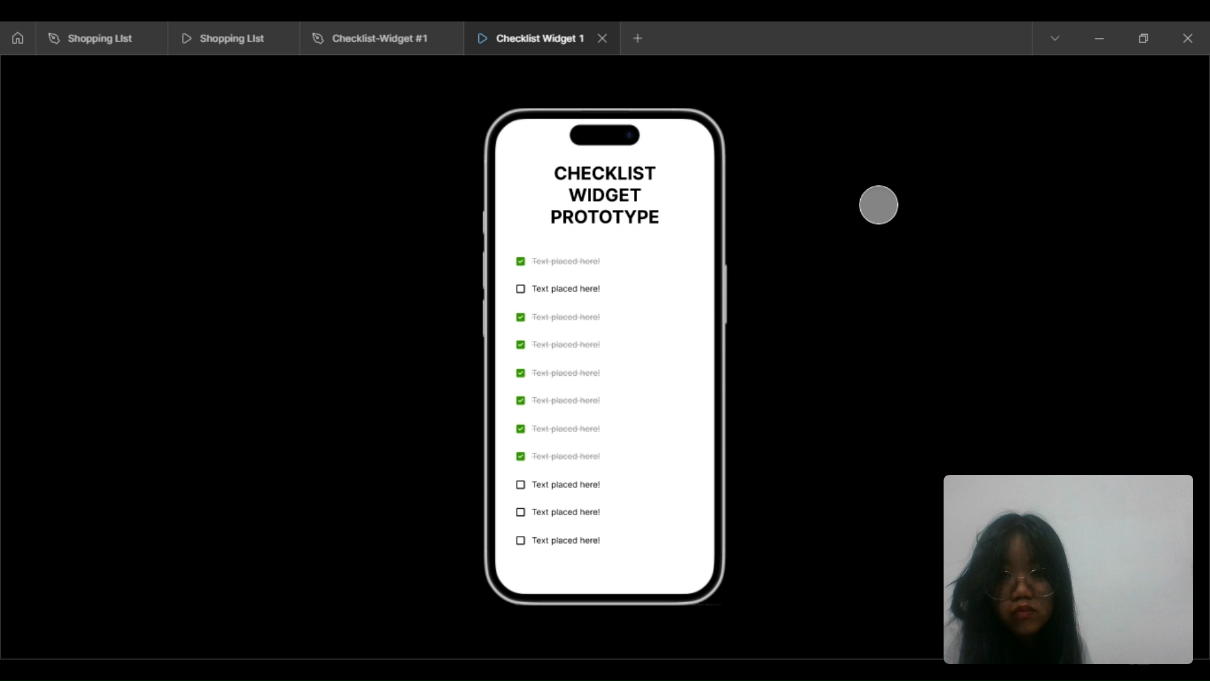
Suryana, D. (2018). *Android Studio Belajar Android Studio*. https://www.google.co.id/books/edition/Android\_Studio/wIhyDwAAQBAJ?hl=en&gbpv=0

Suryono, B. S., & Hardiansah. (2020). *Panduan Lengkap membuat aplikasi Android dan iOS dari Nol*.

