TinyTap: Educational App for Kids

Group Leader:
Trixie C. Mediran

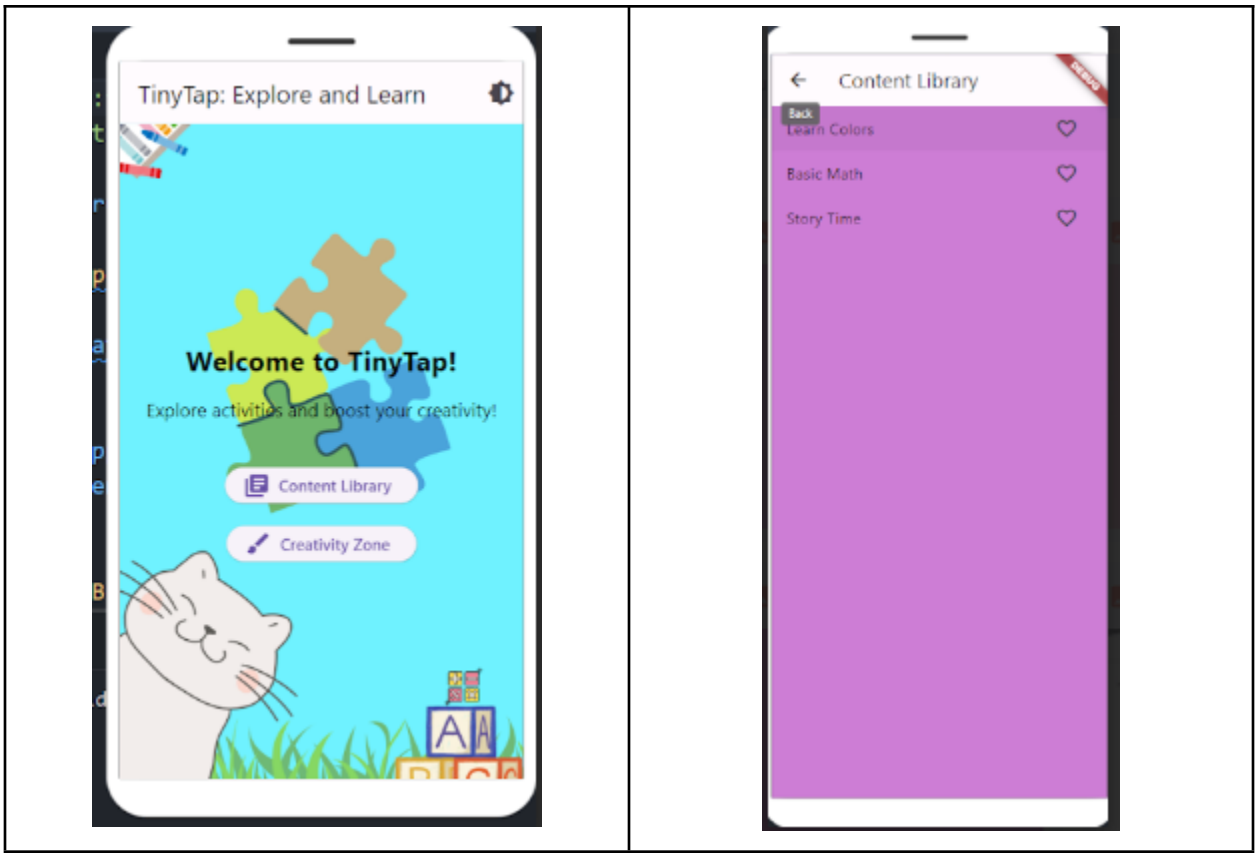
Members:
Angela Jean A. Matawaran
Gian H. Solis

Summary of Mobile Application

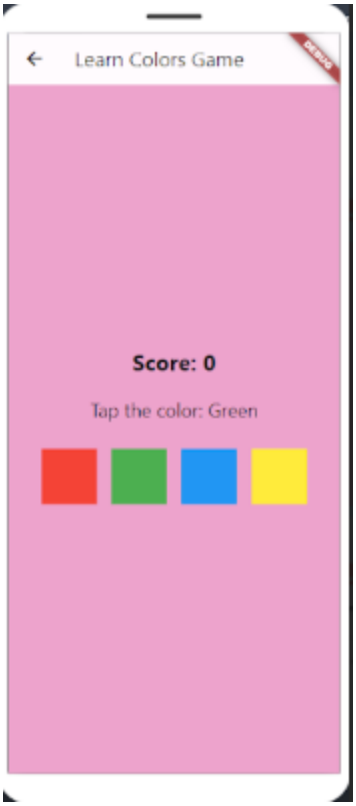
TinyTap is a basic educational mobile application which empowers children to learn from exploring any kinds of activities. Some studies suggest that preschool-age children can learn specific skills and concepts through interactive apps, such as problem-solving skills.

