



Project Practice Report

For Mobile Programming

**By:**

王佳佳

**Full name:**

Patricia Fiona

2021-12

# Table of Contents

[Table of Contents ii](#_Toc90932266)

[Table of Figures iv](#_Toc90932267)

[Table of Tables vi](#_Toc90932268)

[Chapter I 1](#_Toc90932269)

[1.1 Background 1](#_Toc90932270)

[1.2 Identification of Problem 1](#_Toc90932271)

[1.3 Limitation of the Problem 2](#_Toc90932272)

[1.4 The Purpose of the Research 2](#_Toc90932273)

[1.5 The Significance of the Research 3](#_Toc90932274)

[Chapter II 4](#_Toc90932275)

[2.1 Research Stages 4](#_Toc90932276)

[2.2 Application Development Stages 5](#_Toc90932277)

[2.3 Business Modelling 6](#_Toc90932278)

[2.3.1 Use Case Diagram 6](#_Toc90932279)

[2.3.2 Use Case Detail 7](#_Toc90932280)

[2.3.3 Activity Diagram 9](#_Toc90932281)

[2.4 System Requirements Analysis 24](#_Toc90932282)

[2.5 Analysis and Design 25](#_Toc90932283)

[2.5.1 Class Diagram 25](#_Toc90932284)

[2.5.2 User Interface Design 25](#_Toc90932285)

[2.6 Implementation 33](#_Toc90932286)

[2.6.1 Jetpack Component & Architecture 33](#_Toc90932287)

[2.6.2 Repository Pattern 33](#_Toc90932288)

[Chapter III 34](#_Toc90932289)

[Chapter IV 35](#_Toc90932290)

[References 36](#_Toc90932291)

[Appendix 37](#_Toc90932292)

# Table of Figures

[**Figure 1** Software Development 4](#_Toc90978914)

[**Figure 2** Use Case for Nike Application 6](#_Toc90978915)

[**Figure 3** See Product By Category – Home Activity Diagram 9](#_Toc90978916)

[**Figure 4** See All Product By Category Activity Diagram 10](#_Toc90978917)

[**Figure 5** See Product Details Activity Diagram 11](#_Toc90978918)

[**Figure 6** See Product Image Detail Activity Diagram 12](#_Toc90978919)

[**Figure 7** See Product By Color Option Activity Diagram 12](#_Toc90978920)

[**Figure 8** Add-Remove Product to Favorite List Activity Diagram 13](#_Toc90978921)

[**Figure 9** Add Product to Shopping Cart Activity Diagram 14](#_Toc90978922)

[**Figure 10** Update Product in Shopping Cart – Shopping Cart Page Activity Diagram 14](#_Toc90978923)

[**Figure 11** Update Product in Shopping Cart – Product Detail Page Activity Diagram 15](#_Toc90978924)

[**Figure 12** Remove Product from Shopping Cart Activity Diagram 16](#_Toc90978925)

[**Figure 13** See Favorite Products Activity Diagram 16](#_Toc90978926)

[**Figure 14** See Shopping Cart Activity Diagram 17](#_Toc90978927)

[**Figure 15** Search Product Activity Diagram 18](#_Toc90978928)

[**Figure 16** Sign Up Account Activity Diagram 19](#_Toc90978929)

[**Figure 17** Login Activity Diagram 20](#_Toc90978930)

[**Figure 18** Edit User Data Activity Diagram 21](#_Toc90978931)

[**Figure 19** Delete User Account Activity Diagram 22](#_Toc90978932)

[**Figure 20** Checkout Activity Diagram 23](#_Toc90978933)

[**Figure 21** Class Diagram for NIKE Database 25](#_Toc90978934)

[**Figure 22** Overall Design For Nike Apps 26](#_Toc90978935)

[**Figure 23** Splash Screen Design 26](#_Toc90978936)

[**Figure 24** Welcome Scree for New User Page 27](#_Toc90978937)

[**Figure 25** Home Screen Design 27](#_Toc90978938)

[**Figure 26** Search Page Design 28](#_Toc90978939)

[**Figure 27** Favorite Page – Not Empty Design 28](#_Toc90978940)

[**Figure 28** Favorite Page – Empty Design 29](#_Toc90978941)

[**Figure 29** Shopping Cart – Not Empty Design 29](#_Toc90978942)

[**Figure 30** Shopping Cart Page – Empty Design 30](#_Toc90978943)

[**Figure 31** Product Detail Page Design 30](#_Toc90978944)

[**Figure 32** Product Image Preview Page Design 31](#_Toc90978945)

[**Figure 33** Login and Signup Page Design 31](#_Toc90978946)

[**Figure 34** User Profile Page Design 32](#_Toc90978947)

[**Figure 35** User Edit Data Page Design 32](#_Toc90978948)

[**Figure 36** Android Jetpack Component Illustration 33](#_Toc90978949)

[**Figure 37** Beyond Phone Category in Jetpack Components 34](#_Toc90978950)

[**Figure 38** Data Category in Jetpack Components 34](#_Toc90978951)

[**Figure 39** Graphic Category in Jetpack Components 34](#_Toc90978952)

[**Figure 40** Lifecycle Category in Jetpack Components 35](#_Toc90978953)

[**Figure 41** Media Category in Jetpack Components 35](#_Toc90978954)

[**Figure 42** Navigation Category in Jetpack Components 35](#_Toc90978955)

[**Figure 43** Security Category in Jetpack Components 35](#_Toc90978956)

[**Figure 44** Performance/Test Category in Jetpack Components 36](#_Toc90978957)

[**Figure 45** UI Category in Jetpack Components 36](#_Toc90978958)

[**Figure 46** Library import in the build.gradle 38](#_Toc90978959)

[**Figure 47** MVVM Architecture Model 38](#_Toc90978960)

[**Figure 48** The Directory Structure that Store Each Code Based on Their Function 39](#_Toc90978961)

[**Figure 49** Some View Model Location That Related to Each View 40](#_Toc90978962)

[**Figure 50** Repository Pattern Architecture Illustration 41](#_Toc90978963)

[**Figure 51** Implementation of Repository Pattern in Data Package 42](#_Toc90978964)

[**Figure 52** Code Implementation of Dependency Injection for Provie Nike Repository 43](#_Toc90978965)

[**Figure 53** One Example of Application of Idling Resource with Espresso in the getSearchResult() 44](#_Toc90978966)

[**Figure 54** View Binding Code in build.gradle(:app) 44](#_Toc90978967)

[**Figure 55** View Binding Implementation in UserEditActivity() 45](#_Toc90978968)

# Table of Tables

[***Table 1*** *Customer Description* 7](#_Toc90932226)

[***Table 2*** *Interactions that can be done by the customer* 7](#_Toc90932227)

# Chapter I

**Introduction**

## Background

Shopping is an important aspect of everyday life. By shopping, customers get the goods or services they want. In addition, shopping can improve the economy of a region or country. This is because by shopping we have carried out a buying and selling process where this process can trigger the circulation of money.

With the Covid-19 pandemic, many business owners are required to think about new solutions and breakthroughs so that their business can continue to run like normal or even better than before. One of the main problems they have to solve is how customers can access the goods or services they sell without having to communicate and transact offline or in traditional way.

The presence of an online store in today’s life can be solution that can overcome various transaction difficulties in times like today. This is one from various ways to normalize the transaction process in society. Supported by the rapid development of internet and technology, it is expected to have a positive impact on new business and existing businesses.

Inspired by the Nike products, in this research I using this company and their product to build the shoes online shop platform that has been developed in Android platform using Kotlin programming language. The use of Kotlin as a programming language in this development of this application is based on a simple, concise, and easier-to-understand syntax. Apart from that, Kotlin is robust statically type and much less verbose than Java. Many programming features are available in Kotlin and are very helpful for programmers [1]. All views in the application will be redesigned to the UI that I have designed to the prototype in the Figma application. In the backend side, I used J2EE as the backend services and MYSQL as database.

## Identification of Problem

The formula of the problems if this research is as follows:

1. Developing Online Shop Case Study Nike in Android Platform with Kotlin Programming Language for show product to customers
2. Developing Online Shop Case Study Nike in Android Platform with Kotlin Programming Language in Android-based platform
3. Developing Online Shop Case Study Nike in Android Platform with Kotlin Programming Language for discover and explore product in the store
4. Developing Online Shop Case Study Nike in Android Platform with Kotlin Programming Language for adding-removing product to the favorite list
5. Developing Online Shop Case Study Nike in Android Platform with Kotlin Programming Language for adding-removing product to the shopping cart
6. Developing Online Shop Case Study Nike in Android Platform with Kotlin Programming Language for adding-removing shopping product in cart to the transaction data

## Limitation of the Problem

The problem limitations of this study are as follows:

1. This research discusses the development of Nike online shop for Android platform using Kotlin language
2. The scope of application users is for users in Indonesia
3. This study discusses the process of viewing products in the Nike application
4. This study discusses the process of adding-removing product to user local favorite list
5. This study discusses the process of adding-removing product to user local shopping list
6. This study discusses the process of adding-removing product from user shopping cart to the transaction database and history
7. This study used RESTful API that developed in J2EE to provide data transfer from database to the application
8. This study implemented Android Jetpack components and Architecture
9. This study using Repository Pattern to developed this application
10. This study used MVVM architecture in this development
11. This study used View Binding in the android application

## The Purpose of the Research

This document describes the systems inside the application. This product is an Android-based software for customer. This application product has main functions, such as:

1. **Explore and Discover Product**

This function served to display the shoes sold in the Nike store.

1. **View Product Details**

This function shows the detail of the product that user see

1. **Add and Remove Product to/from Favorite List**

This function serves to add and remove items that customer like in the favorite list.

1. **Add and Remove Product to/from Shopping Cart**

This function serves to add and remove items that customer wants to buy later into the shopping cart list.

1. **Add and Cancel Transaction to/from Transaction Database**

This function serves to add and remove product in the shopping cart to the transaction database

## The Significance of the Research

The benefits of this research are as follows:

1. Provides example of shoes online shop using Kotlin language.
2. Provides example of developing RESTful API using J2EE and MYSQL database.
3. Provide good Android Architecture that using Jetpack Pro library and architecture
4. Provide good example of implementing repository pattern in the Android project.

# Chapter II

**Research Methodology**

## Research Stages

**Figure 1** Software Development

In this study the authors adapt software development, typically following a six-step process known as the software development life cycle (SDLC). The six steps are [2]:

1. **Program specification**

The program’s objectives, outputs, inputs, and processing requirements are determined.

1. **Program design**

A solution is created using programming techniques such as top-down program design, pseudocode, flowcharts, and logic structures.

1. **Program code**

The program is written or coded using a programming language.

1. **Program test**

The program is tested or debugged by looking for syntax and logic errors.

1. **Program documentation**

Documentation is an ongoing process throughout the programming process. This phase focuses on formalizing the written description and processes used in the program.

1. **Program maintenance**

Completed programs are periodically reviewed to evaluate their accuracy, efficiency, standardization, and ease of use. Changes are made to the program’s code as needed.

## Application Development Stages

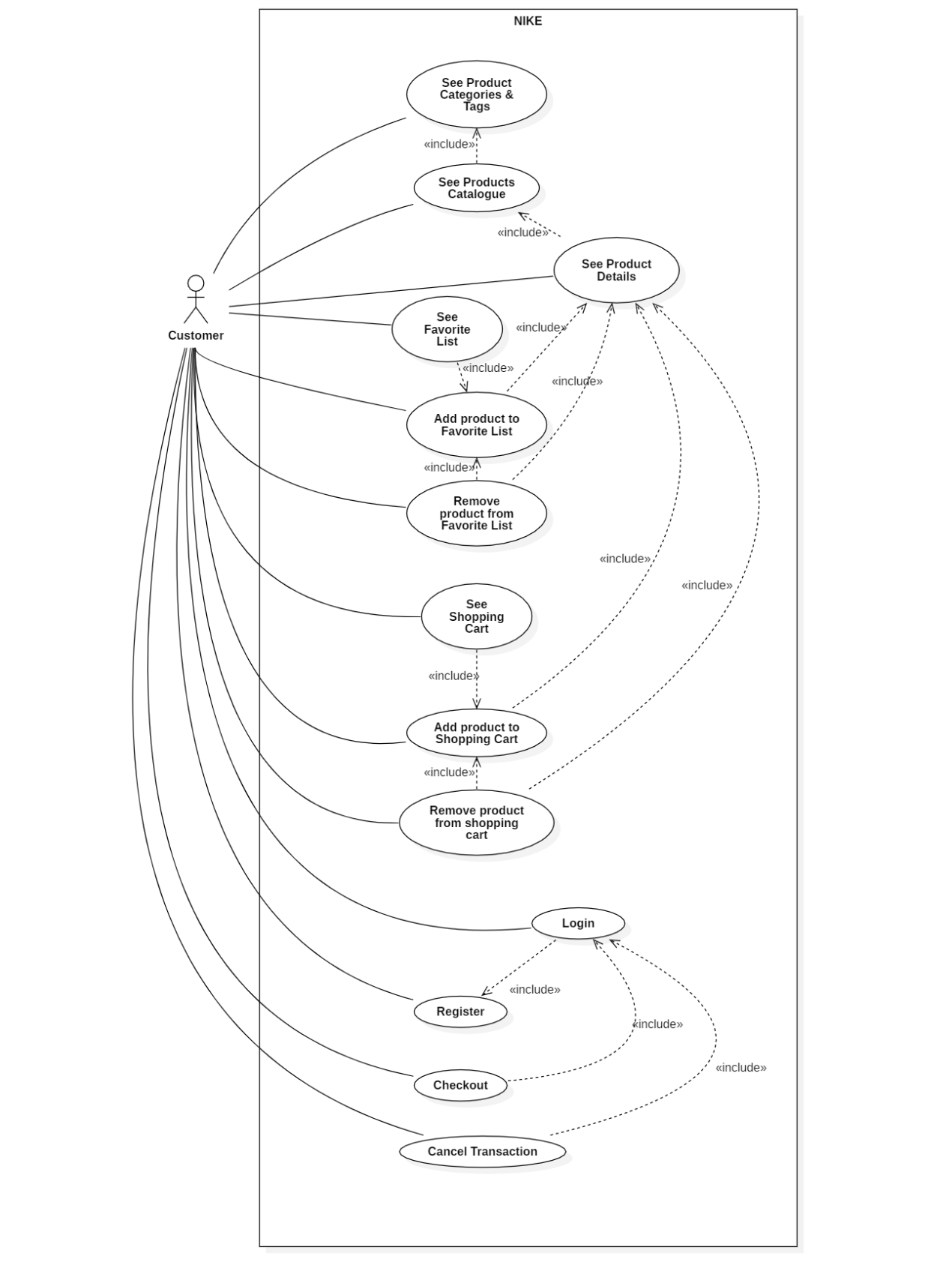
The methodology used in developing this application is to use the prototype methodology. Prototyping Model is a software development model in which prototype is built, tested, and reworked until an acceptable prototype is achieved [3]. There are several stages of the prototype that I did, including:

1. **Prototype 1** – Implement Welcome Layout, Splash Screen, and Basic Layout
2. **Prototype 2** – Finished Home Design and Integrate Data in This Page
3. **Prototype 3** – Add Favorite Page and Data Logic
4. **Prototype 4** – Add See All Product Page and Data Integration
5. **Prototype 5** – Create and Integrate Product Details in Product Details Page
6. **Prototype 6** – Adding Setting page, About page
7. **Prototype 7** – Adding Shopping Cart Page and Data Logic
8. **Prototype 8** – Adding and show Discount Price in the Product Preview
9. **Prototype 9** – Adding Sign Up, Login, Edit, and Delete User Account Feature
10. **Prototype 10** – Add Checkout & Cancel Transaction Feature (Finish V 1.0.0)

This progress of prototyping can be seen in my Github commit history with repository name ‘*Mobile-Programming-Final-Project*’ that can be access thought this link: <https://github.com/patriciafiona/Mobile-Programming-Final-Project> .

## Business Modelling

### Use Case Diagram



**Figure 2** Use Case for Nike Application

### Use Case Detail

1. **Actor Description**

The following is a definition of several actors in an Online Shop Case Study Nike With Kotlin programming language:

***Table 1*** *Customer Description*

|  |  |
| --- | --- |
| **Actor** | **Description** |
| Customer | A customer is a person who visits the Nike apps to view a list of goods as well as a person who makes a purchase transaction for goods sold on this application. |

1. **Customer Use Case Definition**

The following is a use case scenario in Nike Android-based Application with Kotlin Programming Language that will be used by the customer:

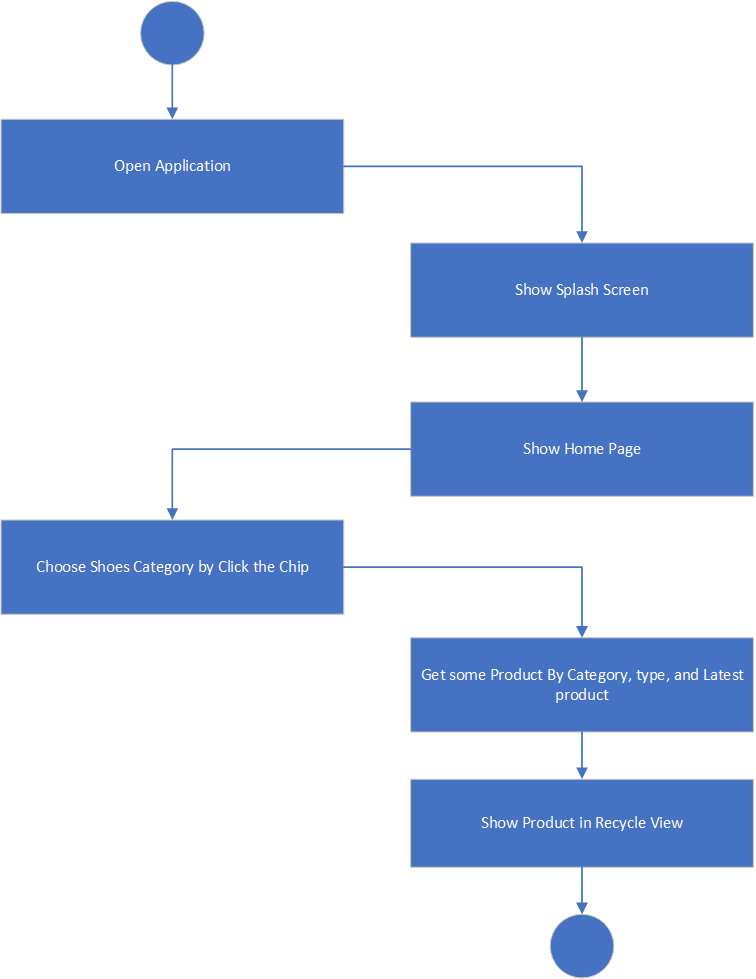
***Table 2*** *Interactions that can be done by the customer*

|  |  |  |
| --- | --- | --- |
| **Number** | **Actor** | **Description** |
| 1. | View product categories | Is process to see and select the way to show product based on the product category |
| 2. | View Products | Is a process to see the product sold by Nike |
| 3. | View Product Details | Is a process to see the detail of the product |
| 4. | Add Product to Favorite List | Is a process to add product that customer want to see later or bookmark it to the favorite list |
| 5. | Remove Product from Favorite List | Is a process to remove product from favorite list |
| 6. | See Favorite List | Is a process to see products that user want to see later in the favorite list |
| 7. | See Shopping Cart | Is a process to see products that user want to buy later in the shopping cart |
| 8. | Add Product to Shopping Cart | Is a process to add product that customer want to buy later or bookmark it to the shopping cart |
| 9. | Remove Product from Shopping Cart | Is a process to remove product from shopping cart |
| 10. | Sign Up | Is a process to add user data and register in the Nike system by using email, password, and other user data |
| 11. | Login | Is a process to get in and connect user data into the application |
| 12. | Logout | Is a process to disconnected user data into the application |
| 13. | Edit Use Account | Is a process to change user data from database |
| 14. | Delete User Account | Is a process to remove user data and information from Nike database |
| 15. | Checkout Product | Is a process to add product in the shopping cart to be process as a transaction |
| 16. | Cancel transaction in the Pending Payment Stage | Is a process to remove transaction made by user that still in the Pending Payment stage |

### Activity Diagram

The following is an activity diagram in the application of Nike Android-based application with Kotlin programming language. Where the description of the activity is explained in each actor, namely customer activity.

