**A**

**Project I**

**Report** **On**

**Minimal Shop**



**Submitted for Partial Fulfillment of the Award of Bachelor of Technology (B.Tech) in CSE Kurukshetra University, Kurukshetra**

**Submitted By:**

**Nikhil Kumar**

**(1221263)**

**Submitted To:**

Er. Pinki Tanwar

(A.P, Deptt. of CSE)

**Department of Computer Science and Engineering**



**Seth Jai Parkash Mukand Lal Institute of Engineering & Technology**

**Affiliated to Kurukshetra University Kurukshetra**

# 

# Declaration

We hereby certify that the work which is being presented in the Project I Report entitled, **“Minimal Shop”** by me, Nikhil Kumar (1221263) in partial fulfillment of the requirements for the award of degree of **Bachelor of Technology in Computer Science Engineering** submitted in the **Department of Computer Science and Engineering at JMIT Radaur (Affiliated to Kurukshetra University, Kurukshetra, Haryana (India))** is an authentic record of my own work carried out under the supervision of **Er. Pinki Tanwar.** The matter presented in the report has not been submitted in any other University/Institute for the award of any degree.

NIKHIL KUMAR

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

**Er. Pinki Tanwar.**

A.P, Department of CSE. **JMIT Radaur**

**Countersigned By**

**Dr. Gaurav Sharma(HOD,CSE)**

The B.Tech Project I Viva-voce examination of **Nikhil Kumar (1221263)** was held on 24/05/2024 and is accepted.

**Er. Pinki Tanwar**

(Supervisor)

i

# 

# Acknowledgement

The writing of this project report has been assisted by the generous help of many people. I feel that I was very fortunate to receive assistance from them. I wish to express our sincere appreciation to them.

First and foremost, I am indebted to our principal supervisor, **Er. Pinki Tanwar** ,A.P , (Department of Computer Science and Engineering) of JMIT Radaur, who has been very supportive at every stage of our project completion. i wish to express our utmost gratitude to him for the invaluable advice and patience in reading, correcting and commenting on the drafts of this report and, more importantly, for his generosity which we have received throughout our project completion.

I wish to express our thanks to all staff members of JMIT Radaur, who also helped us in conducting this study.

Finally, I am particularly indebted to my dearest parents/guardians as without their generous assistance and love; this project could never have been completed.

**Nikhil Kumar 1221263**

**ii**

# Abstract

The "Minimal Shop" project is a mobile shopping application developed using Flutter, designed to enable small and local businesses to establish and expand their online presence. In the digital era, many small businesses struggle with the complexities of transitioning to online platforms. Minimal Shop addresses this challenge by providing a streamlined, efficient, and secure shopping solution tailored for small business needs.

The application features a minimalist design, focusing on essential functionalities to simplify the management of online stores for business owners and enhance the shopping experience for customers. Key features include streamlined product catalogs, secure user authentication, efficient search and filtering options, smooth checkout processes, and real-time order tracking. By integrating robust security measures, the application ensures safe and trustworthy transactions, addressing concerns about the protection of personal and financial data.

Leveraging Flutter's cross-platform capabilities, Minimal Shop offers a consistent user experience across both Android and iOS devices, reducing development time and cost. This approach ensures a high-performing and responsive application that meets modern consumer expectations.

By simplifying the process of going online, Minimal Shop empowers small and local businesses to reach a broader audience, increase sales, and compete effectively in the digital marketplace. This project represents a harmonious blend of minimalist design and powerful functionality, aimed at supporting the growth and success of small businesses in the competitive online environment.

iii

**Table of Contents**

[Declaration](#_gjdgxs) 2[Acknowledgement ii](#_30j0zll)

[Abstract iii](#_1fob9te)

[Table of Contents iv](#_2et92p0)

[List of Figures v](#_3znysh7)

|  |  |  |
| --- | --- | --- |
| **Chapter 1.** | 1. | Problem Background and its Context |
|  | 2.  3.. 4.  5. | System Objective  Functionality (Core and Enhanced) Academic, Technical & Economic Feasibility Risk Factors Identification & their Mitigation |
| **Chapter 2.** | 1.  2. | Primary Research Techniques and Analysis Secondary Research |
| **Chapter 3.** | 1.  2. | Requirement analysis  System Design |

**Chapter 4.** System Implementation

**Chapter 5.** Future Scope & Limitations

**Chapter 6.** Conclusion

**Chapter 7.** Bibliography

**Chapter 8.** Appendix

iv

# 

# List of figures

|  |  |  |
| --- | --- | --- |
| **Figure Number** | **Description** | **Page Number** |
| 4.1 | Flow diagram | 13 |
| 4.2 | Intro Page | 17 |
| 4.3 | Shop Page | 20 |
| 4.4 | Cart Page | 24 |
| 4.5 | My Drawer | 28 |
| 4.6 | My Button | 30 |
| 4.7 | My Product Tile | 35 |

v

**1.1 Problem Background and its Context**

**Small and local businesses often face significant challenges when attempting to establish and expand their online presence. Many lack the necessary resources and technical expertise to create and maintain a robust online platform, limiting their ability to reach a broader audience and compete with larger, more established e-commerce entities. This gap in digital capability often leaves small businesses struggling to keep pace with the rapid advancements in online shopping technologies.**

**The complexity of existing e-commerce solutions further exacerbates these difficulties. Current platforms are often loaded with numerous features and settings that can be overwhelming for small business owners, who may not have the time or technical background to manage them effectively. This complexity can result in inefficient store management, poor customer experiences, and ultimately, lost sales. A simpler, more intuitive solution is needed to help these businesses thrive online.**

