Shreyash G Dhekane D15A Roll No:16

EXPERIMENT NO :- 05

AIM: - To apply navigation, routing and gestures in Flutter App.

Theory: -

In Flutter, the screens and pages are known as routes, and these routes are just a widget. In Android, a route is similar to an Activity.
In any mobile app, navigating to different pages defines the workflow of the application, and the way to handle the navigation is known as routing. Flutter provides a basic routing class MaterialPageRoute and two methods Navigator.push() and Navigator.pop() that shows how to navigate between two routes. The following steps are required to start navigation in your application. Gestures enable the app to respond to user interactions, making the application more dynamic and responsive.

Navigation and Routing in Flutter

Navigation is the process of moving between different screens or pages in an app. Flutter provides a simple and effective way to handle this through the use of the Navigator widget and routes.

1.Using Navigator Widget
The Navigator widget manages a stack of routes, allowing for pushing and popping routes on the stack.

Pushing a Route: To navigate to a new screen, use Navigator.push().
Popping a Route: To go back to the previous screen, use Navigator.pop().

```
ElevatedButton(
onPressed: () {
Navigator.push(
context,
MaterialPageRoute(builder: (context) => SecondScreen()),
);},
);
```

2.Named Routes
Flutter also allows the use of named routes to navigate, which can make the routing process cleaner, especially in larger applications.

```
MaterialApp(
initialRoute: '/',
routes: {
'/': (context) => HomeScreen(),
'/second': (context) => SecondScreen(),
},
);
```

Navigate to the route using Navigator.pushNamed()

```
Navigator.pushNamed(context, '/second');
```

Handling Gestures in Flutter

Gestures refer to user interactions with the app, such as taps, swipes, pinches, and drags. Flutter provides several widgets and gesture detectors to handle these interactions.

Tap Gestures
The most common gesture is the tap, which can be handled using the GestureDetector widget or specific buttons like InkWell or ElevatedButton.

Long Press Gesture
For long press gestures, Flutter provides the onLongPress callback in GestureDetector or InkWell.

Swipe and Drag Gestures
Flutter also provides swipe and drag gesture handling. The onHorizontalDragUpdate and onVerticalDragUpdate callbacks are used for dragging gestures.

Code:

main.dart

```
import 'package:flutter/material.dart';
import 'home_screen.dart';
import 'log_food_screen.dart';
import 'settings_screen.dart';

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

  @override
  State<MainScreen> createState() =>
_MainScreenState();
}

class _MainScreenState extends
State<MainScreen> {
  int _currentIndex = 0;

  // List of pages corresponding to each
bottom nav item.
  final List<Widget> _pages = const [
    HomeScreen(),
    LogFoodScreen(),
    SettingsScreen(),
  ];

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.black, // Ensures
a black background for the screen
      body: _pages[_currentIndex], // Display
selected page
      bottomNavigationBar:
BottomNavigationBar(
        backgroundColor: Colors.grey[850], //
Ensure a distinct background color
        currentIndex: _currentIndex,
        unselectedItemColor: Colors.white70,
        selectedItemColor: Colors.blue,
        showUnselectedLabels: true,
        items: const [
          BottomNavigationBarItem(
            icon: Icon(Icons.home),
            label: "Home",
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.fastfood),
            label: "Log Food",
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.settings),
            label: "Settings",
          ),
        ],
        onTap: (index) {
          setState(() {
            _currentIndex = index; // Update the
current selected index
          });
        },
      ),
    );
  }
}
```

log_food_screen.dart

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

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

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

class _LogFoodScreenState extends
State<LogFoodScreen> {
  // Dummy list to store custom recipes added
via the custom recipe screen.
  List<Map<String, dynamic>> customRecipes
= [];