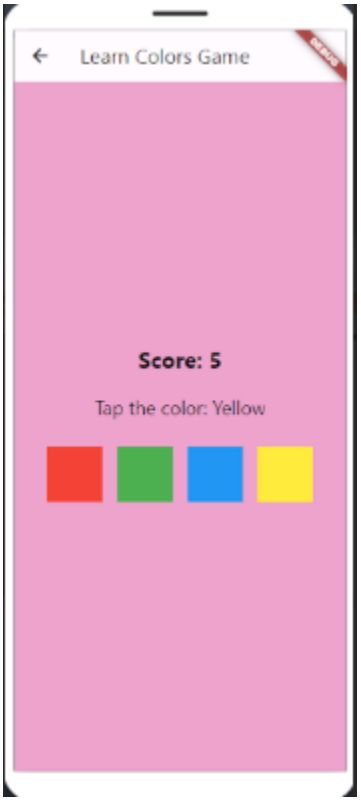
An application is a software program that's designed to perform a specific function directly for the user. Empowers creativity and library of educational games for kids handmade by the group.

Screenshot of Mobile Application

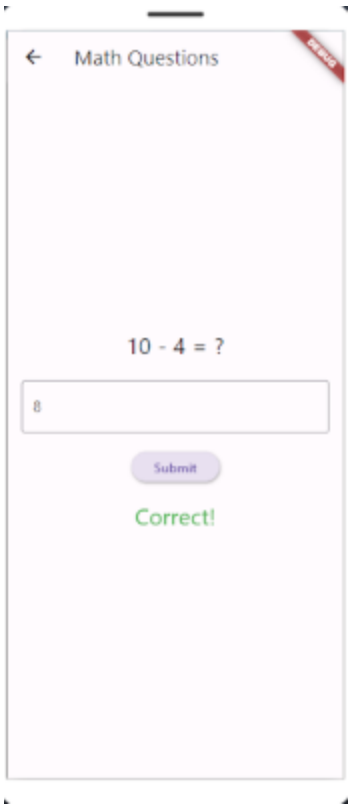


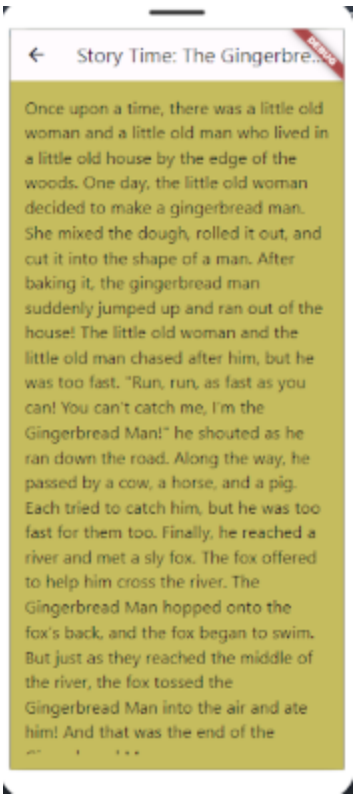

Content Library
Back
Learn Colors
Basic Math
Story Time

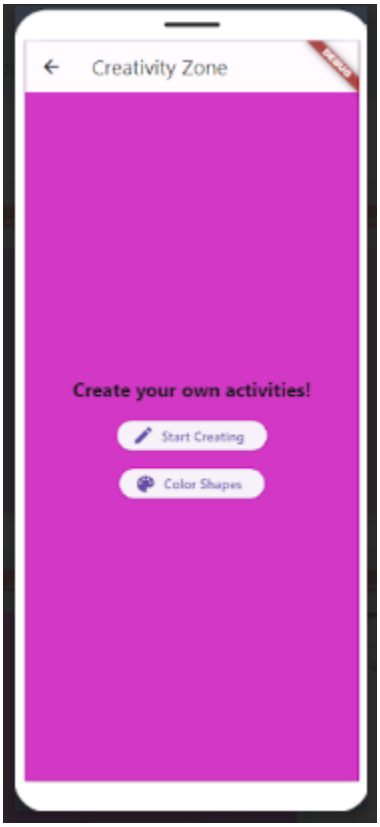

Learn Colors Game
Score: 0
Tap the color: Green


Learn Colors Game
Score: 5
Tap the color: Yellow


Math Questions
5 + 3 = ?
8
Submit

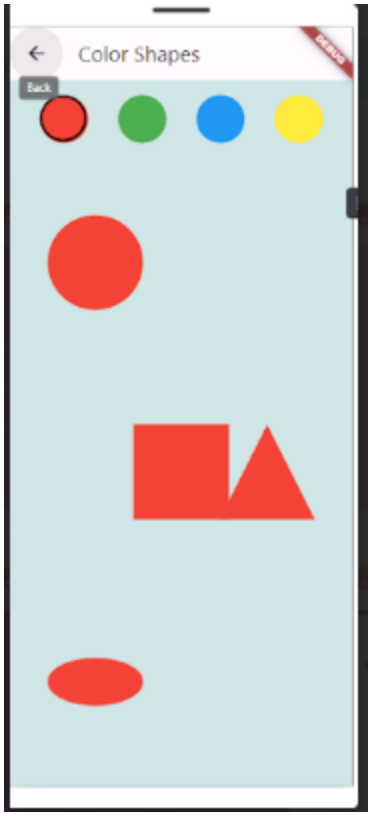

Math Questions
10 - 4 = ?
8
Submit
Correct!

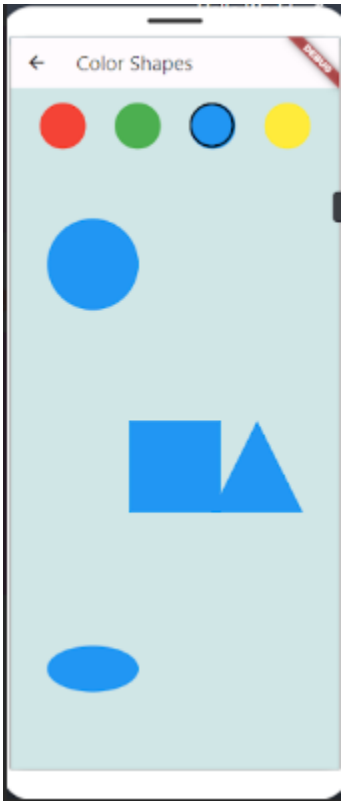

Story Time: The Gingerbre...
Once upon a time, there was a little old woman and a little old man who lived in a little old house by the edge of the woods. One day, the little old woman decided to make a gingerbread man. She mixed the dough, rolled it out, and cut it into the shape of a man. After baking it, the gingerbread man suddenly jumped up and ran out of the house! The little old woman and the little old man chased after him, but he was too fast. "Run, run, as fast as you can! You can't catch me, I'm the Gingerbread Man!" he shouted as he ran down the road. Along the way, he passed by a cow, a horse, and a pig. Each tried to catch him, but he was too fast for them too. Finally, he reached a river and met a sly fox. The fox offered to help him cross the river. The Gingerbread Man hopped onto the fox's back, and the fox began to swim. But just as they reached the middle of the river, the fox tossed the Gingerbread Man into the air and ate him! And that was the end of the


