Flutter Discovery App Documentation

1. Project Overview

The Flutter Discovery App is designed to showcase a list of items fetched from a mock API. It features a paginated list with unlimited scroll, providing a visually appealing user interface and smooth user experience.

2. Technologies Used

- Flutter
- Dart
- Android Studio

3. Project Structure

The project is organized as follows:

- lib/: Contains the Dart source code.
- api/: API integration files.
- models/: Data model classes.
- widgets/: Reusable UI components.
- screens/: Flutter screens or pages.

4. Setup Instructions

To run the app locally, follow these steps:

```
# Clone the repository

git clone https://github.com/amancd/api_data_fetch_flutter


# Navigate to the project directory

cd flutter_discovery_app


# Install dependencies

flutter pub get


# Run the app
```

```
flutter run
```

5. API Integration

The app fetches data from the mock API endpoint:

- Endpoint: https://api-stg.together.buzz/mocks/discovery
- Query Parameters: page and limit

6. Pagination Implementation

Pagination is implemented by using the page and limit query parameters in API requests. The app loads additional items as the user scrolls and stops when the API returns an empty response.

7. UI/UX Design

The app features a visually appealing design with smooth animations and transitions. It includes a search bar, a paginated list of discovery items, and a "Load More" button for seamless navigation.

8. Code Quality and Best Practices

The code follows Flutter best practices and is structured for readability and maintainability. Meaningful variable names and comments are used to enhance code understanding.

9. Error Handling

The app gracefully handles network failures or API errors, displaying appropriate error messages to the user when necessary.

10. Testing

The app has been tested under various scenarios, including edge cases, to ensure correct functionality.

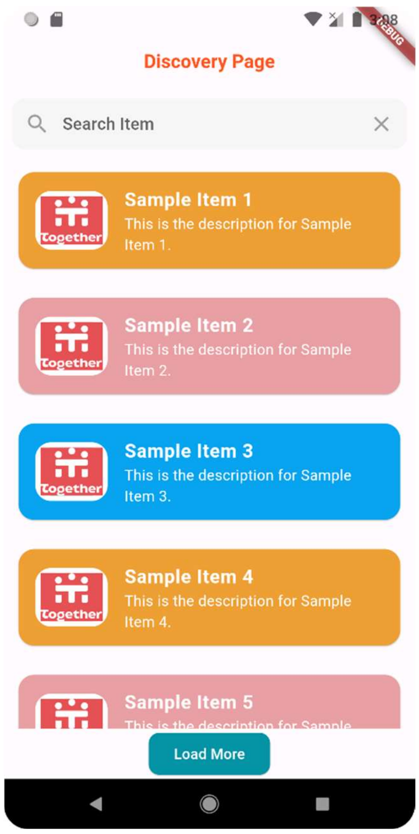
11. Additional Features

Search feature has been implemented.

12. Project Completion

The Flutter Discovery App is complete and meets the basic requirements. Future plans may include enhancements or additional features based on project needs.

13. Screenshot


Discovery Page

Search Item

Sample Item 1
This is the description for Sample Item 1.

Sample Item 2
This is the description for Sample Item 2.

Sample Item 3
This is the description for Sample Item 3.

Sample Item 4
This is the description for Sample Item 4.

Sample Item 5
This is the description for Sample

API file.dart

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

class DiscoveryApi {
  static const baseUrl = 'https://api-stg.together.buzz/mocks/discovery';

  Future<List<DiscoveryItem>> fetchItems(int page, int limit) async {
    try {
      final response = await
http.get(Uri.parse('$baseUrl?page=$page&limit=$limit'));

      if (response.statusCode == 200) {
        final dynamic data = json.decode(response.body);

        // Check if 'data' key exists and its value is not null
        if (data != null && data['data'] != null) {
```

```dart
          // Use 'as List<dynamic>' to handle dynamic data
          final List<dynamic> itemsList = data['data'] as List<dynamic>;

          // Map items and convert them to DiscoveryItem
          final List<DiscoveryItem> items = itemsList.map((item) =>
DiscoveryItem(
            id: item['id'],
            title: item['title'],
            description: item['description'],
            imageUrl: item['image_url'],
          )).toList();

          return items;
        } else {
          // Handle the case where 'data' key is null or doesn't exist
          return [];
        }
      } else {
        throw Exception('Failed to load items');
      }
    } catch (e) {
      // Handle other exceptions
      print('Error: $e');
      throw Exception('Failed to load items');
    }
  }
}
```

Discovery_item.dart

```dart
class DiscoveryItem {
  final int id;
  final String title;
  final String description;
  final String imageUrl;

  DiscoveryItem({
    required this.id,
    required this.title,
    required this.description,
    required this.imageUrl,
  });
}
```

Discovery_page.dart

```dart
import 'package:flutter/material.dart';
import '../api/discovery_api.dart';
import '../models/discovery_item.dart';
import '../widgets/discovery_card.dart';

class DiscoveryPage extends StatefulWidget {
  @override
  _DiscoveryPageState createState() => _DiscoveryPageState();
}

class _DiscoveryPageState extends State<DiscoveryPage> {
  final DiscoveryApi _api = DiscoveryApi();
  final TextEditingController _searchController = TextEditingController();
  List<DiscoveryItem> _items = [];
  int _page = 1;
  final int _limit = 10;

