Experiment No 5

Name : Kunal Pal **Div/Roll no :** D15B/50

Lab : MAD & PWA Lab

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

Navigation and Routing in Flutter:

➢ Navigation in Flutter refers to moving between different screens or pages in your app.
➢ Flutter uses a navigation stack to manage the screens, where each screen added to the stack is pushed on top of the previous screen.
➢ You can use the Navigator class and its methods like push, pop, pushReplacement, etc., to navigate between screens.
➢ Routing in Flutter involves defining routes for different screens in your app. You can use named routes to easily navigate to specific screens using their names.

Gestures in Flutter:

➢ Flutter provides a rich set of gesture recognizers for handling user interactions such as tapping, swiping, dragging, etc.
➢ Some commonly used gestures are GestureDetector for detecting taps, DragGestureRecognizer for dragging gestures, ScaleGestureRecognizer for pinch-to-zoom gestures, etc.
➢ You can handle gestures using callbacks like onTap, onLongPress, onPanUpdate, etc., to perform specific actions based on user interactions.

BMI Calculator:

➢ Body Mass Index (BMI) is a measure of body fat based on height and weight. It's commonly used to assess a person's health status related to weight.
➢ In your app, the BMI calculator section would involve taking input from the user for height and weight, calculating the BMI using the formula BMI = weight (kg) / (height (m))^2, and then displaying the calculated BMI along with a health status indicator (e.g., underweight, normal weight, overweight).

Water Intake Calculator:

➢ Calculating daily water intake is based on factors such as age, weight, activity level, and climate.
➢ Your water intake calculator section would require input from the user for these factors and then calculating the recommended daily water intake based on established guidelines or formulas.

Monthly Data Graph:

➢ The monthly data graph would visualize the user's fitness-related data over time, such as BMI changes, water intake levels, activity levels, etc.
➢ You can use Flutter libraries like charts_flutter to create interactive and visually appealing graphs to display this data.

Activity Tracking:

➢ Activity tracking involves monitoring and recording the user's physical activities such as workouts, steps taken, calories burned, etc.
➢ You can use sensors like accelerometers, pedometers, or integrate with fitness APIs to track and display this data in your app.

Notifications:

- ➢ Implement notifications using the flutter_local_notifications package to remind users about their last meal and exercises they have done.
- ➢ Schedule notifications based on user activities such as recording a meal or completing an exercise session.
- ➢ Use notification channels and specific details for different types of notifications to ensure a consistent and user-friendly experience.

Code :

Custom button

```
import 'package:flutter/material.dart';
import 'package:kunal_s_application1/core/app_export.dart';

class CustomBottomAppBar extends StatefulWidget {
  CustomBottomAppBar({this.onChanged});

  Function(BottomBarEnum)? onChanged;

  @override
  CustomBottomAppBarState createState() => CustomBottomAppBarState();
}

class CustomBottomAppBarState extends State<CustomBottomAppBar> {
  List<BottomMenuModel> bottomMenuList = [
    BottomMenuModel(
        icon: ImageConstant.imgDownload,
        activeIcon: ImageConstant.imgDownload,
        type: BottomBarEnum.Download,
        isSelected: true),
    BottomMenuModel(
      icon: ImageConstant.imgSettings,
      activeIcon: ImageConstant.imgSettings,
      type: BottomBarEnum.Settings,
    )
  ];
  @override
  Widget build(BuildContext context) {
    return ClipRRect(
      borderRadius: BorderRadius.vertical(
        bottom: Radius.circular(40.h),
      ),
      child: BottomAppBar(
        shape: CircularNotchedRectangle(),
        color: appTheme.whiteA700,
        child: SizedBox(
          height: 94.v,
          child: Row(
            mainAxisAlignment: MainAxisAlignment.spaceAround,
            children: List.generate(
              bottomMenuList.length,
              (index) {
                return InkWell(
                  onTap: () {
                    for (var element in bottomMenuList) {
                      element.isSelected = false;
                    }
                    bottomMenuList[index].isSelected = true;

widget.onChanged?.call(bottomMenuList[index].type);
                    setState(() {});
                  },
                  child: bottomMenuList[index].isSelected
                      ? CustomImageView(
                          imagePath:
bottomMenuList[index].activeIcon,
                          height: 28.v,
                          width: 96.h,
                          color: appTheme.gray40002,
                        )
                      : CustomImageView(
                          imagePath: bottomMenuList[index].icon,
                          height: 24.v,
                          width: 96.h,
                          color: appTheme.gray40001,
                        ),
                );
              },
            ),
          ),
        ),
      ),
    );
  }
}

enum BottomBarEnum {
  Download,
  Settings,
}

class BottomMenuModel {
  BottomMenuModel({
    required this.icon,
    required this.activeIcon,
    required this.type,
    this.isSelected = false,
  });

  String icon;

  String activeIcon;

  BottomBarEnum type;

  bool isSelected;
}

class DefaultWidget extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Container(
      color: Color(0xffffffff),
      padding: EdgeInsets.all(10),
      child: Center(
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Text(
              'Please replace the respective Widget here',
              style: TextStyle(
                fontSize: 18,
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

Custom floating button

