

**PROFESSIONAL TRAINING REPORT
AT
Sathyabama Institute of Science and Technology
(Deemed to be University)**

**Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering Degree in
Computer Science and Engineering**

By

**KALARI LIKITHA
Reg.No:39120061**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600119**

NOVEMBER 2021



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade “A” by NAAC
(Established under Section 3 of UGC Act, 1956)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119
www.sathyabamauniversity.ac.in



**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING**

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **KALARI LIKITHA(Reg.No:39120061)** who carried out the project entitled “**HOTEL BOOKING APP (FLUTTER)**” under my supervision SEPTEMBER 2021 to OCTOBER 2021.

Internal Guide Name

Dr.S.Gowri

Head of the Department

Dr.R.Subhashini

Submitted for Viva voce Examination held on

Internal Examiner

External Examiner

DECLARATION

I **KALARI LIKITHA (Reg.no.39120061)** hereby declare that the Project Report entitled “E – COMMERCE APP (FLUTTER)” done by me under the guidance of **Dr.S.Gowri** and at **ALGO FUSION** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

K. Likitha.

PLACE: Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr.T.SASIKALA., M.E.,Ph.D.**, Dean, School of Computing **Dr.S.Vigneshwari M.E,Ph.D**, **Dr.L.Lakshmanan.,M.E.,Ph.D.**, Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.S.Gowri** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE



ABSTRACT

Mobile and e-commerce applications are tools for accessing the Internet and for buying products and services. These applications are constantly evolving due to the high rate of technological advances being made. This paper provides a new perspective on the types of applications that can be used. It describes and analyses device requirements, provides a literature review of important aspects of mobile devices that can use such applications and the requirements of websites designed for e-commerce. The design and security aspects of mobile devices are also investigated. As an alternative to existing e-commerce applications, this paper also investigates the characteristics and potential of the PhoneGap cross-mobile platform application. The results suggest that effective mobile applications do exist for various Smart phones, and web applications on mobile devices should be effective.

Ecommerce or electronic commerce actually means the use of an electronic medium for commercial transactions, but it is commonly used to refer to selling products and services over the internet to consumers or other businesses.

According to another definition, ecommerce refers to the sharing of business information, maintenance of business relationships and conducting business transactions with the help of computers that are connected to a telecommunication network, and without the use of paper documents.

Table of Content

Chapter No	TITLE	Page No
1	INTRODUCTION	
	1.1 ABOUT FLUTTER	
	1.2 ABOUT DART	
2	AIM AND SCOPE	
	2.1 AIM	
	2.2 SCOPE OF THE PROJECT	
	2.3 FUTURE WORK	
3	EXPERIMENTAL AND METHOD ALGORITHM	
	3.1 MODULAR DESIGN	
	3.2 INSTALL FLUTTER	
	3.3 SOURCE CODE	
4	RESULT AND DISCUSSION	
	4.1 OUTPUT SCREEN SHOTS	
	4.2 PERFORMANCE ANALYSICS	
5	SUMMARY AND CONCLUSION	
	5.1 SUMMARY	
	5.2 CONCLUSION	

CHAPTER 1

INTRODUCTION

1.1 ABOUT FLUTTER

Flutter has been around already since 2015 when Google first introduced it, but the buzz around it has grown stronger only lately. It's a cross-platform tool intended for creating Android and iOS apps from a single code base by using a modern, reactive framework. Flutter apps are built using Dart, a simple object-oriented programming language. The central idea of Flutter revolves around widgets. The entire UI is made of combining different widgets, each of which defines a structural element (like a button or menu), a stylistic element (like a font or color scheme), an aspect of layout (like padding), and so on. Flutter does not use OEM widgets, but provides its own ready-made widgets which look native either to Android (Material Design) or iOS apps (Cupertino). It's also possible to create custom widgets. Look here for a technical overview.

In terms of popularity, Flutter is making good progress. While Flutter had made it to the top 100 software repos based on GitHub stars by the time release preview 1 was announced in June 2018, it has risen in the ranks and is now among the top 30. Two years later Flutter SDK ranks as top 10 software repo on Github, following giants like Linux, vue and vscode. This, without a doubt, is a promising trend. Thousands of Flutter apps have made its way to app stores, among these the Alibaba app with 50 million users.

