**UGANDA MARTYRS UNIVERSITY, NKOZI**

**NAME: BIRUNGI EMMANUEL**

**REG NO; 2019-B071-11861**

**NAME; MARIA OLIVIA, YEAR THREE**

**FINAL EXAM**

**Project Title: Book Store App**

**Overview**

The Book Store app is an e-commerce application that allows users to browse and purchase books online. The app will provide a user-friendly interface for customers to search, discover, and buy books from a vast collection. The app will also enable bookstores to manage their inventory, process orders, and track sales.

**Features**

**1. User Registration and Login:**

- Users can create an account and log in to the app.

- Users can reset their passwords and update their account information.

**2. Book Search and Discovery:**

- Users can search for books by title, author, genre, and keyword.

- The app will suggest books based on the user's search history and preferences.

**3. Book Details and Reviews:**

- Users can view book details, including title, author, description, price, and ratings.

- Users can read and write reviews for books.

**4. Shopping Cart and Checkout:**

- Users can add books to their cart and view their cart summary.

- Users can proceed to checkout and pay for their orders.

**5. Order Management:**

- Users can view their order history and track their orders.

- Users can cancel or return orders.

**6. Bookstore Management:**

- Bookstores can manage their inventory and update book information.

- Bookstores can process orders and track sales.

**7. Payment Gateway:**

- The app will integrate a secure payment gateway for transactions.

**8. Push Notifications:**

- The app will send push notifications for order updates, promotions, and new book releases.

**Technical Requirements:**

1. Programming Language: Flutter (for mobile app)

2. Database: MongoDB

3. Payment Gateway: Stripe

4. Push Notification Service: Firebase Cloud Messaging

5. Development Tools: Android Studio, Visual Studio Code, and GitHub

**Design Requirements:**

1. User Interface: Clean, intuitive, and responsive design

2. Color Scheme: A combination of blue and white, with accents of orange

3. Logo: A stylized book icon with the app name

4. Fonts: Open Sans and Lato

5. Imagery: High-quality book covers and author images

**Timeline:**

1. Research and Planning: 2 weeks

2. Design and Prototyping: 4 weeks

3. Development: 16 weeks

4. Testing and Debugging: 4 weeks

5. Launch and Deployment: 2 weeks

**Conclusion:**

The Book Store app will provide a seamless and enjoyable experience for book lovers to discover and purchase books online. With its user-friendly interface, robust features, and secure payment gateway, the app will become a go-to destination for book enthusiasts. We believe that this project will be a great success and look forward to working with you to bring it to life.

**Here are some common challenges faced while implementing a mobile app:**

**1. Technical Challenges:**

- Integrating with third-party APIs and services

- Handling device and OS variations

- Optimizing performance and battery life

- Ensuring security and data privacy

**2. Design and User Experience Challenges:**

- Creating an intuitive and user-friendly interface

- Balancing functionality and simplicity

- Adapting to different screen sizes and resolutions

- Ensuring consistency across platforms (iOS and Android)

**3. Development Time and Resource Challenges:**

- Managing timelines and deadlines

- Allocating resources (time, budget, personnel)

- Prioritizing features and functionality

- Handling changes in project scope or requirements

**4. Testing and Quality Assurance Challenges:**

- Ensuring thorough testing and debugging

- Handling different device and OS variations

- Conducting effective user testing and feedback

- Meeting app store guidelines and regulations

**5. Marketing and Distribution Challenges:**

- Creating effective marketing and promotion strategies

- Reaching target audience and increasing visibility

- Managing app store optimization (ASO)

- Handling user feedback and ratings

**6. Maintenance and Update Challenges:**

- Regularly updating and improving the app

- Fixing bugs and issues

- Adding new features and functionality

- Ensuring compatibility with future OS and device updates

**7. Budget and Resource Constraints:**

- Managing development costs and budget

- Allocating resources (time, personnel, equipment)

- Prioritizing features and functionality

- Handling changes in project scope or requirements

**8. Team Collaboration and Communication Challenges:**

- Ensuring effective communication and collaboration

- Managing different roles and responsibilities

- Handling conflicts and disagreements

**-** Meeting project goals and deadlines

**9. User Adoption and Retention Challenges:**

- Encouraging user engagement and retention

- Providing effective onboarding and tutorials

- Handling user feedback and suggestions

- Ensuring consistent user experience

**10. Data Security and Privacy Challenges:**

- Ensuring secure data storage and transmission

- Handling user data and privacy concerns

- Meeting data protection regulations and guidelines

- Preventing data breaches and cyber attacks

**Here are the key features and functionality of the Book Store app:**

**1. User Registration and Login:**

- Users can create an account and log in to the app.

