

# Experiment 5

Name: Umesh Artani

Div: D15A

Roll no: 03

Aim: To apply navigation, routing and gestures in Flutter App

Theory:

## 1. Navigation:

- Navigation refers to the process of moving between different screens or pages within a Flutter app.
- In Flutter, navigation is typically managed using the Navigator class, which maintains a stack of routes.
- Each route represents a screen or page in the app, and the navigator manages the navigation stack, allowing users to move forward and backward between routes.
- Navigation can be triggered by user actions such as tapping buttons, selecting items from lists, or swiping between pages.

## 2. Routing:

- Routing is the mechanism used to define and manage the routes within a Flutter app.
- Routes are defined using route names and associated with corresponding widgets or screens.
- Flutter provides several routing mechanisms, including named routes, on-the-fly routes, and nested routes.
- Named routes allow developers to define routes with unique names and navigate to them using the Navigator based on these names.
- On-the-fly routes are created dynamically at runtime and pushed onto the navigation stack as needed.
- Nested routes involve embedding navigators within other navigators to create complex navigation structures, such as tab-based navigation or drawer navigation.

## 3. Gestures:

- Gestures refer to user interactions such as tapping, dragging, swiping, pinching, and rotating on the screen.
- Flutter provides a rich set of gesture recognition widgets and APIs to handle user gestures effectively.
- Common gesture recognition widgets include GestureDetector, InkWell, InkResponse, Draggable, Dismissible, etc.
- These widgets allow developers to detect various user gestures and trigger corresponding actions or animations in response.

- Gestures can be used to implement interactive UI elements, such as buttons, sliders, swipers, drag-and-drop interfaces, and more.

#### 4. Gesture Detection:

- Gesture detection in Flutter involves registering gesture recognizers on widgets to detect specific user interactions.
- Gesture recognizers analyze touch input and determine whether a specific gesture has occurred, such as a tap, double-tap, long-press, drag, etc.
- Once a gesture is detected, Flutter invokes the corresponding callback function associated with the gesture recognizer.
- Developers can customize gesture detection by configuring properties such as gesture sensitivity, velocity thresholds, and touch area boundaries.

#### 5. Gesture Handling:

- After a gesture is detected, developers can handle it by performing various actions, such as updating UI state, navigating between screens, triggering animations, or executing business logic.
- Gesture handling involves responding to user interactions in a way that provides feedback and enhances the user experience.
- Flutter's declarative programming model makes it easy to update UI elements in response to user gestures, ensuring a smooth and responsive user interface.

### **home\_screen.dart**

```
import 'dart:math';
import 'package:animated_text_kit/animated_text_kit.dart';
import 'package:carousel_slider/carousel_slider.dart';
import 'package:flutter/foundation.dart';
import 'package:flutter/material.dart';
import 'package:flutter_screenutil/flutter_screenutil.dart';
import
'package:flutter_staggered_grid_view/flutter_staggered_grid_view.dart';
import 'package:google_fonts/google_fonts.dart';
import 'package:liquid_pull_to_refresh/liquid_pull_to_refresh.dart';
import 'package:dribbble_clone/screens/images_details_screen.dart';
import 'package:dribbble_clone/widget/api_call_waiting_widget.dart';
import 'package:shimmer/shimmer.dart';
import
'../api/pixel_api_class.dart';
import '../model/pixel_model.dart';

import '../themes/container_random_colors.dart';
import '../widget/loading_Widget.dart';
import 'onboard_screen.dart';

class HomeScreen extends StatefulWidget {
```

```

const HomeScreen({super.key});

@override
State<HomeScreen> createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  int currentIndex = 1;
  ScrollController scrollController = ScrollController();
  bool isStarted = false;
  List<Photo> curatedPhotos = [];
  bool isRefreshing = false;

  final GlobalKey<LiquidPullToRefreshState> _refreshIndicatorKey =
    GlobalKey<LiquidPullToRefreshState>();
  final GlobalKey<ScaffoldState> _scaffoldKey = GlobalKey<ScaffoldState>();

  final PexelsApi pexelsApi = PexelsApi(
    apiKey: 'MsB5RufxehOuViswlLug2dA0qBtlXvyrRdfA0n1GnLL2MCIMbA2eD8Sk');

  final List<String> quotes = [
    "Exploring moments, capturing dreams where every frame tells a story of "
    "timeless beauty.",
    "Seeking moments in pixels, where every frame tells a story, and each "
    "image is a masterpiece waiting to be discovered.",
    "Discover moments, capture stories. Elevate your world with curated "
    "creativity."
  ];

  void handleRefresh() async {
    setState(() {
      isRefreshing = true;
    });

    try {
      List<Photo> newCuratedPhotos =
        await pexelsApi.getCuratedPhotos(page: 1, perPage: 15);

      setState(() {
        curatedPhotos = newCuratedPhotos;
      });
    } catch (error) {
      if (kDebugMode) {
        print('Error refreshing curated photos: $error');
      }
    }
  }
}