Without making any comparisons with other platforms, here's a list of some of the features and qualities that may make you consider having a go at Flutter:

- High productivity. Since Flutter is cross-platform, you can use the same code base for your iOS and Android app. This can definitely save you both time and resources.
- Great performance. Dart compiles into native code and there is no need to access OEM widgets as Flutter has its own. This means less mediated communication between the app and the platform. As Wm Leler puts it: "Flutter is the only mobile SDK that provides reactive views without requiring a JavaScript bridge." All of this contributes to fast app startup times and less performance issues.
- Fast and simple development. One of the most lauded features of Flutter is hot reload which allows you to instantly view the changes made in the code on emulators, simulators and hardware. In less than a second, the changed code is reloaded while the app is running with no need for a restart. This is great not just for building UIs or adding features but also for bug fixing. As far as simplicity is concerned, Flutter claims in its docs that programming with Flutter is so easy that no prior programming knowledge is required: "Experience with object-oriented languages is definitely helpful, but even non-programmers have made Flutter apps!" There's only one way to find out whether this is true.
- Compatibility. Since widgets are part of the app and not the platform, you'll likely experience less or no compatibility issues on different OS versions. This in turn means less time spent on testing.
- Open-source. Both Flutter and Dart are open-source and free to use, and provide extensive documentation and community support to help out with any issues you may encounter.

1.2 ABOUT DART

Dart is a client-optimized language for developing fast apps on any platform. Its goal is to offer the most productive programming language for multi-platform development, paired with a flexible execution runtime platform for app frameworks.

Languages are defined by their technical envelope — the choices made during development that shape the capabilities and strengths of a language. Dart is designed for a technical envelope that is particularly suited to client development, prioritizing both development (sub-second stateful hot reload) and high-quality production experiences across a wide variety of compilation targets (web, mobile, and desktop).

Dart also forms the foundation of Flutter. Dart provides the language and runtimes that power Flutter apps, but Dart also supports many core developer tasks like formatting, analyzing, and testing code.

Dart: The language

The Dart language is type safe; it uses static type checking to ensure that a variable's value *always* matches the variable's static type. Sometimes, this is referred to as sound typing. Although types are mandatory, type annotations are optional because of type inference. The Dart typing system is also flexible, allowing the use of a dynamic type combined with runtime checks, which can be useful during experimentation or for code that needs to be especially dynamic.

Dart offers sound null safety, meaning that values can't be null unless you say they can be. With sound null safety, Dart can protect you from null exceptions at runtime through static code analysis. Unlike many other null-safe languages, when Dart determines that a variable is

non-nullable, that variable is always non-nullable. If you inspect your running code in the debugger, you'll see that non-nullability is retained at runtime (hence *sound* null safety).

The following code sample showcases several Dart language features, including libraries, async calls, nullable and non-nullable types, arrow syntax, generators, streams, and getters. To find examples of using additional Dart features, see the samples page. To learn more about the language, take the Dart language tour.

Dart: The libraries

Dart has a rich set of core libraries, providing essentials for many everyday programming tasks:

- Built-in types, collections, and other core functionality for every Dart program (`dart:core`)
- Richer collection types such as queues, linked lists, hashmaps, and binary trees (`dart:collection`)
- Encoders and decoders for converting between different data representations, including JSON and UTF-8 (`dart:convert`)
- Mathematical constants and functions, and random number generation (`dart:math`)
- File, socket, HTTP, and other I/O support for non-web applications (`dart:io`)
- Support for asynchronous programming, with classes such as `Future` and `Stream` (`dart:async`)
- Lists that efficiently handle fixed-sized data (for example, unsigned 8-byte integers) and SIMD numeric types (`dart:typed_data`)
- Foreign function interfaces for interoperability with other code that presents a C-style interface (`dart:ffi`)

- Concurrent programming using isolates — independent workers that are similar to threads but don't share memory, communicating only through messages (`dart:isolate`)
- HTML elements and other resources for web-based applications that need to interact with the browser and the Document Object Model (DOM) (`dart:html`)

Beyond the core libraries, many APIs are provided through a comprehensive set of packages. The Dart team publishes many useful supplementary packages, such as these:

- `characters`
- `intl`
- `http`
- `crypto`
- `markdown`

Additionally, third-party publishers and the broader community publish thousands of packages, with support for features like these:

- XML
- Windows integration
- SQLite
- compression

DART: The platforms

- **Native platform:** For apps targeting mobile and desktop devices, Dart includes both a Dart VM with just-in-time (JIT) compilation and an ahead-of-time (AOT) compiler for producing machine code.
- **Web platform:** For apps targeting the web, Dart includes both a development time compiler (`dartdevc`) and a production time compiler (`dart2js`). Both compilers translate Dart into JavaScript.