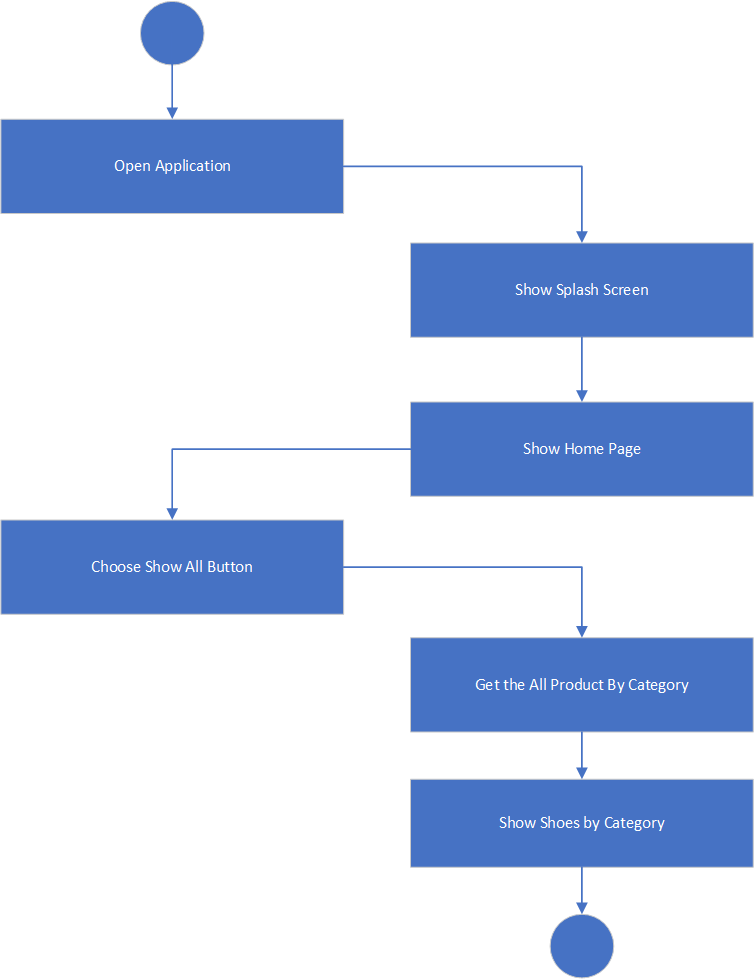
1. **System Feature: See Latest Product, Product By Category and Types – Home**



**Figure 3** See Product By Category – Home Activity Diagram

This function section is used by customer to see product by product categories in home page. By click the category chip in the home page, user can change product preview based on the selected category. Later this application will show some numbers of products for each category and for latest shoes.

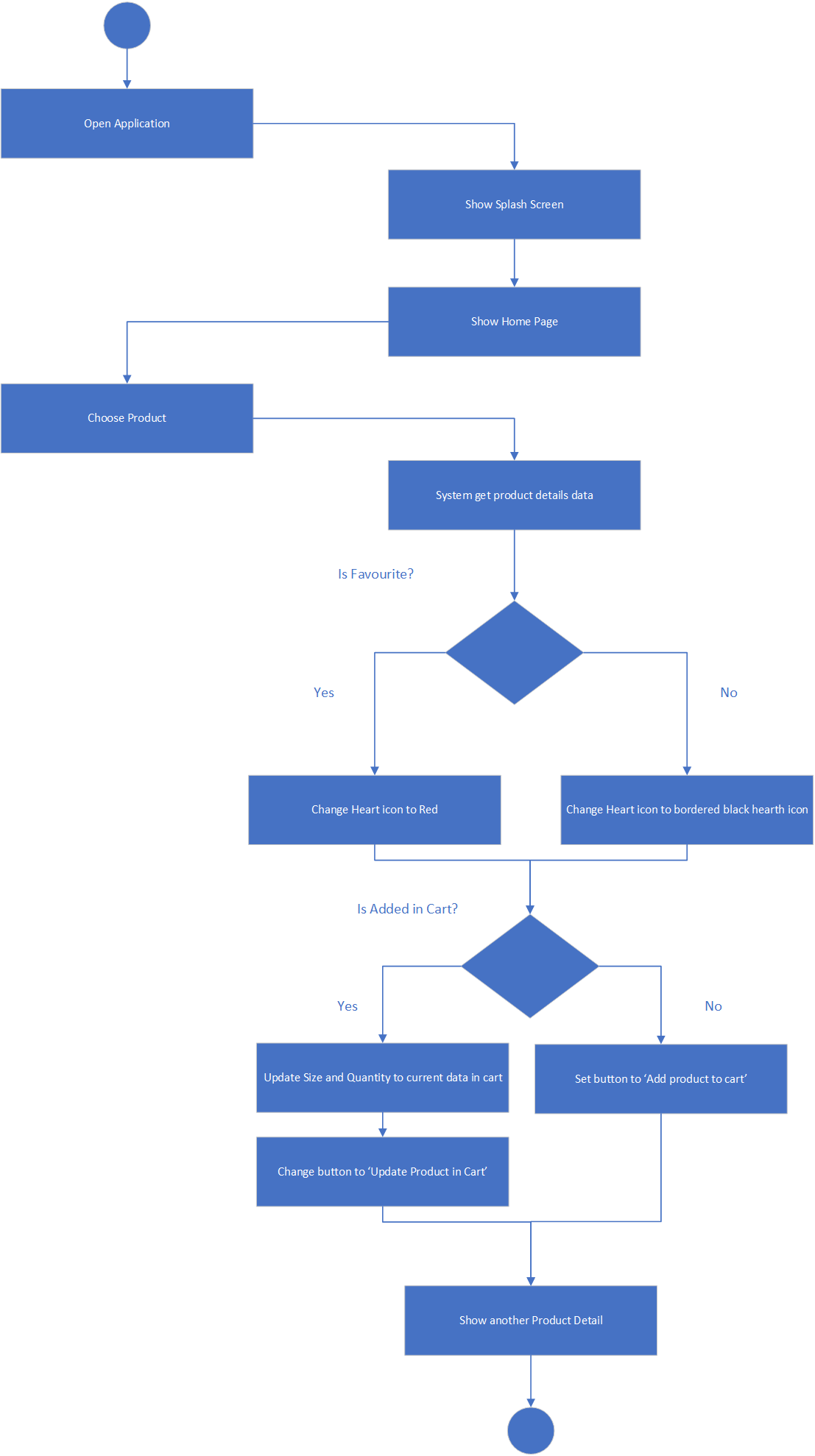
1. **System Feature: See All Product By Category**

****

**Figure 4** See All Product By Category Activity Diagram

This function section is used by customer to see all product based on their category. To see all product by category, user can click ‘See All’ button in the home page.

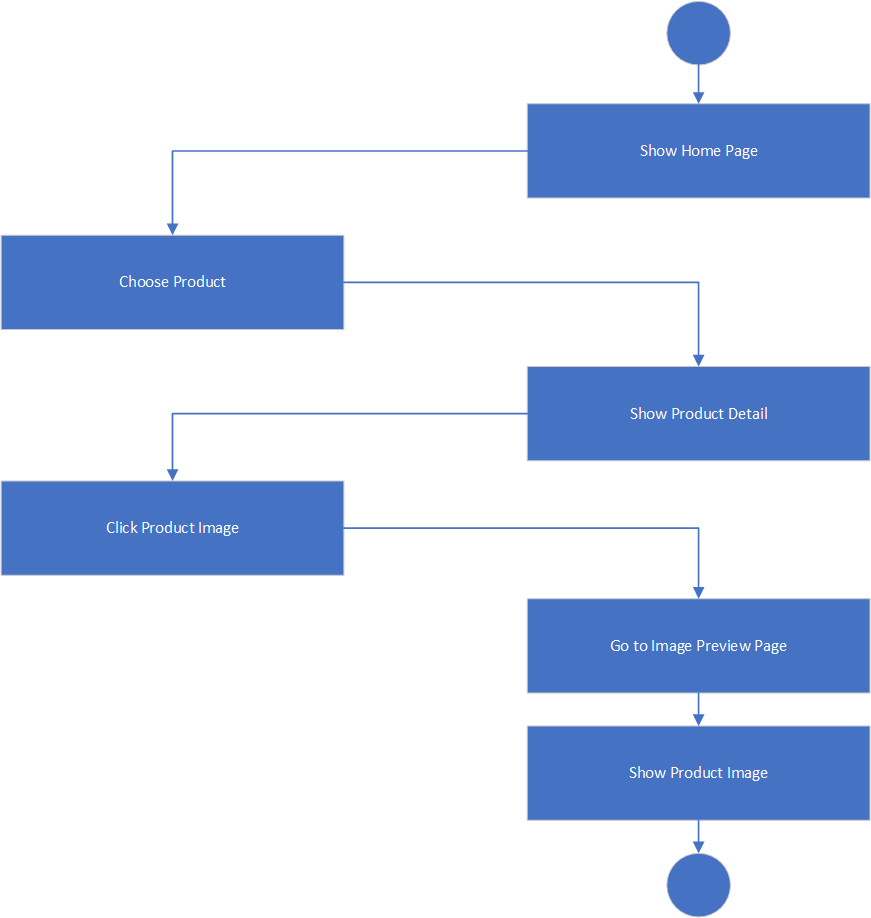
1. **System Feature: See Product Details**

****

**Figure 5** See Product Details Activity Diagram

This function section is used by customer to see product information in more details. To show the details, suer can choose the product that they want to see. Later in this page they can see the information like price, product name, product color options, product description, etc.

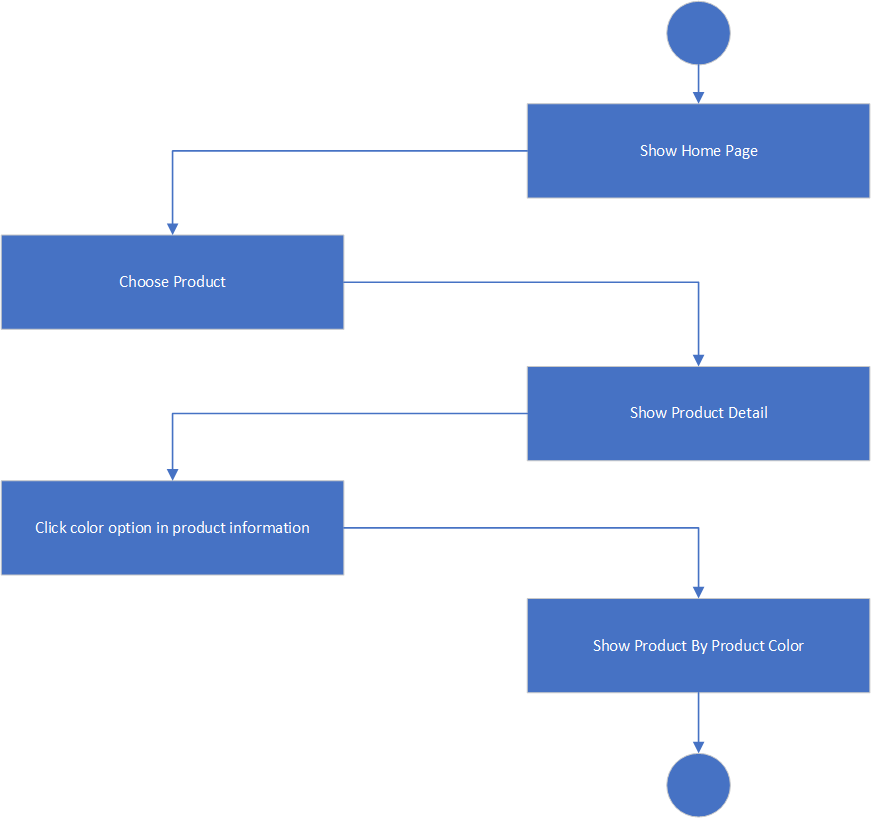
1. **System Feature: See Product Image Detail**



**Figure 6** See Product Image Detail Activity Diagram

This function section is used by customer to see product image in more detail. To see it user can click the product image in the product detail page. Later the program will show the product image that user can zoom in and zoom out.

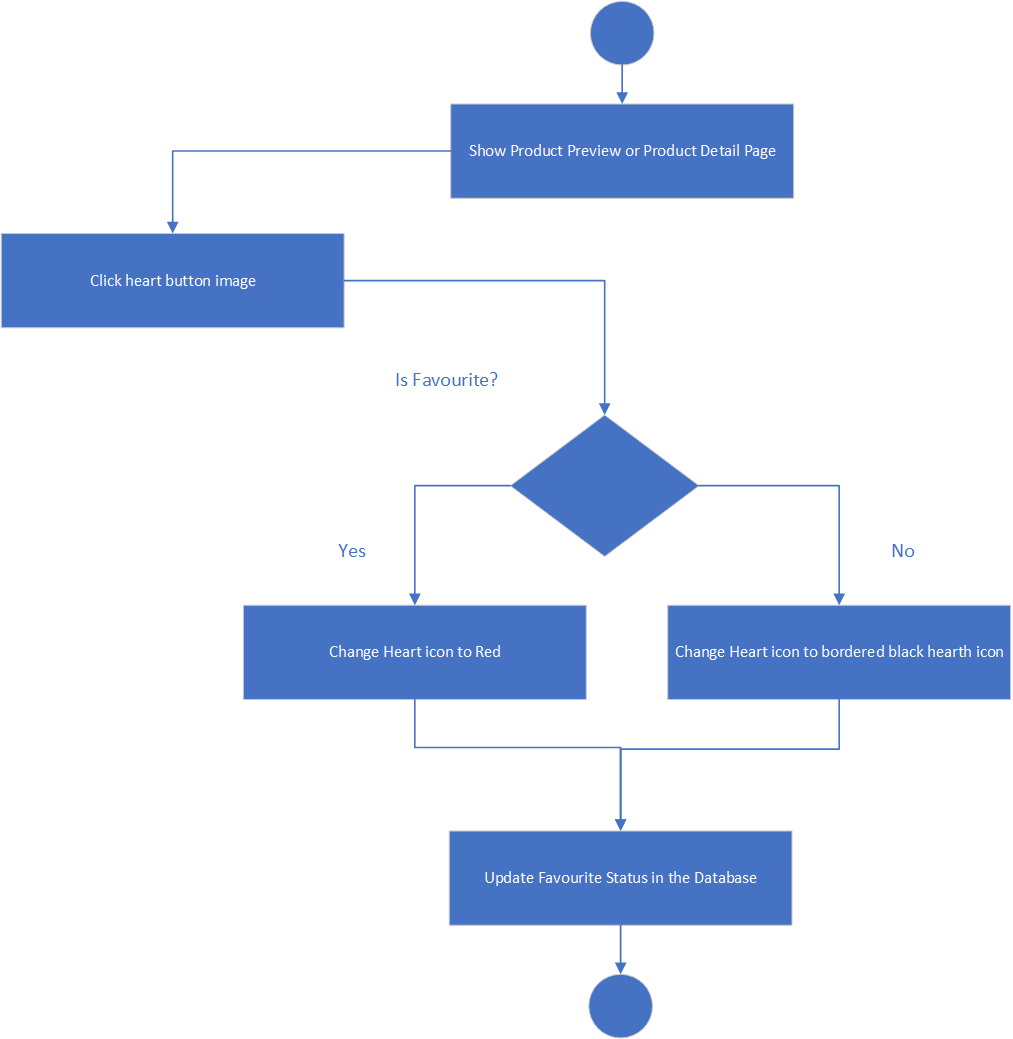
1. **System Feature: See Product By Color Option**



**Figure 7** See Product By Color Option Activity Diagram

This function section is used by customer to show product data based on their color. To see the different color of product image, user can select on of the product color in the color options.

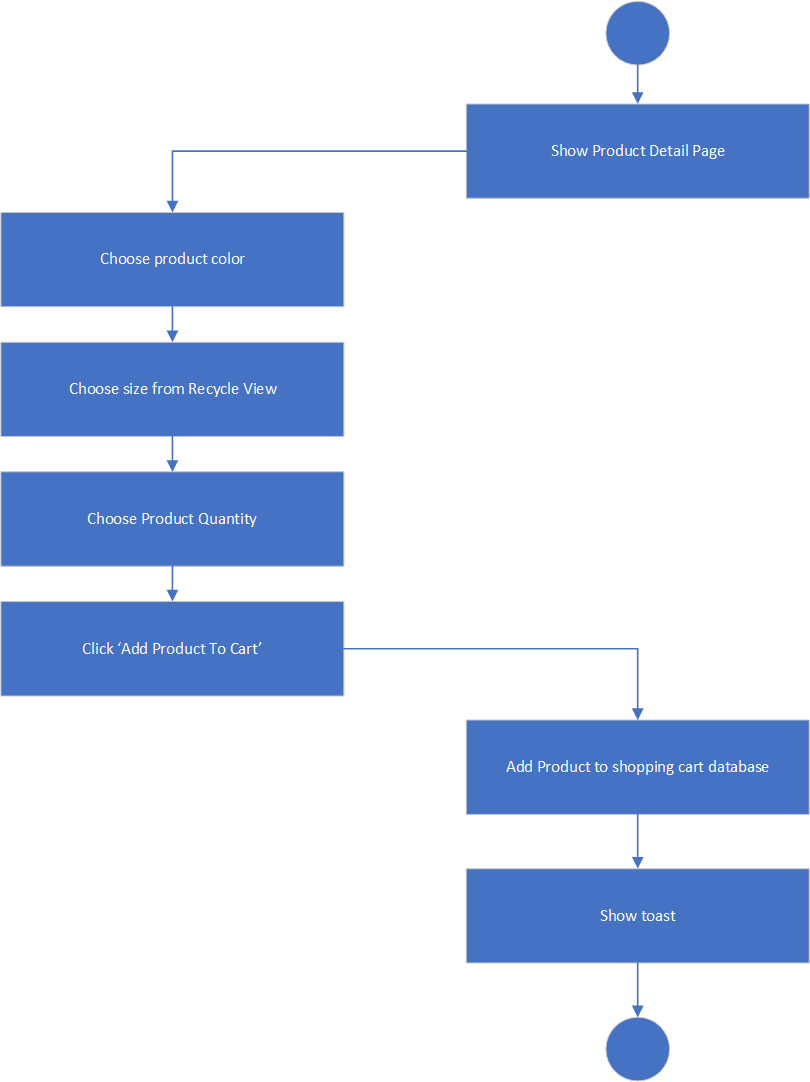
1. **System Feature: Add-Remove Product to Favorite List**

****

**Figure 8** Add-Remove Product to Favorite List Activity Diagram

This function section is used by customer to add or remove product to favorite list. To add product, user can click the bordered black hearth button icon in the product details page, or in the product preview. To remove product from list of favorite, user can click again the red heart button icon and the product will removed from database.