**In addition to usability issues, small businesses frequently encounter problems with providing a consistent user experience across different devices. Customers expect seamless interactions whether they are shopping on a smartphone, tablet, or computer. However, many small business online stores fail to deliver this consistency, leading to customer frustration and decreased engagement. This inconsistency can hinder the business's ability to retain customers and grow its online presence.**

**Security concerns also pose a significant threat to small businesses venturing into online sales. Limited resources and expertise in cybersecurity make these businesses more vulnerable to data breaches and cyberattacks. Such incidents can severely damage a business's reputation and erode customer trust. Therefore, a secure, easy-to-manage platform is essential to protect both business and customer data, ensuring**

**1.2 System Objective**

**1.2.1 Simplified Online Store Management**

Enable small and local business owners to easily manage their online stores through a streamlined interface. The system will provide essential functionalities for inventory management, order processing, and customer engagement, minimizing the need for extensive technical knowledge.

**1.2.2 Consistent User Experience**

Ensure a consistent and seamless shopping experience for customers across different devices, including smartphones and tablets. By utilizing Flutter's cross-platform capabilities, the system will maintain a uniform look and feel, enhancing user satisfaction and engagement.

**1.2.3 Enhanced Security**

Incorporate robust security measures to protect both business and customer data. The system will feature secure payment gateways, encrypted data storage, and stringent authentication processes to safeguard transactions and build trust.

**1.2.4 Efficient Product Search and Navigation**

Provide users with efficient search functionalities and intuitive navigation options. The system will include advanced filtering and sorting mechanisms to help customers quickly find products, thereby improving the overall shopping experience.

**1.2.5 Scalability and Flexibility**

Develop a scalable system that can accommodate the growth of small and local businesses. The system will be designed to handle increasing product listings, customer data, and transaction volumes, as well as adapt to future business needs and technological advancements.

**1.2.6 Mobile Integration**

Seamlessly integrate the system into mobile applications developed using Flutter for both Android and iOS platforms. By leveraging the widespread use of smartphones, the system will extend its reach to a broader audience, including customers in remote areas.

**1.3 Functionality (Core and Enhanced)**

**1.3.1 Product Catalog Management**

Core Functionality: Business owners can easily manage their product catalog within the Minimal Shop system. They can add new products, update existing ones, and remove items as needed. Product information such as name, description, price, and images can be easily edited through an intuitive interface.

**1.3.2 Secure Payment Processing**

Core Functionality: The system ensures secure payment processing for both business owners and customers. It integrates with trusted payment gateways to facilitate smooth and reliable transactions. Customers can securely make purchases using various payment methods, including credit/debit cards, digital wallets, and other online payment options.

**1.3.3 Search and Filtering Options**

Enhanced Functionality: The system provides advanced search and filtering options to help customers find products quickly and easily. Users can search for products by keywords, categories, price range, and other relevant criteria. Additionally, they can apply filters to narrow down search results based on specific preferences, such as size, color, or brand.

**1.3.4 Personalized Recommendations**

Enhanced Functionality: Utilizing machine learning algorithms, the system offers personalized product recommendations to customers based on their browsing and purchasing history. By analyzing past interactions and preferences, the system suggests relevant products that align with each customer's interests and needs, enhancing the overall shopping experience.

**1.3.5 Inventory Management**

Enhanced Functionality: Business owners can efficiently manage their inventory levels and track stock availability within the Minimal Shop system. They receive notifications when stock levels are low, allowing them to restock products in a timely manner and avoid out-of-stock situations. Additionally, the system provides insights into sales trends and popular products to help optimize inventory management strategies.

**1.4 Academic, Technical & Economic Feasibility**

**1.4.1 Academic Feasibility**

The Minimal Shop project holds significant academic potential, serving as an educational tool for individuals interested in small business management and e-commerce. By providing a simplified platform for learning about online store operations, marketing strategies, and customer engagement, the project caters to both beginners and experts in the field. Interactive tutorials and guides enhance experiential learning, fostering engagement with essential business concepts. This academic focus positions the project as a valuable resource for self-learners, entrepreneurs, and educational institutions seeking practical insights into small business operations and digital entrepreneurship.

**1.4.2 Technical Feasibility**

The project demonstrates strong technical feasibility through the utilization of advanced mobile app development technologies. Leveraging frameworks such as Flutter, the project ensures seamless functionality and compatibility across various devices and operating systems. The integration of secure payment processing systems and robust backend technologies enhances the user experience and data security. Furthermore, compatibility with emerging technologies and industry standards reinforces the project's technical robustness and future readiness, all achievable by a single developer.

**1.4.3 Economic Feasibility**

The project exhibits promising economic prospects for sustainable growth and revenue generation. Adopting a freemium model, offering basic shop management functionalities for free and premium features through subscription or in-app purchases, establishes a viable monetization strategy. Value-added services, such as advanced analytics and personalized recommendations, attract a dedicated user base willing to invest in their business growth. Moreover, strategic partnerships with industry stakeholders, such as payment processors and business service providers, create additional revenue streams through referral programs and collaborations. As a solo developer, the project presents a compelling economic opportunity for entrepreneurship and sustainable revenue generation.

**1.5 Risk Factors Identification & their Mitigation**

**1.5.1 Technical Challenges**

* **Identification**

Technical complexities may emerge during the development and maintenance phases, potentially causing delays or system malfunctions.