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text("Log Food"),
        backgroundColor: Colors.black,
        centerTitle: true,
      ),
      backgroundColor: Colors.black,
      body: SingleChildScrollView(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          crossAxisAlignment:
CrossAxisAlignment.start,
          children: [
            // --------------------------
            // Existing Log Food UI Content
            // --------------------------
            const Text(
              "Enter the details of your meal:",
              style: TextStyle(color: Colors.white,
fontSize: 18),
            ),
            const SizedBox(height: 20),
            TextField(
              style: const TextStyle(color:
Colors.white),
              decoration: InputDecoration(
                labelText: "Food Name",
                labelStyle: const TextStyle(color:
Colors.white70),
                enabledBorder: OutlineInputBorder(
                  borderSide: const
BorderSide(color: Colors.white38),
                  borderRadius:
BorderRadius.circular(8),
                ),
                focusedBorder: OutlineInputBorder(
                  borderSide: const
BorderSide(color: Colors.blue),
                  borderRadius:
BorderRadius.circular(8),
                ),
              ),
            ),
            const SizedBox(height: 20),
            TextField(
              style: const TextStyle(color:
Colors.white),
              decoration: InputDecoration(
                labelText: "Calories",
                labelStyle: const TextStyle(color:
Colors.white70),
                enabledBorder: OutlineInputBorder(
                  borderSide: const
BorderSide(color: Colors.white38),
                  borderRadius:
BorderRadius.circular(8),
                ),
                focusedBorder: OutlineInputBorder(
                  borderSide: const
BorderSide(color: Colors.blue),
                  borderRadius:
BorderRadius.circular(8),
                ),
              ),
            ),
            const SizedBox(height: 20),
            ElevatedButton(
              style: ElevatedButton.styleFrom(
                backgroundColor: Colors.blue,
                minimumSize: const
Size(double.infinity, 50),
              ),
              onPressed: () {
```

```
          // Add logic to log the food entry.
        },
        child: const Text("Log Food", style:
TextStyle(color: Colors.white)),
      ),
      const SizedBox(height: 30),