```

```

    }

    setState(() {
      isRefreshing = false;
    });
  }

  // void shareImage(String imageUrl, String photographer) {
  //   Share.share(
  //     'Check out this photo by $photographer: $imageUrl',
  //     subject: 'Photo Sharing',
  //   );
  // }

  @override
  void initState() {
    super.initState();
    // Connectivity().checkConnectivity().then((ConnectivityResult result) {
    //   if (result == ConnectivityResult.none) {
    //     showNoInternetSnackBar();
    //   }
    // });
    //
    // Connectivity().onConnectivityChanged.listen((ConnectivityResult result) {
    //   if (result == ConnectivityResult.none) {
    //     showNoInternetSnackBar();
    //   }
    // });
  }

  // void showNoInternetSnackBar() {
  //   final snackBar = SnackBar(
  //     backgroundColor: Colors.transparent,
  //     elevation: 0,
  //     padding: const EdgeInsets.only(bottom: 750),
  //     dismissDirection: DismissDirection.endToStart,
  //     content: Container(
  //       width: 190.w,
  //       height: 56.h,
  //       decoration: BoxDecoration(
  //         shape: BoxShape.rectangle,
  //         borderRadius: BorderRadius.circular(45),
  //         color: Colors.red,
  //       ),
  //       child: Center(
  //         child: Text(
  //           "Hmm... you're not connected to the \n internet",

```

```

//      style: GoogleFonts.poppins(
//      color: Colors.black,
//      fontWeight: FontWeight.w500,
//      fontSize: 17.sp),
//      textAlign: TextAlign.center,
//    ),
//  )),
//  duration: const Duration(seconds: 2),
// );
//
// ScaffoldMessenger.of(context).showSnackBar(snackBar);
// }

```

@override

```

Widget build(BuildContext context) {
  // Connectivity().checkConnectivity().then((ConnectivityResult result) {
  // if (result == ConnectivityResult.none) {
  //   showNoInternetSnackBar();
  // }
  // });
}

```

```

return LiquidPullToRefresh(
  key: _refreshIndicatorKey,
  onRefresh: () async {
    handleRefresh;
  },
  showChildOpacityTransition: true,
  color: Colors.grey.shade900,
  springAnimationDurationInMilliseconds: 800,
  backgroundColor: Colors.grey.shade600,
  child: Scaffold(
    key: _scaffoldKey,
    backgroundColor: Colors.transparent,
    body: NestedScrollView(
      physics: const AlwaysScrollableScrollPhysics(),
      headerSliverBuilder: (BuildContext context, bool isScrolled) {
        return [
          SliverAppBar(
            expandedHeight:
              260.h, pinned: false,
            flexibleSpace: Stack(children: [
              FutureBuilder<List<Photo>>>(
                future: pexelsApi.getCuratedPhotos(page: 1, perPage: 10),
                builder: (context, snapshot) {
                  if (snapshot.connectionState == ConnectionState.waiting) {
                    return Shimmer.fromColors(