The Flutter framework is a popular, multi-platform UI toolkit that's powered by the Dart platform, and that provides tooling and UI libraries to build UI experiences that run on iOS, Android, macOS, Windows, Linux, and the web.

Dart Native (machine code JIT and AOT)

During development, a fast developer cycle is critical for iteration. The Dart VM offers a just-in-time compiler (JIT) with incremental recompilation (enabling hot reload), live metrics collections (powering DevTools), and rich debugging support.

When apps are ready to be deployed to production — whether you're publishing to an app store or deploying to a production backend — the Dart AOT compiler enables ahead-of-time compilation to native ARM or x64 machine code. Your AOT-compiled app launches with consistent, short startup time.

The AOT-compiled code runs inside an efficient Dart runtime that enforces the sound Dart type system and manages memory using fast object allocation and a generational garbage collector.

More information:

- Get started: Command-line and server apps
- `dart` tool for running with JIT or AOT compiling to machine code
- Write command-line apps
- Write HTTP servers

Dart Web (JavaScript dev & prod)

Dart Web enables running Dart code on web platforms powered by JavaScript. With Dart Web, you compile Dart code to JavaScript code, which in turn runs in a browser — for example, V8 inside Chrome.

Dart web contains both an incremental dev compiler enabling a fast developer cycle, and an optimizing production compiler, `dart2js`, which compiles Dart code to fast, compact, deployable JavaScript using techniques such as dead-code elimination.

More information:

- [Get started: Web apps](#)
- [dartdevc compiler](#)
- [webdev tool](#)
- [Web deployment tips](#)

The Dart runtime

Regardless of which platform you use or how you compile your code, executing the code requires a Dart runtime. This runtime is responsible for the following critical tasks:

- **Managing memory:** Dart uses a managed memory model, where unused memory is reclaimed by a garbage collector (GC).
- **Enforcing the Dart type system:** Although most type checks in Dart are static (compile-time), some type checks are dynamic (runtime). For example, the Dart runtime enforces dynamic checks by type check and cast operators.
- **Managing isolates:** The Dart runtime controls the main isolate (where code normally runs) and any other isolates that the app creates.

On native platforms, the Dart runtime is automatically included inside self-contained executables, and is part of the Dart VM provided by the `dart run` command.

Learning Dart

You have many choices for learning Dart. Here are a few that we recommend:

- Explore Dart in the browser through DartPad, a web-based execution environment for Dart code.
- Take a tour of the Dart language, which shows you how to use each major Dart feature.
- Complete a Dart tutorial that covers the basics of using Dart to build for the command line.
- Work through extensive online training from Dart experts.
- Explore the API documentation that describes the Dart core libraries.
- Read a book about Dart programming.

CHAPTER 2

AIM AND SCOPE

2.1 AIM

2.2 SCOPE OF THE PROJECT

With mobile applications, customers can interact with numerous products and services in the simplest and easiest of ways. They can easily view the products and services on sale along with their descriptions and all the other details. All that users would have to do for connecting to these apps is touch, speak or type on their smart phones. With mobile applications, users are always connected to diverse companies and institutions like banks, insurance agencies, hospitals, education centers, and so forth.

This makes it easier for companies to market their products more precisely, and in the right fashion. With ecommerce mobile applications, it is estimated that the sales might increase by 30 times in the next five years. Ecommerce mobile applications offer notifications and other important details that send out alerts to customers as soon as a particular event takes place. These notifications could be everything from product news, sale offers, events, or things that a company implements for taking on its strategies.

Besides, ecom mobile apps are also a solution to customer grievances. It is indeed a pleasing experience for customers to interact with the company directly as they don't have to wait for company representatives to register a complaint, take care of their demands, or address their product related queries or payments.

Benefits of E-com Mobile Apps

1. Enables users to shop from anywhere, anytime, while they travel, during workout, etc.
2. The presence of more mobile applications influence the purchase decisions of those who wish to buy products only after comparing the prices, features and other attributes of all other products.
3. Collects the customer feedback and data for survey campaigns.

4. Keeps a track of the usage pattern and shopping history of customers.
5. Facilitates easy shopping of different items, and sharing of various company details without customers being present physically.

2.3 FUTURE WORK

B2B mobile e-commerce apps are focused on interactions between companies. To be precise, one company (business) sells a product (or provides a service) to another company (which is business too).

