Q2. Assignment- Flutter:

Here's a code outline for developing the e-commerce application in Flutter based on your requirements. This outline will cover each feature step-by-step.

1. Project Setup

1. Create a New Flutter Project

```bash
flutter create ecommerce_app
cd ecommerce_app
```

2. Add Dependencies

Update `pubspec.yaml` with necessary packages:

```yaml
dependencies:
  flutter:
    sdk: flutter
  http: ^0.14.0
  provider: ^6.0.0
  flutter_secure_storage: ^5.0.2 # For storing user session tokens
  shared_preferences: ^2.0.7    # For storing user login state
```

3. Create Folder Structure

```
lib/
├── models/
├── providers/
├── screens/
├── widgets/
```

└── services/

```

## 2. Implementing the Homepage with Infinite Scrolling

**Model for Product:**

```dart
class Product {
  final int id;
  final String title;
  final String description;
  final double price;
  final String image;
  final double rating;

  Product({required this.id, required this.title, required this.description, required this.price, required this.image, required this.rating});

  factory Product.fromJson(Map<String, dynamic> json) {
    return Product(
      id: json['id'],
      title: json['title'],
      description: json['description'],
      price: json['price'].toDouble(),
      image: json['image'],
      rating: json['rating']['rate'].toDouble(),
    );
  }
}
```
```

Fetching Products (Service):

```dart
import 'dart:convert';
import 'package:http/http.dart' as http;
import '../models/product.dart';

class ProductService {
  final String apiUrl = 'https://fakestoreapi.com/products';

  Future<List<Product>> fetchProducts(int start, int limit) async {
    final response = await http.get(Uri.parse('$apiUrl?limit=$limit&offset=$start'));
    if (response.statusCode == 200) {
      List<dynamic> data = json.decode(response.body);
      return data.map((item) => Product.fromJson(item)).toList();
    } else {
      throw Exception('Failed to load products');
    }
  }
}
```

Provider for Product State Management:

```dart
import 'package:flutter/material.dart';
import '../models/product.dart';
import '../services/product_service.dart';

class ProductProvider with ChangeNotifier {
```

```
  final ProductService _productService = ProductService();
  List<Product> _products = [];
  bool _isLoading = false;

  List<Product> get products => _products;
  bool get isLoading => _isLoading;

  Future<void> fetchProducts(int start, int limit) async {
    _isLoading = true;
    notifyListeners();

    try {
      List<Product> newProducts = await _productService.fetchProducts(start, limit);
      _products.addAll(newProducts);
    } catch (e) {
      print(e);
    } finally {
      _isLoading = false;
      notifyListeners();
    }
  }
}
```

Homepage with Infinite Scrolling:

```dart
import 'package:flutter/material.dart';
import 'package:provider/provider.dart';
import '../providers/product_provider.dart';
```

```
class HomePage extends StatefulWidget {
  @override
  _HomePageState createState() => _HomePageState();
}

class _HomePageState extends State<HomePage> {
  final ScrollController _scrollController = ScrollController();
  int _currentLimit = 10;

  @override
  void initState() {
    super.initState();
    final productProvider = Provider.of<ProductProvider>(context, listen: false);
    productProvider.fetchProducts(0, _currentLimit);

    _scrollController.addListener(() {
      if (_scrollController.position.pixels == _scrollController.position.maxScrollExtent) {
        _currentLimit += 10;
        productProvider.fetchProducts(productProvider.products.length, _currentLimit);
      }
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('E-commerce App')),
      body: Consumer<ProductProvider>(
```

```
      builder: (context, productProvider, child) {
        if (productProvider.isLoading && productProvider.products.isEmpty) {
          return Center(child: CircularProgressIndicator());
        }
        return ListView.builder(
          controller: _scrollController,
          itemCount: productProvider.products.length + (productProvider.isLoading ? 1 : 0),
          itemBuilder: (context, index) {
            if (index == productProvider.products.length) {
              return Center(child: CircularProgressIndicator());
            }
            final product = productProvider.products[index];
            return ListTile(
              leading: Image.network(product.image, width: 50),
              title: Text(product.title),
              subtitle: Text('\$${product.price} - Rating: ${product.rating}'),
              onTap: () {
                Navigator.pushNamed(context, '/product', arguments: product.id);
              },
            );
          },
        );
      },
    ),
  );
 }
}
```