* **Mitigation**

As a solo developer, it's crucial to plan thoroughly and utilize agile development methodologies to address challenges promptly. Implement comprehensive testing procedures to identify and resolve issues early. Utilize online resources, forums, and developer communities for technical support. Establish contingency plans to mitigate potential technical issues.

**1.5.2 Security Concerns**

* **Identification**

Ensuring data security for both business owners and customers can be challenging, especially in handling sensitive information like payment details.

* **Mitigation**

Implement robust security measures such as encrypted data storage, secure payment gateways, and stringent authentication processes. Regularly update security protocols and stay informed about the latest security threats and best practices. Use trusted third-party security tools and services when necessary.

**1.5.3 User Adoption and Engagement**

* **Identification**

Low user adoption rates or lack of sustained engagement may impede the project's success.

* **Mitigation**

Conduct extensive user research and usability testing to understand user needs and preferences. Incorporate user feedback to improve the application's features and overall user experience. Develop marketing strategies to raise awareness and highlight the benefits of the app to the target audience, ensuring it meets their expectations and requirements.

**1.5.4 Dependency on Third-Party Services**

* **Identification**

Reliance on third-party services or APIs for critical functionalities may expose the project to risks such as service outages or changes in service terms.

* **Mitigation**

Diversify dependencies where feasible and maintain contingency plans to mitigate disruptions in third-party services. Regularly monitor service performance and establish communication channels with service providers to address issues promptly.

**2.1 Primary Research Techniques and Analysis**

**2.1.1 Surveys**

* **Technique**

Conduct online or in-person surveys to gather quantitative data on user preferences, needs, and behaviors related to managing small business online stores and e-commerce platforms.

* **Analysis**

Analyze survey responses using statistical methods to identify trends, patterns, and correlations. Use tools such as spreadsheets or survey analysis software to organize and visualize data for insights.

**2.1.3 Focus Groups**

* **Technique**

Organize focus group discussions with small business owners or e-commerce experts to facilitate in-depth discussions on specific topics related to online store management and customer engagement.

* **Analysis**

Record focus group sessions and analyze them thematically to identify key insights, preferences, and suggestions. Use structured frameworks or coding techniques to analyze group dynamics and interactions.

**2.1.4 Observational Studies**

* **Technique**

Conduct observational studies in real-world settings, such as local markets or small business environments, to observe how business owners manage their operations and interact with customers.

* **Analysis**

Record observational data and analyze it to identify patterns, trends, and insights into business owner behavior and customer interactions. Use observational data to inform design decisions and user experience enhancements in the application.

**2.2 Secondary Research**

**2.2.1 Literature Review**

* **Source**

Academic journals, books, and research papers related to small business management, e-commerce, and mobile application development.

* **Analysis**

Synthesize findings from literature sources to identify trends, best practices, and potential challenges in the field of online store management and digital entrepreneurship. Explore academic insights to inform our project's feature set and user experience design.

**2.2.2 Market Analysis**

* **Source**

Industry reports, market research publications, and online databases focusing on the e-commerce, small business, and mobile application sectors.

* **Analysis**

Identify market opportunities, competitive landscape, and consumer behavior trends to inform our project's positioning, differentiation strategy, and target market segmentation. Assess market dynamics to identify potential challenges and mitigate risks in launching and scaling the application.

**2.2.3 Case Studies**

* **Source**

Case studies, success stories, and user testimonials from existing e-commerce platforms, online marketplaces, and small business management solutions.

* **Analysis**

Extract insights, best practices, and lessons learned from case studies to inform our project's design decisions, feature prioritization, and user engagement strategies. Learn from successful implementations and avoid common pitfalls to enhance the project's chances of success.

**2.2.4 Technology Trends**

* **Source**

Technology blogs, industry reports, and online forums discussing emerging technologies relevant to mobile application development and e-commerce.

* **Analysis**

Evaluate the potential applications, benefits, and limitations of emerging technologies such as artificial intelligence, augmented reality, and blockchain for our project's development roadmap and long-term strategy. Stay informed about technological advancements to leverage innovative solutions and stay competitive in the market.

**2.2.5 User Feedback and Reviews**

* **Source**

User forums, app store reviews, and social media discussions related to existing e-commerce platforms, online stores, and mobile applications.

* **Analysis**

Analyze user feedback and reviews to identify common pain points, feature requests, and usability concerns in existing e-commerce platforms. Incorporate user insights into our project's design iterations and feature prioritization to ensure a user-centric approach and maximize user satisfaction and engagement..

**3.1 Requirement Analysis**

**3.1.1 User Requirements**

Users of the application expect a seamless experience in managing their online stores, including easy product listing, secure payment processing, and efficient order management. Additionally, they require access to insights and analytics to track sales performance, customer behavior, and inventory levels. User-friendly features such as intuitive navigation, personalized recommendations, and responsive customer support are essential to enhance the user experience. The application should cater to users with varying levels of technical proficiency, ensuring accessibility and ease of use.

**3.1.2 Functional Requirements**

The application must facilitate easy product catalog management, allowing users to add, edit, and delete product listings effortlessly. Secure payment processing through trusted payment gateways is essential to ensure smooth and secure transactions for both business owners and customers. Efficient order management functionalities, including order tracking, status updates, and automated notifications, are necessary to streamline the fulfillment process. Additionally, integration with analytics tools to provide insights into sales trends, customer behavior, and inventory management is crucial for informed decision-making and business growth.