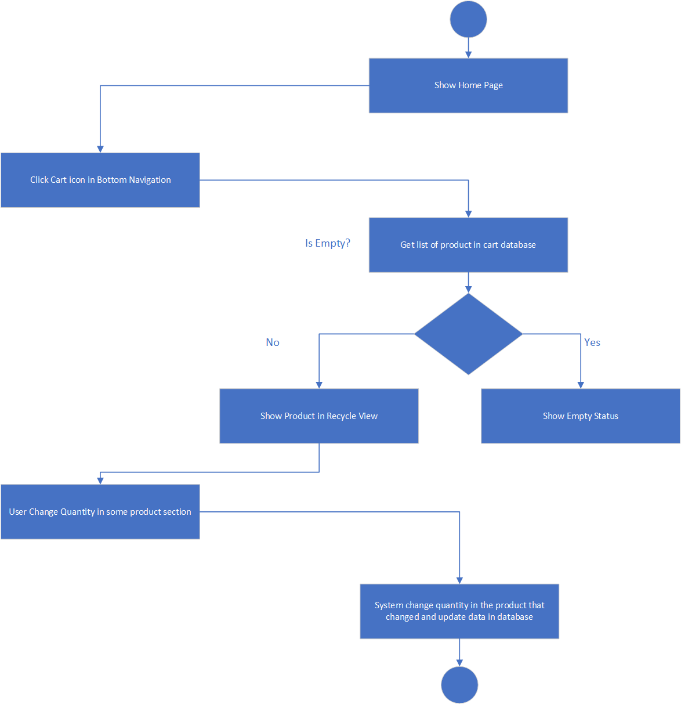
1. **System Feature: Add Product to Shopping Cart**



**Figure 9** Add Product to Shopping Cart Activity Diagram

This function section is used by customer to add product to shopping cart. To do this, user can go to product detail page, select size, color, product quantity, and click ‘Add product to Cart’ button.

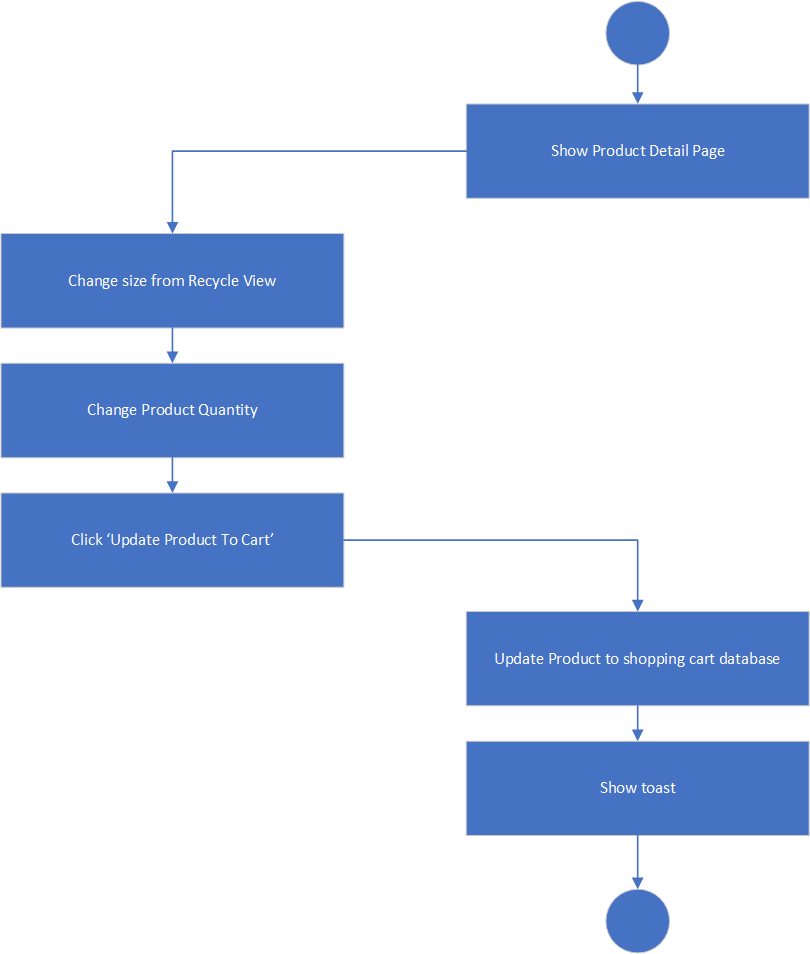
1. **System Feature: Update Product in Shopping Cart – Shopping Cart Page**



**Figure 10** Update Product in Shopping Cart – Shopping Cart Page Activity Diagram

This function section is used by customer to update product in the shopping cart through shopping cart page. To do this, user can change the quantity directly by click add button or reduce button in the product that they want to update. To change the size and color of the product, user must to change it through product detail page. They can access it by click the product in the shopping cart data.

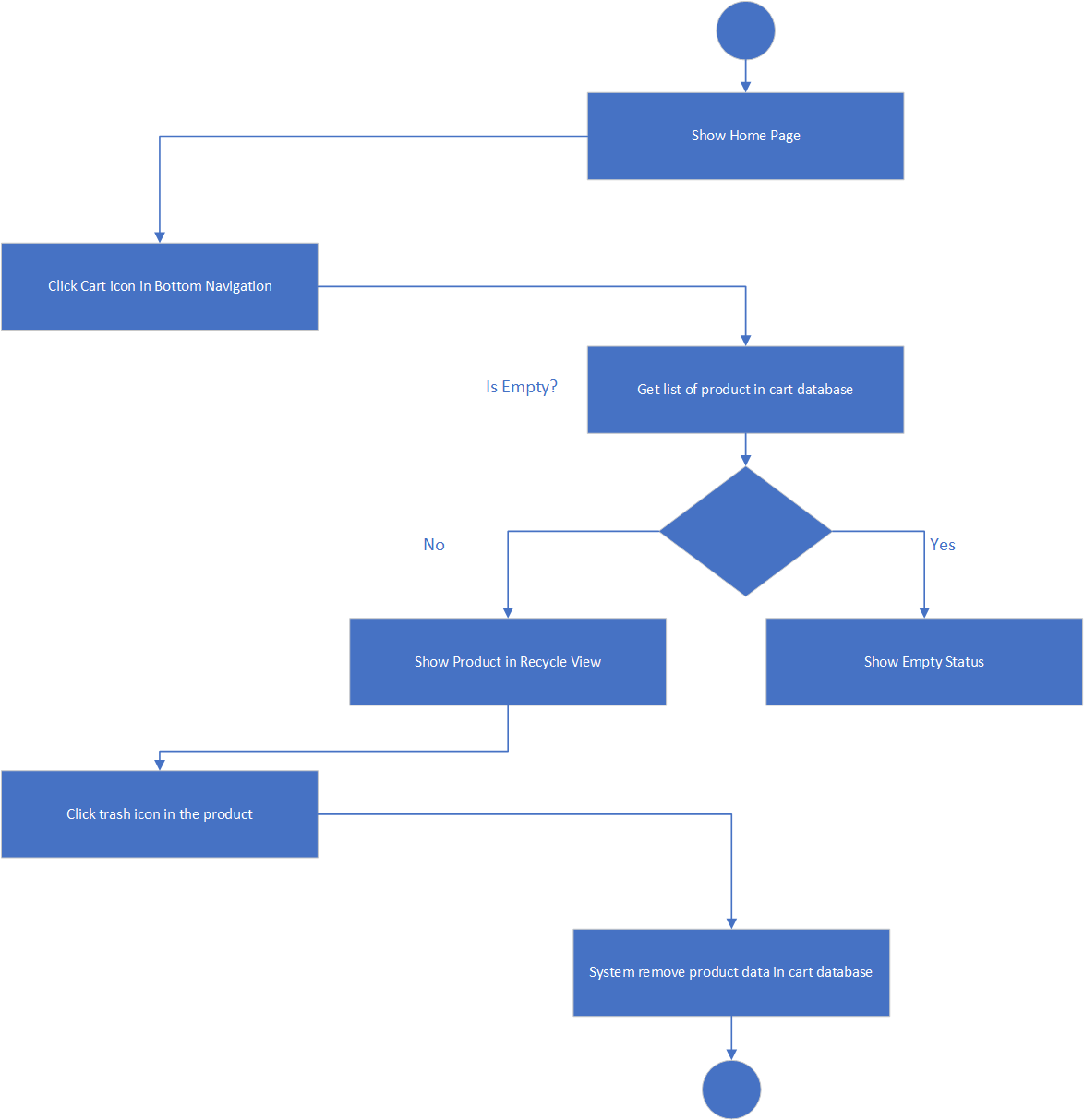
1. **System Feature: Update Product in Shopping Cart – Product Detail Page**



**Figure 11** Update Product in Shopping Cart – Product Detail Page Activity Diagram

This function section is used by customer to update product in shopping cart through product detail page. To change the data in cart, user can select product that they want to change in the shopping cart, change the size, color, and/or quantity. After they change it, click ‘Update product to Cart’ button to update the data in the shopping cart database.

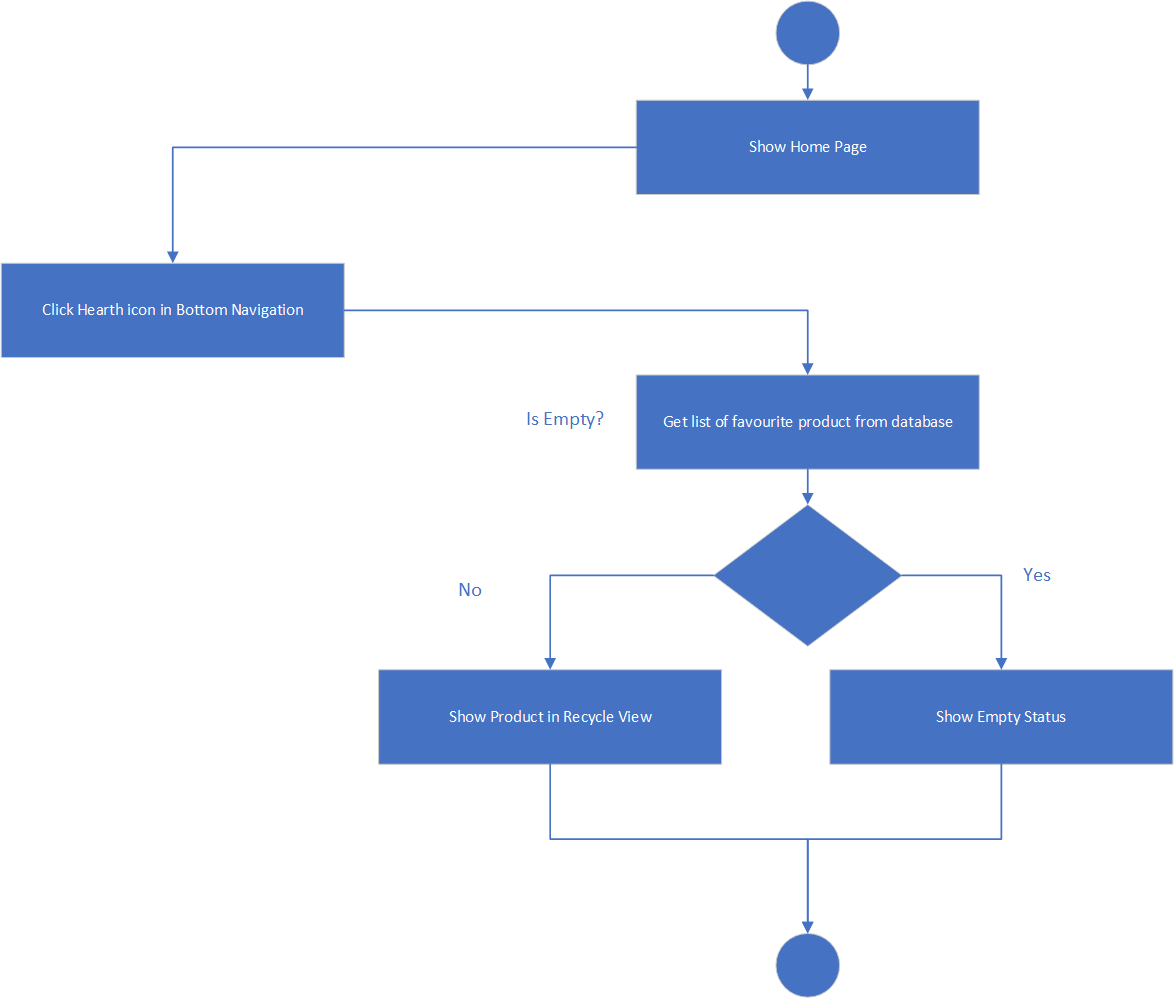
1. **System Feature: Remove Product from Shopping Cart**



**Figure 12** Remove Product from Shopping Cart Activity Diagram

This function section is used by customer to remove product in shopping cart. To do this action, user can click the trash bin icon in the product that they want to remove from sopping cart. Later, system will remove product from cart database.

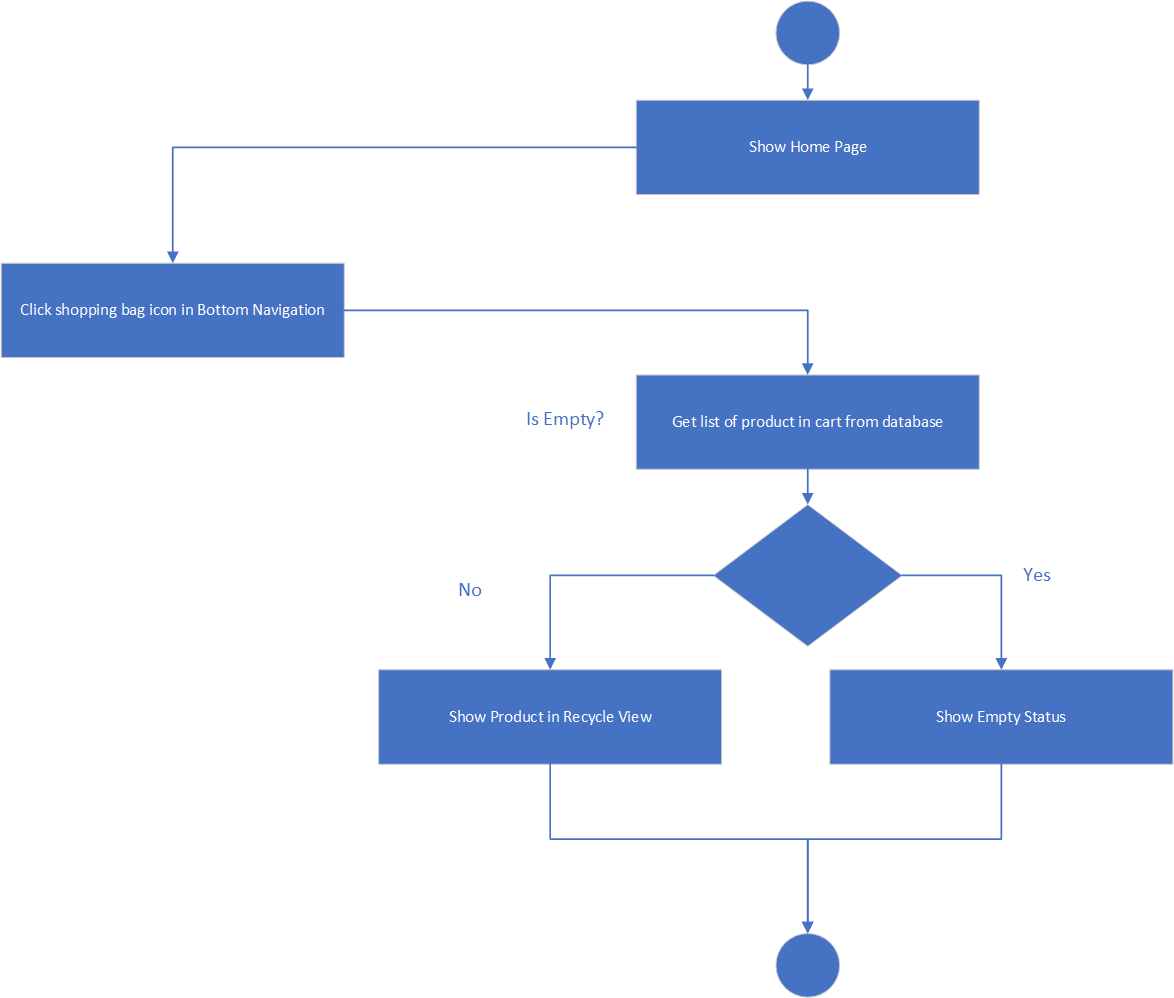
1. **System Feature: See Favorite Products**



**Figure 13** See Favorite Products Activity Diagram

This function section is used by customer to see list of the favorite product in the favorite page. To see this page, user can click the heart icon in the bottom navigation. Later, this app will change the current page to the favorite page.