```

```

baseColor: Colors.grey[300]!,
highlightColor: Colors.grey[100]!,
child: CarouselSlider(
  options: CarouselOptions(
    height: 380.h, initialPage:
    1, enlargeCenterPage:
    true, viewportFraction: 1,
    enlargeStrategy: CenterPageEnlargeStrategy.scale,
    enableInfiniteScroll: true,
  ),
  items: List.generate(3, (index) {
    return Container(
      decoration: BoxDecoration(
        color: Colors.grey.shade800,
      ),
    );
  }),
),
);
} else if (snapshot.hasError) {
  return Text('Error: ${snapshot.error}');
} else if (snapshot.hasData) {
  final List<Photo> curatedPhotos = snapshot.data!;
  final middleIndex = curatedPhotos.length ~/ 2;
  return CarouselSlider(
    options: CarouselOptions(
      height: 380.h,
      initialPage: middleIndex,
      autoPlay: true,
      viewportFraction: 1,
      enlargeStrategy: CenterPageEnlargeStrategy.scale,
      enableInfiniteScroll: true,
    ),
    items: curatedPhotos.map((photo) {
      return Container(
        decoration: BoxDecoration(
          color: Colors.grey.shade800,
          image: DecorationImage(
            fit: BoxFit.cover,
            image: NetworkImage(photo.src["large"]),
          ),
        ),
      );
    }).toList(),
  );
}

```

```

    } else {
      return YourLoadingWidget(
        child: Center(
          child: Text(
            "Error Loading",
            style: GoogleFonts.poppins(
              color: Colors.white,
              fontWeight:
                FontWeight.w500),
          ),
        ),
      );
    }
  },
),
Center(
  child: AnimatedTextKit(
    animatedTexts: quotes.map((quote) {
      return TypewriterAnimatedText(
        quote,
        textStyle: GoogleFonts.poppins(
          color: Colors.white,
          fontSize: 20.sp,
          fontWeight: FontWeight.w500,
        ),
        speed: const Duration(milliseconds: 100),
        textAlign: TextAlign.center,
      );
    }).toList(),
  )),
),
],
},
body: Padding(
  padding: const EdgeInsets.all(8.0),
  child: SizedBox(
    height: MediaQuery.of(context).size.height,
    child: FutureBuilder<List<Photo>>(
      future: pexelsApi.getCuratedPhotos(page: 3, perPage: 150),
      builder: (context, snapshot) {
        if (snapshot.connectionState == ConnectionState.waiting) {
          return const WaitingContainer();
        } else if (snapshot.hasError) {
          return Center(
            child: Text('Error: ${snapshot.error}'),
          );
        }
      }
    )
  )
)

```

```

} else if (snapshot.hasData) {
  final List<Photo> curatedPhotos = snapshot.data!;
  return MasonryGridView.builder(
    mainAxisSpacing: 10.0,
    crossAxisSpacing: 10.0,
    physics: const AlwaysScrollableScrollPhysics(),
    itemCount: curatedPhotos.length,
    itemBuilder: (context, index) {
      final photo = curatedPhotos[index];
      final double imageAspectRatio =
        photo.width / photo.height;
      return GestureDetector(
        onTap: () {
          Navigator.of(context).push(PageRouteBuilder(
            pageBuilder: (context, animation,
              secondaryAnimation)
              => ImageDetailsScreen(
                photo: photo,
                curatedPhotos: curatedPhotos,
                initialIndex: index,
              ),
            transitionsBuilder: (context, animation,
              secondaryAnimation, child) {
              const begin = Offset(0.0, 1.0);
              const end = Offset.zero;
              const curve = Curves.easeInOut;

              var tween = Tween(begin: begin, end: end)
                .chain(CurveTween(curve: curve));
              var offsetAnimation =
                animation.drive(tween);

              return SlideTransition(
                position: offsetAnimation,
                child: child);
            },
            transitionDuration:
              const Duration(milliseconds: 500),
          ));
        },
        child: Column(
          children: [
            AspectRatio(
              aspectRatio: imageAspectRatio,
              child: Container(
                margin: const EdgeInsets.all(5),

```



```

decoration: BoxDecoration(
  borderRadius: BorderRadius.circular(15),
  color: ColorList.colorList[Random().nextInt(ColorList.colorList.length)],
  image: DecorationImage(
    fit: BoxFit.cover,
    image: NetworkImage(
      photo.src["large"] ?? ""
    ),
  ),
),
child: (photo.url != null)
  ? null
  : const Center(
    child: Icon(
      Icons.image_not_supported,
      size: 50,
      color: Colors.white,
    ),
  ),
),
),
Row(
  mainAxisAlignment:
    MainAxisAlignment.spaceBetween,
  children: [
    Flexible(
      child: Text(
        photo.alt,
        style: GoogleFonts.poppins(
          fontWeight: FontWeight.w500,
          fontSize: 14.sp,
          color: Colors.white,
        ),
      ),
    ),
    IconButton(
      icon: const Icon(
        Icons.more_horiz_sharp,
        color: Colors.white,
      ),
      onPressed: () {
        // shareImage(photo.src["large"] ?? "",
        //   photo.photographer);
      },
    ),
  ],
),
),