**3.1.3 Non-Functional Requirements**

Efficient performance is paramount, requiring fast and responsive functionality for product listing, payment processing, and order management processes. Robust data encryption, authentication mechanisms, and stringent data privacy measures must be implemented to safeguard sensitive business and customer information. Scalable architecture and infrastructure are necessary ts, operating systems, and screen sizes is crucial to maximize accessibility and user reach.

**3.1.4 Integration Requirements**

Integration with external APIs for payment processing, shipping providers, and inventory management systems will enhance the application's functionality and efficiency. Enabling users to integrate with third-party analytics tools and marketing platforms will provide valuable insights into business performance and customer engagement. Incorporating social media integration for sharing product listings, promotions, and customer reviews will enhance brand visibility and facilitate customer acquisition. Additionally, integration with customer relationship management (CRM) systems will enable seamless customer interactions and relationship management.

**3.2 System Design**

**3.2.1 Architecture**

The system will follow a client-server architecture, where the mobile application serves as the client and interacts with a centralized server. The server will host the application logic, database, and business logic.

**3.2.2 Components**

The client-side application will be developed using Flutter for cross-platform compatibility, ensuring a consistent user experience across different devices. Backend services will be implemented using frameworks such as Node.js or Django, facilitating data processing and business logic handling.

**3.2.3 Functional Modules**

The system will consist of several functional modules. The product management module will handle operations related to adding, editing, and deleting product listings. The payment processing module will facilitate secure transactions using trusted payment gateways. The order management module will manage order processing, tracking, and notifications. Additionally, the analytics module will provide insights into sales performance, customer behavior, and inventory management.

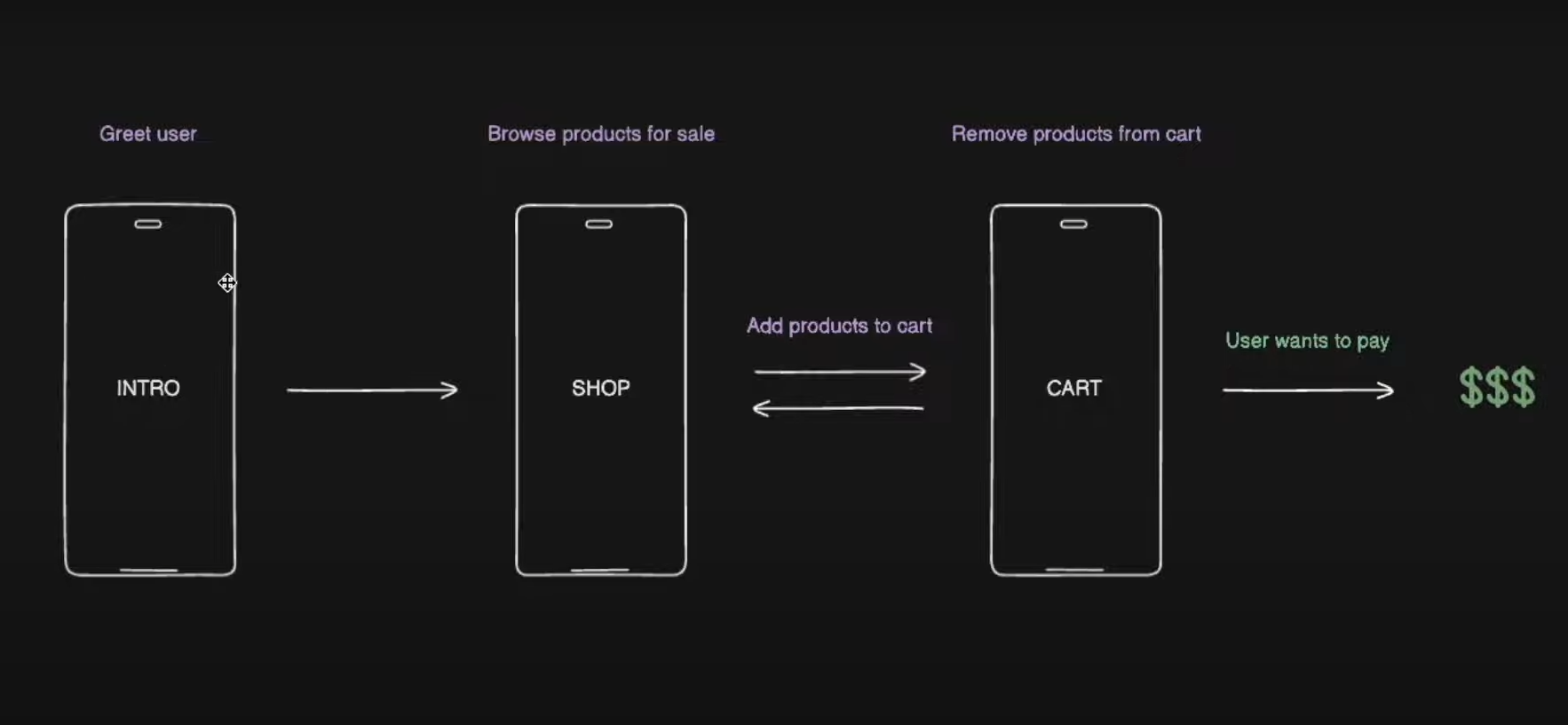
**3.2.4 Integration Points**

Integration with external APIs for payment processing, shipping providers, and analytics tools will enhance the application's functionality and efficiency. Social media integration will enable users to share product listings, promotions, and customer reviews, enhancing brand visibility and customer engagement. Integration with customer relationship management (CRM) systems will facilitate seamless customer interactions and relationship management.