Creativity Zone
Create your own activities!
Start Creating
Color Shapes


Create Your Drawing

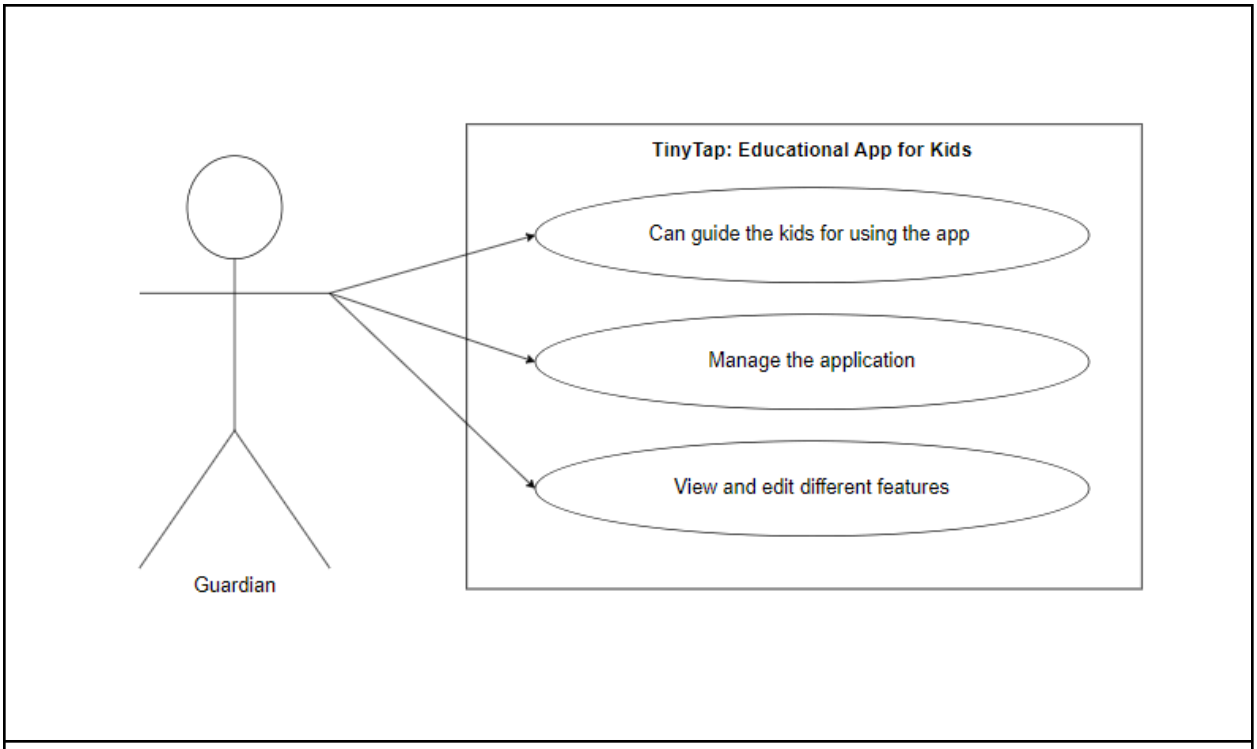

Create Your Drawing

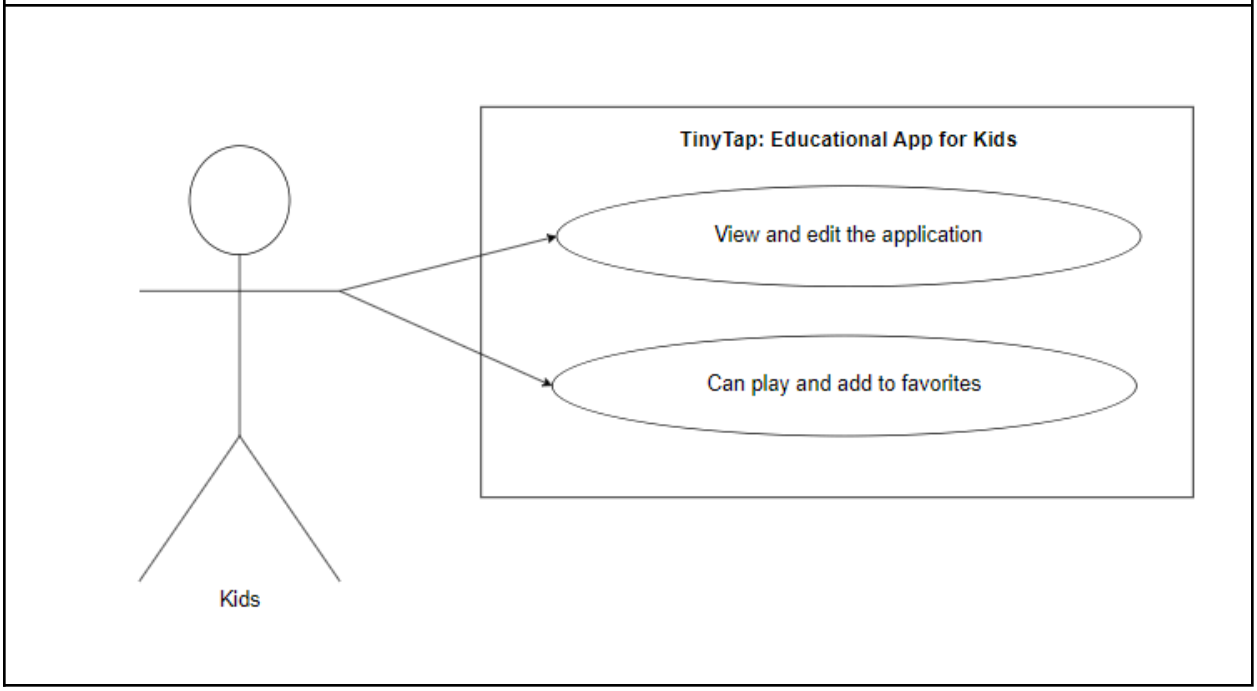

Color Shapes
Back


Color Shapes

Use Case of Mobile Application

1. Use- Case of Guardian/Parents.
2. Use- Case of Kids





Source Code of Mobile Application

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

void main() => runApp(TinyTapApp());

class TinyTapApp extends StatefulWidget {
  @override
  _TinyTapAppState createState() => _TinyTapAppState();
}

class _TinyTapAppState extends State<TinyTapApp> {
  bool isDarkTheme = false;