```
import 'package:flutter/material.dart';
import 'package:kunal_s_application1/core/app_export.dart';

class CustomBottomAppBar extends StatefulWidget {
  CustomBottomAppBar({this.onChanged});

  Function(BottomBarEnum)? onChanged;

  @override
  CustomBottomAppBarState createState() => CustomBottomAppBarState();
}

class CustomBottomAppBarState extends State<CustomBottomAppBar> {
  List<BottomMenuModel> bottomMenuList = [
    BottomMenuModel(
        icon: ImageConstant.imgDownload,
        activeIcon: ImageConstant.imgDownload,
        type: BottomBarEnum.Download,
        isSelected: true),
    BottomMenuModel(
      icon: ImageConstant.imgSettings,
      activeIcon: ImageConstant.imgSettings,
      type: BottomBarEnum.Settings,
    )
  ];

  @override
  Widget build(BuildContext context) {
    return ClipRRect(
      borderRadius: BorderRadius.vertical(
        bottom: Radius.circular(40.h),
      ),
      child: BottomAppBar(
        shape: CircularNotchedRectangle(),
        color: appTheme.whiteA700,
        child: SizedBox(
          height: 94.v,
          child: Row(
            mainAxisAlignment: MainAxisAlignment.spaceAround,
            children: List.generate(
              bottomMenuList.length,
              (index) {
                return InkWell(
                  onTap: () {
                    for (var element in bottomMenuList) {
                      element.isSelected = false;
                    }
                    bottomMenuList[index].isSelected = true;

widget.onChanged?.call(bottomMenuList[index].type);
                    setState(() {});
                  },
                  child: bottomMenuList[index].isSelected
                      ? CustomImageView(
                          imagePath:
bottomMenuList[index].activeIcon,
                          height: 28.v,
                          width: 96.h,
                          color: appTheme.gray40002,
                        )
                      : CustomImageView(
                          imagePath: bottomMenuList[index].icon,
                          height: 24.v,
                          width: 96.h,
                          color: appTheme.gray40001,
                        ),
                );
              },
            ),
          ),
        ),
      ),
    );
  }
}

enum BottomBarEnum {
  Download,
  Settings,
}

class BottomMenuModel {
  BottomMenuModel({
    required this.icon,
    required this.activeIcon,
    required this.type,
    this.isSelected = false,
  });
  String icon;

  String activeIcon;

  BottomBarEnum type;

  bool isSelected;
}

class DefaultWidget extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Container(
      color: Color(0xffffffff),
      padding: EdgeInsets.all(10),
      child: Center(
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Text(
              'Please replace the respective Widget here',
              style: TextStyle(
                fontSize: 18,
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