Suppose you manufacture and sell building materials, and your customers are construction firms (after all, they are the ones who use your product). Thus, you need some kind of platform to promote your company and find new clients in a better way. You can create a mobile application and work through it... or you're welcome to find an existing one and take full advantage of it (for a fee, of course).

By the by, here's a great startup idea: build a e-Commerce project that serves as a meeting place for business representatives, and start getting your commission

CHAPTER 3

EXPERIMENTAL AND METHOD ALGORITHM

3.1 MODULAR DESIGN

#1. Earning money

Mobile e-commerce apps will boost your income. Why? Because any customer now has an opportunity to make a purchase in your e-store (instead of buying goods in a real shop). Therefore, you'll have more acting customers, which is equal to more money.

Anyway, a good chance to increase your profitability is a very impressive reason to create an e-commerce mobile application, what do you think?

#2. Strengthening of business

Here we're discussing mobile apps that can augment e-commerce success.

Suppose, you, as the owner of a clothing store, are quite happy with the level of your income. But, of course, you're unlikely to be against increasing your business profitability (and who would be?).

There are many options to improve your earnings, and one of them is by making a mobile responsive e-commerce site and an application. Thus, you offer your customers several ways to go shopping: in a real store or via the digital platform. Whatever your customer likes and finds convenient!

Moreover, having a real (physical) store isn't a must-have. A virtual one will do just fine.

#3. Following the latest trends

Successful businesses are those companies that follow the latest trends and implement the most relevant ones (such as building a mobile application for e-commerce projects) in their processes.

Ordinary stores have become outdated, they're not enough in order to be among the market leaders. The mobile e-commerce era has come.

#4. Mobility

Developing a mobile app for an e-commerce store implies getting maximum mobility. And we're talking not only about customers who can shop anytime and anywhere but also about you! After all, you, too, have a chance to manage your business on the go (literally!).

#5. Tet-a-tet with the client

The store manager rarely interacts with his customers directly, in a confidential manner. Thanks to the app for an e-commerce store, such friendly communication becomes possible, which leads us to the next factor, namely...

#6. Increased customer loyalty

So, you communicate with your clients in a confidential and friendly way using your mobile shopping service. This means you get to know them better. You understand what to offer them and how to interest them.

In other words, you've managed to build customers loyalty.

#7. Informing consumers about company news

If, when building an e-commerce app, you develop a push notification feature, you'll receive another tool for communicating with customers. You'll be able to inform them of your news, promote discounts and loyalty programs, and so on... simply put, you start reminding clients of yourself (or, rather, of your store).

#8. Better promotion

Your mobile app solution is also an advertising platform; what's more, you've already purchased it (sounds cool, right?). And you're free to use it the way you like! So why not promote your business with it?

Let's say you can send sale notifications to the users passing by your store (the geolocation feature might help you!), thereby increasing shopping activity.

#9. Business growth

The mobile commerce market offers you an additional opportunity to develop and improve your company, and it'd be foolish to miss your chance to grow! This is an undeniable advantage for e-commerce businesses, so don't ignore it.

#10. Cutting costs

Completing the description of the e-commerce app development benefits, we'd like to add that the mobile platform allows you to reduce the cost of doing business. Working through a digital platform means you have a smaller staff and can give up your renting space (say, if you choose to work on a dropshipping model).

3.2 INSTALL FLUTTER

System requirements

To install and run Flutter, your development environment must meet these minimum requirements:

- Operating Systems: Windows 7 SP1 or later (64-bit), x86-64 based.
- Disk Space: 1.64 GB (does not include disk space for IDE/tools).
- Tools: Flutter depends on these tools being available in your environment.
 - Windows PowerShell 5.0 or newer (this is pre-installed with Windows 10)
 - Git for Windows 2.x, with the Use Git from the Windows Command Prompt option.

If Git for Windows is already installed, make sure you can run git commands from the command prompt or PowerShell.

Get the Flutter SDK

1. Download the following installation bundle to get the latest stable release of the Flutter SDK:

For other release channels, and older builds, see the SDK releases page.

2. Extract the zip file and place the contained flutter in the desired installation location for the Flutter SDK (for example, C:\Users\<your-user-name>\Documents).

Warning: Do not install Flutter in a directory like C:\Program Files\ that requires elevated privileges.

If you don't want to install a fixed version of the installation bundle, you can skip steps 1 and 2. Instead, get the source code from the Flutter repo on GitHub, and change branches or tags as needed. For example:

content_copy

```
C:\src>git clone https://github.com/flutter/flutter.git -b stable
```

You are now ready to run Flutter commands in the Flutter Console.