```

```

        ],
      ));
    },
    gridDelegate:
      const SliverSimpleGridDelegateWithFixedCrossAxisCount(
        crossAxisCount: 2));
  } else {
    return YourLoadingWidget(
      child: Container(),
    );
  }
},
),
),
),
),
),
);
}
}

```

### search\_screen.dart

```

import 'dart:math';
import 'package:flutter/foundation.dart';
import 'package:flutter/material.dart';
import 'package:flutter_screenutil/flutter_screenutil.dart';
import
'package:flutter_staggered_grid_view/flutter_staggered_grid_view.dart';
import 'package:google_fonts/google_fonts.dart';
import 'package:dribbble_clone/screens/search_Screen_bottomsheet.dart';
import 'package:dribbble_clone/widget/api_call_waiting_widget.dart'; import
'../api/pexel_api_class.dart';
import '../model/pexel_model.dart';
import '../themes/container_random_colors.dart';
import '../widget/loading_Widget.dart';
import '../widget/loading_indicator.dart';
import 'images_details_screen.dart';

```

```

class SearchScreen extends StatefulWidget {
  const SearchScreen({Key? key}) : super(key: key);

```

```

  @override

```

```

    State<SearchScreen> createState() => _SearchScreenState();
}

class _SearchScreenState extends State<SearchScreen> {
    List<Photo> _searchResults = [];
    bool _isLoading = false;
    // void shareImage(String imageUrl, String photographer) {
    //   Share.share(
    //     'Check out this photo by $photographer: $imageUrl',
    //     subject: 'Photo Sharing',
    //   );
    // }
    @override
    void initState() {
      super.initState();
    }
    final PexelsApi pexelsApi = PexelsApi(
      apiKey: 'MsB5RufxehOuViswILug2dAoqBtlXvyrRdfA0n1GnLL2MCIMbA2eD8Sk');

    Future<void> _searchPhotos(String query) async {
      try {
        // Existing code...

        final List<Photo> results = await PexelsApi(
          apiKey:
            'MsB5RufxehOuViswILug2dAoqBtlXvyrRdfA0n1GnLL2MCIMbA2eD8Sk')
          .searchPhotos(query);

        if (kDebugMode) {
          print("Search Results: $results");
        } // Add this line for debugging

        setState(() {
          _searchResults = results;
          _isLoading = false; // Move this line inside setState to ensure proper state update.
        });
      } catch (e) {
        // Handle error
        if (kDebugMode) {
          print("Error during search: $e");
        } // Add this line for debugging
        if (kDebugMode) {
          print(e.toString());
        }
        setState(() {
          _isLoading = false;

```

```

    });
  }
}

```

```

@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.black,
    appBar: AppBar(
      title: GestureDetector(
        onTap: () {
          showModalBottomSheet(
            context: context,
            isDismissible: true,
            isScrollControlled: true,
            useSafeArea: true,
            shape: const RoundedRectangleBorder(
              borderRadius: BorderRadius.only(
                topLeft: Radius.circular(15),
                topRight: Radius.circular(15),
              ),
            ),
            builder: (BuildContext context) {
              return StatefulBuilder(
                builder: (BuildContext context,
                  void Function(void Function()) setState) {
                  return _searchBottomSheet();
                },
              );
            },
          );
        },
      ),
    child: Container(
      width: double.infinity,
      height: 46.h,
      padding: const EdgeInsets.all(8),
      decoration: BoxDecoration(
        shape: BoxShape.rectangle,
        borderRadius: BorderRadius.circular(45),
        color: Colors.grey.shade900),
      child: Row(
        children: [
          Icon(Icons.search, color: Colors.grey.shade300,),
          Text(
            "Search Dribbble",
            style: GoogleFonts.poppins(
              color: Colors.grey.shade300,