Images view code :

```
// ignore_for_file: must_be_immutable
```

```dart
import 'dart:io';

import 'package:cached_network_image/cached_network_image.dart';
import 'package:flutter/material.dart';
import 'package:flutter_svg/flutter_svg.dart';

class CustomImageView extends StatelessWidget {
  ///[imagePath] is required parameter for showing image
  String? imagePath;

  double? height;
  double? width;
  Color? color;
  BoxFit? fit;
  final String placeHolder;
  Alignment? alignment;
  VoidCallback? onTap;
  EdgeInsetsGeometry? margin;
  BorderRadius? radius;
  BoxBorder? border;

  ///a [CustomImageView] it can be used for showing any type of
images
  /// it will shows the placeholder image if image is not found on
network image
  CustomImageView({
    this.imagePath,
    this.height,
    this.width,
    this.color,
    this.fit,
    this.alignment,
    this.onTap,
    this.radius,
    this.margin,
    this.border,
    this.placeHolder = 'assets/images/image_not_found.png',
  });

  @override
  Widget build(BuildContext context) {
    return alignment != null
        ? Align(
            alignment: alignment!,
            child: _buildWidget(),
          )
        : _buildWidget();
  }

  Widget _buildWidget() {
    return Padding(
      padding: margin ?? EdgeInsets.zero,
      child: InkWell(
        onTap: onTap,
        child: _buildCircleImage(),
      ),
    );
  }

  ///build the image with border radius
  _buildCircleImage() {
    if (radius != null) {
      return ClipRRect(
        borderRadius: radius ?? BorderRadius.zero,
        child: _buildImageWithBorder(),
      );
    } else {
      return _buildImageWithBorder();
    }
  }

  ///build the image with border and border radius style
  _buildImageWithBorder() {
    if (border != null) {
      return Container(
        decoration: BoxDecoration(
          border: border,
          borderRadius: radius,
        ),
        child: _buildImageView(),
      );
    } else {
      return _buildImageView();
    }
  }

  Widget _buildImageView() {
    if (imagePath != null) {
      switch (imagePath!.imageType) {
        case ImageType.svg:
          return Container(
            height: height,
            width: width,
            child: SvgPicture.asset(
              imagePath!,
              height: height,
              width: width,
              fit: fit ?? BoxFit.contain,
              colorFilter: color != null
                  ? ColorFilter.mode(
                      this.color ?? Colors.transparent,
BlendMode.srcIn)
                  : null,
            ),
          );
        case ImageType.file:
          return Image.file(
            File(imagePath!),
            height: height,
            width: width,
            fit: fit ?? BoxFit.cover,
            color: color,
          );
```

```dart
        case ImageType.network:
          return CachedNetworkImage(
            height: height,
            width: width,
            fit: fit,
            imageUrl: imagePath!,
            color: color,
            placeholder: (context, url) => Container(
              height: 30,
              width: 30,
              child: LinearProgressIndicator(
                color: Colors.grey.shade200,
                backgroundColor: Colors.grey.shade100,
              ),
            ),
            errorWidget: (context, url, error) => Image.asset(
              placeHolder,
              height: height,
              width: width,
              fit: fit ?? BoxFit.cover,
            ),
          );
        case ImageType.png:
        default:
          return Image.asset(
            imagePath!,
            height: height,
            width: width,
            fit: fit ?? BoxFit.cover,
            color: color,
          );
      }
    }
    return SizedBox();
  }
}

extension ImageTypeExtension on String {
  ImageType get imageType {
    if (this.startsWith('http') || this.startsWith('https')) {
      return ImageType.network;
    } else if (this.endsWith('.svg')) {
      return ImageType.svg;
    } else if (this.startsWith('file://')) {
      return ImageType.file;
    } else {
      return ImageType.png;
    }
  }
}

enum ImageType { svg, png, network, file, unknown }
```