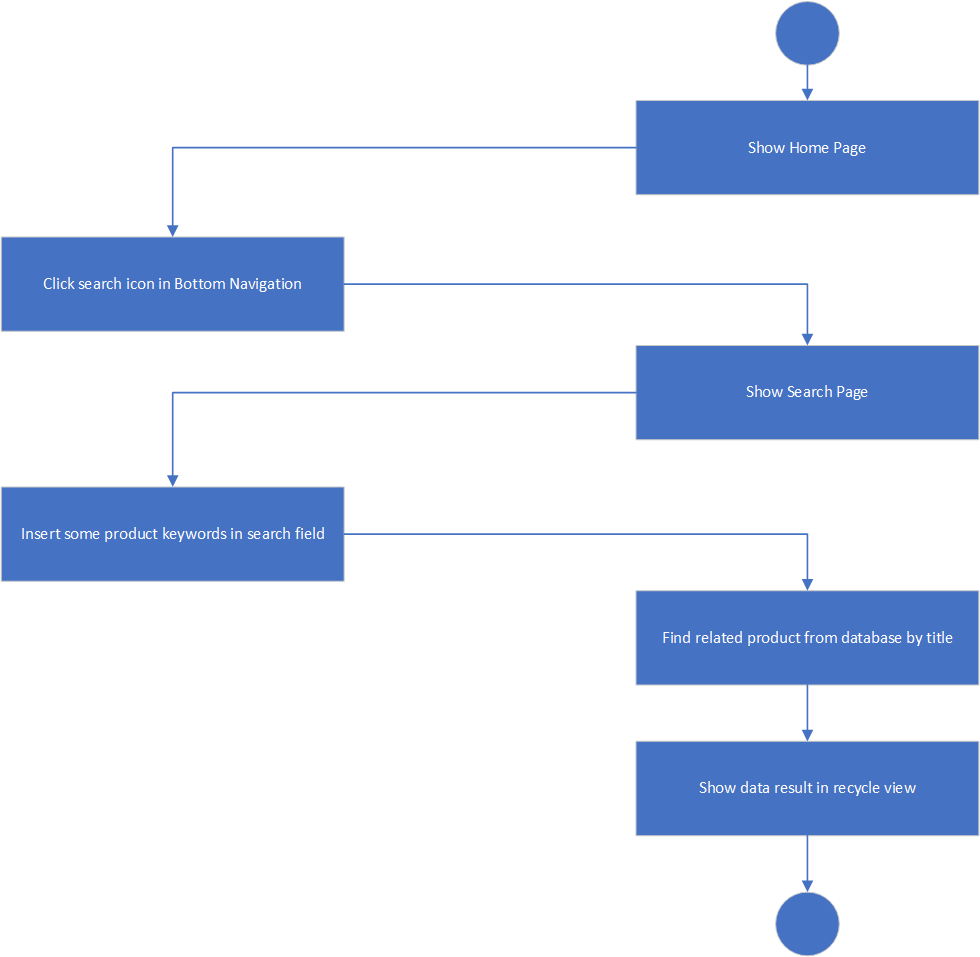
1. **System Feature: See Shopping Cart**



**Figure 14** See Shopping Cart Activity Diagram

This function section is used by customer to see list of the shopping cart product in the shopping cart page. To see this page, user can click the shopping bag icon in the bottom navigation. Later, this app will change the current page to the shopping cart page.

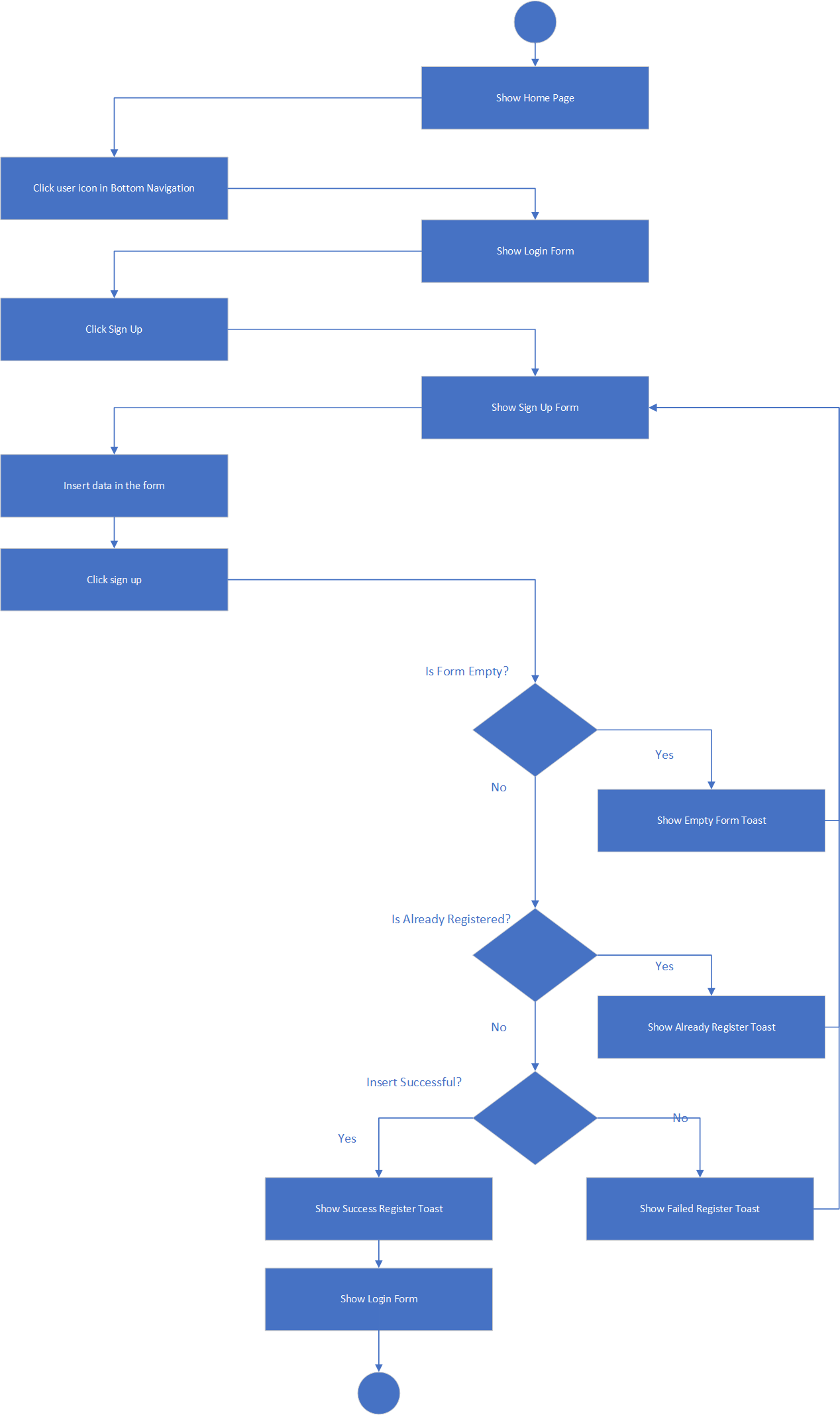
1. **System Feature: Search Product**



**Figure 15** Search Product Activity Diagram

This function section is used by customer to search the product by their title. To do this action, user can go to search page by click the magnifier/search icon in the bottom navigation, after that they can search product by fill the search form in this page.

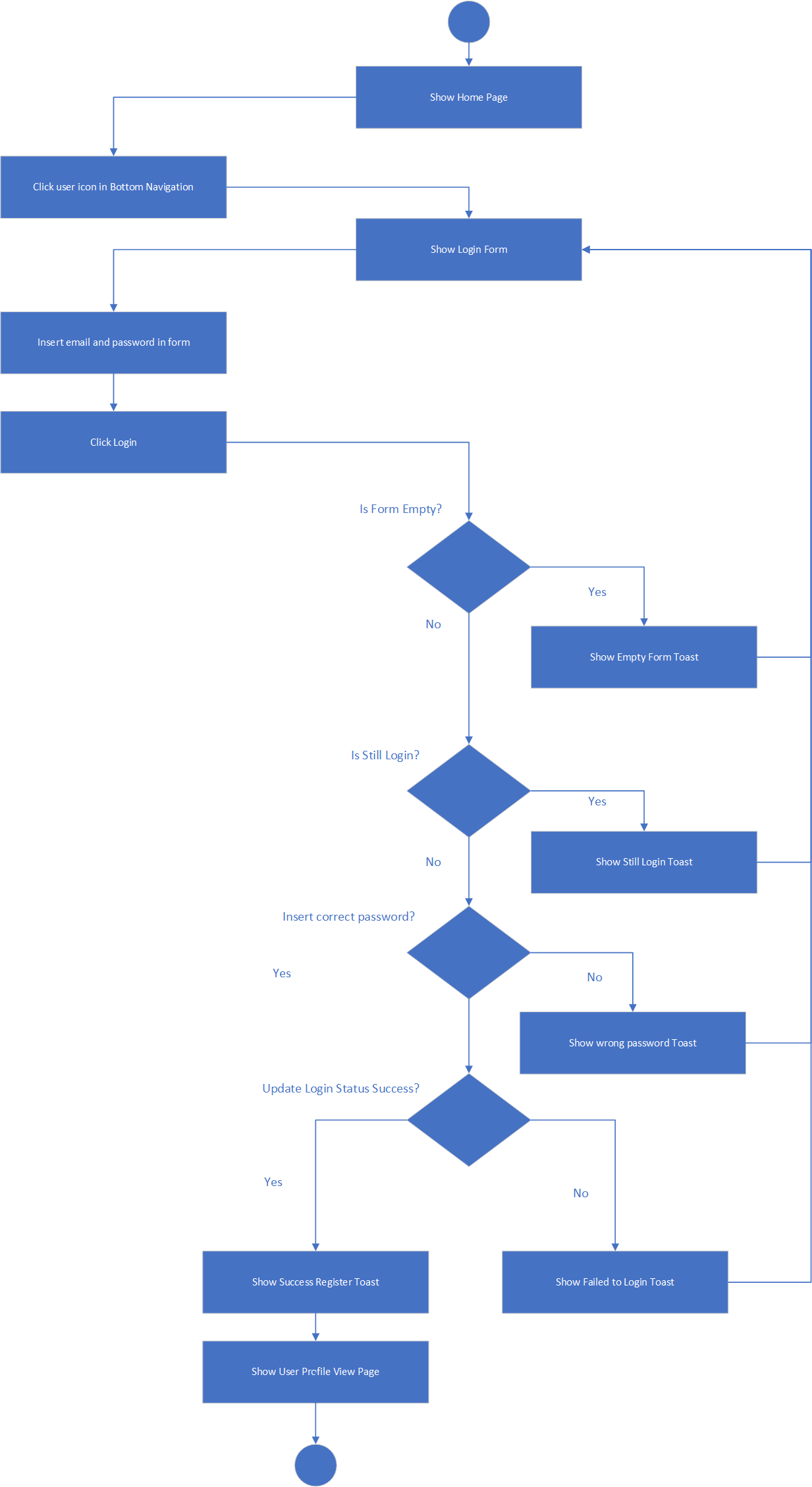
1. **System Feature: Sign Up Account**



**Figure 16** Sign Up Account Activity Diagram

This function section is used by customer to sign up to the Nike account. To do this action, user can go to the user account page by click the user icon in the bottom navigation. After user will see the login page if they aren’t login yet. They can click sign up button, fill the sign up form, and submit it to process the account register.

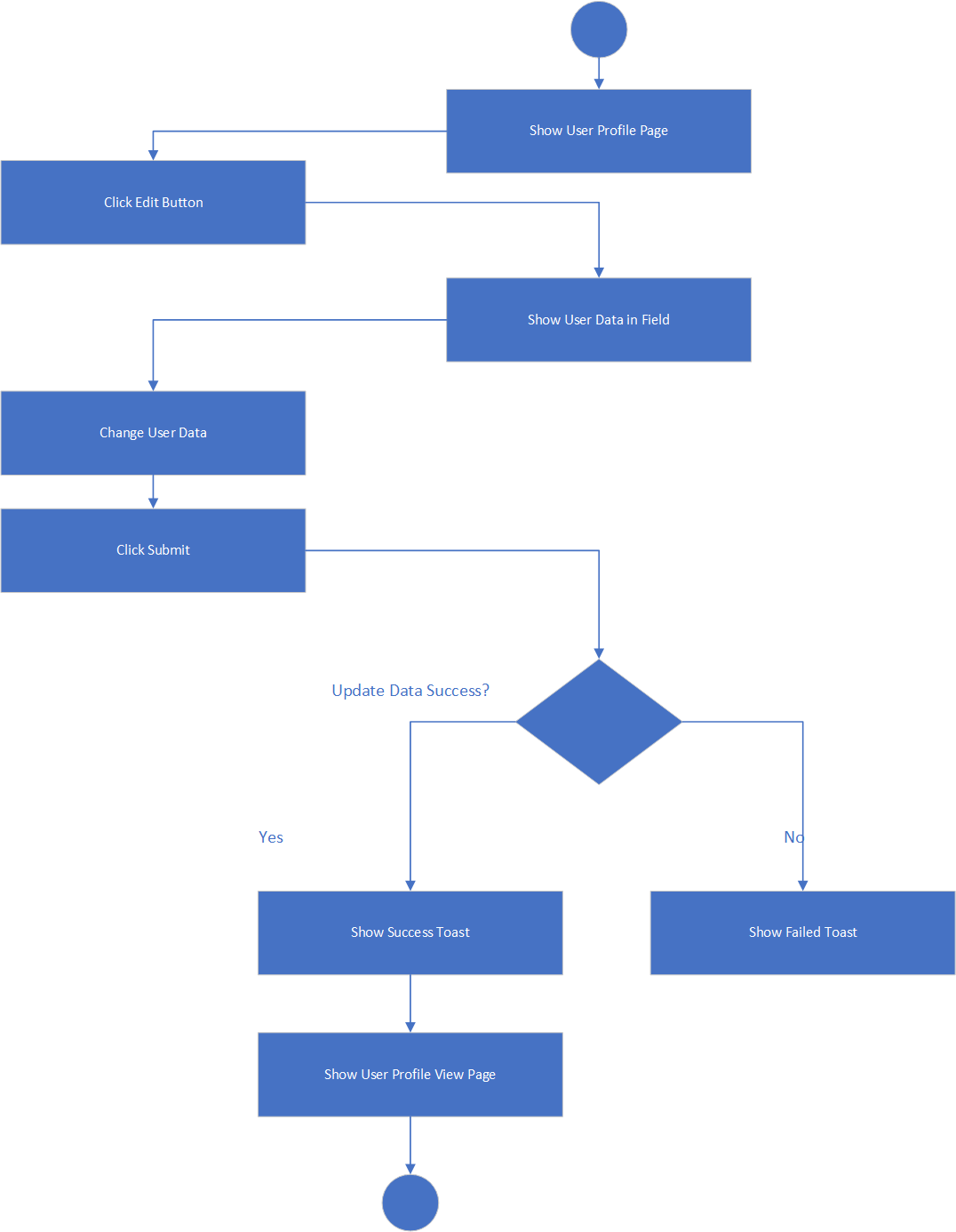
1. **System Feature: Login User**



**Figure 17** Login Activity Diagram

This function section is used by customer to login to Nike account. To do this, user that already register can login by insert their email and password in the login field, and after that click the login button. If the user already registered, apps will redirect the page to the user profile page.

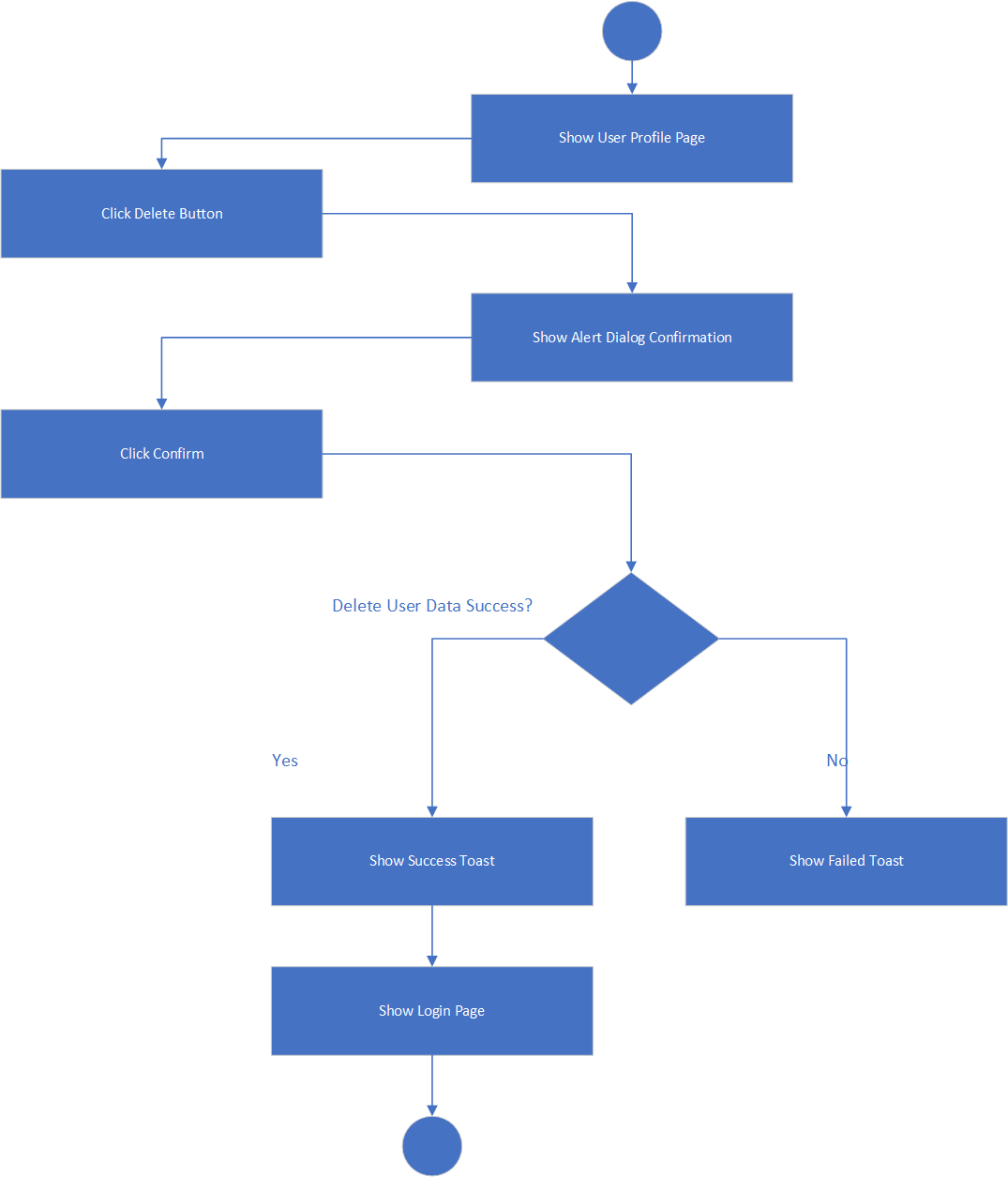
1. **System Feature: Edit User Data**



**Figure 18** Edit User Data Activity Diagram

This function section is used by customer to edit user data in Nike account. To change the user data, after they login, they can click the ‘Edit’ button to redirect to the edit user data page. Later inside this page they must to fill the current password or change it if they want to change, and change the other data that they wan to change. After change the data user can click ‘Submit’ or ‘Confirm’ button to process the update data.