**3.2.5 Scalability and Performance**

Integration with external APIs for payment processing, shipping providers, and analytics tools will enhance the application's functionality and efficiency. Social media integration will enable users to share product listings, promotions, and customer reviews, enhancing brand visibility and customer engagement. Integration with customer relationship management (CRM) systems will facilitate seamless customer interactions and relationship management.

**4.1 Flow Diagram of Minimal Shop**



**Fig 4.1 Flow Diagram of Minimal Shop**

The user starts at the introduction page, where they learn about the Minimal Shop app's features. They move to the shop page to browse products, select items, and add them to their cart. From there, they proceed to the cart page to review their selections. Finally, they move to the payment process, where they choose a payment method, confirm their order, and receive a confirmation message after successful payment.

**4.2 Project code**

**Main fuction:- main.dart**

import 'package:flutter/material.dart';

import 'package:minimalshop/models/shop.dart';

import 'package:minimalshop/pages/cart\_page.dart';

import 'package:minimalshop/pages/shop\_page.dart';

import 'package:minimalshop/themes/light\_modes.dart';

import 'package:provider/provider.dart';

import 'pages/intro\_page.dart';

void main() {

  runApp(

    ChangeNotifierProvider(

      create: (context) => Shop(),

      child: const MyApp(),

    ),

  );

}

class MyApp extends StatelessWidget {

  const MyApp({super.key});

  @override

  Widget build(BuildContext context) {

    return MaterialApp(

        debugShowCheckedModeBanner: false,

        home: IntroPage(),

        theme: lightMode,

        routes: {

          '/intro\_page': (context) => const IntroPage(),

          '/shop\_page': (context) => const ShopPage(),

          '/cart\_page': (context) => const CartPage(),

        });

  }

}

**Intro page :- intro\_page.dart**

import 'package:flutter/material.dart';

import '../components/my\_button.dart';

class IntroPage extends StatelessWidget {

  const IntroPage({super.key});

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      backgroundColor: Theme.of(context).colorScheme.background,

      body: Center(

        child: Column(

          mainAxisAlignment: MainAxisAlignment.center,

          children: [

            //logo

            Icon(

              Icons.shopping\_bag,

              size: 72,

              color: Theme.of(context).colorScheme.inversePrimary,

            ),

            const SizedBox(height: 25),

            //title

            Text(

              "MInimal Shop",

              style: TextStyle(

                fontWeight: FontWeight.bold,

                fontSize: 24,

              ),

            ),

            const SizedBox(height: 10),

            //subtitle

            Text(

              "Premium Quality product",

              style: TextStyle(

                color: Theme.of(context).colorScheme.inversePrimary,

              ),

            ),

            const SizedBox(height: 25),

            //button

            MyButton(

              onTap: () => Navigator.pushNamed(context, '/shop\_page'),

              child: Icon(Icons.arrow\_forward),

            )

          ],

        ),

      ),

    );

  }

}



**Fig 4.2 Intro Page**

**Shop page :- shop\_page.dart**

import 'package:flutter/material.dart';

import 'package:minimalshop/components/my\_drawer.dart';

import 'package:minimalshop/components/my\_product\_tile.dart';

//import 'package:minimalshop/models/product.dart';

import 'package:minimalshop/models/shop.dart';

import 'package:provider/provider.dart';

class ShopPage extends StatelessWidget {

  const ShopPage({super.key});

  @override

  Widget build(BuildContext context) {

    //access the product in shop

    final products = context.watch<Shop>().shop;

    return Scaffold(

        appBar: AppBar(

          backgroundColor: Colors.transparent,

          elevation: 0,

          foregroundColor: Theme.of(context).colorScheme.inversePrimary,

          title: const Text("Shop page"),

          actions: [

            // go to cart

            IconButton(

                onPressed: () => Navigator.pushNamed(context, '/cart\_page'),

                icon: const Icon(Icons.shopping\_cart))

          ],

        ),

        drawer: const MyDrawer(),

        backgroundColor: Theme.of(context).colorScheme.background,

        body: ListView(

          children: [

            const SizedBox(height: 25),

            //shop subtitle

            Center(

                child: Text(

              "Pick from a selected list of premium products",

              style: TextStyle(

                  color: Theme.of(context).colorScheme.inversePrimary),

            )),

            //product list

            SizedBox(

              height: 550,

              child: ListView.builder(

                  itemCount: products.length,

                  scrollDirection: Axis.horizontal,

                  padding: const EdgeInsets.all(15),

                  itemBuilder: (context, index) {

                    // get each individual product from aoo

                    final product = products[index];

                    //return as aproject tile UI

                    return MyProductTile(product: product);

                  }),

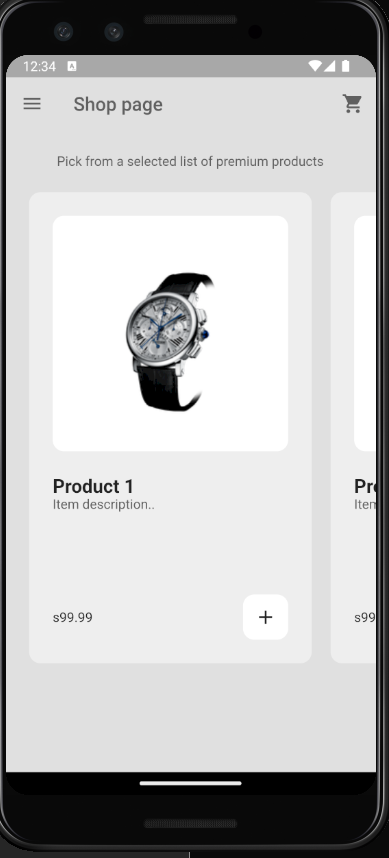
            ),

          ],

        ));