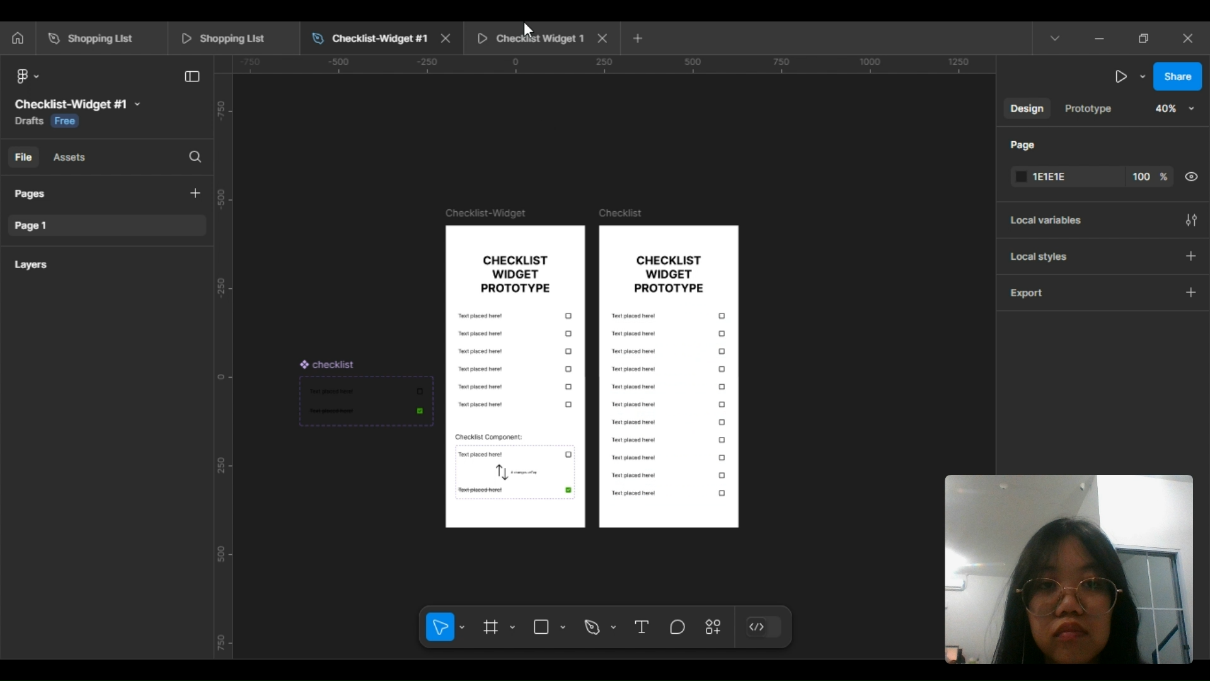
# LAMPIRAN



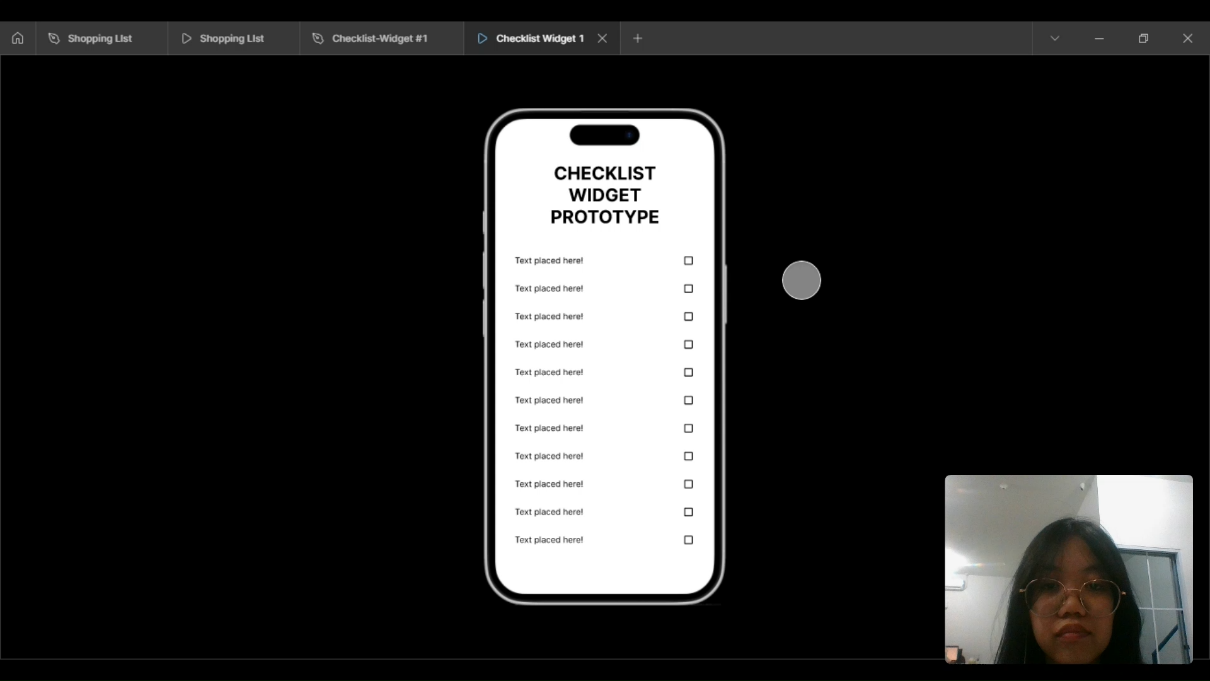
Lampiran 1. Merancang Desain Awal *Widget Checklist*