```

```

        fontWeight: FontWeight.w500,
        fontSize: 17.sp),
        textAlign: TextAlign.center,
    ),
    ],
  ),
),
),
backgroundColor: Colors.black,

),
body: Padding(
  padding: const EdgeInsets.all(8.0),
  child: SizedBox(
    height: MediaQuery.of(context).size.height,
    child: FutureBuilder<List<Photo>>(
      future: pexelsApi.getCuratedPhotos(page: 5, perPage: 150),
      builder: (context, snapshot) {
        if (snapshot.connectionState == ConnectionState.waiting) {
          return const WaitingContainer();
        } else if (snapshot.hasError) {
          return Center(
            child: Text('Error: ${snapshot.error}'),
          );
        } else if (snapshot.hasData) {
          final List<Photo> curatedPhotos = snapshot.data!;
          return MasonryGridView.builder(
            mainAxisSpacing: 10.0,
            crossAxisSpacing: 10.0,
            physics: const AlwaysScrollableScrollPhysics(),
            itemCount: curatedPhotos.length,
            itemBuilder: (context, index) {
              final photo = curatedPhotos[index];
              final double imageAspectRatio =
                photo.width / photo.height;
              return GestureDetector(
                onTap: () {
                  Navigator.of(context).push(PageRouteBuilder(
                    pageBuilder: (context, animation,
                      secondaryAnimation) =>
                      ImageDetailsScreen(
                        photo: photo,
                        curatedPhotos: curatedPhotos,
                        initialIndex: index,
                      ),
                    transitionsBuilder: (context, animation,

```

```

        secondaryAnimation, child) {
      const begin = Offset(0.0, 1.0);
      const end = Offset.zero;
      const curve = Curves.easeInOut;

      var tween = Tween(begin: begin, end: end)
        .chain(CurveTween(curve: curve));
      var offsetAnimation =
        animation.drive(tween);

      return SlideTransition(
        position: offsetAnimation,
        child: child);
    },
    transitionDuration:
      const Duration(milliseconds: 500),
  ));
},
child: Column(
  children: [
    AspectRatio(
      aspectRatio: imageAspectRatio,
      child: Container(
        margin: const EdgeInsets.all(5),
        decoration: BoxDecoration(
          borderRadius: BorderRadius.circular(15),
          color: ColorList.colorList[Random().nextInt(ColorList.colorList.length)],
          image: DecorationImage(
            fit: BoxFit.cover,
            image: NetworkImage(
              photo.src["large"] ?? ""
            ),
          ),
        ),
        child: (photo.url != null)
          ? null
          : const Center(
              child: Icon(
                Icons.image_not_supported,
                size: 50,
                color: Colors.white,
              ),
            ),
      ),
    ),
  ),
  Row(
    mainAxisAlignment:

```

```

MainAxisAlignment.spaceBetween,
children: [
  Flexible(
    child: Text(
      photo.alt,
      style: GoogleFonts.poppins(
        fontWeight: FontWeight.w500,
        fontSize: 14.sp,
        color: Colors.white,
      ),
    ),
  ),
  IconButton(
    icon: const Icon(
      Icons.more_horiz_sharp,
      color: Colors.white,
    ),
    onPressed: () {
      // shareImage(photo.src["large"] ?? "",
      //   photo.photographer);
    },
  ),
],
),
],
));
},
gridDelegate:
const SliverSimpleGridDelegateWithFixedCrossAxisCount(
  crossAxisCount: 2));
} else {
  return YourLoadingWidget(
    child: Container(),
  );
}
},
),
),
),
);
}
Widget _searchBottomSheet() {
  return Container(
    decoration: const BoxDecoration(
      color: Colors.black,
      borderRadius: BorderRadius.only(