  }

}

****

**Fig 4.3 Shop Page**

**Cart page :- cart\_page.dart**

import 'package:flutter/material.dart';

import 'package:minimalshop/components/my\_button.dart';

import 'package:minimalshop/models/product.dart';

import 'package:minimalshop/models/shop.dart';

import 'package:provider/provider.dart';

class CartPage extends StatelessWidget {

  const CartPage({super.key});

  // remove item from cart

  void removeItemFromCart(BuildContext context, Product product) {

    {

      //show a dailog box for confirmation to remove

      showDialog(

          context: context,

          builder: (context) => AlertDialog(

                content: const Text("Remove this item from your cart?"),

                actions: [

                  //cancel button

                  MaterialButton(

                    onPressed: () => Navigator.pop(context),

                    child: Text("Cancel"),

                  ),

                  // yes button

                  MaterialButton(

                    onPressed: () {

                      Navigator.pop(context);

                      context.read<Shop>().removeFromCart(product);

                    },

                    child: Text("yes"),

                  ),

                ],

              ));

    }

  }

  // user pressed the pay button

  void payButtonPressed(BuildContext context) {

    showDialog(

      context: context,

      builder: (context) => const AlertDialog(

        content: Text("user wants to pay! connect it payment backend...."),

      ),

    );

  }

  @override

  Widget build(BuildContext context) {

    // get access to the cart

    final cart = context.watch<Shop>().cart;

    return Scaffold(

      appBar: AppBar(

        backgroundColor: Colors.transparent,

        elevation: 0,

        foregroundColor: Theme.of(context).colorScheme.inversePrimary,

        title: Text("Cart Page"),

      ),

      backgroundColor: Theme.of(context).colorScheme.background,

      body: Column(children: [

        //cart list

        Expanded(

          child: cart.isEmpty

              ? const Center(child: Text("your cart is empty.."))

              : ListView.builder(

                  itemCount: cart.length,

                  itemBuilder: (context, index) {

                    // get individual item

                    final item = cart[index];

                    //return as alist tile

                    return ListTile(

                      title: Text(item.name),

                      subtitle: Text(item.price.toStringAsFixed(2)),

                      trailing: IconButton(

                        icon: const Icon(Icons.remove),

                        onPressed: () => removeItemFromCart(context, item),

                      ),

                    );

                  }),

        ),

        Padding(

          padding: const EdgeInsets.all(50),

          child: MyButton(

            onTap: () => payButtonPressed(context),

            child: Text("pay now"),

          ),

        ),

      ]),

    );

  }

}

****

**Fig 4.3 Cart Page**

**My Components**

**my\_drawer.dart**

import 'package:flutter/material.dart';

import 'package:minimalshop/components/my\_list\_tile.dart';

//import 'my\_list\_tile.dart';

class MyDrawer extends StatelessWidget {

  const MyDrawer({super.key});

  @override

  Widget build(BuildContext context) {

    return Drawer(

      backgroundColor: Theme.of(context).colorScheme.background,

      child: Column(

        mainAxisAlignment: MainAxisAlignment.spaceBetween,

        children: [

          Column(

            children: [

              //drawe header: logo

              DrawerHeader(

                child: Center(

                  child: Icon(

                    Icons.shopping\_bag,

                    size: 72,

                    color: Theme.of(context).colorScheme.inversePrimary,

                  ),

                ),

              ),

              const SizedBox(height: 25),

              //shop tile

              MyListTile(

                text: "Shop",

                icon: Icons.home,

                onTap: () => Navigator.pop(context),

              ),

              //cart tile

              MyListTile(

                text: "Cart",

                icon: Icons.shopping\_cart,

                onTap: () {

                  // pop drawer first

                  Navigator.pop(context);

                  //go to cart page

                  Navigator.pushNamed(context, '/cart\_page');

                },

              ),

            ],

          ),

          //exit shop tile

          Padding(

            padding: const EdgeInsets.only(bottom: 25.0),

            child: MyListTile(

              text: "Exit",

              icon: Icons.logout,

              onTap: () => Navigator.pushNamedAndRemoveUntil(

                  context, '/intro\_page', (route) => false),

            ),

          ),

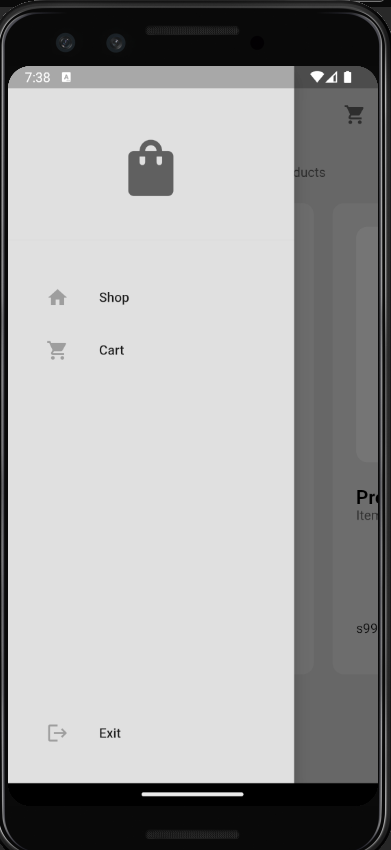
        ],

      ),

    );