Update your path

If you wish to run Flutter commands in the regular Windows console, take these steps to add Flutter to the PATH environment variable:

- From the Start search bar, enter 'env' and select Edit environment variables for your account.
- Under User variables check if there is an entry called Path:
 - If the entry exists, append the full path to flutter\bin using ; as a separator from existing values.
 - If the entry doesn't exist, create a new user variable named Path with the full path to flutter\bin as its value.

Android setup

Note: Flutter relies on a full installation of Android Studio to supply its Android platform dependencies. However, you can write your Flutter apps in a number of editors; a later step discusses that.

Install Android Studio

1. Download and install Android Studio.
2. Start Android Studio, and go through the 'Android Studio Setup Wizard'. This installs the latest Android SDK, Android SDK Command-line Tools, and Android SDK Build-Tools, which are required by Flutter when developing for Android.
3. Run `flutter doctor` to confirm that Flutter has located your installation of Android Studio. If Flutter cannot locate it, run `flutter config --android-studio-dir <directory>` to set the directory that Android Studio is installed to.

Set up your Android device

To prepare to run and test your Flutter app on an Android device, you need an Android device running Android 4.1 (API level 16) or higher.

1. Enable Developer options and USB debugging on your device. Detailed instructions are available in the Android documentation.
2. Windows-only: Install the Google USB Driver.
3. Using a USB cable, plug your phone into your computer. If prompted on your device, authorize your computer to access your device.
4. In the terminal, run the flutter devices command to verify that Flutter recognizes your connected Android device. By default, Flutter uses the version of the Android SDK where your adb tool is based. If you want Flutter to use a different installation of the Android SDK, you must set the ANDROID_SDK_ROOT environment variable to that installation directory.

Set up the Android emulator

To prepare to run and test your Flutter app on the Android emulator, follow these steps:

1. Enable VM acceleration on your machine.
2. Launch Android Studio, click the AVD Manager icon, and select Create Virtual Device...
 - In older versions of Android Studio, you should instead launch Android Studio > Tools > Android > AVD Manager and select Create Virtual Device.... (The Android submenu is only present when inside an Android project.)
 - If you do not have a project open, you can choose Configure > AVD Manager and select Create Virtual Device...
3. Choose a device definition and select Next.

4. Select one or more system images for the Android versions you want to emulate, and select Next. An x86 or x86_64 image is recommended.
5. Under Emulated Performance, select Hardware - GLES 2.0 to enable hardware acceleration.
6. Verify the AVD configuration is correct, and select Finish.

For details on the above steps, see Managing AVDs.

7. In Android Virtual Device Manager, click Run in the toolbar. The emulator starts up and displays the default canvas for your selected OS version and device.

Agree to Android Licenses

Before you can use Flutter, you must agree to the licenses of the Android SDK platform. This step should be done after you have installed the tools listed above.

Make sure that you have a version of Java 8 installed and that your JAVA_HOME environment variable is set to the JDK's folder.

Android Studio versions 2.2 and higher come with a JDK, so this should already be done.

3.3 SOURCE CODE

DART.DART

```
import 'package:shopping_app/models/categorie_model.dart';
import 'package:shopping_app/models/product_model.dart';
import
'package:shopping_app/models/trending_productmodel.dart';
List<TrendingProductModel> getTrendingProducts() {
List<TrendingProductModel> trendingProducts = new
```

```

List<TrendingProductModel>(); TrendingProductModel
productModel = new TrendingProductModel(); //1
productModel.storename = "Store Name";
productModel.productName = "Product"; productModel.noOfRating
= 1; productModel.rating = 4; productModel.priceInDollars = 75;
trendingProducts.add(productModel); productModel = new
TrendingProductModel(); //2 productModel.storename = "Store
Name"; productModel.productName = "Product";
productModel.noOfRating = 3; productModel.rating = 4;
productModel.priceInDollars = 30;
trendingProducts.add(productModel); productModel = new
TrendingProductModel(); //3 productModel.storename = "Store
Name"; productModel.productName = "Product";
productModel.noOfRating = 3; productModel.rating = 4;
productModel.priceInDollars = 30;
trendingProducts.add(productModel); productModel = new
TrendingProductModel(); //4 productModel.storename = "Store
Name"; productModel.productName = "Product";
productModel.noOfRating = 301; productModel.rating = 4;
productModel.priceInDollars = 30;
trendingProducts.add(productModel); productModel = new
TrendingProductModel(); return trendingProducts; }
List<ProductModel> getProducts(){ List<ProductModel> products =
new List(); ProductModel productModel = new ProductModel(); //1
productModel.productName = "Special gift card";
productModel.noOfRating = 4; productModel.imgUrl = "";
productModel.rating = 4; productModel.priceInDollars = 20;
products.add(productModel); productModel = new ProductModel();
//1 productModel.productName = "Special gift card";
productModel.noOfRating = 4; productModel.imgUrl = "";
productModel.rating = 4; productModel.priceInDollars = 20;
products.add(productModel); productModel = new ProductModel();
//1 productModel.productName = "Special gift card";
productModel.noOfRating = 4; productModel.imgUrl = "";