```

```

        topLeft: Radius.circular(15),
        topRight: Radius.circular(15),
    ),
),
child: Padding(
  padding: const EdgeInsets.all(8.0),
  child: Column(
    children: [
      Row(
        mainAxisAlignment:
        MainAxisAlignment.spaceBetween, children: [
          SearchBottomSheetContent(
            onSearch: (query) {
              setState(() {
                _isLoading = true;
              });
              _searchPhotos(query).then((_) {
                setState(() {
                  _isLoading = false;
                });
              });
            },
          ),
          GestureDetector(
            onTap: () {
              Navigator.pop(context);
            },
            child: Text(
              "Cancel",
              style: GoogleFonts.poppins(
                color: Colors.white,
                fontWeight: FontWeight.w500,
                fontSize: 18.sp,
              ),
            ),
          ),
        ],
      ),
      SizedBox(height: 10.h),
      if (_isLoading)
        const Center(child: AnimatedCircularContainer())
      else
        Expanded(child: _buildSearchResults()),
    ],
  ),
),
),

```



```
);  
}
```

```
Widget _buildSearchResults() {  
  return MasonryGridView.builder(  
    mainAxisSpacing: 10.0,  
    crossAxisSpacing: 10.0,  
    physics: const AlwaysScrollableScrollPhysics(),  
    itemCount: _searchResults.length,  
    itemBuilder: (context, index) {  
      final photo = _searchResults[index];  
      final double imageAspectRatio =  
        photo.width / photo.height;  
      return GestureDetector(  
        onTap: () {  
          Navigator.of(context).push(PageRouteBuilder(  
            pageBuilder: (context, animation,  
              secondaryAnimation) =>  
              ImageDetailsScreen(  
                photo: photo,  
                curatedPhotos: _searchResults,  
                initialIndex: index,  
              ),  
            transitionsBuilder: (context, animation,  
              secondaryAnimation, child) {  
              const begin = Offset(0.0, 1.0);  
              const end = Offset.zero;  
              const curve = Curves.easeInOut;  
  
              var tween = Tween(begin: begin, end: end)  
                .chain(CurveTween(curve: curve));  
              var offsetAnimation =  
                animation.drive(tween);  
  
              return SlideTransition(  
                position: offsetAnimation,  
                child: child);  
            },  
            transitionDuration:  
              const Duration(milliseconds: 500),  
          ));  
        },  
        child: Column(  
          children: [  
            AspectRatio(  

```

```
aspectRatio: imageAspectRatio,
child: Container(
  margin: const EdgeInsets.all(5),
  decoration: BoxDecoration(
    borderRadius: BorderRadius.circular(15),
    color: ColorList.colorList[Random().nextInt(ColorList.colorList.length)],
    image: DecorationImage(
      fit: BoxFit.cover,
      image: NetworkImage(
        photo.src["large"] ?? ""
      ),
    ),
  ),
  child: (photo.url != null)
    ? null
    : const Center(
      child: Icon(
        Icons.image_not_supported,
        size: 50,
        color: Colors.white,
      ),
    ),
),
Row(
  mainAxisAlignment:
MainAxisAlignment.spaceBetween,
children: [
  Flexible(
    child: Text(
      photo.alt,
      style: GoogleFonts.poppins(
        fontWeight: FontWeight.w500,
        fontSize: 14.sp,
        color: Colors.white,
      ),
    ),
  ),
  IconButton(
    icon: const Icon(
      Icons.more_horiz_sharp,
      color: Colors.white,
    ),
    onPressed: () {
      // shareImage(photo.src["large"] ?? "",
      //   photo.photographer);
    },
  ),
],
)
```

```

        ),
      ],
    ),
  ],
));
},
gridDelegate:
const SliverSimpleGridDelegateWithFixedCrossAxisCount(
  crossAxisCount: 2));
}
}

```

### upload\_screen.dart

```

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

class MessageScreen extends StatefulWidget {
  const MessageScreen({super.key});

  @override
  State<MessageScreen> createState() => _MessageScreenState();
}

class _MessageScreenState extends State<MessageScreen> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.black,
      body: Column(
        crossAxisAlignment: CrossAxisAlignment.center,
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Center(
            child: Image.asset("assets/images/pinterst.png",
              width: 60.w,
              height: 60.h,),
          ),
        ],
      ),
    );
  }
}

```

## message\_screen.dart

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

class ProfileScreen extends StatefulWidget {
  const ProfileScreen({super.key});