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'TinyTap Learning App',
      theme: isDarkTheme ? ThemeData.dark() : ThemeData.light(),
      home: HomeScreen(
        toggleTheme: () {
          setState(() {
            isDarkTheme = !isDarkTheme;
          });
        },
      ),
    );
  }
}

class HomeScreen extends StatelessWidget {
  final VoidCallback toggleTheme;
```

```dart
HomeScreen({required this.toggleTheme});

@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Color(0xff30c1db),
    appBar: AppBar(
      title: Text('TinyTap: Explore and Learn'),
      actions: [
        IconButton(
          icon: Icon(Icons.brightness_6),
          onPressed: toggleTheme,
        ),
      ],
    ),
    body: Stack(
      fit: StackFit.expand,
      children: [
        Image.asset(
          "assets/tinytap-new.png",
          fit: BoxFit.fitWidth,
        ),
        SafeArea(
          child: Padding(
            padding: const EdgeInsets.all(16.0),
            child: Column(
              mainAxisAlignment: MainAxisAlignment.center,
              children: <Widget>[
                Text(
                  'Welcome to TinyTap!',
                  style: TextStyle(
                      fontSize: 24,
                      fontWeight: FontWeight.bold,
```

```
                color: Colors.black),
          ),
          SizedBox(height: 15),
          Text(
            'Explore activities and boost your creativity!',
            textAlign: TextAlign.center,
            style: TextStyle(fontSize: 16, color:
olors.black),
          ),
          SizedBox(
            height: 40,
          ),
          ElevatedButton.icon(
            icon: Icon(Icons.library_books),
            label: Text('Content Library'),
            onPressed: () {
              Navigator.push(
                context,
                PageRouteBuilder(
                  pageBuilder:
                      (context, animation, secondaryAnimation)
,
                          ContentLibraryScreen(),
                  transitionsBuilder:
                      (context, animation, secondaryAnimation,
ild) {
                    return FadeTransition(
                      opacity: animation,
                      child: child,
                    );
                  },
                ),
              );
```

```
            },
          ),
          SizedBox(height: 20),
          ElevatedButton.icon(
            icon: Icon(Icons.brush),
            label: Text('Creativity Zone'),
            onPressed: () {
              Navigator.push(
                context,
                PageRouteBuilder(
                  pageBuilder:
                      (context, animation, secondaryAnimation)
,
                          CreativityScreen(),
                  transitionsBuilder:
                      (context, animation, secondaryAnimation,
hild) {
                    return FadeTransition(
                      opacity: animation,
                      child: child,
                    );
                  },
                ),
              );
            },
          ),
        ],
      ),
    ),
  );
```

```
  }
}

class ContentLibraryScreen extends StatefulWidget {
  @override
  _ContentLibraryScreenState createState() =>
_ContentLibraryScreenState();
}

class _ContentLibraryScreenState extends State<ContentLibraryScreen> {
  final List<String> activities = [
    'Learn Colors',
    'Basic Math',
    'Story Time',
  ];

  final Set<String> favoriteActivities = {};

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Color(0xffcd7dd5),
      appBar: AppBar(
        title: Text('Content Library'),
      ),
      body: ListView.builder(
        itemCount: activities.length,
        itemBuilder: (context, index) {
          final activity = activities[index];
          final isFavorite = favoriteActivities.contains(activity);
          return ListTile(
            title: Text(activity),
            trailing: IconButton(
              icon: Icon(
                isFavorite ? Icons.favorite : Icons.favorite_border,
                color: isFavorite ? Colors.red : null,
              ),
              onPressed: () {
                setState(() {
                  if (isFavorite) {
                    favoriteActivities.remove(activity);
                  } else {
                    favoriteActivities.add(activity);
                  }
                });
              },
            ),
            onTap: () {
              Navigator.push(
                context,
                MaterialPageRoute(
                  builder: (context) {
                    switch (activity) {
                      case 'Story Time':
                        return StoryTimeScreen();
                      case 'Basic Math':
                        return MathQuestionsScreen();
                      case 'Learn Colors':
                        return ColorLearningGameScreen();
                      default:
                        return ActivityScreen(activityName: activity);
                    }
                  },
                ),
              );
            },
```