```

```

productModel.rating = 4; productModel.priceInDollars = 20;
products.add(productModel); productModel = new ProductModel();
//1 productModel.productName = "Special gift card";
productModel.noOfRating = 4; productModel.imgUrl = "";
productModel.rating = 4; productModel.priceInDollars = 20;
products.add(productModel); productModel = new ProductModel();
//1 productModel.productName = "Special gift card";
productModel.noOfRating = 4; productModel.imgUrl = "";
productModel.rating = 4; productModel.priceInDollars = 20;
products.add(productModel); productModel = new ProductModel();
//1 productModel.productName = "Special gift card";
productModel.noOfRating = 4; productModel.imgUrl = "";
productModel.rating = 4; productModel.priceInDollars = 57;
products.add(productModel); productModel = new ProductModel();
return products; } List<CategorieModel> getCategories(){
List<CategorieModel> categories = new List(); CategorieModel
categorieModel = new CategorieModel(); //1
categorieModel.categorieName = "Regular Gift";
categorieModel.color1 = "0xff8EA2FF"; categorieModel.color2 =
"0xff557AC7"; categorieModel.imgAssetPath =
"assets/categorie.png"; categories.add(categorieModel);
categorieModel = new CategorieModel(); //2
categorieModel.categorieName = "Box Gift"; categorieModel.color1
= "0xff50F9B4"; categorieModel.color2 = "0xff38CAE9";
categorieModel.imgAssetPath = "assets/boxgift.png";
categories.add(categorieModel); categorieModel = new
CategorieModel(); //3 categorieModel.categorieName =
"Chocolate"; categorieModel.color1 = "0xffFFB397";
categorieModel.color2 = "0xffF46AA0";
categorieModel.imgAssetPath = "assets/choclade.png";
categories.add(categorieModel); categorieModel = new
CategorieModel(); //4 categorieModel.categorieName = "Gift with
card"; categorieModel.color1 = "0xff8EA2FF"; categorieModel.color2
= "0xff557AC7"; categorieModel.imgAssetPath =

```

```

"assets/categorie.png"; categories.add(categorieModel);
categorieModel = new CategorieModel(); //5
categorieModel.categorieName = "Regular Gift";
categorieModel.color1 = "0xff8EA2FF"; categorieModel.color2 =
"0xff557AC7"; categorieModel.imgAssetPath =
"assets/categorie.png"; categories.add(categorieModel);
categorieModel = new CategorieModel(); return categories;
}

```

HOME.DART

```

import 'package:flutter/material.dart';
import 'package:shopping_app/main.dart';

class Home extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SingleChildScrollView(
        child: Container(
          padding: EdgeInsets.symmetric(horizontal: 24),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: <Widget>[
              SizedBox(height: 60,),
              SizedBox(height: 4,),
              Text("Shopping App\nBuy everthing", style: TextStyle(
                color: Colors.black87.withOpacity(0.7),
                fontWeight: FontWeight.w500,
                fontSize: 28),),
              SizedBox(height: 40,),
              Image.asset("assets/illustration.png"),
              SizedBox(height: 40,),
              Container(
                width: MediaQuery.of(context).size.width,

```

```

child: Column(
  children: <Widget>[
    GestureDetector(
      onTap: (){
        Navigator.push(context, MaterialPageRoute(
          builder: (context) => MyHomePage()
        ));
      },
      child: Container(
        width: MediaQuery.of(context).size.width - 70,
        alignment: Alignment.center,
        padding: EdgeInsets.symmetric(horizontal: 24,vertical:
17),
        decoration: BoxDecoration(
          gradient: LinearGradient(
            colors: [const Color(0xff8EA2FF), const
Color(0xff557AC7)]
          ),
          boxShadow: <BoxShadow>[
            BoxShadow(
              offset: Offset(5.0, 5.0),
              blurRadius: 15.0,
              color: Color(0xff8EA2FF).withOpacity(0.5),
            ),
          ],
          borderRadius: BorderRadius.circular(6)
        ),
        child: Text("Explore Shop".toUpperCase(), style:
TextStyle(
          color: Colors.white
        )),
      ),
    ),
    SizedBox(height: 30,),