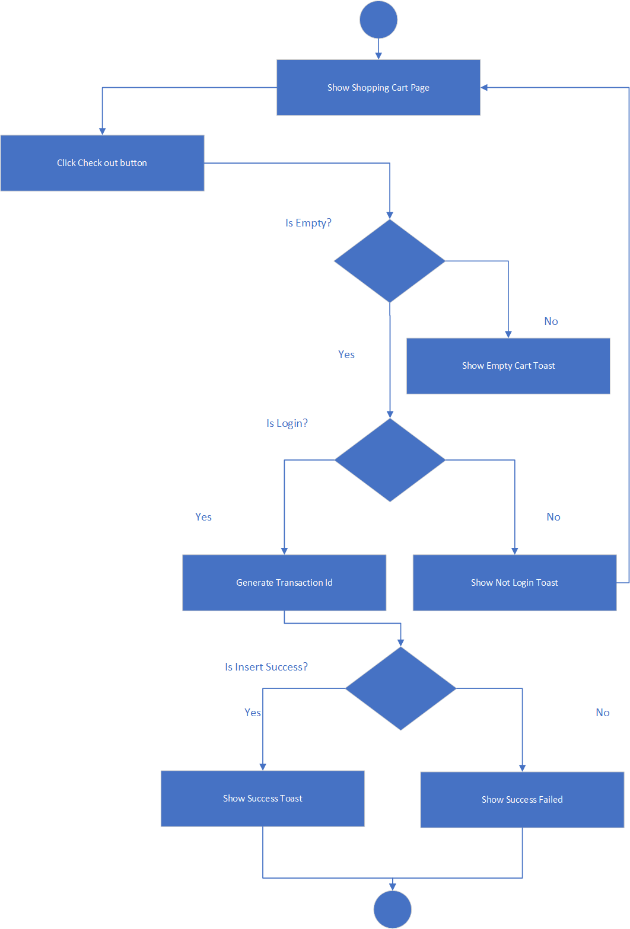
1. **System Feature:** **Delete User Account**



**Figure 19** Delete User Account Activity Diagram

This function section is used by customer to delete user account from Nike database. To do this action user must make sure to clear and finish their running transaction, if not they can’t delete their account. Through user profile page, they can click the delete button to process the delete account.

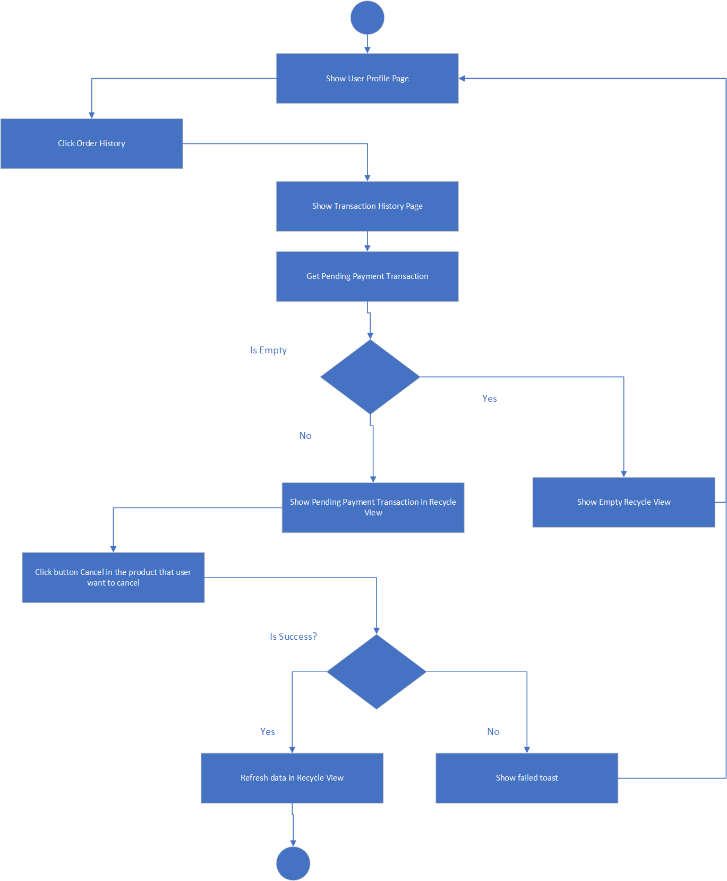
1. **System Feature: Checkout**



**Figure 20** Checkout Activity Diagram

This function section is used by customer to add product to the transaction process. To do this action user can go to shopping cart page and click the check out button. Product will be processes into the transaction process if user already login and the shopping cart is not empty.

1. **System Feature: Cancel Transaction**



This function section is used by customer to remove or cancel transaction in pending payment stage. To do this user can open the transaction history page, and click transaction history button. Later this app will show the page that have tab button to navigate to one transaction page to the another. In the pending payment page user can click ‘Cancel’ button in the transaction that they want to cancel.

## System Requirements Analysis

System requirements analysis consist of hardware and software used in this study. The following s the hardware used in this study:

1. Laptop or Computer. This device is the device that used in the development of the Nike Android-based Application With Kotlin Programing Language. In this research I used Asus A456UQ laptop with Windows 11 Pro 64-bit version 21H2, 12 GB RAM, processor Intel(R) Core (TM) i7-7500U CPU @ 2.70GHz 2.90 GHz, 512 SSD SATA and 1 TB hard disk caddy.

The following is the software used in this research:

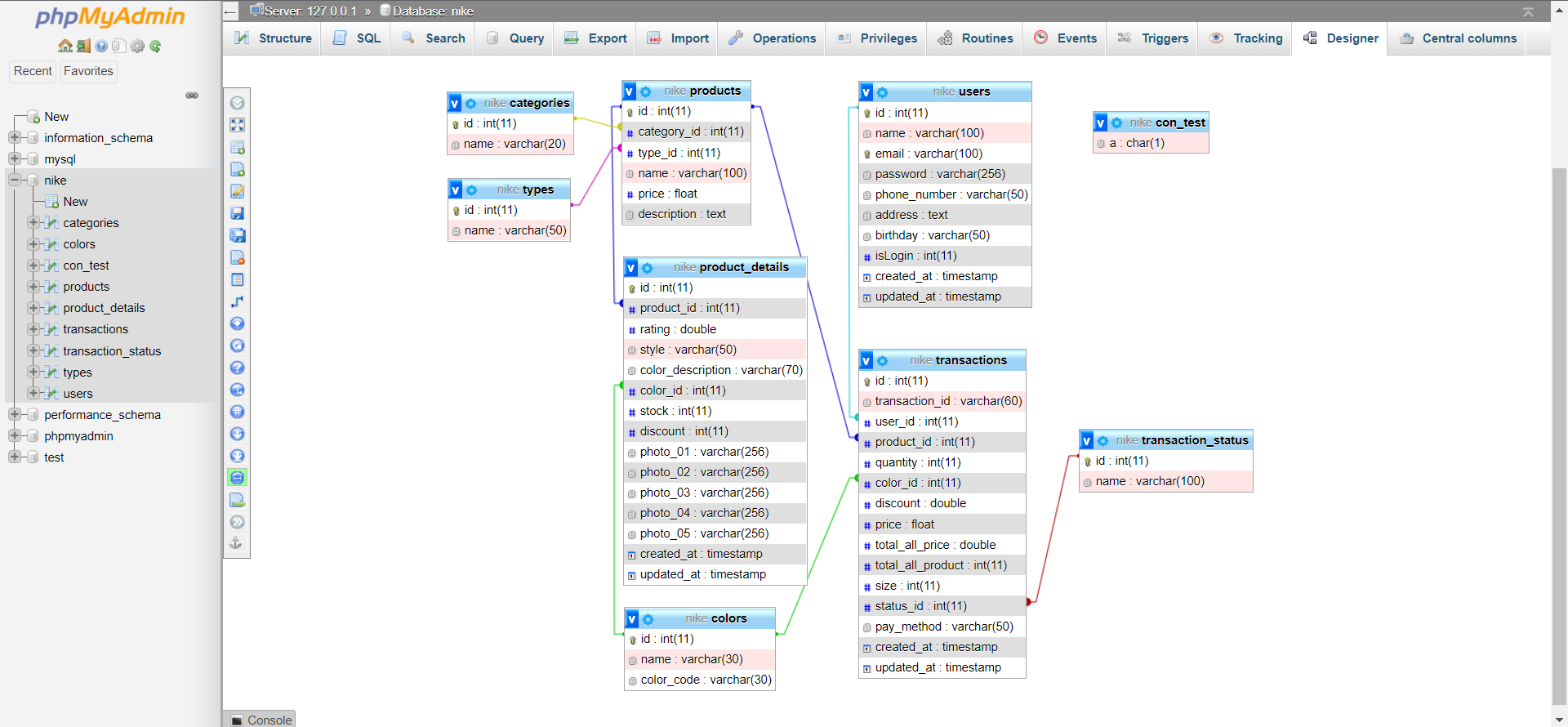
1. XAMPP. Used as local server for Nike Android-based Application.
2. Eclipse. Used as IDE for developed the RESTful service for the Nike Android-based Application.
3. Android Studio. Used to create an Android-based Nike Online Shop Application.
4. Postman. Used to testing get the results from RESTful service that already developed with J2EE.

## Analysis and Design

In this step, this research will analyze using class diagram modeling and interface design to provide an initial overview of the application to be made.

### Class Diagram

Figure 21 is the class diagram from Nike Android-based Application with Kotlin Programming language. This figure describes the classes contained in the application created. All these classes are connected directly to the database connection.

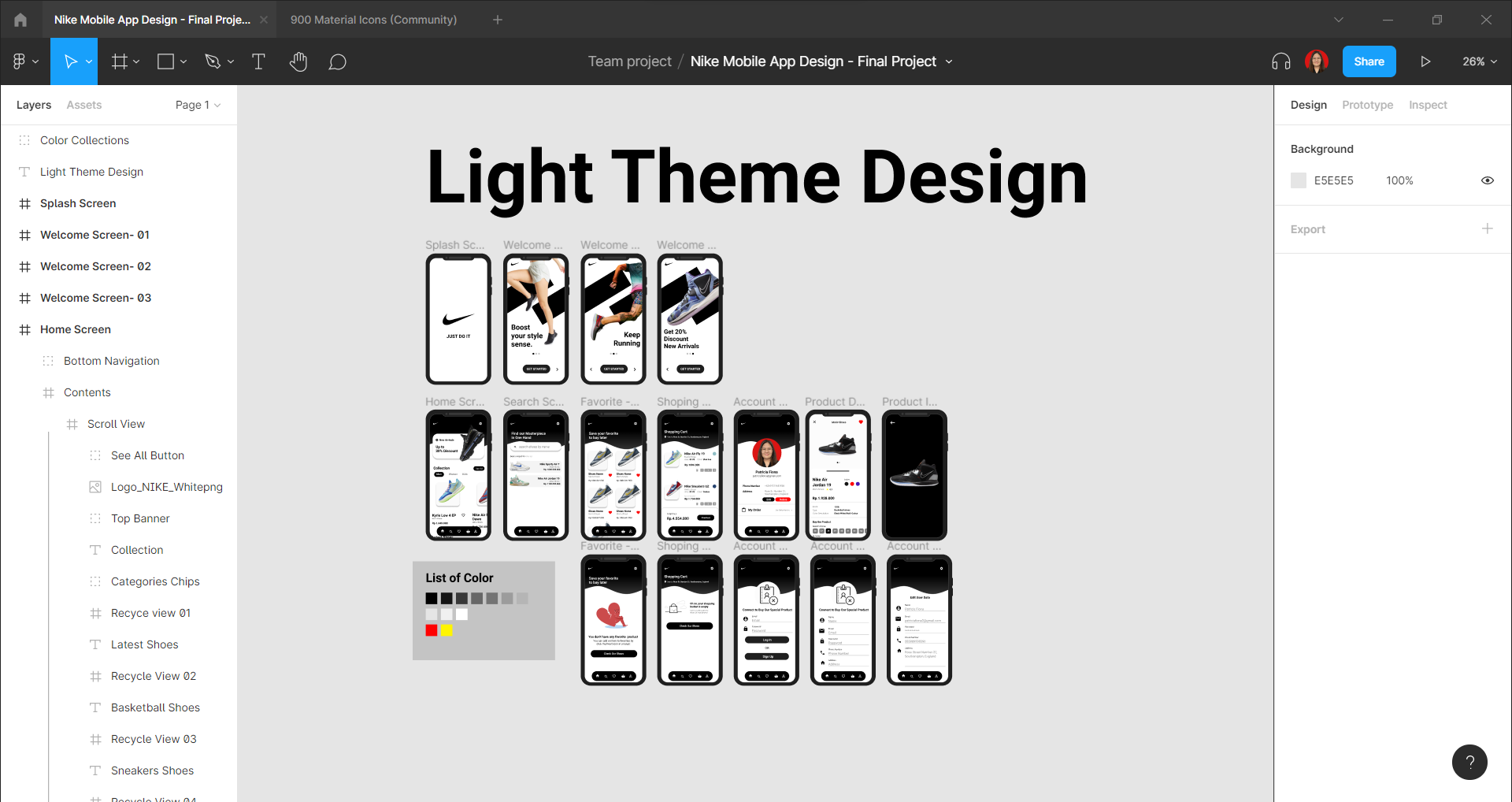


**Figure 21** Class Diagram for NIKE Database

### User Interface Design

User interface help researcher for give an overview in general for the result of the application that we want to develop. In this research I create one user interface design for the user side of Nike Online Shop in Android-Based Application.

In Figure 22 we can see the overview of user interface design of Nike Android-Based Application. This design used Figma application to create the design and the prototype the application. With Figma, we can simulate how the app will run or work later. Here I share my design link of my UI design in Figma: <https://www.figma.com/file/nYKeeZFbhUI7wUoaD1mHQK/Nike-Mobile-App-Design---Final-Project?node-id=0%3A1>.



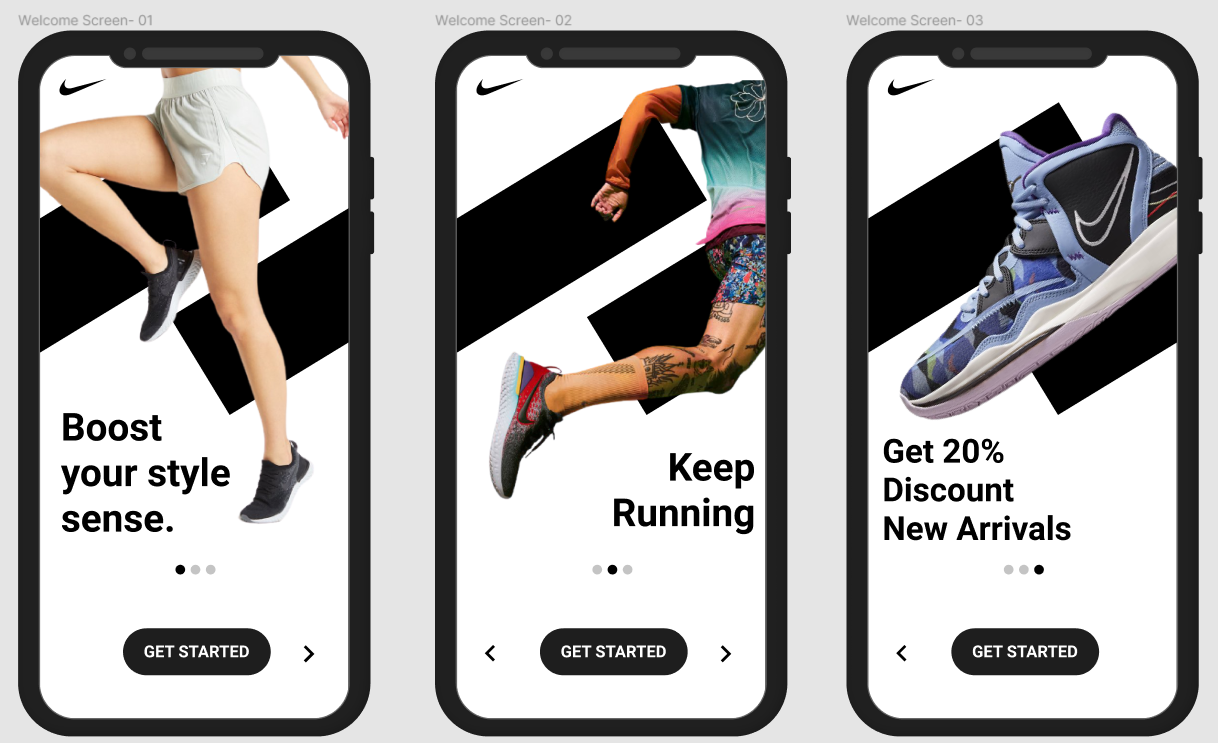
**Figure 22** Overall Design For Nike Apps

Figure 23 is the design for splash screen. This screen will show every time user open the application and this screen will show the logo of Nike and the slogan in a few second.



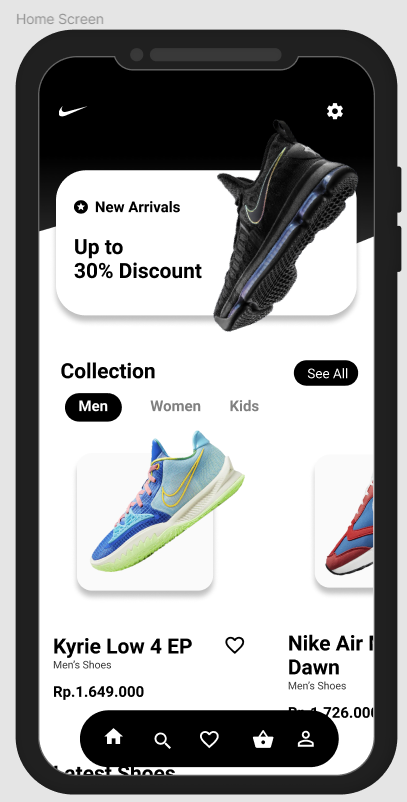
**Figure 23** Splash Screen Design

Figure 24 is the design for welcome page. This is the page for the new users that download this application. Inside this page there are some welcome remarks and picture that are poured into the form of a slide show.



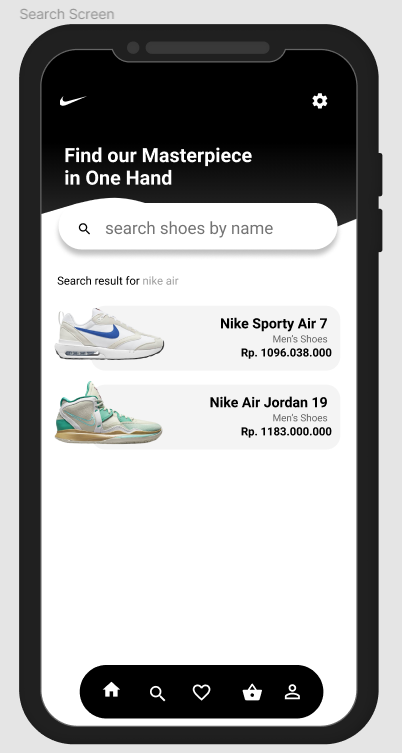
**Figure 24** Welcome Scree for New User Page

Figure 25 is the design for home page. In this page I put 2 banners in the top and bottom of the page. Beside that in this page I show some collections, latest products, and some shoes types based on the selected category.



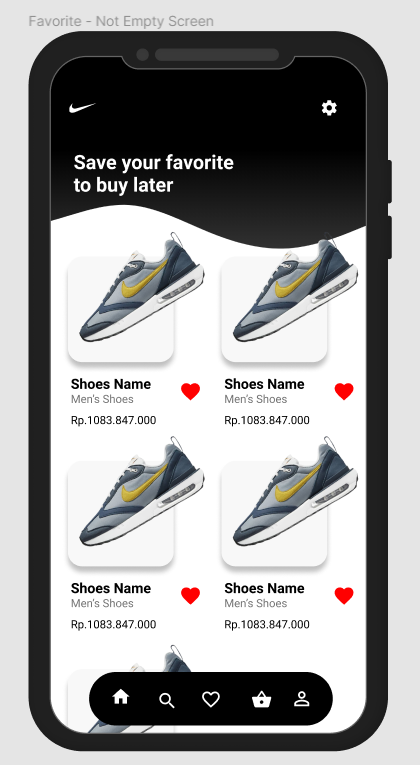
**Figure 25** Home Screen Design

Figure 26 is the design for the result of the search feature. To search product, user can search by some keywords related to the product title/name.