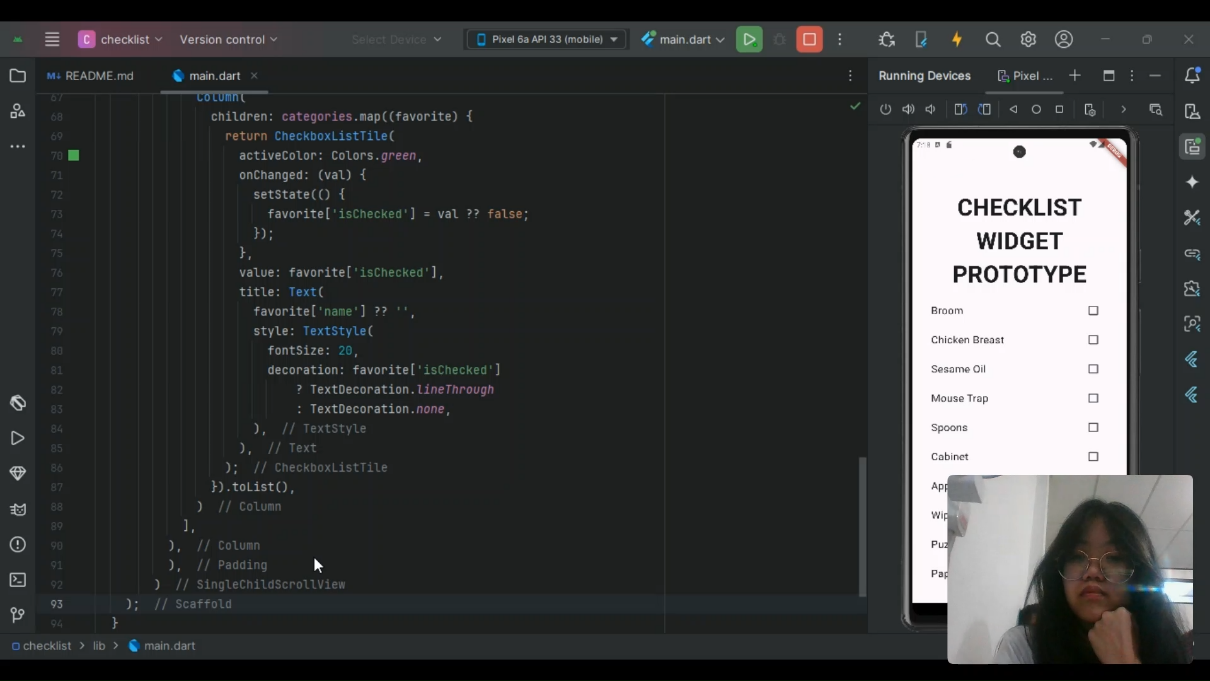
Lampiran 2. Menjalankan Desain Awal *Widget Checklist*



Lampiran 3. Revisi Desain *Widget Checklist*



Lampiran 4. Menjalankan Revisi Desain *Widget Checklist*



Lampiran 5. *Coding* *Widget Checklist* pada *Android Studio*