- Users can reset their passwords and update their account information.

**2. Book Search and Discovery:**

- Users can search for books by title, author, genre, and keyword.

- The app will suggest books based on the user's search history and preferences.

**3. Book Details and Reviews:**

- Users can view book details, including title, author, description, price, and ratings.

- Users can read and write reviews for books.

**4. Shopping Cart and Checkout:**

- Users can add books to their cart and view their cart summary.

- Users can proceed to checkout and pay for their orders.

**5. Order Management:**

- Users can view their order history and track their orders.

- Users can cancel or return orders.

**6. Bookstore Management:**

- Bookstores can manage their inventory and update book information.

- Bookstores can process orders and track sales.

**7. Payment Gateway:**

- The app will integrate a secure payment gateway for transactions.

**8. Push Notifications:**

- The app will send push notifications for order updates, promotions, and new book releases.

**9. Book Recommendations:**

- The app will suggest books to users based on their reading preferences and purchase history.

**10. Wishlists:**

- Users can create and manage wishlists of books they want to read.

**Here are the benefits of the Book Store app:**

1. Convenience: Users can browse and purchase books anytime, anywhere, and have them delivered to their doorstep.

2. Wide Selection: The app offers a vast collection of books from various genres, authors, and publishers, making it a one-stop shop for book lovers.

3. Personalized Recommendations: The app provides personalized book recommendations based on users' reading preferences and purchase history, helping them discover new books and authors.

4. Easy Discovery: Users can search for books by title, author, genre, or keyword, and browse through curated lists and recommendations.

5. Book Reviews and Ratings: Users can read and write reviews, and rate books on a 5-star scale, helping them make informed purchasing decisions.

6. Community Building: The app offers book clubs, discussion forums, and author events, allowing users to connect with fellow readers and authors.

7. Author and Book Information: The app provides detailed information about authors and books, including biographies, book summaries, and reviews.

8. Book Samples: Users can read samples of books before purchasing, ensuring they find the perfect book for their interests.

9. Audiobooks and E-books: The app offers audiobook and e-book versions of books, catering to different reading preferences.

10. Exclusive Offers and Discounts: The app provides exclusive offers, discounts, and promotions, making book purchasing more affordable and rewarding.

11. Order Management: Users can track their orders, cancel or return books, and manage their account information easily.

12. Secure Payment: The app ensures secure payment transactions, protecting users' financial information.

13. Push Notifications: The app sends push notifications for order updates, promotions, and new book releases, keeping users informed and engaged.

14. User-Friendly Interface: The app has a clean, intuitive design, making it easy for users to navigate and find what they're looking for.

15. Accessibility: The app is accessible on multiple devices, including smartphones, tablets, and desktops, allowing users to access their books anywhere.

**Here are the highlights of the value that the Book Store app adds to users:**

- Discover new books and authors: The app's personalized recommendations and curated lists help users discover new books and authors they may not have found otherwise.

- Save time and effort: The app's search functionality and filters make it easy for users to find books that match their interests, saving them time and effort.

- Access a vast collection: The app offers a vast collection of books from various genres, authors, and publishers, making it a one-stop shop for book lovers.

- Read anywhere, anytime: The app allows users to access their books on multiple devices, making it easy to read anywhere, anytime.

- Connect with fellow readers: The app's book clubs, discussion forums, and author events provide a platform for users to connect with fellow readers and authors.

- Track orders and manage account: The app's order management and account management features make it easy for users to track their orders and manage their account information.

- Exclusive offers and discounts: The app provides exclusive offers and discounts, making book purchasing more affordable and rewarding.

- Secure payment: The app ensures secure payment transactions, protecting users' financial information.

- Personalized experience: The app's personalized recommendations and curated lists provide a personalized experience for each user.

- Access to book reviews and ratings: The app's book reviews and ratings feature helps users make informed purchasing decisions.

- Author and book information: The app provides detailed information about authors and books, including biographies, book summaries, and reviews.

- Book samples: The app's book samples feature allows users to read samples of books before purchasing, ensuring they find the perfect book for their interests.

**Here are some questions that the audience may have, along with answers that address their concerns:**

Q: How do I find books that match my interests?

Q: Can I read books on multiple devices?

Q: How do I know if a book is good?

Q: Can I try before I buy?

Q: How do I connect with other readers?

Q: Is my financial information secure?

Q: Can I manage my account and orders easily?

Q: Are there exclusive offers and discounts?

Q: Can I access my books offline?

Q: Is the app user-friendly?

Q: Can I customize my reading experience?

Q: Are there any additional features?