  }

}



**Fig 4.4 My Drawer**

**my\_button.dart**

import 'package:flutter/material.dart';

class MyButton extends StatelessWidget {

  final void Function()? onTap;

  final Widget child;

  const MyButton({super.key, required this.onTap, required this.child});

  @override

  Widget build(BuildContext context) {

    return GestureDetector(

      onTap: onTap,

      child: Container(

        decoration: BoxDecoration(

          color: Theme.of(context).colorScheme.primary,

          borderRadius: BorderRadius.circular(12),

        ),

        padding: const EdgeInsets.all(25),

        child: child,

      ),

    );

  }

}



**Fig 4.4 My Button**

**my\_product\_tile.dart**

import 'package:flutter/material.dart';

import 'package:minimalshop/models/product.dart';

import 'package:minimalshop/models/shop.dart';

import 'package:provider/provider.dart';

class MyProductTile extends StatelessWidget {

  final Product product;

  const MyProductTile({

    super.key,

    required this.product,

  });

  //add to cart button pressed

  void addToCart(BuildContext context) {

    //show a dailog box for confirmation

    showDialog(

        context: context,

        builder: (context) => AlertDialog(

              content: Text("add this item to your cart?"),

              actions: [

                MaterialButton(

                  onPressed: () => Navigator.pop(context),

                  child: Text("Cancel"),

                ),

                // yes button

                MaterialButton(

                  onPressed: () {

                    Navigator.pop(context);

                    context.read<Shop>().addToCart(product);

                  },

                  child: Text("yes"),

                ),

              ],

            ));

  }

  @override

  Widget build(BuildContext context) {

    return Container(

      decoration: BoxDecoration(

        color: Theme.of(context).colorScheme.primary,

        borderRadius: BorderRadius.circular(12),

      ),

      margin: const EdgeInsets.all(10),

      padding: const EdgeInsets.all(25),

      width: 300,

      child: Column(

          crossAxisAlignment: CrossAxisAlignment.start,

          mainAxisAlignment: MainAxisAlignment.spaceBetween,

          children: [

            Column(

              crossAxisAlignment: CrossAxisAlignment.start,

              children: [

                //product image

                AspectRatio(

                  aspectRatio: 1,

                  child: Container(

                      decoration: BoxDecoration(

                        color: Theme.of(context).colorScheme.secondary,

                        borderRadius: BorderRadius.circular(12),

                      ),

                      width: double.infinity,

                      padding: const EdgeInsets.all(25),

                      child: Image.asset(product.imagepath)),

                ),

                const SizedBox(height: 25),

                //product name

                Text(

                  product.name,

                  style: TextStyle(

                    fontWeight: FontWeight.bold,

                    fontSize: 20,

                  ),

                ),

                //product description

                Text(

                  product.description,

                  style: TextStyle(

                    color: Theme.of(context).colorScheme.inversePrimary,

                  ),

                ),

              ],

            ),

            //product price + add to cart button

            Row(

              mainAxisAlignment: MainAxisAlignment.spaceBetween,

              children: [

                // product price

                Text('\s' + product.price.toStringAsFixed(2)),

                //add to cart button

                Container(

                  decoration: BoxDecoration(

                    color: Theme.of(context).colorScheme.secondary,

                    borderRadius: BorderRadius.circular(12),

                  ),

                  child: IconButton(

                    onPressed: () => addToCart(context),

                    icon: Icon(Icons.add),

                  ),

                )

              ],

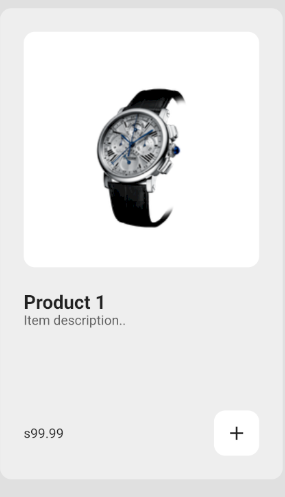
            ),

          ]),

    );

  }

}



**Fig 4.4 My Product Tile**

**my\_list\_tile.dart**

import 'package:flutter/material.dart';

class MyListTile extends StatelessWidget {

  final String text;

  final IconData icon;

  final void Function()? onTap;

  const MyListTile({

    super.key,

    required this.text,

    required this.icon,

    required this.onTap,

  });

  @override

  Widget build(BuildContext context) {

    return Padding(

      padding: const EdgeInsets.only(left: 25.0),

      child: ListTile(

        leading: Icon(

          icon,

          color: Colors.grey,

        ),

        title: Text(text),

        onTap: onTap,

      ),

    );

  }

}

**5.1 Future Scope**

#### The Minimal Shop project aims to bring small and local businesses online, helping them expand their reach and streamline their operations. As the project progresses, several future enhancements can be considered to ensure continued growth and improvement.

#### 5.1.1. Mobile Application Integration

#### Real-Time Order Management: Business owners can receive real-time notifications about new orders, inventory levels, and customer inquiries, allowing for efficient management.

#### Support for Remote Businesses: The mobile application will support businesses in remote areas, providing them with tools to reach a wider customer base and manage their operations effectively.

#### Intuitive User Interface

#### A user-friendly interface will guide business owners through setting up their online store, adding products, managing orders, and processing payments.

#### 5.1.2. Expansion of Business Services

#### Diverse Product Listings: Initially, the platform may cater to specific types of businesses. In the future, the range will expand to include various categories, from retail and food services to handcrafted goods and more

#### Regular Feature Updates: Establishing partnerships with business associations and user communities will help gather feedback and regularly update the platform with new features and improvements.