```
        );
      },
    ),
  );
 }
}

class CreativityScreen extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
   return Scaffold(
     backgroundColor: Color(0xffd237c5),
     appBar: AppBar(
       title: Text('Creativity Zone'),
     ),
     body: Center(
       child: Column(
         mainAxisAlignment: MainAxisAlignment.center,
         children: <Widget>[
           Text(
             'Create your own activities!',
             style: TextStyle(fontSize: 20, fontWeight:
FontWeight.bold),
           ),
           SizedBox(height: 20),
           ElevatedButton.icon(
             icon: Icon(Icons.create),
             label: Text('Start Creating'),
             onPressed: () {
               Navigator.push(
                 context,
                 MaterialPageRoute(
                   builder: (context) => DrawingScreen(),
```

```
               ),
             );
           },
         ),
         SizedBox(height: 20),
         ElevatedButton.icon(
           icon: Icon(Icons.color_lens),
           label: Text('Color Shapes'),
           onPressed: () {
             Navigator.push(
               context,
               MaterialPageRoute(
                 builder: (context) => ShapeColoringScreen(),
               ),
             );
           },
         ),
       ],
     ),
   ),
 );
}


lass DrawingScreen extends StatefulWidget {
@override
_DrawingScreenState createState() => _DrawingScreenState();


lass _DrawingScreenState extends State<DrawingScreen> {
List<Offset?> points = [];

@override
```

```dart
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Create Your Drawing'),
      actions: [
        IconButton(
          icon: Icon(Icons.clear),
          onPressed: () {
            setState(() {
              points.clear();
            });
          },
        ),
      ],
    ),
    body: GestureDetector(
      onPanUpdate: (details) {
        setState(() {
          points.add(details.localPosition);
        });
      },
      onPanEnd: (details) {
        points.add(null);
      },
      child: CustomPaint(
        painter: DrawingPainter(points),
        child: Container(),
      ),
    ),
  );
}
```

```dart
class DrawingPainter extends CustomPainter {
  final List<Offset?> points;
  DrawingPainter(this.points);

  @override
  void paint(Canvas canvas, Size size) {
    Paint paint = Paint()
      ..color = Colors.blue
      ..strokeCap = StrokeCap.round
      ..strokeWidth = 5.0;

    for (int i = 0; i < points.length - 1; i++) {
      if (points[i] != null && points[i + 1] != null) {
        canvas.drawLine(points[i]!, points[i + 1]!, paint);
      }
    }
  }

  @override
  bool shouldRepaint(DrawingPainter oldDelegate) => true;
}

class ActivityScreen extends StatelessWidget {
  final String activityName;

  ActivityScreen({required this.activityName});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(activityName),
      ),
```

```
    body: Center(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Text(
            'This is the $activityName activity',
            style: TextStyle(fontSize: 18),
          ),
          SizedBox(height: 20),
          ActivityProgressTracker(),
        ],
      ),
    ),
  );
}


lass ActivityProgressTracker extends StatefulWidget {
 @override
 _ActivityProgressTrackerState createState() =>
     _ActivityProgressTrackerState();


lass _ActivityProgressTrackerState extends
tate<ActivityProgressTracker> {
 double _progress = 0.0;

 @override
 Widget build(BuildContext context) {
   return Column(
     children: [
       Text(
         'Progress: ${(_progress * 100).toInt()}%',
```

```
        style: TextStyle(fontSize: 16),
      ),
      Slider(
        value: _progress,
        onChanged: (value) {
          setState(() {
            _progress = value;
          });
        },
      ),
    ],
  );
}


lass ShapeColoringScreen extends StatefulWidget {
@override
_ShapeColoringScreenState createState() =>
hapeColoringScreenState();


lass _ShapeColoringScreenState extends State<ShapeColoringScreen> {
Color selectedColor = Colors.red;

@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Color(0xffd0e6e6),
    appBar: AppBar(
      title: Text('Color Shapes'),
    ),
    body: Column(
      children: [
```

```
        Padding(
          padding: const EdgeInsets.all(16.0),
          child: Row(
            mainAxisAlignment: MainAxisAlignment.spaceAround,
            children: [
              ColorPickerButton(
                color: Colors.red,
                isSelected: selectedColor == Colors.red,
                onTap: () {
                  setState(() {
                    selectedColor = Colors.red;
                  });
                },
              ),
              ColorPickerButton(
                color: Colors.green,
                isSelected: selectedColor == Colors.green,
                onTap: () {
                  setState(() {
                    selectedColor = Colors.green;
                  });
                },
              ),
              ColorPickerButton(
                color: Colors.blue,
                isSelected: selectedColor == Colors.blue,
                onTap: () {
                  setState(() {
                    selectedColor = Colors.blue;
                  });
                },
              ),
              ColorPickerButton(
```