```

```

        Text("Learn More", style: TextStyle(
            color: Color(0xff8EA2FF),
            fontSize: 17
        )),
    ],
),
)
],
),
),
),
);
}
}

```

MAIN.DART

```

import 'package:flutter/material.dart';
import 'package:flutter/material.dart';
import 'package:shopping_app/data/data.dart';
import 'package:shopping_app/models/categorie_model.dart';
import 'package:shopping_app/models/product_model.dart';
import
'package:shopping_app/models/trending_productmodel.dart';
import 'package:shopping_app/resources/colors.dart';
import 'package:shopping_app/views/home.dart';

void main() => runApp(MyApp());
class MyApp extends StatelessWidget{
  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',

```

```

        debugShowCheckedModeBanner: false,
        theme: ThemeData(
          primarySwatch: Colors.blue,
        ),
        home: MyHomePage(),
      );
    }
  }
class MyHomePage extends StatefulWidget {
  @override
  _MyHomePageState createState() => _MyHomePageState();
}
class _MyHomePageState extends State<MyHomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SingleChildScrollView(
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: <Widget>[
            SizedBox(height: 50,),
            // here we will add our containers
          ],
        ),
      ),
    );
  }
}

```

BESTSELLING.DART

```

class ProductTile extends StatelessWidget {
  final int priceInDollars;
  final String productName;
  final int rating;

```

```

final String imgUrl;
final int noOfRating;
ProductTile({this.priceInDollars, this.imgUrl,
this.rating,this.productName, this.noOfRating});
@override
Widget build(BuildContext context) {
  return Container(
    margin: EdgeInsets.only(right: 16),
    child: Column(
      children: <Widget>[
        Container(
          child: Stack(
            children: <Widget>[
              Image.asset("assets/productImage.png",height: 150,fit:
BoxFit.cover,) ,
              Container(
                height: 25,
                width: 45,
                margin: EdgeInsets.only(left: 8,top: 8),
                alignment: Alignment.center,
                decoration: BoxDecoration(
                  borderRadius: BorderRadius.circular(6) ,
                  gradient: LinearGradient(
                    colors: [const Color(0xff8EA2FF).withOpacity(0.5),
const Color(0xff557AC7).withOpacity(0.5)]
                )
              ),
            ],
          child: Text("\$$priceInDollars",style: TextStyle(
            color: Colors.white
          ),
        ),
      ],
    ),
  ),

```

```

    ),
    Text(productName),
    SizedBox(height: 8,),
    Row(
      children: <Widget>[
        StarRating(rating: rating,),
        SizedBox(width: 10,),
        Text("($noOfRating)", style: TextStyle(
          color: textGrey,
          fontSize: 12
        )),
      ],
    ),
  ],
),
],
),
);
}
}

```

TRENDINGPRODUCT.DART

```

class TrendingTile extends StatelessWidget {
  final String productName;
  final String storename;
  final String imgUrl;
  final int noOfRating;
  final int priceInDollars;
  final int rating;
  TrendingTile({this.productName,this.storename,this.imgUrl,this.rati
ng,this.noOfRating,this.priceInDollars});
  @override
  Widget build(BuildContext context) {
    return Container(
      width: MediaQuery.of(context).size.width - 70,

```

```

margin: EdgeInsets.only(right: 13),
decoration: BoxDecoration(
  color: Colors.white,
  borderRadius: BorderRadius.circular(4),
  boxShadow: <BoxShadow>[
    BoxShadow(
      offset: Offset(1.0, 1.0),
      blurRadius: 15.0,
      color: Colors.black87.withOpacity(0.05),
    ),
  ],
),
child: Row(
  children: <Widget>[
    Container(
      child: Stack(
        children: <Widget>[
          Image.asset("assets/image.png",height: 150,fit:
BoxFit.cover,) ,
          Container(
            height: 25,
            width: 50,
            margin: EdgeInsets.only(left: 5,top: 5),
            alignment: Alignment.center,
            decoration: BoxDecoration(
              borderRadius: BorderRadius.circular(6) ,
              gradient: LinearGradient(
                colors: [const Color(0xff8EA2FF).withOpacity(0.5),
const Color(0xff557AC7).withOpacity(0.5)]
              )
            ),
            child: Text("\$$priceInDollars",style: TextStyle(
              color: Colors.white
            ),