  @override
  void initState() {
    super.initState();
    _loadData();
  }

  Future<void> _loadData() async {
    try {
      final List<DiscoveryItem> newItems = await _api.fetchItems(_page, _limit);

      setState(() {
        _items.addAll(newItems);
        _page++;
      });
    } catch (e) {
      // Handle errors gracefully
      print('Error: $e');
    }
  }

  void _loadMore() {
    _loadData();
  }

  void _searchItems(String query) {
    setState(() {
      _items = _items.where((item) {
        final titleLowerCase = item.title.toLowerCase();
        final descriptionLowerCase = item.description.toLowerCase();
        final searchLowerCase = query.toLowerCase();
        return titleLowerCase.contains(searchLowerCase) ||
            descriptionLowerCase.contains(searchLowerCase);
      }).toList();
    });
  }

  void _clearSearch() {
    setState(() {
      _searchController.clear();
      _loadData();
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Discovery Page', style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold,
color: Colors.deepOrange)),
        centerTitle: true,
      ),
      body: Column(
        children: [
          Padding(
            padding: const EdgeInsets.all(8.0),
            child: TextField(
              onChanged: _searchItems,
              decoration: InputDecoration(
                hintText: 'Search Item',
                filled: true,
                fillColor: Theme.of(context).brightness == Brightness.light
                    ? const Color(0xFFf6f6f6)
                    : const Color(0xFF282828),
                enabledBorder: OutlineInputBorder(
                  borderSide: BorderSide.none,
                  borderRadius: BorderRadius.circular(12.0),
                ),
                focusedBorder: OutlineInputBorder(
                  borderSide: BorderSide.none,
                  borderRadius: BorderRadius.circular(12.0),
                ),
                contentPadding:
                const EdgeInsets.symmetric(horizontal: 16.0),
                prefixIcon:
                const Icon(Icons.search, color: Colors.grey),
                suffixIcon: GestureDetector(
                  onTap: () {
                    _clearSearch();
                  },
                  child: const Padding(
                    padding: EdgeInsets.all(8.0),
                    child: Icon(Icons.clear, color: Colors.grey),
                  )
                ),
              ),
            ),
          ),
          Expanded(
            child: ListView.builder(
              itemCount: _items.length,
              itemBuilder: (context, index) => DiscoveryCard(item: _items[index]),
            ),
          ),
          ElevatedButton(
            onPressed: _loadMore,
            style: ElevatedButton.styleFrom(
              backgroundColor: Colors.cyan.shade700,
              foregroundColor: Colors.white,
              shape: RoundedRectangleBorder(
                borderRadius: BorderRadius.circular(10), // Button border radius
              ),
            ),
            child: const Text('Load More', style: TextStyle(color: Colors.white)),
          ),
        ],
      ),
    );
  }
}
```

Discovery_card.dart

```dart
import 'package:flutter/material.dart';
import '../models/discovery_item.dart';

class DiscoveryCard extends StatelessWidget {
  final DiscoveryItem item;

  const DiscoveryCard({super.key, required this.item});

  @override
  Widget build(BuildContext context) {
    List<Color> lightColors = [
      Colors.lightBlue,
      const Color(0xFFf6a530),
      const Color(0xFFf1a5a5),
    ];

    List<Color> darkColors = [
      const Color(0xFF282828),
    ];

    ThemeData currentTheme = Theme.of(context);
    List<Color> containerColors =
    currentTheme.brightness == Brightness.light ? lightColors : darkColors;
    int petId = item.id;
    Color selectedColor = containerColors[petId % containerColors.length];

    Color borderColor =
    currentTheme.brightness == Brightness.light ? Colors.white : Colors.teal;
    return Padding(

      padding: const EdgeInsets.all(10.0),
      child: SizedBox(
        width: double.infinity,
        height: 100,
        child: Card(
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(15.0),
          ),
          color: selectedColor,
          child: Center(
            child: ListTile(
              title: Text(
                item.title,
                style: const TextStyle(color: Colors.white, fontSize: 18, fontWeight: FontWeight.bold),
              ),
              subtitle: Text(
                item.description,
                style: const TextStyle(color: Colors.white),
              ),
              leading: Container(
                  decoration: BoxDecoration(
                    borderRadius: BorderRadius.circular(15.0),
                    border: Border.all(
                      color: Colors.white,
                      width: 5.0,
                    ),
                  ),
                  child: Image.network(
                    item.imageUrl,
                    fit: BoxFit.contain,
                  ),
                ),
              ),
            ),
          ),
        ),
      ),
    );
  }
}
```

About Me:

https://github.com/amancd

https://www.linkedin.com/in/aman-singh-095439234/

My Projects:

https://github.com/amancd/MedApp

https://github.com/amancd/petapp (Similar Task)

https://play.google.com/store/apps/details?id=com.atomdyno.allcalculators

https://play.google.com/store/apps/details?id=com.apna.au.app

Play Store:

https://play.google.com/store/search?q=pub%3Aatomdyno&c=apps