```
                color: Colors.yellow,
                isSelected: selectedColor == Colors.yellow,
                onTap: () {
                  setState(() {
                    selectedColor = Colors.yellow;
                  });
                },
              ),
            ],
          ),
        ),
        Expanded(
          child: GestureDetector(
            onTapUp: (details) {
              setState(() {
                // Detect the area tapped and color the shape
accordingly
                // For simplicity, we'll assume shapes are
positioned at fixed coordinates
              });
            },
            child: CustomPaint(
              painter: ShapePainter(selectedColor),
              child: Container(),
            ),
          ),
        ),
      ],
    ),
  );
 }
}
```

```
class ColorPickerButton extends StatelessWidget {
 final Color color;
 final bool isSelected;
 final VoidCallback onTap;

 ColorPickerButton({
   required this.color,
   required this.isSelected,
   required this.onTap,
 });

 @override
 Widget build(BuildContext context) {
   return GestureDetector(
     onTap: onTap,
     child: Container(
       width: 50,
       height: 50,
       decoration: BoxDecoration(
         color: color,
         border: isSelected ? Border.all(color: Colors.black, width:
3) : null,
         shape: BoxShape.circle,
       ),
     ),
   );
 }
}

class ShapePainter extends CustomPainter {
 final Color color;
 ShapePainter(this.color);
```

```
 @override
 void paint(Canvas canvas, Size size) {
   final paint = Paint()
     ..color = color
     ..style = PaintingStyle.fill;

   // Draw a circle
   canvas.drawCircle(Offset(size.width / 4, size.height / 6), 50,
paint);

   // Draw a rectangle
   canvas.drawRect(
       Rect.fromCenter(
           center: Offset(size.width / 2, size.height / 2),
           width: 100,
           height: 100),
       paint);

   // Draw a triangle
   var path = Path();
   path.moveTo(size.width * 3 / 4, size.height / 2 - 50);
   path.lineTo(size.width * 3 / 4 - 50, size.height / 2 + 50);
   path.lineTo(size.width * 3 / 4 + 50, size.height / 2 + 50);
   path.close();
   canvas.drawPath(path, paint);

   // Draw an oval
   canvas.drawOval(
       Rect.fromCenter(
           center: Offset(size.width / 4, size.height * 5 / 6),
           width: 100,
           height: 50),
       paint);
```

```
    // Draw a line
    canvas.drawLine(Offset(size.width / 2, size.height * 5 / 6 + 50),
        Offset(size.width * 3 / 4, size.height * 5 / 6 + 50), paint);
  }

  @override
  bool shouldRepaint(ShapePainter oldDelegate) {
    return oldDelegate.color != color;
  }
}

class StoryCard extends StatelessWidget {
  final String title;
  final String story;
  final VoidCallback onTap; // Add this parameter

  StoryCard({
    required this.title,
    required this.story,
    required this.onTap, // Add this parameter
  });

  @override
  Widget build(BuildContext context) {
    return GestureDetector(
      onTap: onTap, // Use the callback here
      child: Card(
        margin: const EdgeInsets.symmetric(vertical: 8.0),
        child: Padding(
          padding: const EdgeInsets.all(16.0),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
```

```
            children: [
              Text(
                title,
                style: TextStyle(fontSize: 22, fontWeight:
ontWeight.bold),
              ),
              SizedBox(height: 10),
              Text(
                story,
                style: TextStyle(fontSize: 16),
              ),
            ],
          ),
        ),
      ),
    );
  }


lass StoryTimeScreen extends StatelessWidget {
  final String storyText =
      'Once upon a time, there was a little old woman and a little old
an who lived in a little old house by the edge of the woods. '
      'One day, the little old woman decided to make a gingerbread
an. She mixed the dough, rolled it out, and cut it into the shape of
 man. '
      'After baking it, the gingerbread man suddenly jumped up and ran
ut of the house! '
      'The little old woman and the little old man chased after him,
ut he was too fast. '
      '"Run, run, as fast as you can! You can't catch me, I'm the
ingerbread Man!" he shouted as he ran down the road. '
```

```
      'Along the way, he passed by a cow, a horse, and a pig. Each
ried to catch him, but he was too fast for them too. '
      'Finally, he reached a river and met a sly fox. The fox offered
o help him cross the river. '
      'The Gingerbread Man hopped onto the fox's back, and the fox
egan to swim. '
      'But just as they reached the middle of the river, the fox
ossed the Gingerbread Man into the air and ate him! '
      'And that was the end of the Gingerbread Man.';

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Color(0xffc5bc5e),
      appBar: AppBar(
        title: Text('Story Time: The Gingerbread Man'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: SingleChildScrollView(
          child: Text(
            storyText,
            style: TextStyle(fontSize: 18, height: 1.5),
          ),
        ),
      ),
    );
  }


lass StoryDetailScreen extends StatelessWidget {
  final String title;
  final String story;
```

