Name : Pranav Chindage Class : D15B Roll no. : 10

Experiment 3: Include Icons and Images to Flutter App

Aim:
To understand how to integrate icons and images into a Flutter application and utilize them effectively to enhance the user interface.

Theory:

Icons and images are essential elements in mobile app development that contribute to the visual appeal and functionality of the user interface.

Icons:
- Icons are small graphical representations used to convey actions, features, or information visually.
- Flutter provides a wide range of built-in icons that can be easily integrated into the app using the `Icon` widget.
- Icons can be customized with various properties such as size, color, and alignment.

Images:
- Images are visual assets that can be used to represent graphics, logos, illustrations, or other visual elements in the app.
- Flutter supports various image formats such as PNG, JPEG, GIF, WebP, etc.
- Images can be loaded into Flutter applications using the `Image` widget, specifying the source of the image (asset, network, file, etc.).
- Flutter also provides advanced features for caching, resizing, and displaying images efficiently.

Code :

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

void main() {
  runApp(FootballTournamentApp());
}

class FootballTournamentApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
```

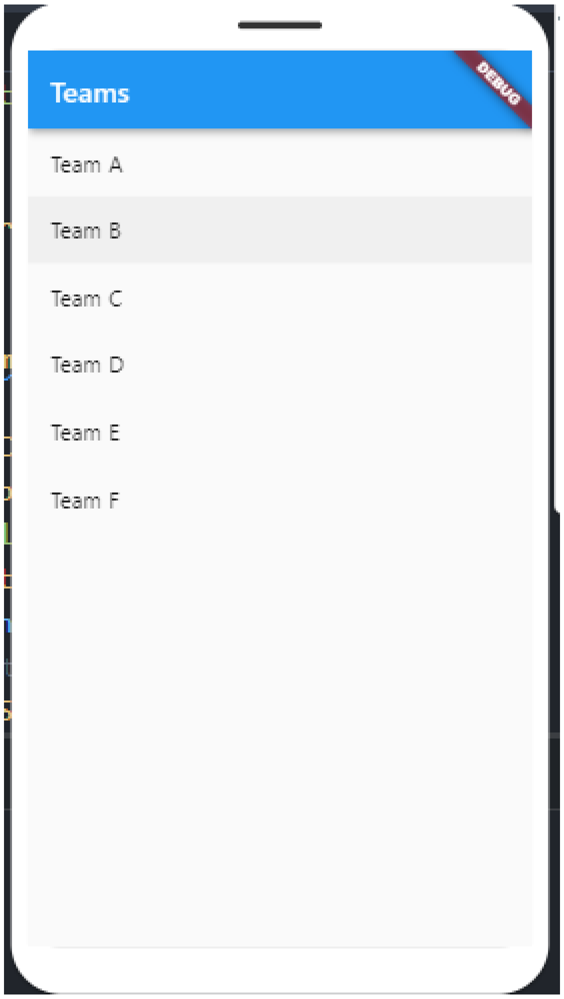
Name : Pranav Chindage Class : D15B Roll no. : 10

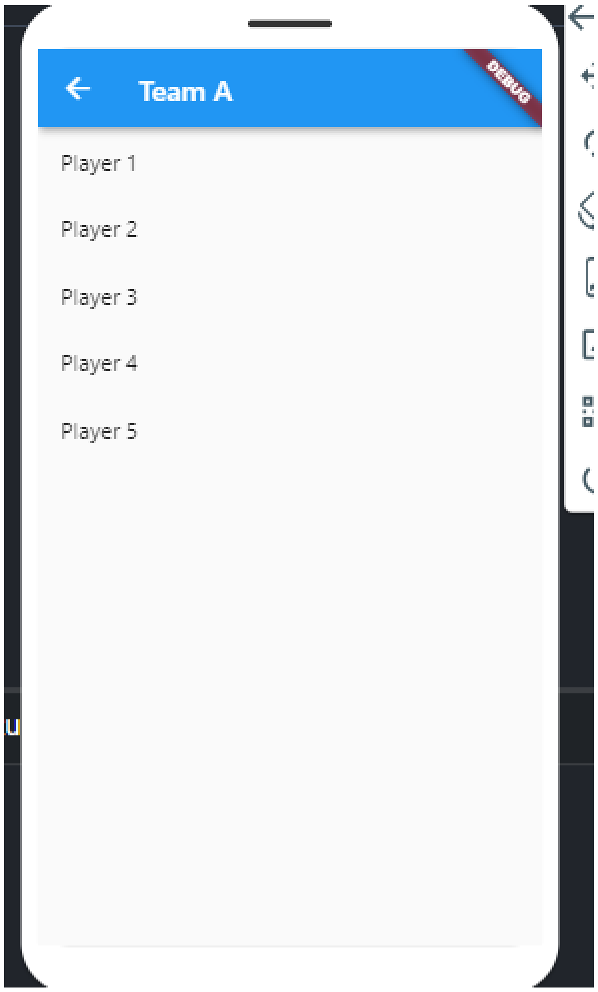
```
    return MaterialApp(
      title: 'Football Tournament',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: TeamListScreen(),
    );
  }
}

class TeamListScreen extends StatelessWidget {
  final List<String> teams = [
    'Team A',
    'Team B',
    'Team C',
    'Team D',
    'Team E',
    'Team F',
  ];

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Teams'),
      ),
      body: ListView.builder(
        itemCount: teams.length,
        itemBuilder: (context, index) {
          return ListTile(
            title: Text(teams[index]),
            onTap: () {
              // Navigate to team details screen with team name
              Navigator.push(
                context,
                MaterialPageRoute(
                  builder: (context) =>
                      TeamDetailsScreen(teamName: teams[index]),
                ),
              );
            },
          );
        },
      ),
    );
  }
}

class TeamDetailsScreen extends StatelessWidget {
  final String teamName;
  final List<String> players;

  TeamDetailsScreen({required this.teamName})
      : players = _getPlayerList(teamName); // Get players for
the selected team

  static List<String> _getPlayerList(String teamName) {
    // Add logic to fetch players for the team from a database
or API
    // For demonstration, using hardcoded players
    if (teamName == 'Team A') {
      return ['Player 1', 'Player 2', 'Player 3', 'Player 4',
'Player 5'];
    } else {
      // Return an empty list if team not found
      return [];
    }
  }
```

Name : Pranav Chindage Class : D15B Roll no. : 10

```
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(teamName),
      ),
      body: ListView.builder(
        itemCount: players.length,
        itemBuilder: (context, index) {
          return ListTile(
            title: Text(players[index]),
          );
        },
      ),
    );
  }
}
```

Name : Pranav Chindage Class : D15B Roll no. : 10





Conclusion :

This experiment illustrates how icons and images can be effectively incorporated to provide a more engaging user experience, reinforcing the app's branding and conveying essential information efficiently. Going forward, optimizing image assets and leveraging custom icons can further refine the app's aesthetics and usability.