**CODES**

**SPLASHSCREEN**

import 'package:flutter/material.dart';  
  
class SplashScreen extends StatefulWidget {  
 @override  
 \_SplashScreenState createState() => \_SplashScreenState();  
}  
  
class \_SplashScreenState extends State<SplashScreen> {  
 @override  
 void initState() {  
 super.initState();  
 Future.delayed(Duration(seconds: 3), () {  
 Navigator.pushReplacement(  
 context,  
 MaterialPageRoute(builder: (context) => LoginPage()),  
 );  
 });  
 }  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 body: Center(  
 child: Text("Welcome to my App"),  
 ),  
 );  
 }  
}

**LOGIN PAGE**

import 'package:flutter/material.dart';  
  
class LoginPage extends StatefulWidget {  
 @override  
 \_LoginPageState createState() => \_LoginPageState();  
}  
  
class \_LoginPageState extends State<LoginPage> {  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text('Login Page'),  
 ),  
 body: Center(  
 child: Text('Login Page Content'),  
 ),  
 );  
 }  
}  
class LoginPage extends StatefulWidget {  
 @override  
 \_LoginPageState createState() => \_LoginPageState();  
}  
  
class \_LoginPageState extends State<LoginPage> {  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text('Login Page'),  
 ),  
 body: Center(  
 child: Column(  
 children: [  
 // Login form  
 TextButton(  
 onPressed: () {  
 // Navigate to SignUpPage  
 Navigator.push(  
 context,  
 MaterialPageRoute(builder: (context) => SignUpPage()),  
 );  
 },  
 child: Text('Sign Up'),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

**SIGN UP PAGE**

package org.example  
  
class SignUp {  
 class SignUpPage extends StatefulWidget {  
 @override  
 \_SignUpPageState createState() => \_SignUpPageState();  
 }  
  
 class \_SignUpPageState extends State<SignUpPage> {  
 final \_formKey = GlobalKey<FormState>();  
 final \_storage = FlutterSecureStorage();  
 String \_username, \_email, \_password, \_confirmPassword;  
  
 @override  
 Widget build(BuildContext context) **{** return Scaffold(  
 appBar: AppBar(  
 title: Text('Sign Up Page'),  
 ),  
 body: Padding(  
 padding: const EdgeInsets.all(20.0),  
 child: Form(  
 key: \_formKey,  
 child: Column(  
 children: [  
 TextFormField(  
 decoration: InputDecoration(  
 labelText: 'Username',  
 border: OutlineInputBorder(),  
 ),  
 validator: (value) **{** if (value.isEmpty) {  
 return 'Please enter a username';  
 }  
 return null;  
 **}**,  
 onSaved: (value) => \_username = value,  
 ),  
 SizedBox(height: 20),  
 TextFormField(  
 decoration: InputDecoration(  
 labelText: 'Email',  
 border: OutlineInputBorder(),  
 ),  
 validator: (value) **{** if (value.isEmpty || !value.contains('@')) {  
 return 'Please enter a valid email';  
 }  
 return null;  
 **}**,  
 onSaved: (value) => \_email = value,  
 ),  
 SizedBox(height: 20),  
 TextFormField(  
 decoration: InputDecoration(  
 labelText: 'Password',  
 border: OutlineInputBorder(),  
 ),  
 obscureText: true,  
 validator: (value) **{** if (value.isEmpty || value.length < 8) {  
 return 'Please enter a password with at least 8 characters';  
 }  
 return null;  
 **}**,  
 onSaved: (value) => \_password = value,  
 ),  
 SizedBox(height: 20),  
 TextFormField(  
 decoration: InputDecoration(  
 labelText: 'Confirm Password',  
 border: OutlineInputBorder(),  
 ),  
 obscureText: true,  
 validator: (value) **{** if (value.isEmpty || value != \_password) {  
 return 'Please enter the same password';  
 }  
 return null;  
 **}**,  
 onSaved: (value) => \_confirmPassword = value,  
 ),  
 SizedBox(height: 20),  
 ElevatedButton(  
 onPressed: () async **{** if (\_formKey.currentState.validate()) {  
 \_formKey.currentState.save();  
 // Sign up logic goes here  
 // ...  
 await \_storage.write(key: 'username', value: \_username);  
 await \_storage.write(key: 'email', value: \_email);  
 await \_storage.write(key: 'password', value: \_password);  
 Navigator.pushReplacement(  
 context,  
 MaterialPageRoute(builder: (context) => HomePage()),  
 );  
 }  
 **}**,  
 child: Text('Sign Up'),  
 ),  
 SizedBox(height: 20),  
 TextButton(  
 onPressed: () **{** // Login logic goes here  
 // ...  
 **}**,  
 child: Text('Login'),  
 ),  
 TextButton(  
 onPressed: () **{** // Forgot password logic goes here  
 // ...  
 **}**,  
 child: Text('Forgot Password'),  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 **}** }  
  