      // --------------------------
      // Recipe Catalog Section
      // --------------------------
      const Text(
        "Recipe Catalog",
        style: TextStyle(
          color: Colors.white,
          fontSize: 20,
          fontWeight: FontWeight.bold,
        ),
      ),
      const SizedBox(height: 10),
      customRecipes.isEmpty
          ? const Text(
        "No custom recipes added yet.",
        style: TextStyle(color:
Colors.white70),
      )
          : ListView.builder(
        shrinkWrap: true,
        physics: const
NeverScrollableScrollPhysics(),
        itemCount: customRecipes.length,
        itemBuilder: (context, index) {
          final recipe = customRecipes[index];
          return Card(
            color: Colors.grey[900],
            child: ListTile(
              title: Text(recipe['name'],
                  style: const TextStyle(color:
Colors.white)),
                subtitle: Text(
                  "${recipe['ingredients'].length}
ingredients",
                  style: const TextStyle(color:
Colors.white70),
                ),
              ),
            );
          },
        ),
        const SizedBox(height: 80), // Extra
space for the FAB
      ],
    ),
  ),
  floatingActionButton:
FloatingActionButton(
    backgroundColor: Colors.blue,
    onPressed: () async {
      // Navigate to the
AddCustomRecipeScreen and await the
result.
      final result = await Navigator.push(
        context,
        MaterialPageRoute(builder: (context)
=> const AddCustomRecipeScreen()),
      );
      if (result != null && result is Map<String,
dynamic>) {
        setState(() {
          customRecipes.add(result);
        });
      }
    },
    child: const Icon(Icons.add),
  ),
);
}
}
```

add_custom_recipe_screen.dart

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

/// A helper class to hold
TextEditingControllers for each ingredient.
class IngredientField {
  final TextEditingController nameController;
  final TextEditingController caloriesController;
  final TextEditingController proteinController;
  final TextEditingController fibreController;
  final TextEditingController fatController;

  IngredientField({
    String? name,
    String? calories,
    String? protein,
    String? fibre,
    String? fat,
  })  : nameController =
TextEditingController(text: name),
        caloriesController =
TextEditingController(text: calories),
        proteinController =
TextEditingController(text: protein),
        fibreController =
TextEditingController(text: fibre),
        fatController = TextEditingController(text:
fat);

  void dispose() {
    nameController.dispose();
    caloriesController.dispose();
    proteinController.dispose();
    fibreController.dispose();
    fatController.dispose();
  }
}

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

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

class _AddCustomRecipeScreenState
extends State<AddCustomRecipeScreen> {
  final TextEditingController
recipeNameController =
TextEditingController();
  List<IngredientField> ingredients = [];

  @override
  void initState() {
    super.initState();
    // Start with one ingredient input row.
    ingredients.add(IngredientField());
  }

  @override
  void dispose() {
    recipeNameController.dispose();
    for (var ingredient in ingredients) {
      ingredient.dispose();
    }
    super.dispose();
  }

  void addIngredient() {
    setState(() {
      ingredients.add(IngredientField());
    });
  }

  void removeIngredient(int index) {
    setState(() {
      ingredients[index].dispose();
      ingredients.removeAt(index);
    });
  }

  void submitRecipe() {
    final recipeName =
recipeNameController.text.trim();
    if (recipeName.isEmpty) {

ScaffoldMessenger.of(context).showSnackBar
(
        const SnackBar(content: Text("Please
enter a recipe name")),
      );
      return;
    }
    List<Map<String, String>> ingredientList =
[];
    for (var ingredient in ingredients) {
      final name =
ingredient.nameController.text.trim();
      final calories =
ingredient.caloriesController.text.trim();
      final protein =
ingredient.proteinController.text.trim();
      final fibre =
ingredient.fibreController.text.trim();
      final fat =
ingredient.fatController.text.trim();
      if (name.isEmpty ||
          calories.isEmpty ||
          protein.isEmpty ||
        fibre.isEmpty ||
        fat.isEmpty) {

ScaffoldMessenger.of(context).showSnackBar
(
        const SnackBar(content: Text("Please
fill out all fields for each ingredient")),
      );
      return;
     }
     ingredientList.add({
      'name': name,
      'calories': calories,
      'protein': protein,
      'fibre': fibre,
      'fat': fat,
     });
    }

    // For UI purposes, simulate submission.
    // In a real app, this data would be sent to a
database for admin verification.
    final recipeData = {
     'name': recipeName,
     'ingredients': ingredientList,
    };

    // Show a success message and pop with
the recipe data.

ScaffoldMessenger.of(context).showSnackBar
(
     const SnackBar(content: Text("Recipe
submitted for verification")),
    );
    Navigator.pop(context, recipeData);
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
     appBar: AppBar(
      title: const Text("Add Custom Recipe"),
      backgroundColor: Colors.black,
      centerTitle: true,
     ),
     backgroundColor: Colors.black,
     body: SingleChildScrollView(
      padding: const EdgeInsets.all(16.0),
      child: Column(
         crossAxisAlignment:
CrossAxisAlignment.start,
         children: [
          // Recipe Name Field
          TextField(
           controller: recipeNameController,
           style: const TextStyle(color:
Colors.white),
           decoration: InputDecoration(
            labelText: "Recipe Name",
            labelStyle: const TextStyle(color:
Colors.white70),
            enabledBorder: OutlineInputBorder(
             borderSide: const
BorderSide(color: Colors.white38),
             borderRadius:
BorderRadius.circular(8),
            ),
            focusedBorder: OutlineInputBorder(
             borderSide: const
BorderSide(color: Colors.blue),
             borderRadius:
BorderRadius.circular(8),
            ),
           ),
          ),
          const SizedBox(height: 20),
          // Dynamic List of Ingredient Fields
          const Text(
           "Ingredients",
           style: TextStyle(color: Colors.white,
fontSize: 18, fontWeight: FontWeight.bold),