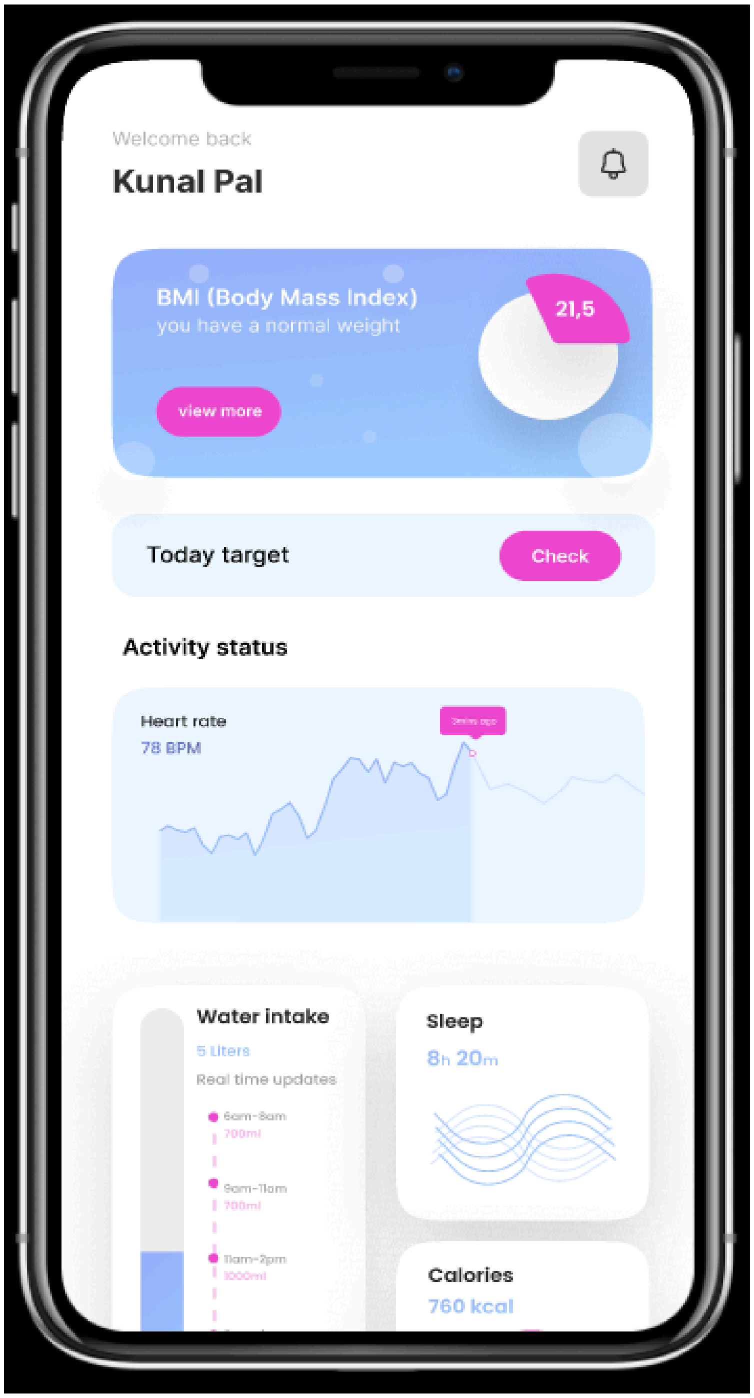
main.dart

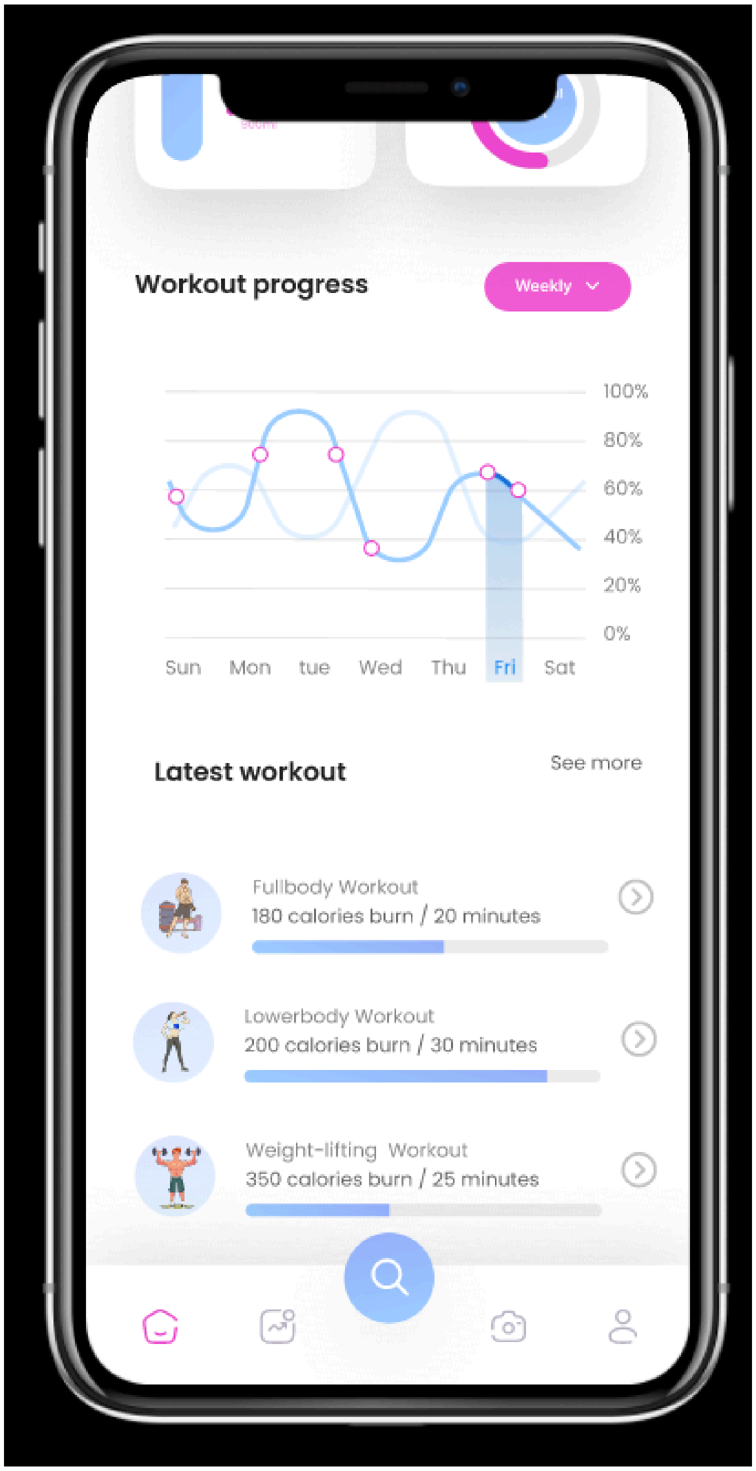
```dart
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:flutter_localizations/flutter_localizations.dart';
import 'core/app_export.dart';

var globalMessengerKey = GlobalKey<ScaffoldMessengerState>();
void main() {
  WidgetsFlutterBinding.ensureInitialized();
  Future.wait([
    SystemChrome.setPreferredOrientations([
      DeviceOrientation.portraitUp,
    ]),
    PrefUtils().init()
  ]).then((value) {
    runApp(MyApp());
  });
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Sizer(
      builder: (context, orientation, deviceType) {
        return BlocProvider(
          create: (context) => ThemeBloc(
            ThemeState(
              themeType: PrefUtils().getThemeData(),
            ),
          ),
          child: BlocBuilder<ThemeBloc, ThemeState>(
            builder: (context, state) {
              return MaterialApp(
                theme: theme,
                title: 'kunal_s_application1',
                navigatorKey: NavigatorService.navigatorKey,
                debugShowCheckedModeBanner: false,
                localizationsDelegates: [
                  AppLocalizationDelegate(),
                  GlobalMaterialLocalizations.delegate,
                  GlobalWidgetsLocalizations.delegate,
                  GlobalCupertinoLocalizations.delegate,
                ],
                supportedLocales: [
                  Locale(
                    'en',
                    '',
                  ),
                ],
                initialRoute: AppRoutes.initialRoute,
                routes: AppRoutes.routes,
              );
            },
          ),
        );
      },
    );
  }
}
```

OutPut :


Welcome back
Kunal Pal
BMI (Body Mass Index)
you have a normal weight
21,5
view more
Today target
Check
Activity status
Heart rate
78 BPM
Water intake
5 Liters
Real time updates
6am-8am
700ml
9am-11am
700ml
11am-2pm
1000ml
Sleep
8h 20m
Calories
760 kcal