}  
class SignUpPage extends StatefulWidget {  
 @override  
 \_SignUpPageState createState() => \_SignUpPageState();  
}  
  
class \_SignUpPageState extends State<SignUpPage> {  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text('Sign Up Page'),  
 ),  
 body: Center(  
 child: Text('Sign Up Page Content'),  
 ),  
 );  
 }  
}  
class SignUpPage extends StatefulWidget {  
 @override  
 \_SignUpPageState createState() => \_SignUpPageState();  
}  
  
class \_SignUpPageState extends State<SignUpPage> {  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text('Sign Up Page'),  
 ),  
 body: Center(  
 child: Column(  
 children: [  
 // Sign-up form  
 TextButton(  
 onPressed: () {  
 // Navigate to HomePage  
 Navigator.pushReplacement(  
 context,  
 MaterialPageRoute(builder: (context) => HomePage()),  
 );  
 },  
 child: Text('Sign Up'),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}  
  
class \_SignUpPageState extends State<SignUpPage> {  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text('Sign Up Page'),  
 ),  
 body: Center(  
 child: Column(  
 children: [  
 // Sign-up form  
 TextButton(  
 onPressed: () {  
 // Navigate to HomePage  
 Navigator.pushReplacement(  
 context,  
 MaterialPageRoute(builder: (context) => HomePage()),  
 );  
 },  
 child: Text('Sign Up'),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

**HOME PAGE**

import 'package:flutter/material.dart';  
  
class MyApp extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 home: Scaffold(  
 appBar: AppBar(  
 title: Text('Card Design'),  
 ),  
 body: ListView.builder(  
 itemCount: 10,  
 itemBuilder: (context, index) {  
 return Card(  
 elevation: 5,  
 shape: RoundedRectangleBorder(  
 borderRadius: BorderRadius.circular(10),  
 ),  
 color: Colors.white,  
 margin: EdgeInsets.all(16),  
 child: Column(  
 mainAxisSpacing: 30,  
 crossAxisSpacing: 40,  
 crossAxisCount: 2,  
 children: [  
 itemDashboard('Health', CupertinoIcons.play\_rectangle, Colors.deepOrange),  
 itemDashboard('Sports', CupertinoIcons.play\_rectangle, Colors.green),  
 itemDashboard('Education', CupertinoIcons.play\_rectangle, Colors.purple),  
 itemDashboard('Entertainment', CupertinoIcons.play\_rectangle, Colors.red),  
 Button(  
 onPressed: () {  
 // Action  
 },  
 child: Text('Button $index'),  
 ),  
 ],  
 ),  
 );  
 },  
 ),  
 ),  
 );  
 }  
}  
  
class \_HomePageState extends State<HomePage> {  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text('Home Page'),  
 ),  
 body: Center(  
 child: Text('Home Page Content'),  
 ),  
 );  
 }  
}

**MAIN DART**

import 'package:flutter/material.dart';  
import 'SplashScreen.dart';  
import 'LoginPage.dart';  
  
void main() {  
 runApp(MyApp());  
}  
  
class MyApp extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 title: 'Flutter Demo',  
 theme: ThemeData(  
 primarySwatch: Colors.blue,  
 ),  
 home: SplashScreen(), // Set SplashScreen as the initial route  
 routes: {  
 '/login': (context) => LoginPage(),  
 },  
 );  
 }  
}  
class MyApp extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 title: 'Flutter Demo',  
 theme: ThemeData(  
 primarySwatch: Colors.blue,  
 ),  
 home: LoginPage(), // Set LoginPage as the initial route  
 routes: {  
 '/login': (context) => LoginPage(),  
 '/signUp': (context) => SignUpPage(),  
 },  
 );  
 }  
}  
class MyApp extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 title: 'Flutter Demo',  
 theme: ThemeData(  
 primarySwatch: Colors.blue,  
 ),  
 home: LoginPage(), // Set LoginPage as the initial route  
 routes: {  
 '/login': (context) => LoginPage(),  
 '/signUp': (context) => SignUpPage(),  
 },  
 );  
 }  
}  
class MyApp extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 title: 'Flutter Demo',  
 theme: ThemeData(  
 primarySwatch: Colors.blue,  
 ),  
 home: SignUpPage(), // Set SignUpPage as the initial route  
 routes: {  
 '/signUp': (context) => SignUpPage(),  
 '/home': (context) => HomePage(),  
 },  
 );  
 }  
}