          ),
          const SizedBox(height: 10),
          ListView.builder(
           shrinkWrap: true,
           physics: const
NeverScrollableScrollPhysics(),
           itemCount: ingredients.length,
           itemBuilder: (context, index) {
            return IngredientInputCard(
             ingredientField: ingredients[index],
             onRemove: ingredients.length > 1
? () => removeIngredient(index) : null,
            );
           },
          ),
          const SizedBox(height: 10),
          TextButton.icon(
           onPressed: addIngredient,
            icon: const Icon(Icons.add, color:
Colors.blue),
            label: const Text("Add Ingredient",
style: TextStyle(color: Colors.blue)),
          ),
          const SizedBox(height: 20),
          ElevatedButton(
            style: ElevatedButton.styleFrom(
              backgroundColor: Colors.blue,
              minimumSize: const
Size(double.infinity, 50),
            ),
            onPressed: submitRecipe,
            child: const Text("Submit Recipe",
style: TextStyle(color: Colors.white)),
          ),
        ],
      ),
    ),
  );
 }
}

/// A widget that displays input fields for one
ingredient.
class IngredientInputCard extends
StatelessWidget {
 final IngredientField ingredientField;
 final VoidCallback? onRemove;

 const IngredientInputCard({
   Key? key,
   required this.ingredientField,
   this.onRemove,
 }) : super(key: key);

 @override
 Widget build(BuildContext context) {
   return Card(
     color: Colors.grey[900],
     margin: const
EdgeInsets.symmetric(vertical: 8),
     child: Padding(
       padding: const EdgeInsets.all(8.0),
       child: Column(
         children: [
           // Row for Ingredient Name and
remove button
           Row(
             children: [
               Expanded(
                 child: TextField(
                   controller:
ingredientField.nameController,
                   style: const TextStyle(color:
Colors.white),
                   decoration: InputDecoration(
                     labelText: "Ingredient Name",
                     labelStyle: const TextStyle(color:
Colors.white70),
                     enabledBorder:
OutlineInputBorder(
                       borderSide: const
BorderSide(color: Colors.white38),
                       borderRadius:
BorderRadius.circular(8),
                     ),
                     focusedBorder:
OutlineInputBorder(
                       borderSide: const
BorderSide(color: Colors.blue),
                       borderRadius:
BorderRadius.circular(8),
                     ),
                   ),
                 ),
               ),
               if (onRemove != null)
                 IconButton(
                   icon: const
Icon(Icons.remove_circle, color: Colors.red),
                   onPressed: onRemove,
                 ),
             ],
           ),
           const SizedBox(height: 10),
           // Row for Calories and Protein
           Row(
             children: [
               Expanded(
                 child: TextField(
                   controller:
ingredientField.caloriesController,
                   style: const TextStyle(color:
Colors.white),
                   decoration: InputDecoration(
                     labelText: "Calories",
                     labelStyle: const TextStyle(color:
Colors.white70),
                enabledBorder:
OutlineInputBorder(
                  borderSide: const
BorderSide(color: Colors.white38),
                  borderRadius:
BorderRadius.circular(8),
                ),
                focusedBorder:
OutlineInputBorder(
                  borderSide: const
BorderSide(color: Colors.blue),
                  borderRadius:
BorderRadius.circular(8),
                ),
              ),
              keyboardType:
TextInputType.number,
            ),
          ),
          const SizedBox(width: 8),
          Expanded(
            child: TextField(
              controller:
ingredientField.proteinController,
              style: const TextStyle(color:
Colors.white),
              decoration: InputDecoration(
                labelText: "Protein",
                labelStyle: const TextStyle(color:
Colors.white70),
                enabledBorder:
OutlineInputBorder(
                  borderSide: const
BorderSide(color: Colors.white38),
                  borderRadius:
BorderRadius.circular(8),
                ),
                focusedBorder:
OutlineInputBorder(
                  borderSide: const
BorderSide(color: Colors.blue),
                  borderRadius:
BorderRadius.circular(8),
                ),
              ),
              keyboardType:
TextInputType.number,
            ),
          ),
        ],
      ),
      const SizedBox(height: 10),
      // Row for Fibre and Fat
      Row(
        children: [
          Expanded(
            child: TextField(
              controller:
ingredientField.fibreController,
              style: const TextStyle(color:
Colors.white),
              decoration: InputDecoration(
                labelText: "Fibre",
                labelStyle: const TextStyle(color:
Colors.white70),
                enabledBorder:
OutlineInputBorder(
                  borderSide: const
BorderSide(color: Colors.white38),
                  borderRadius:
BorderRadius.circular(8),
                ),
                focusedBorder:
OutlineInputBorder(
                  borderSide: const
BorderSide(color: Colors.blue),
                  borderRadius:
BorderRadius.circular(8),
                ),
              ),
              keyboardType:
TextInputType.number,
            ),
          ),
          const SizedBox(width: 8),
          Expanded(
            child: TextField(
              controller:
ingredientField.fatController,
              style: const TextStyle(color:
Colors.white),
              decoration: InputDecoration(
                labelText: "Fat",
                labelStyle: const TextStyle(color:
Colors.white70),
                enabledBorder:
OutlineInputBorder(
                  borderSide: const
BorderSide(color: Colors.white38),
            borderRadius:
BorderRadius.circular(8),
           ),
           focusedBorder:
OutlineInputBorder(
            borderSide: const
BorderSide(color: Colors.blue),
            borderRadius:
BorderRadius.circular(8),
           ),
          ),
          keyboardType:
TextInputType.number,
         ),
        ),
       ],
      ),
     ],
    ),
   ),
  );
 }
}
```