Workout progress

Weekly

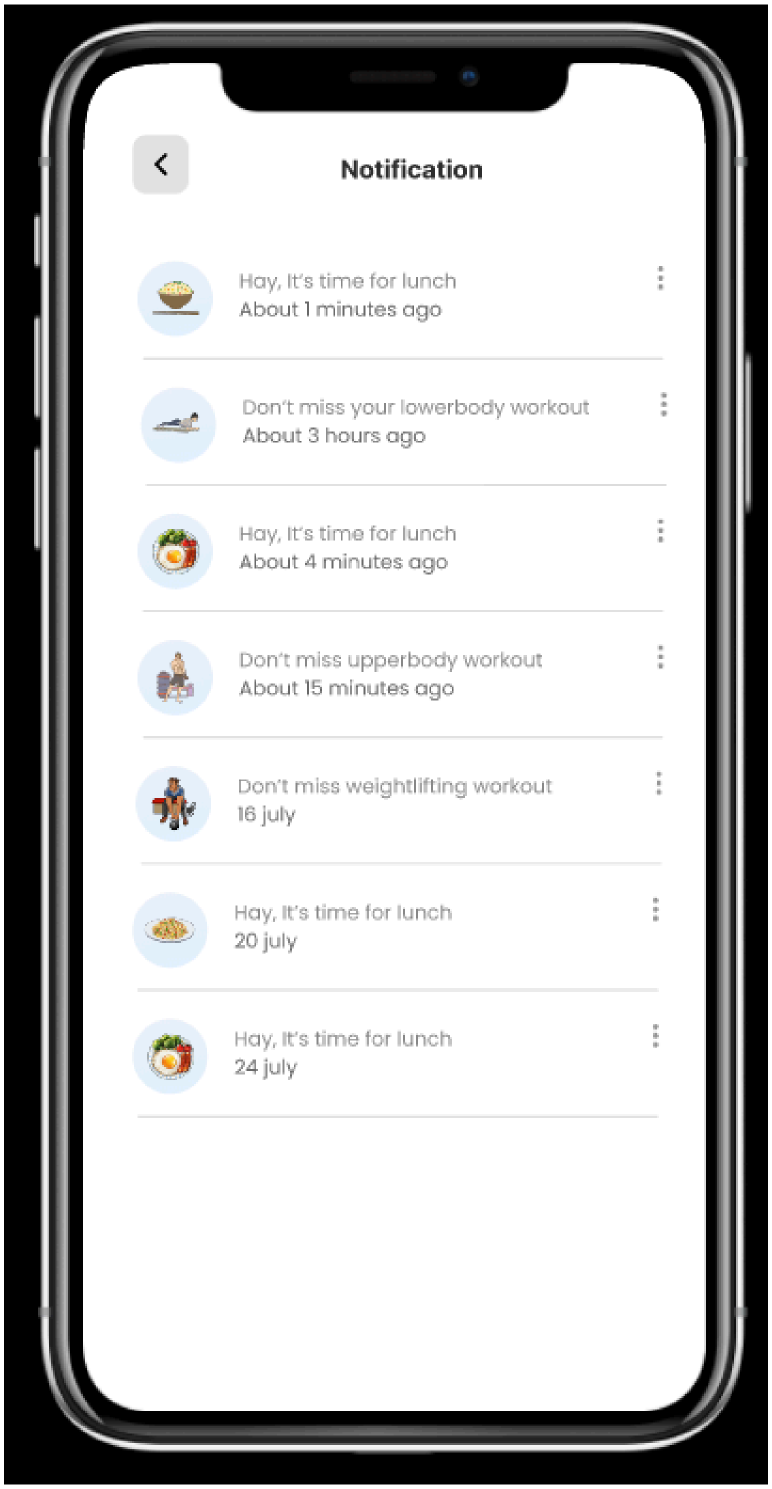
Latest workout

See more

Fullbody Workout
180 calories burn / 20 minutes

Lowerbody Workout
200 calories burn / 30 minutes

Weight-lifting Workout
350 calories burn / 25 minutes



Notification

Hay, It's time for lunch
About 1 minutes ago

Don't miss your lowerbody workout
About 3 hours ago

Hay, It's time for lunch
About 4 minutes ago

Don't miss upperbody workout
About 15 minutes ago

Don't miss weightlifting workout
16 july

Hay, It's time for lunch
20 july

Hay, It's time for lunch
24 july

Conclusion :

In conclusion, developing a fitness tracking app like FitTrack using Flutter involves integrating various features such as BMI and water intake calculators, monthly data graphs, activity tracking, and notifications. Navigation, routing, and gestures are key aspects for seamless user experience. Implementing notifications using the flutter_local_notifications package enhances user engagement by providing timely reminders about their last meal and exercises. This holistic approach ensures a comprehensive and user-friendly fitness tracking experience for users of the FitTrack app.