3. Product Detail Page

Product Detail Screen:

```dart
import 'package:flutter/material.dart';
import 'package:provider/provider.dart';
import '../providers/product_provider.dart';
import '../models/product.dart';

class ProductDetailPage extends StatelessWidget {
  final int productId;

  ProductDetailPage({required this.productId});

  @override
  Widget build(BuildContext context) {
    final product = Provider.of<ProductProvider>(context).products.firstWhere((prod) =>
prod.id == productId);

    return Scaffold(
      appBar: AppBar(title: Text(product.title)),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            Image.network(product.image, height: 250),
            SizedBox(height: 20),
            Text(product.title, style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold)),
            SizedBox(height: 10),
```

```
          Text('\$${product.price}', style: TextStyle(fontSize: 20, color: Colors.green)),
          SizedBox(height: 20),
          Text(product.description),
          SizedBox(height: 20),
          Text('Rating: ${product.rating}'),
        ],
      ),
    ),
  );
 }
}
```

4. Product Search Functionality

Search Functionality:

```dart
class SearchPage extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  final productProvider = Provider.of<ProductProvider>(context);
  final TextEditingController _controller = TextEditingController();

  return Scaffold(
   appBar: AppBar(
    title: TextField(
     controller: _controller,
     decoration: InputDecoration(hintText: 'Search for products...'),
     onChanged: (query) {
```

```
          productProvider.fetchProducts(0, 10); // Update this method to search products based on query.
        },
      ),
    ),
    body: Consumer<ProductProvider>(
      builder: (context, productProvider, child) {
        return ListView.builder(
          itemCount: productProvider.products.length,
          itemBuilder: (context, index) {
            final product = productProvider.products[index];
            return ListTile(
              leading: Image.network(product.image, width: 50),
              title: Text(product.title),
              subtitle: Text('\$${product.price} - Rating: ${product.rating}'),
              onTap: () {
                Navigator.pushNamed(context, '/product', arguments: product.id);
              },
            );
          },
        );
      },
    ),
  );
 }
}
```

5. User Authentication

For user authentication, you can use `flutter_secure_storage` for storing tokens securely.

User Authentication Screen:

```dart
class AuthService {
  final storage = FlutterSecureStorage();

  Future<void> login(String email, String password) async {
    // Implement API call to authenticate user
    // On success:
    await storage.write(key: 'token', value: 'user_token_here');
  }

  Future<void> logout() async {
    await storage.delete(key: 'token');
  }
}
```

Login Screen:

```dart
class LoginPage extends StatelessWidget {
  final TextEditingController _emailController = TextEditingController();
  final TextEditingController _passwordController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    final authService = AuthService();
```

```
    return Scaffold(
      appBar: AppBar(title: Text('Login')),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          children: [
            TextField(controller: _emailController, decoration: InputDecoration(labelText: 'Email')),
            TextField(controller: _passwordController, decoration: InputDecoration(labelText: 'Password'), obscureText: true),
            ElevatedButton(
              onPressed: () async {
                await authService.login(_emailController.text, _passwordController.text);
                Navigator.pushReplacementNamed(context, '/home');
              },
              child: Text('Login'),
            ),
          ],
        ),
      ),
    );
  }
}
```

6. Product Sorting and Filtering

Product Sorting and Filtering:

```dart
// Add sorting and filtering functionality in the ProductProvider and update the fetchProducts method accordingly.
```

```

## 7. Cart Functionality

**Cart Model:**

```dart
class CartItem {
  final Product product;
  int quantity;

  CartItem({required this.product, required this.quantity});
}
```

Cart Provider:

```dart
class CartProvider with ChangeNotifier
```
```