```

```

    ),
  ),
],
),
),
Container(
  padding: EdgeInsets.symmetric(horizontal: 14,vertical: 12),
  child: Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: <Widget>[
      Text(productName, style: TextStyle(
        color: Colors.black87,
        fontSize: 19
      )),
      Text(storename, style: TextStyle(
        color: textGrey
      )),
      SizedBox(height: 8,),
      Row(
        children: <Widget>[
          StarRating(rating: rating,),
          SizedBox(width: 3,),
          Text("($noOfRating)", style: TextStyle(
            color: textGrey,
            fontSize: 12
          )),
        ],
      ),
      SizedBox(height: 13,),
      Container(
        height: 30,
        padding: EdgeInsets.symmetric(horizontal:
12),alignment: Alignment.center,
        decoration: BoxDecoration(

```

```

        borderRadius: BorderRadius.circular(4) ,
        gradient: LinearGradient(
          colors: [const Color(0xff8EA2FF), const
Color(0xff557AC7)]
        )
      ),
      child: Text(
        "Add to cart", style: TextStyle(
          color: Colors.white
        ),
      ),
    ),
  ],
),
)
],
),
),
);
}
}

```

CATEGORI_MODEL.DART

```

class CategorieModel{
  String categorieName; // name of the categorie
  String imgAssetPath; // url of the image
  String color1;
  String color2;

  CategorieModel({this.categorieName,this.imgAssetPath,this.color1,t
his.color2});
}

```

PRODUCT_MODEL.DART

```
class ProductModel{
  int priceInDollars; // price
  String productName; // name of the product
  int rating; // rating
  String imgUrl; // product image url
  int noOfRating; // number of rating
  ProductModel({this.productName,this.imgUrl,this.priceInDollars,thi
s.rating,this.noOfRating});
}
```

TRENDING_PRODUCT_MODEL

```
class TrendingProductModel{
  String productName;
  String storename;
  String imgUrl;
  int noOfRating;
  int priceInDollars;
  int rating;
  TrendingProductModel({this.imgUrl,this.noOfRating,this
.priceInDollars,this.productName,this.storename, this.rating});
}
```

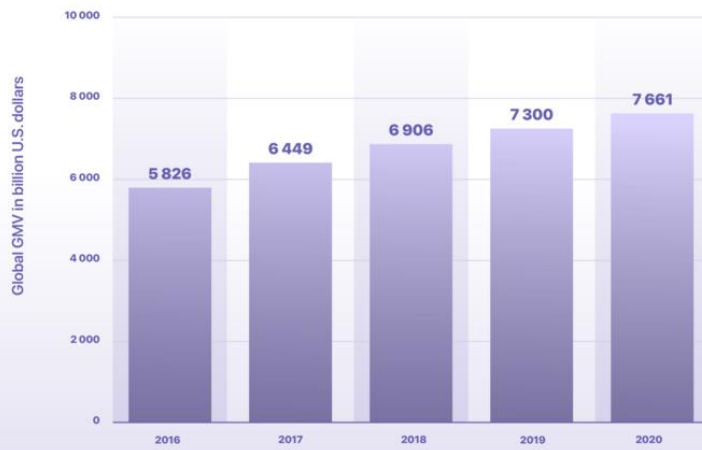
CHAPTER 4

RESULT AND DISCUSSION

4.1 OUTPUT SCREESHOTS

4.2 PERFORMACE ANALYSICS

Global B2B e-commerce gross merchandise volume (GMV) from 2016 to 2020 (in billion U.S. dollars)



World`s largest B2C e-commerce markets

B2C e-commerce turnover in 2019 & 2020 (in billion U.S. dollars)



CHAPTER 5

CONCLUSION

5.1 CONCLUSION

E-Commerce has undeniably become an important part of our society. The successful companies of the future will be those that take E-Commerce seriously, dedicating sufficient resources to its development. E-Commerce is not an IT issue but a whole business undertaking. Companies that use it as a reason for completely re-designing their business processes are likely to reap the greatest benefits. Moreover, E-Commerce is a helpful technology that gives the consumer access to business and companies all over the world.

5.2 REFERENCE

- <https://www.javatpoint.com/flutter>
- <https://flutter.dev/>
- <https://dart.dev/>