#### Continuous Improvement: The system will be periodically updated with the latest business tools and services to improve its capabilities and ensure it meets evolving market demands.

#### 5.1.3. Advanced Features and Enhancements

#### Enhanced Analytics: Future versions of the platform could include advanced analytics tools, providing businesses with deeper insights into customer behavior, sales trends, and market opportunities.

#### Integrated Marketing Tools: By integrating marketing tools, such as social media marketing and email campaigns, businesses can reach a wider audience and engage with their customers more effectively.

#### User Feedback Loop Implement a feedback mechanism where customers can rate and review their shopping experience, helping businesses improve their services.

**5.2 Limitation**

**5.2.2 Model Limitations**

* **Overfitting** The platform's recommendation algorithms may perform well on training data but fail to generalize to new, unseen data, particularly if the training data lacks diversity.
* **Product and Service Range:** Initially, the platform might support a limited range of products and services, which could reduce its attractiveness to a broader audience.
* **Complex Requirements:** Some businesses may have complex requirements that the initial version of the platform may not fully support, potentially limiting its utility for such users.
* **Similar Business Needs:** Different businesses may have similar requirements that are hard to distinguish algorithmically, leading to generic solutions that may not fully satisfy specific business needs.

**5.2.2 Technological Limitations**

* **Device Performance** Running complex functionalities on mobile devices can be resource-intensive and may not perform well on lower-end devices, potentially limiting the platform’s user base.
* **Image Quality** If the platform includes features that rely on image uploads (e.g., for product listings), the quality of the images captured by the user's smartphone can affect the accuracy of any image-based analyses.

**5.2.3 Practical Usage Challenges**

* **User Onboarding:** Users may face challenges in correctly setting up their online stores, including adding products and managing inventory, which can lead to frustration and abandonment of the platform.
* **Interpretation of Analytics:** Business owners might misinterpret the analytics and insights provided by the platform, potentially leading to poor decision-making.

**6.1 Conclusion**

The development of the Minimal Shop project represents a significant step forward in empowering small and local businesses by providing them with an accessible platform to establish and expand their online presence. This Flutter-based mobile application facilitates the transition from traditional brick-and-mortar operations to dynamic digital storefronts, enabling business owners to reach a broader customer base and enhance their market competitiveness.

The future scope of Minimal Shop is highly promising. Plans include adding advanced features such as enhanced analytics, customer relationship management tools, and integration with social media platforms to boost user engagement and business growth. These enhancements will broaden the application's applicability and reliability across diverse business contexts.

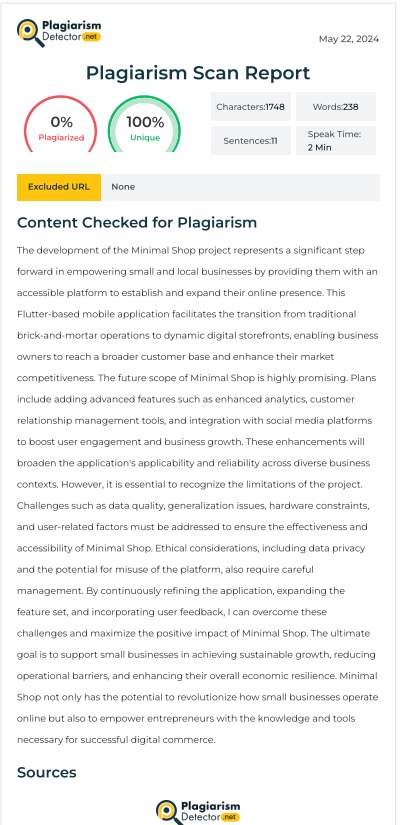
However, it is essential to recognize the limitations of the project. Challenges such as data quality, generalization issues, hardware constraints, and user-related factors must be addressed to ensure the effectiveness and accessibility of Minimal Shop. Ethical considerations, including data privacy and the potential for misuse of the platform, also require careful management.

By continuously refining the application, expanding the feature set, and incorporating user feedback, I can overcome these challenges and maximize the positive impact of Minimal Shop. The ultimate goal is to support small businesses in achieving sustainable growth, reducing operational barriers, and enhancing their overall economic resilience. Minimal Shop not only has the potential to revolutionize how small businesses operate online but also to empower entrepreneurs with the knowledge and tools necessary for successful digital commerce.

**7. Bibliography**

1. Hughes, Google. "Flutter Documentation." Available at: https://flutter.dev/docs
2. Google. "Dart Programming Language." Available at: https://dart.dev/guides
3. Google. "Firebase Documentation." Available at: https://firebase.google.com/docs
4. Karthik Ponnam, "Flutter & Firebase Tutorial for Beginners | Build a Complete App," YouTube, Published on June 12, 2021. Available at: https://www.youtube.com/watch?v=DPniy2dw-Vg
5. Academind, "Flutter Tutorial for Beginners - Build a Flutter App with Firebase," YouTube, Published on September 25, 2019. Available at: https://www.youtube.com/watch?v=9HM1M1R2GC0
6. Udacity. "Flutter & Dart: From Zero to Hero." Available at: https://www.udacity.com/course/flutter-and-dart-from-zero-to-hero--ud905
7. Firebase Summit, "Image Upload and Management with Firebase," YouTube, Published on October 21, 2020. Available at: https://www.youtube.com/watch?v=m\_Lh6NStU10
8. GitHub. "Flutter Awesome - A curated list of Flutter packages and projects." Available at: https://github.com/Solido/awesome-flutter

**Plagiarism Report:**

****