**Figure 26** Search Page Design

Figure 27 is design for favorite page that is not empty. Inside this page user can see the list of the product that they already liked before.



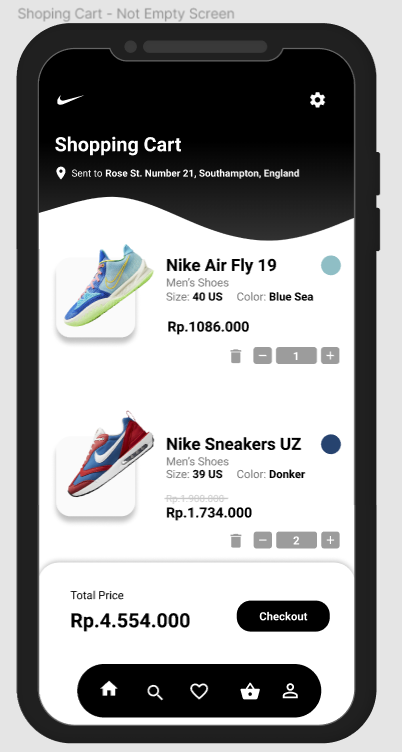
**Figure 27** Favorite Page – Not Empty Design

Figure 28 is the design for favorite page that is empty or still don’t have any like product. With button ’Check Our Shoes’, user can see the list of all product based on their category.



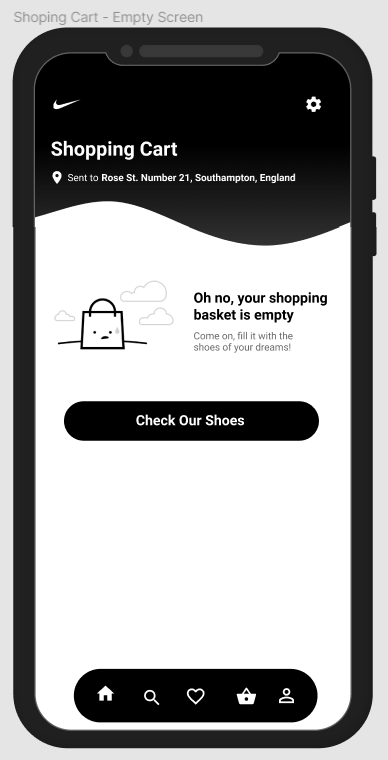
**Figure 28** Favorite Page – Empty Design

Figure 29 is the design of shopping cart – not empty page. All data that already in the cart database will be show using recycle view.



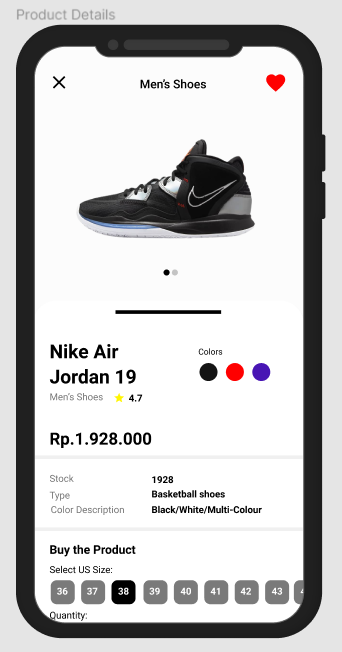
**Figure 29** Shopping Cart – Not Empty Design

Figure 30 is the design for empty shopping cart data. Inside this page we can see the button to explore product in the Nike application.



**Figure 30** Shopping Cart Page – Empty Design

Figure 31 is the design for detail product page. In this page later I implemented slideshow image for show the list of product image and used bottom sheet as information container.



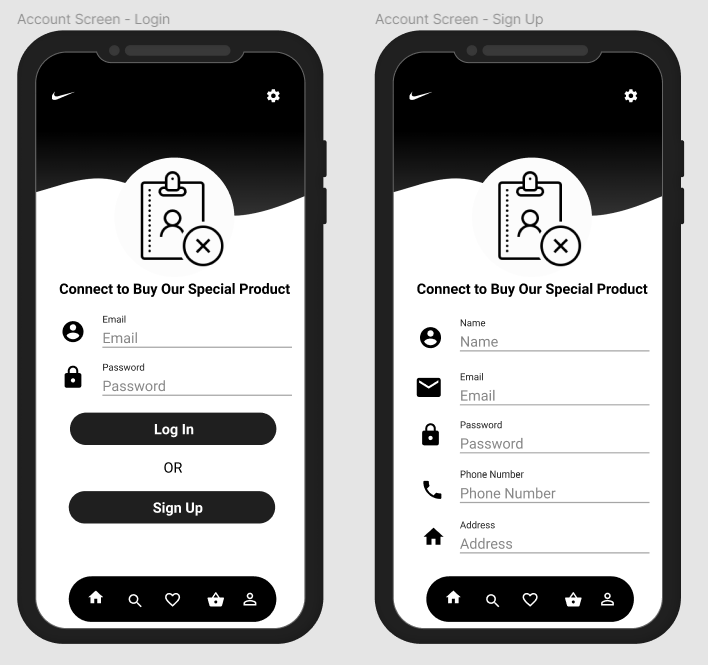
**Figure 31** Product Detail Page Design

Figure 32 is the design for product image detail page. Inside this website user can zoom in and zoom out the image so they can see the detail of the shoes more clearly.



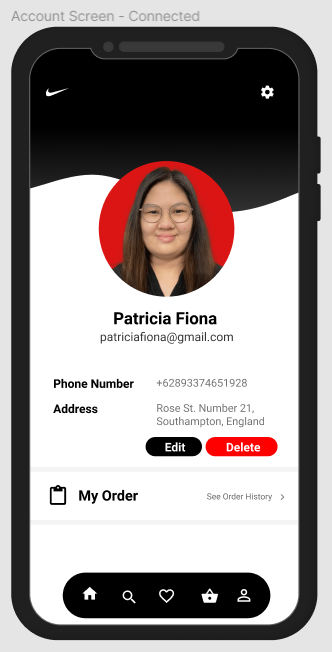
**Figure 32** Product Image Preview Page Design

Figure 33 is the design for login and register. Inside tis page there are some form that user need to filled if they want to create new account or login to their existing Nike account.



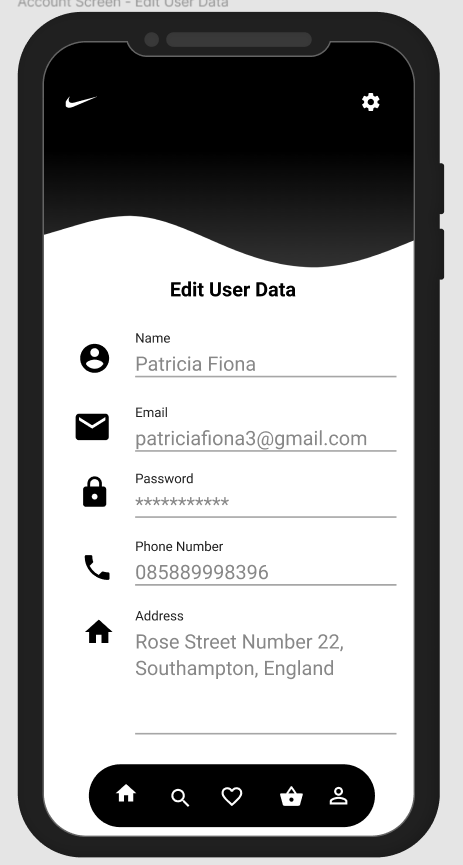
**Figure 33** Login and Signup Page Design

Figure 34 is the design for user profile page. After user login, this page will show. Inside this page, user can change their user data by click the edit button, delete the account by clicking delete button, and see the order history by click See order history button.



**Figure 34** User Profile Page Design

Figure 35 is the design of edit user data page. Inside this page there are some forms that later will filled with the user existing data. User can change their user data by click the submit button.



**Figure 35** User Edit Data Page Design

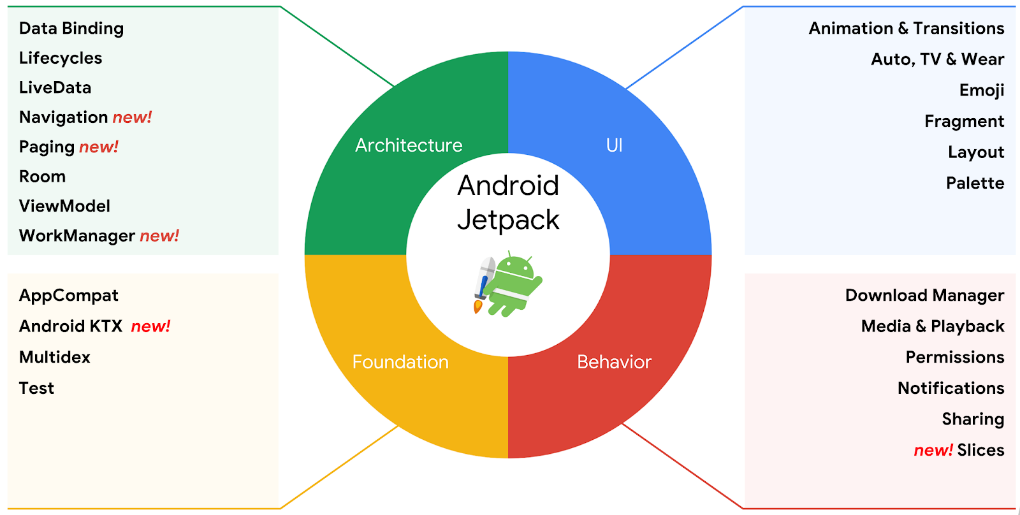
## Implementation

The following is an explanation of the core technology used in the Nike Android-based Application Development Process Using the Kotlin Programming Language.

### Jetpack Component & Architecture

Jetpack is a collection of component libraries, tools, and guides for building great apps [4]. This collection officially published at Google I/O 2018 with the name: Android Jetpack. Jetpack consists of several library packages with the android prefix that are separate from the platform APIs. Jetpack also offers compatibility with previous versions and is updated more frequently than the Android platform. As a result, we can always access the latest and greatest version of Jetpack components regardless of compatibility.

At least we can know 3 benefits by using Jetpack components. First, we will follow the best practice of Android development. By implementing modern best practices, the Android Jetpack component allows us to reduce the rate of crashes that will occur and also avoid the occurrence of memory leaks. Android Jetpack components also have backward compatibility support so you don’t have to worry about version issues. Second, by implement Android Jetpack we will remove boilerplate code because Android Jetpack can manage repetitive code such as background task, navigation, and lifecycle management, so we can focus more on features to make our app attractive. And the last example of the benefits is reducing the fragmentation. Fragmentation means different code for different Android versions, so we have to add branching logic for those different versions. With the Jetpack component, we can reduce this because it supports all versions and produces consistent output on different versions and devices.



**Figure 36** Android Jetpack Component Illustration

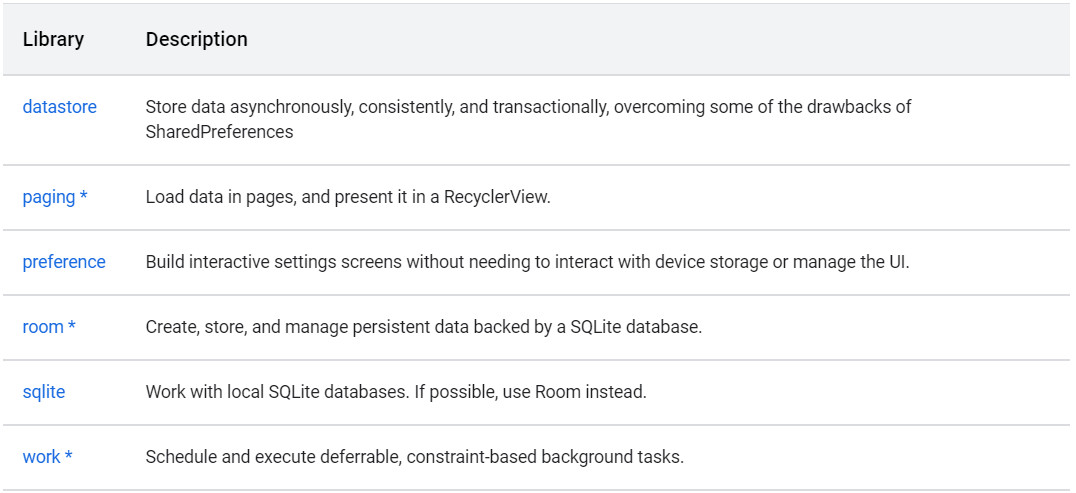
In Figure 36 we can see the image of Android Jetpack components. Until now, Jetpack components are keeps increasing until more than 70 library and divided into 9 parts, namely [5]:

1. **Beyond Phone**



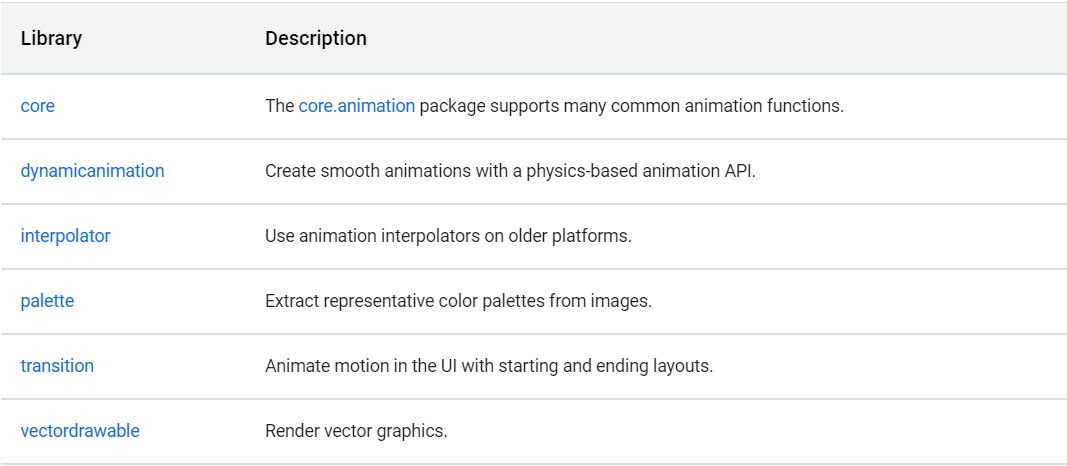
**Figure 37** Beyond Phone Category in Jetpack Components

1. **Data**



**Figure 38** Data Category in Jetpack Components

1. **Graphics**



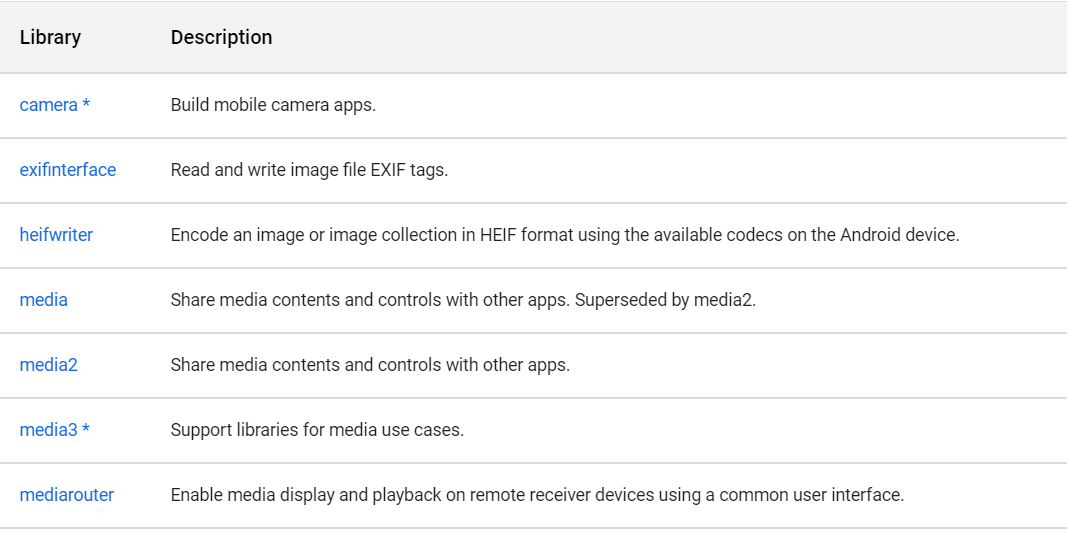
**Figure 39** Graphic Category in Jetpack Components

1. **Lifecycle**



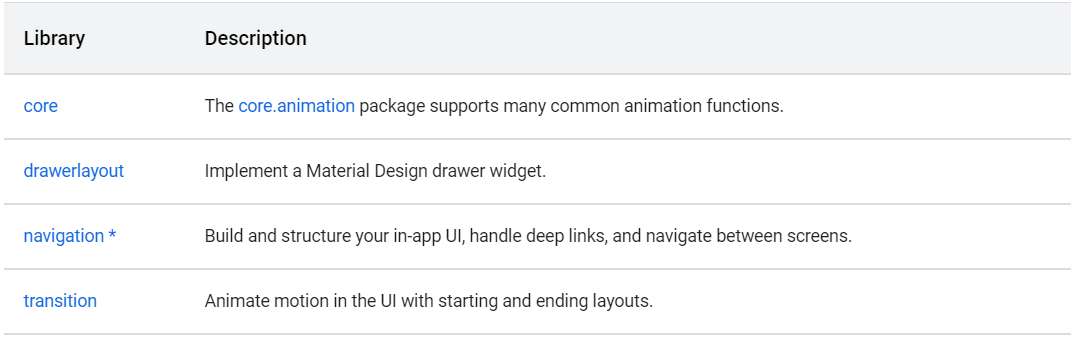
**Figure 40** Lifecycle Category in Jetpack Components

1. **Media**



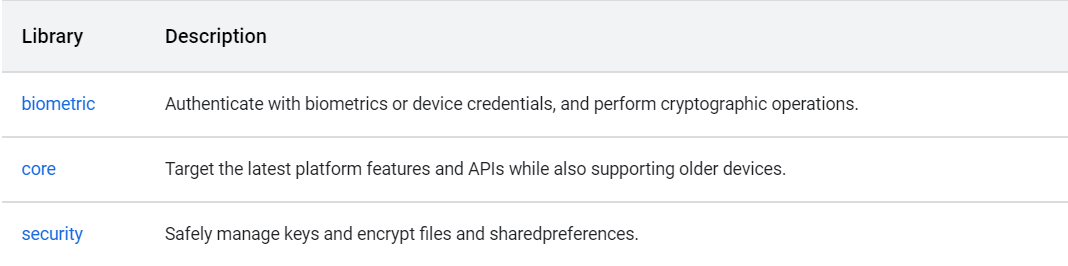
**Figure 41** Media Category in Jetpack Components