```
  StoryDetailScreen({required this.title, required this.story});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(title),
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: SingleChildScrollView(
          child: Text(
            story,
            style: TextStyle(fontSize: 18),
          ),
        ),
      ),
    );
  }
}

lass MathQuestionsScreen extends StatefulWidget {
  @override
  _MathQuestionsScreenState createState() =>
MathQuestionsScreenState();
}

lass _MathQuestionsScreenState extends State<MathQuestionsScreen> {
  final List<Map<String, dynamic>> questions = [
    {'question': '5 + 3 = ?', 'answer': '8'},
    {'question': '10 - 4 = ?', 'answer': '6'},
    {'question': '3 + 8 = ?', 'answer': '11'},
```

```
    {'question': '7 + 6 = ?', 'answer': '13'},
    {'question': '7 - 2 = ?', 'answer': '5'},
    {'question': '2 - 2 = ?', 'answer': '0'},
  ];

  int _currentQuestionIndex = 0;
  String _userAnswer = '';
  bool _isCorrect = false;

  void _checkAnswer() {
    setState(() {
      _isCorrect = _userAnswer ==
uestions[_currentQuestionIndex]['answer'];
      if (_isCorrect) {
        if (_currentQuestionIndex < questions.length - 1) {
          _currentQuestionIndex++;
        } else {
          _currentQuestionIndex = 0; // Reset to the first question
        }
        _userAnswer = '';
      }
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Math Questions'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
```

```
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            Text(
              questions[_currentQuestionIndex]['question'],
              style: TextStyle(fontSize: 24),
            ),
            SizedBox(height: 20),
            TextField(
              keyboardType: TextInputType.number,
              onChanged: (value) {
                _userAnswer = value;
              },
              decoration: InputDecoration(
                border: OutlineInputBorder(),
                hintText: 'Enter your answer',
              ),
            ),
            SizedBox(height: 20),
            ElevatedButton(
              onPressed: _checkAnswer,
              child: Text('Submit'),
            ),
            SizedBox(height: 20),
            if (_isCorrect)
              Text(
                'Correct!',
                style: TextStyle(fontSize: 24, color: Colors.green),
              )
            else if (_userAnswer.isNotEmpty)
              Text(
                'Try Again!',
                style: TextStyle(fontSize: 24, color: Colors.red),
              ),
```

```
          ],
        ),
      ),
    );
  }
}

class ColorLearningGameScreen extends StatefulWidget {
  @override
  _ColorLearningGameScreenState createState() =>
      _ColorLearningGameScreenState();
}

class _ColorLearningGameScreenState extends
State<ColorLearningGameScreen> {
  final List<Color> colors = [
    Colors.red,
    Colors.green,
    Colors.blue,
    Colors.yellow
  ];
  late Color targetColor;
  Color? selectedColor;
  int score = 0;

  @override
  void initState() {
    super.initState();
    _generateNewTargetColor(); // Initialize with a random target
color
  }

  void _generateNewTargetColor() {
```

```
    setState(() {
      targetColor = colors[Random().nextInt(colors.length)];
    });
  }

  void _checkColor(Color color) {
    setState(() {
      selectedColor = color;
      if (selectedColor == targetColor) {
        score++;
        _generateNewTargetColor(); // Generate a new color after a
correct tap
      } else {
        score = 0; // Reset score on incorrect tap (optional)
      }
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Learn Colors Game'),
      ),
      backgroundColor: Color(0xffeda3cc),
      body: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Text(
            'Score: $score',
            style: TextStyle(fontSize: 24, fontWeight:
FontWeight.bold),
          ),
```

```
        SizedBox(height: 20),
        Text(
          'Tap the color: ${_getColorName(targetColor)}',
          style: TextStyle(fontSize: 20),
        ),
        SizedBox(height: 20),
        Row(
          mainAxisAlignment: MainAxisAlignment.center,
          children: colors.map((color) {
            return GestureDetector(
              onTap: () => _checkColor(color),
              child: Container(
                width: 60,
                height: 60,
                margin: EdgeInsets.all(8.0),
                color: color,
              ),
            );
          }).toList(),
        ),
      ],
    ),
  );
}

String _getColorName(Color color) {
  if (color == Colors.red) return 'Red';
  if (color == Colors.green) return 'Green';
  if (color == Colors.blue) return 'Blue';
  if (color == Colors.yellow) return 'Yellow';
  if (color == Colors.pink) return 'Pink';
  return 'Unknown';
}
```

```
}
```