  @override
  State<ProfileScreen> createState() => _ProfileScreenState();
}

class _ProfileScreenState extends State<ProfileScreen> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.black,
      body: Column(
        crossAxisAlignment: CrossAxisAlignment.center,
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Center(
            child: Column(
              children: [
                Image.asset(
                  "assets/images/pinterst.png",
                  width: 60.w,
                  height: 60.h,
                ),
                SizedBox(height: 10.h), // Adding space between image and text
                Text(
                  'No chats yet',
                  style: TextStyle(
                    color: Colors.white,
                    fontSize: 16.sp,
                  ),
                ),
                Text(
                  'Your chats will appear here',
                  style: TextStyle(
                    color: Colors.white,
                    fontSize: 16.sp,
                  ),
                ),
              ],
            ),
          ),
        ],
      ),
    );
  }
}
```

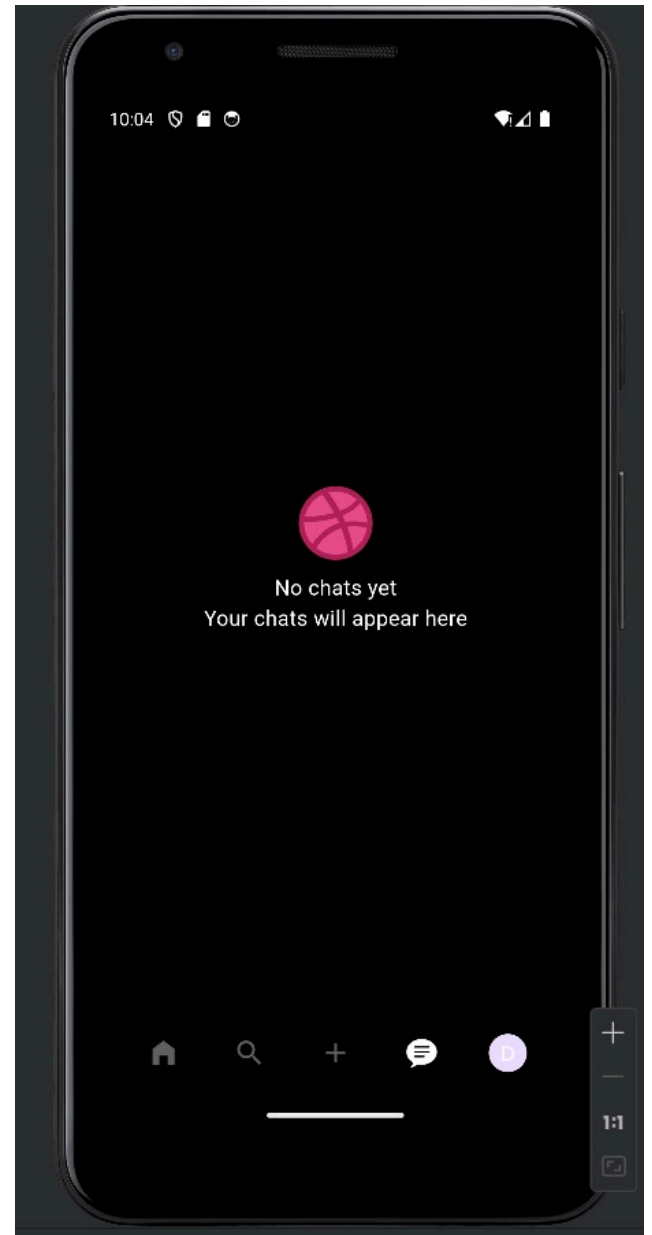
```
),  
)  
],  
)  
);  
}  
}
```

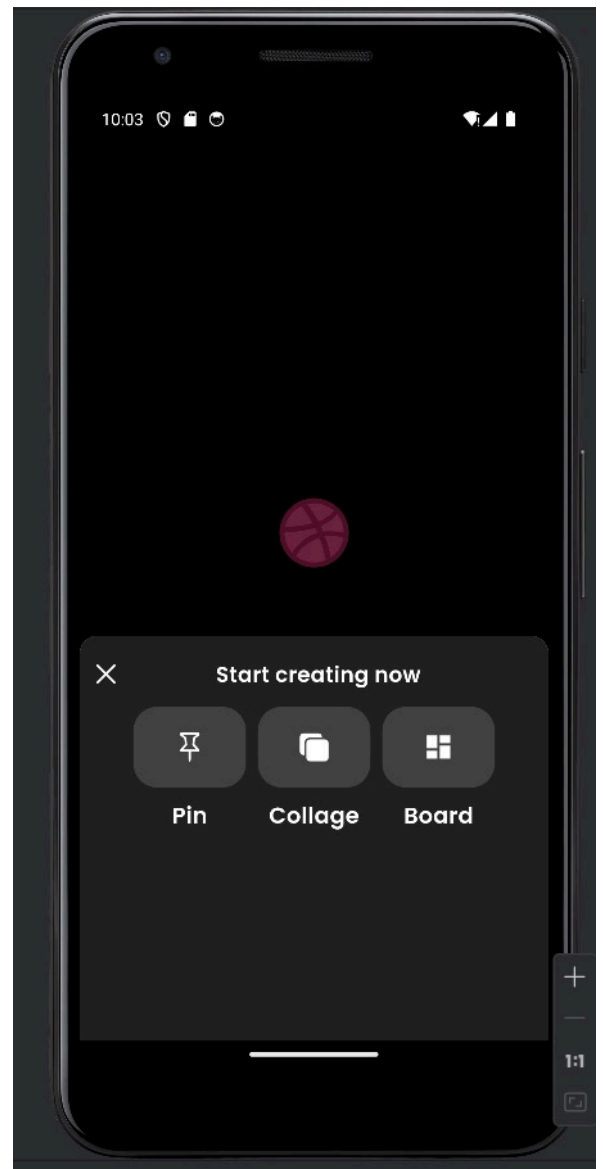
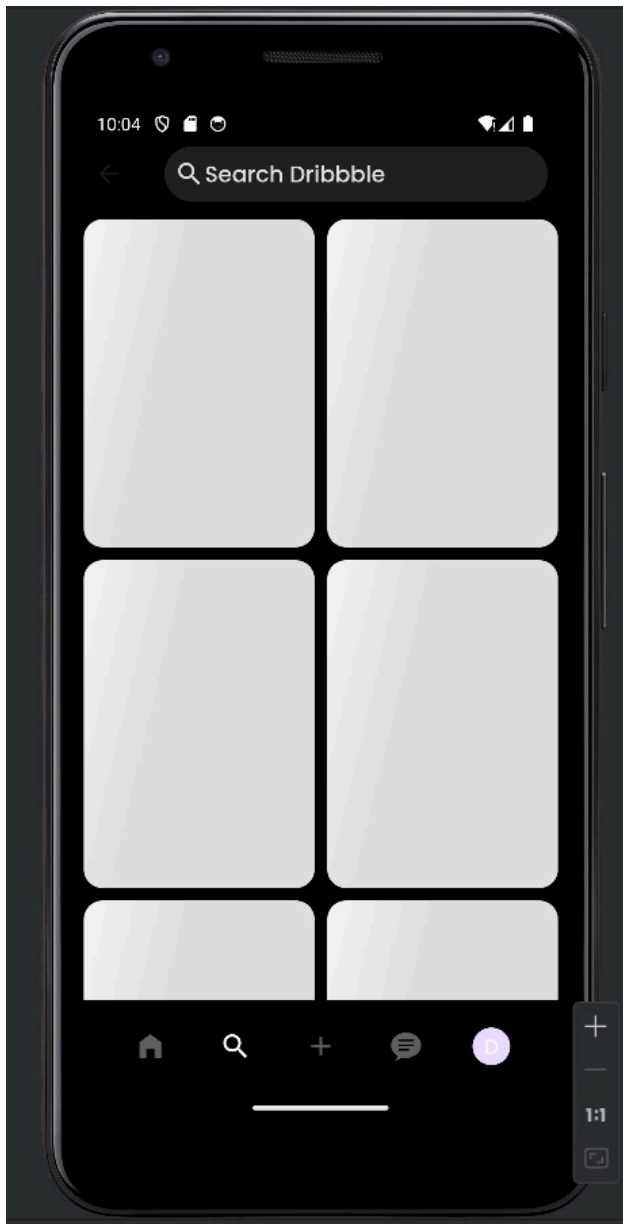
App UI:

\



\





Conclusion: Therefore understood navigation, routing, gesture detection and gesture handling in Flutter and implemented the same in my Flutter application to route different pages.