1. **Navigation**



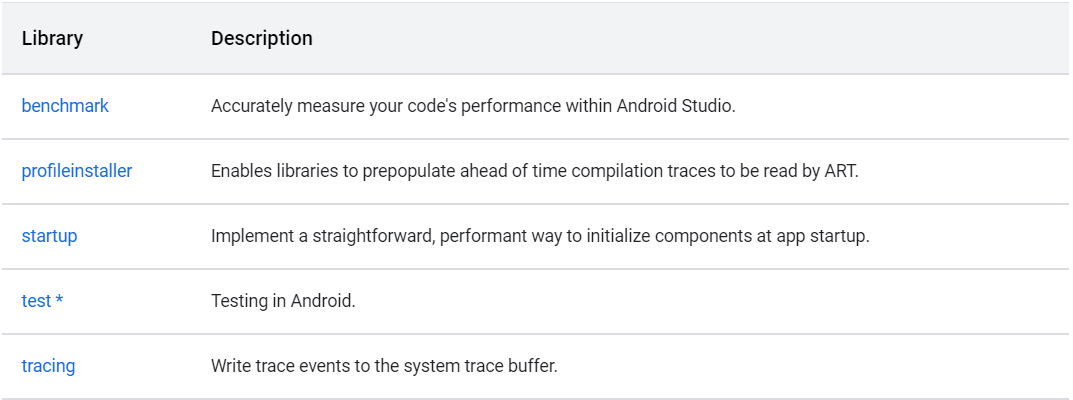
**Figure 42** Navigation Category in Jetpack Components

1. **Security**



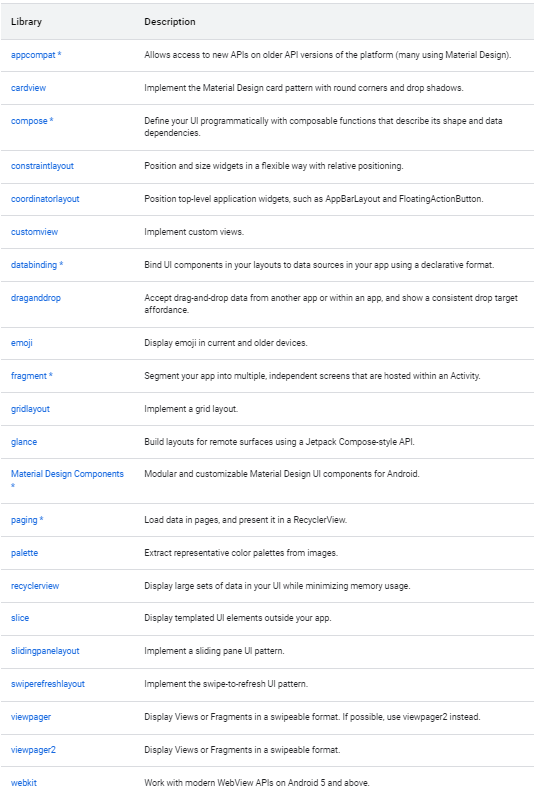
**Figure 43** Security Category in Jetpack Components

1. **Performance/Test**



**Figure 44** Performance/Test Category in Jetpack Components

1. **UI**

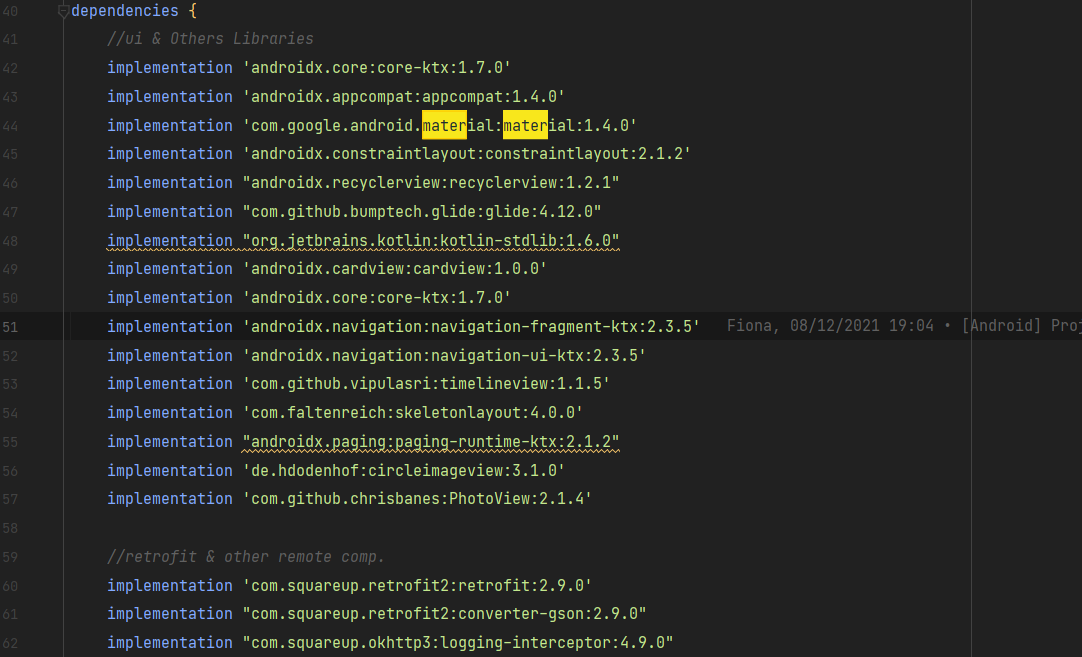


**Figure 45** UI Category in Jetpack Components

From all components, I used some of the Android Jetpack components, and this is the core component that I used in this development:

1. Paging
2. Room
3. Lifecycle
4. Navigation
5. Appcompat
6. Cardview
7. Constraintlayout
8. Coordinatorlayout
9. Databinding
10. Fragment
11. Gridlayout
12. Material design components
13. Paging
14. Recycle view
15. Viewpager2

For the application of this Jetpack component, it is divided into layout design and system design. For further component implementation, see the code snippets or the Android application project, and for some of the import library result can be seen in Figure 46.





**Figure 46** Library import in the build.gradle

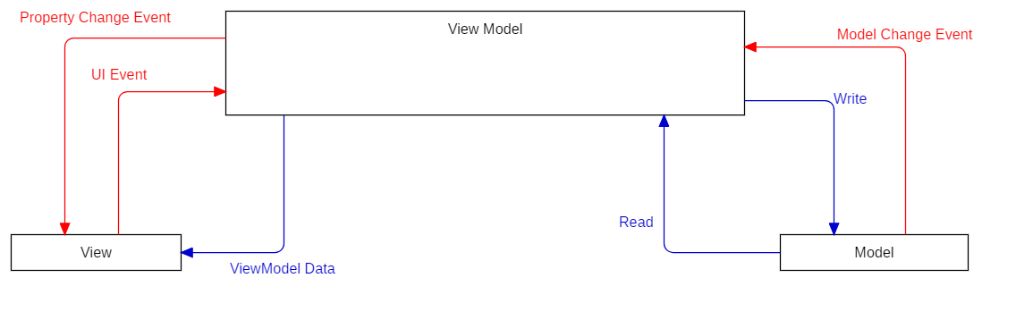
### RESTful API Service

A REST API (also known as RESTful API) is an application programming interface (API or web API) that conforms to the constraints of REST architectural style and allows for interaction with RESTful web services. An API is a set of definitions and protocols for building and integrating application software [6]. REST is a set of architectural constraints, not a protocol or a standard. API developers can implement REST in a variety of ways [6].

When a client request is made via a RESTful API, it transfers a representation of the state of the resource to the requester or endpoint [6]. This information, or representation, is delivered in one of several formats via HTTP: JSON (Javascript Object Notation), HTML, XLT, Python, PHP, or plain text. JSON is the most generally popular file format to use because, despite its name, it’s language-agnostic, as well as readable by both humans and machines [6].

### MVVM Architecture

Model-View-ViewModel (MVVM) is the one of industry-recognized software architecture pattern that overcomes all drawbacks of MVP and MVC design patterns [7]. MVVM suggests separating the data presentation logic (View or User interface) from the core business logic part of the application. These are the details for each component in MVVM architecture [7]:



**Figure 47** MVVM Architecture Model

1. **Model**

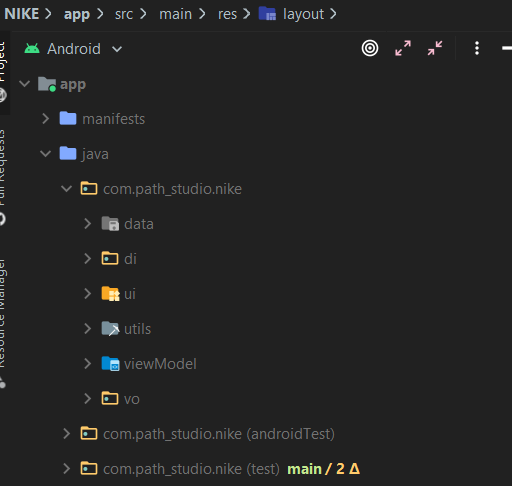
Model is layer that have responsibility to the abstraction of the data sources. Later Model and ViewModel will work together to get and save the data.

1. **View**

This component has a task to inform the ViewModel about the user’s action. This layer observes the ViewModel and doesn’t contain any kind of application logic.

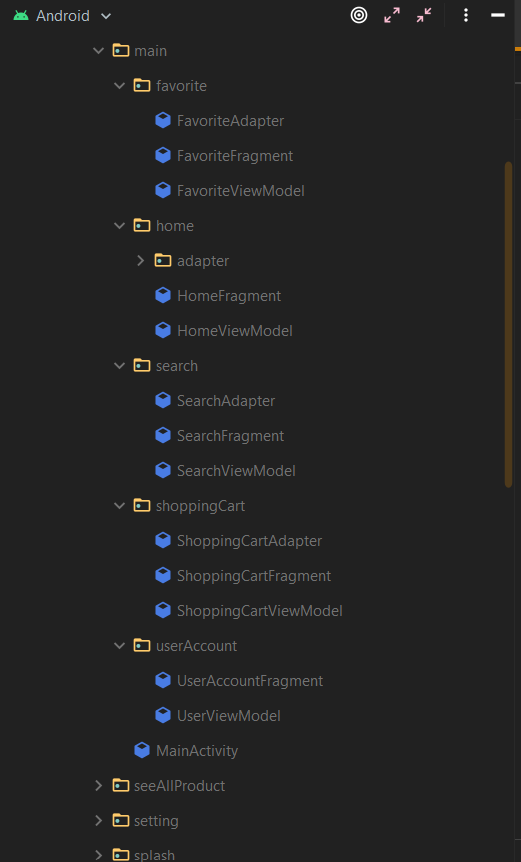
1. **ViewModel**

This component exposes the data streams which are relevant to the view. Moreover, its servers as a link between the Model and the View.



**Figure 48** The Directory Structure that Store Each Code Based on Their Function

To implement this architecture, I separate each code based on their function. We can see from Figure 47 I already separate the code in some of the big packages that have different function from one to another. For the code that related to the Model, I put in the ‘**data**’ package. For the code that related to the View, I put in the ‘**ui**’ package. And for the code that related to the ViewModel, I put the code in **each related View package** like we can see in the Figure 48. If the view model used in the Home page, I will put the view model data in the Home package. I implement in this way because it made me easier to manage the view model related to each View (because each view can have different view model).

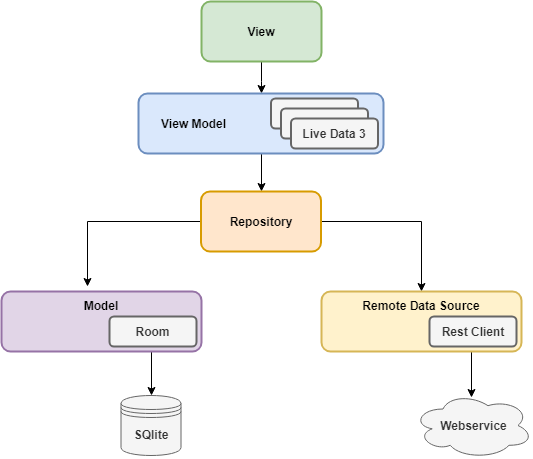


**Figure 49** Some View Model Location That Related to Each View

### Repository Pattern, Dependency Injection, Live Data, and Idling Resource

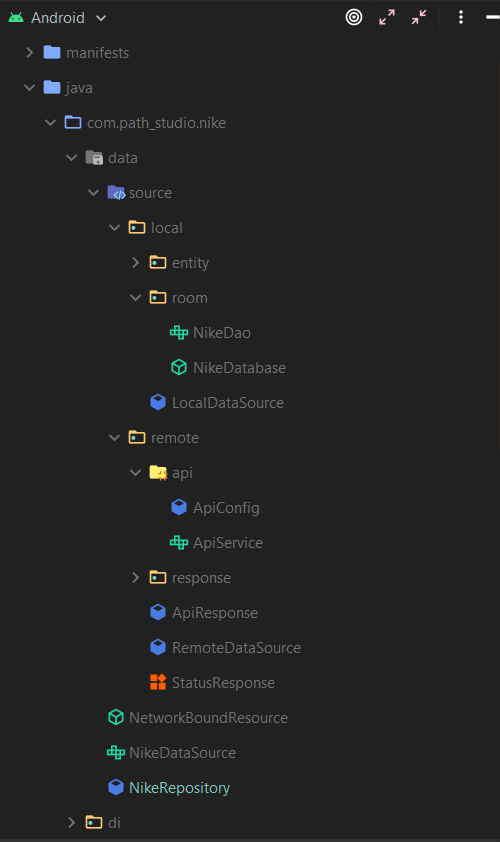
Have you ever thing about the importance of data availability? Imagine if you have a large business and suddenly the server or some part or all of the backend services you have experience problems so that users can’t view and make transactions within a certain time. Of course, this can have a negative impact as well as losses due to the loss of potential transactions that can occur that that time. Due to this problem, I implemented the repository pattern which at least solves some of the problems.

Repository pattern is the pattern or approach pattern for managing data sources so that applications ca run event if they are online or offline. When online, the application will retrieve API data from the network, while when offline the application will display data from a previously stored local database. So, with this pattern implementation, application can continue to be accessed in any condition. This will increase the convenience of our application users.



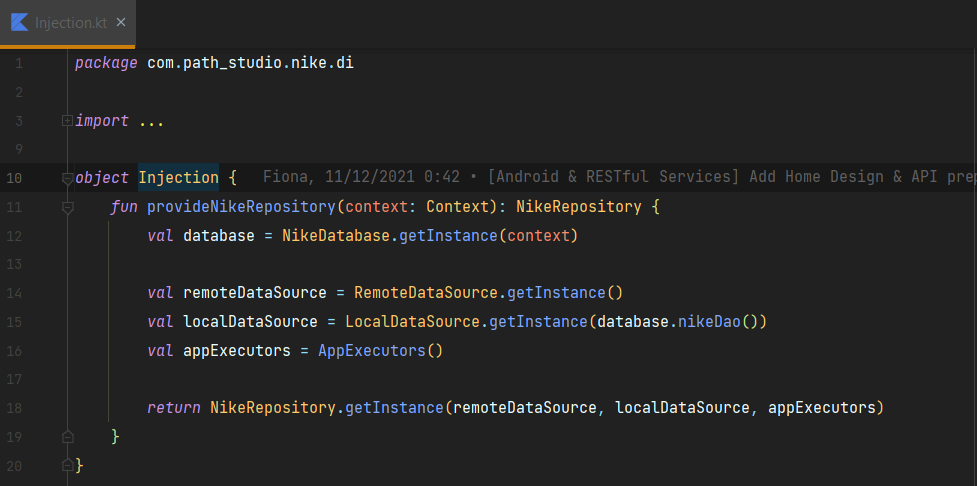
**Figure 50** Repository Pattern Architecture Illustration

In Figure 50 we can see the architecture illustration of repository pattern. If we see in the image, the repository plays a role in managing various resources that exist in the application. In terms, the repository pattern is a pattern or approach pattern where we create an abstraction (with a class) to encapsulate (hide) the query. When we have complex queries, this repository pattern can reduce complexity and provide reusability for data retrieval. For more details of how repository pattern works, the impact of the application of this repository pattern can be seen in the video tutorial that has been attached to the final project report. And for the detail of structure of repository pattern in the data package can be seen in Figure 51.



**Figure 51** Implementation of Repository Pattern in Data Package

The next component that I implemented is dependency injection (DI). Dependency injection is a technique for using another class without having to think about how the class was created, where one object (or a static method) can supply the dependencies of another object. A dependency is an object that can be used (a service). An injection is the transfer of a dependency to a dependent object (client) that will use it. This service is part of the client’s condition. Throwing the service to the client is better than letting the client build or find the service themselves. This is a basic requirement of a pattern. In my project, dependency injection is used for provide some data Nike database instance, remote data source, local data source, and app executor for provide Nike Repository. The detail of the code can be seen in Figure 52.



**Figure 52** Code Implementation of Dependency Injection for Provie Nike Repository

The third component is LiveData. LiveData is an observable class holder that will notify us when data changes. Unlike other observable classes, LiveData is Lifecycle-Aware. This means that LiveData can respond to changes in the Lifecycle such as Activity, Fragment, or Service. LiveData will only update the observer component when the app’s Lifecycle is active.

The last component is idling resource. Idling resource will represent asynchronous operations whose results will affect subsequent operations in the UI Test. By registering an Espresso idling resource, we can validate asynchronous process more easily and flexibly when testing application. Unfortunately, due to time constraints, I was unable to implement application testing using Espresso and JUnit4/JUnit5 in this research, but for the application of idling resources I have put them in the code as can be seen in Figure 53.



**Figure 53** One Example of Application of Idling Resource with Espresso in the getSearchResult()

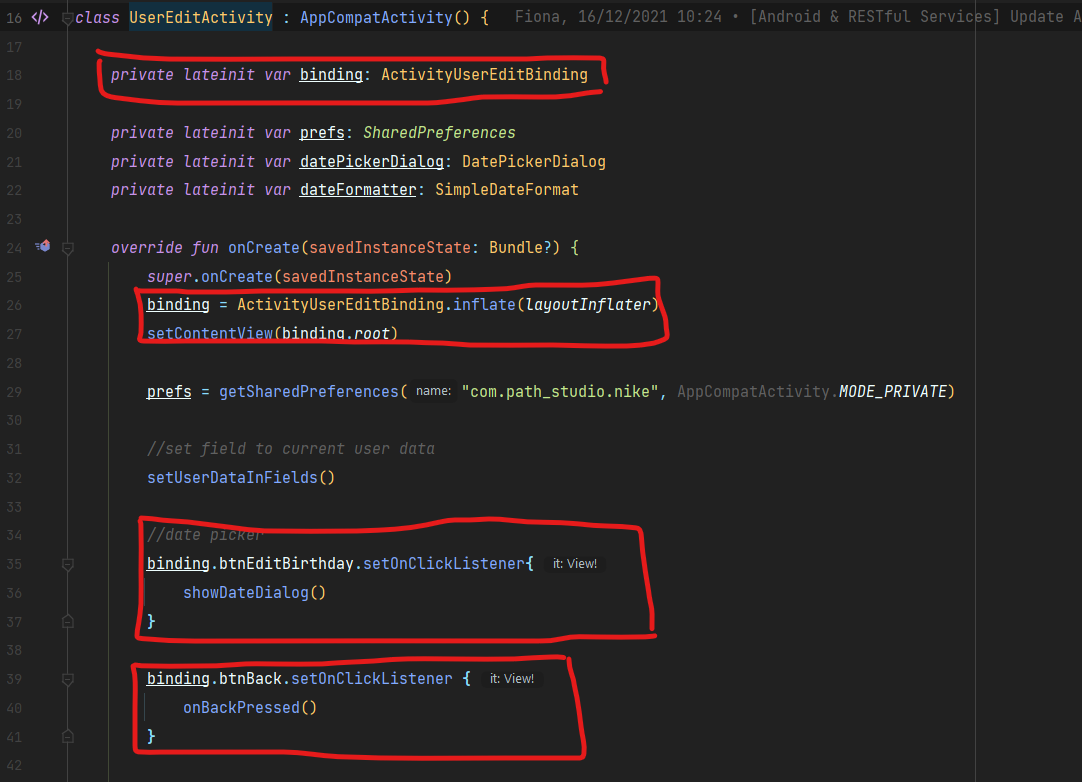
### View Binding

Quoted from Android Developer website, view binding is a feature that allows you to more easily write code that interacts with views [8]. When view binding is enabled in a module, it generates a binding class for each XML layout file present in that module [8]. An instance of a binding class contains direct references to all views that have an ID in the corresponding layout [8]. To enable the view binding feature, we can add code to build.gradle(:app) as can be seen in Figure 54.



**Figure 54** View Binding Code in build.gradle(:app)

The most striking difference from using view binding is that there is no need to import ‘R’ to call the layout and the components in it. Whit view binding, we can initialize binding variables based on the layout associated with the activity/fragment, then call the component we want to call based on its component id name without having to use findById. For more detail, the example of view binding implementation can be seen in Figure 55.



**Figure 55** View Binding Implementation in UserEditActivity()

# Chapter III

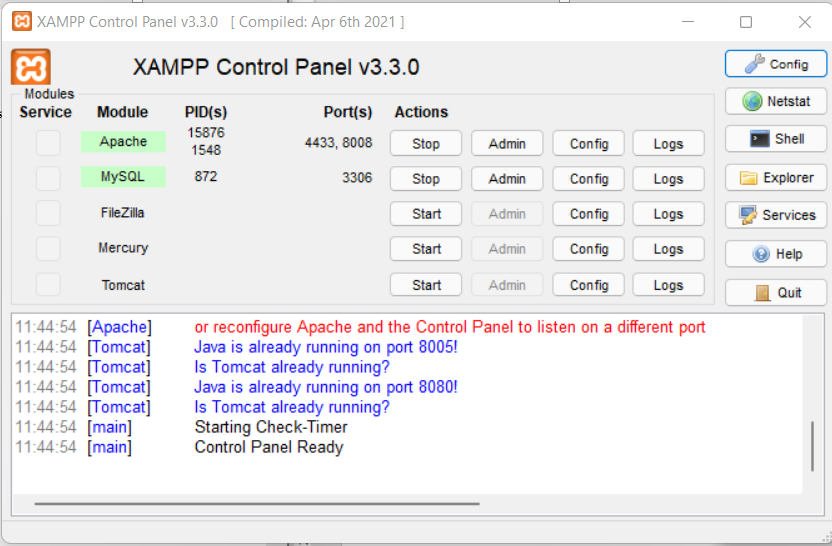
**Result and Evaluation**

## Results of Nike Android-Based Application Development with Kotlin Programming Language

Below is the result of developing an Nike Android-Based Application Development with Kotlin Programming Language. Program results are made based on the analysis and design that has been done in the previous stage. The following are the results of the program that the author has created:

### Preparation for Run Application

Before testing the Android application that has been made, we need to prepare a few things. The first this to do is turn on the local server service. Figure 56 is the screenshot of XAMPP application that I used in this research as local server service.



**Figure 56** Preview of My XAMPP Application in Apache and MySQL Mode On

I use Apache service that using port 4433 and 8008. This is not the default port because in my situation, the default port already used for another application service. Same like Apache, MySQL service also didn’t use default port. Here I used 3306 as MySQL port. With the difference in the application of this port, to do testing on other devices, please pay attention and change the port according to the port you are using.

As an example of the impact of port differences, here are some pieces of code that should be changed according to the port you are using:

1. **c3p0-config.xml**



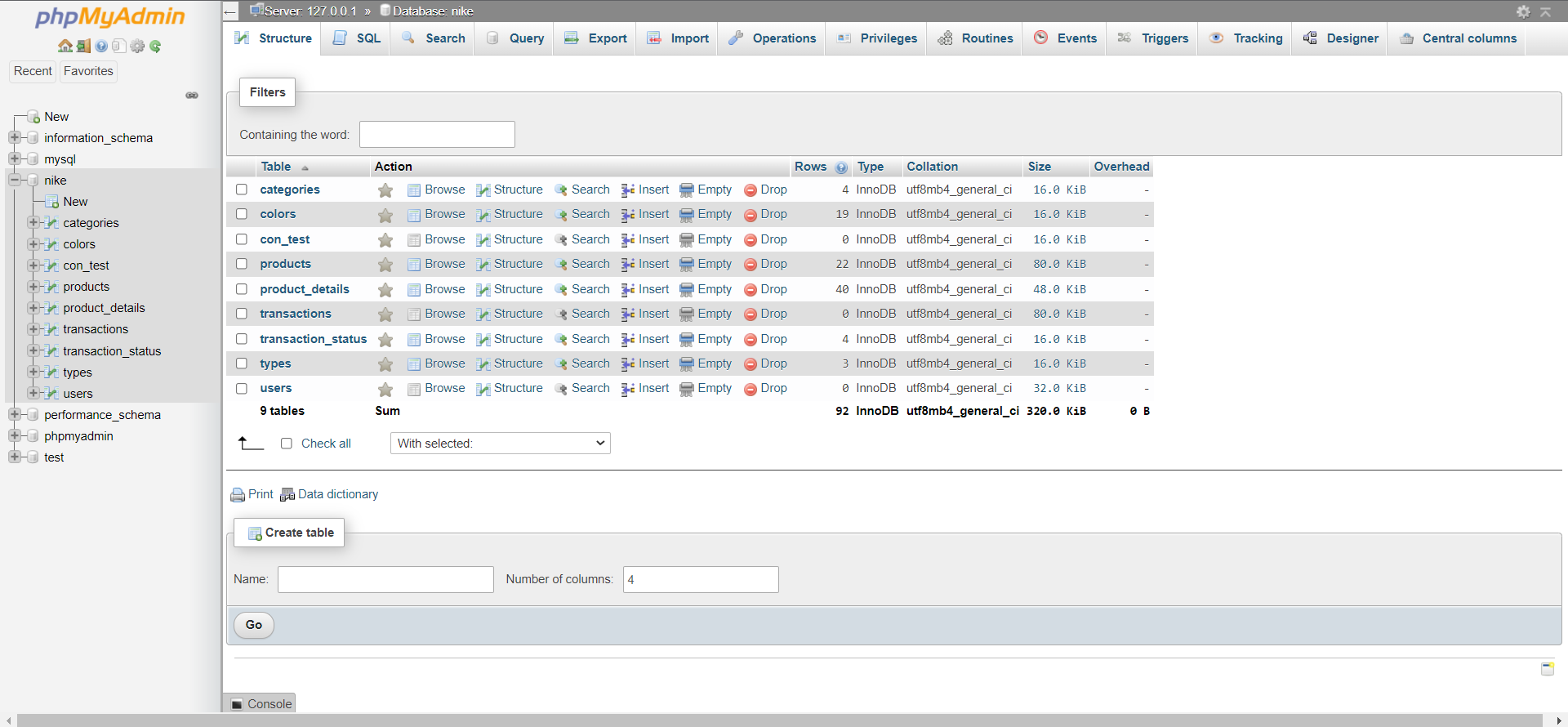
**Figure 57** Change Localhost Port, MySQL User Name, and MySQL User Password Based on the User Data & Port

1. **ApiConfig.kt**



**Figure 58** Change RESTful API Service Port in the ApiConfig.kt

After providing the DBMS, we need to prepare a database containing product data in the Nike store. In this study the author uses MySQL as a database service provider. Later, users can import the database by importing the **nike.sql** file that has been provided in the project folder.



**Figure 59** Nike Database in PHPMyAdmin

The last thing that isn’t less important to prepare is to enable the RESTful API service that I have created with J2EE. To activate this service, it is necessary to ensure that the Apache and MySQL services, as well as database have been enabled and added, so that the backend service can run properly.

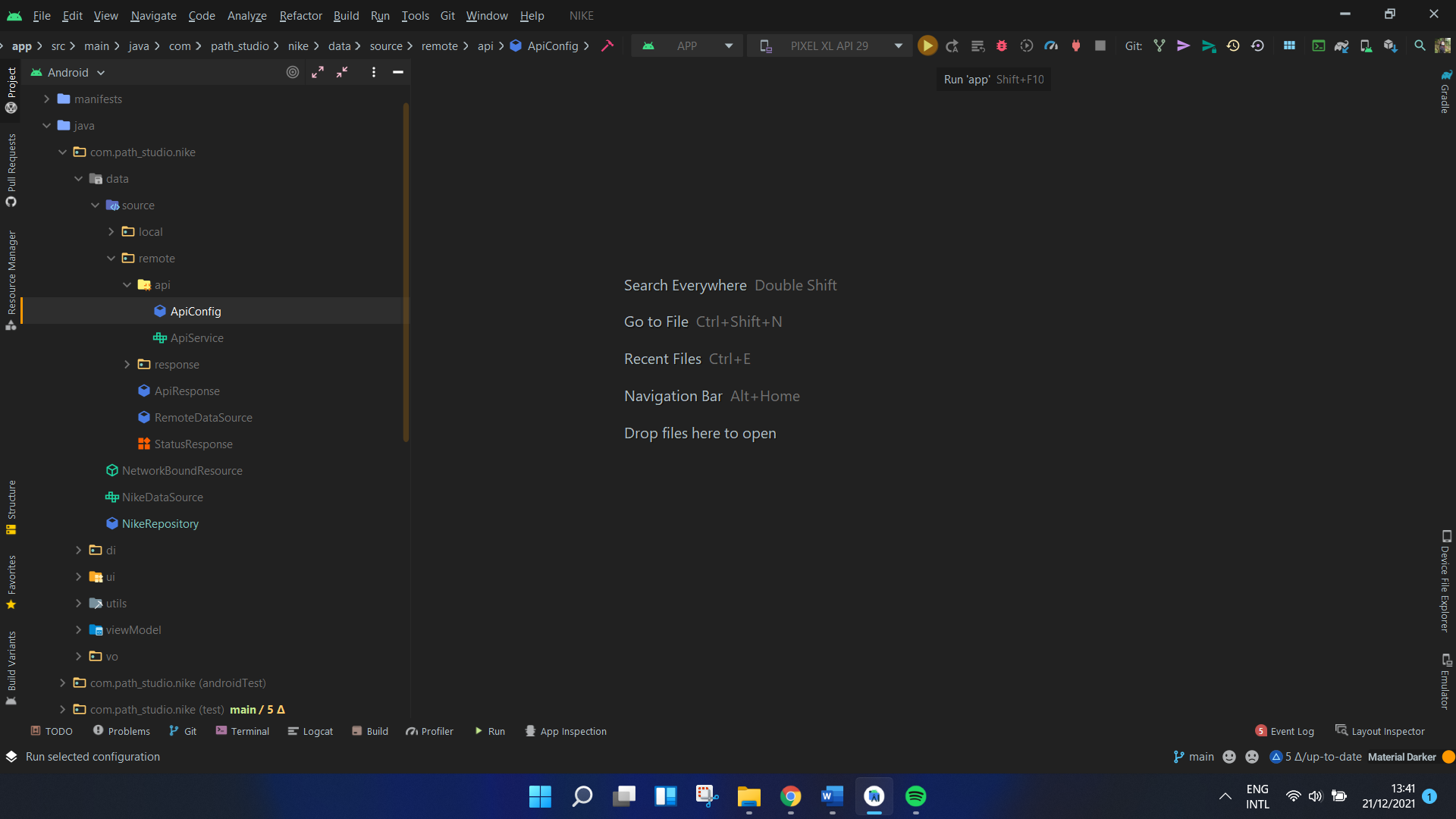
### Results of Android Application

The data service has been completed in the settings; the next step is to test the application. In this study I used at least 4 versions of the simulator to test the responsibility. The simulators used are the Samsung Note 10 with API 30, PIXEL XL with API 29, Samsung Galaxy J7 Prime with API 27, and Samsung Galaxy S10 with API 28.

***Table 3*** *Emulator Information for Testing Application*

|  |  |  |
| --- | --- | --- |
| **Number** | **Device Name** | **Description** |
| 1. | Pixel XL | **API Version**: 29  **Target**: Android 10.0 (Google APIs)  **Screen Resolution**: 1440 x 2560: 560dpi  **Memory**: 1536MB |
| 2. | Samsung Galaxy J7 Prime | **API Version**: 27  **Target**: Android 8.1 (Google APIs)  **Screen Resolution**: 1080 x 1920: xxhdpi  **Memory**: 1536MB |
| 3. | Samsung Galaxy S10 | **API Version**: 28  **Target**: Android 9.0 (Google APIs)  **Screen Resolution**: 1440 x 3040: xxhdpi  **Memory**: 1536MB |
| 4. | Samsung Note 10 | **API Version**: 30/Sv2  **Target**: Android API 32 (Google APIs)  **Screen Resolution**: 1080 x 2280: xxhdpi  **Memory**: 3072MB |

To open the project, we can open the Android Studio app, click file, click open, and go to the location where the Android Project stored. After finish open the project, we can run the project by select the Android Emulator and click run logo in the menu bar. More detail about how to run the project can be seen in Figure 60.



**Figure 60** Run Project Button in Android Studio

The details for each result can be seen in Table 4 until Table 8. For the explanation of how the application work, we can see it more detail in the attached video.

***Table 4*** *Application Result in Samsung Note 10 Emulator*

|  |  |
| --- | --- |
| **Screenshot of Page** | **Description** |
| Welcome Page | This is the page for the user that run this application in the first time. This page consists of slideshow of welcome text and image. To go to the main page, use can click ‘GET STARTED’ button. |
|  | This is the home page of Nike application. I this page user can see some collections of category shoes, latest shoes, and some shoes type like basketball shoes, high tops shoes, and sneakers shoes. For seeing all product based on the category, user can click ‘See All’ button. |
| Home Page |  |
|  | This is the See All result page. In this page user can see all shoes based on the selected category. |
| See All Page |  |
|  | This is search page. In this page user can search product by product name keywords. |
| Search Page with Search Result |  |
|  | This is favorite page that still don’t have any favorite product. In this page user can click ‘Check Our Shoes’ button to go to See All page. |
| Favorite Page with Empty Data |  |
| Favorite Page with Favorite Data | This is the favorite page that already have some product that user already liked. If user want to remove the product from favorite list, they can click the red heart icon. Later the program will remove and refresh the current favorite list. |
|  | This is the preview of shopping cart page that still don’t have any product in cart. If user click the checkout button in this stage (empty cart), this program will show empty cart toast. |
| Shopping Cart Page with No Data |  |
|  | This is the previews of shopping cart with some product in the cart database. After user add some product in the cart database and they try to find the products in this page, later this page will show the list of the product that already added within the total price of all the products in shopping cart. |
| Shopping Cart Page with Data |  |
|  | This is the product detail page. In this page user can like the product, see product image in the slide show, and see other product information such as product name, product color options, price, stock, type, etc. To see the details user can expand the bottom sheet by clicking the arrow icon in the information container. |
| Product Detail Page with Minimize Bottom Sheet |  |
|  | This is the product detail page. In this page user can like the product, see product image in the slide show, and see other product information such as product name, product color options, price, stock, type, etc. To see the details user can minimize the bottom sheet by clicking the arrow icon in the information container. |
| Product Detail Page with Maximize Bottom Sheet |  |
|  | This is the preview of product detail that still don’t put in the cart database. The program will select the first size, set the quantity to 1, and set the button to ‘Add to Bag’ button. |
| Product Detail with Product Not Yet Added in Cart |  |
|  | This is the preview of product detail that already put in the cart database. The program will select the selected size, quantity, and change the button to ‘Update to Bag’ button. |
| Product Detail with Product Already Added in Cart |  |
| Login Page |  |
| Sign Up Page |  |
| User Profile Page |  |
| User Edit Data Page |  |
| Delete Account Confirmation |  |
| Pending Payment with No Data |  |
| Pending Payment Page in Transaction History Page |  |
| Setting Page |  |
| Nike About Page |  |

***Table 5*** *Application Result in Pixel XL, Samsung S10, and Samsung J7 Prime Emulator*

# Chapter IV

**Summary and Suggestion**

# References

|  |  |
| --- | --- |
| [1] | H. Atha, "Java Vs Kotlin – Which Should You Choose For Android Development," moveoapps, 16 August 2018. [Online]. Available: https://www.moveoapps.com/blog/java-vs-kotlin/. [Accessed 20 December 2021]. |
| [2] | T. J. O'Leary, L. I. O'Leary and D. A. O'Leary, Computing Essentials 2017 (Complete Edition), New York: Mc Graw Hill Education, 2017. |
| [3] | M. Martin, "Prototyping Model in Software Engineering: Methodology, Process, Approach," Guru99, 1 November 2021. [Online]. Available: https://www.guru99.com/software-engineering-prototyping-model.html. [Accessed 20 December 2021]. |
| [4] | Android Developer Blog Team, "Use Android Jetpack to Accelerate Your App Development," Android Developer Blog, 08 May 2018. [Online]. Available: https://android-developers.googleblog.com/2018/05/use-android-jetpack-to-accelerate-your.html. [Accessed 21 December 2021]. |
| [5] | Androrid Developer Team, "Explore the Jetpack libraries by type," Androrid Developer, 2021. [Online]. Available: https://developer.android.com/jetpack/androidx/explorer. [Accessed 21 December 2021]. |
| [6] | RedHat Team, "What is a REST API?," RedHat, May 2020. [Online]. Available: https://www.redhat.com/en/topics/api/what-is-a-rest-api. [Accessed 21 December 2021]. |
| [7] | GeeksforGeeks Team, "MVVM (Model View ViewModel) Architecture Pattern in Android," GeeksforGeeks, 28 June 2021. [Online]. Available: https://www.geeksforgeeks.org/mvvm-model-view-viewmodel-architecture-pattern-in-android/. [Accessed 21 December 2021]. |
| [8] | Android Developer Team, "View Binding," Android Developer, 2021. [Online]. Available: https://developer.android.com/topic/libraries/view-binding. [Accessed 21 December 2021]. |

# Appendix