setting_screen.dart

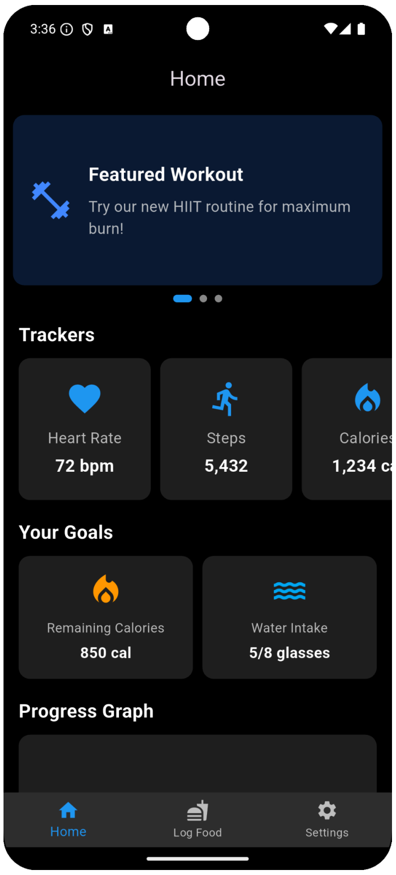
```
import 'package:flutter/material.dart';

class SettingsScreen extends
StatelessWidget {
 const SettingsScreen({super.key});

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: const Text("Settings"),
    backgroundColor: Colors.black,
    centerTitle: true,
   ),
   backgroundColor: Colors.black,
   body: ListView(
    padding: const EdgeInsets.all(16.0),
    children: [
     ListTile(
      leading: const Icon(Icons.person,
color: Colors.blue),
      title: const Text("Account", style:
TextStyle(color: Colors.white)),
      trailing: const
Icon(Icons.arrow_forward_ios, color:
Colors.white70, size: 16),
      onTap: () {
       // Navigate to account settings page
      },
     ),
     const Divider(color: Colors.white24),
     ListTile(
      leading: const Icon(Icons.notifications,
color: Colors.blue),
      title: const Text("Notifications", style:
TextStyle(color: Colors.white)),
      trailing: const
Icon(Icons.arrow_forward_ios, color:
Colors.white70, size: 16),
      onTap: () {
       // Navigate to notifications settings
page
      },
     ),
     const Divider(color: Colors.white24),
     ListTile(
      leading: const Icon(Icons.lock, color:
Colors.blue),
      title: const Text("Privacy", style:
TextStyle(color: Colors.white)),
      trailing: const
Icon(Icons.arrow_forward_ios, color:
Colors.white70, size: 16),
      onTap: () {
       // Navigate to privacy settings page
      },
     ),
     // Add additional settings options as
needed...
    ],
   ),
  );
 }
}
```

Screenshots:

Shreyash G Dhekane D15A Roll No:16


3:36

Home

Featured Workout
Try our new HIIT routine for maximum burn!

Trackers

Heart Rate
72 bpm

Steps
5,432

Calories
1,234 ca

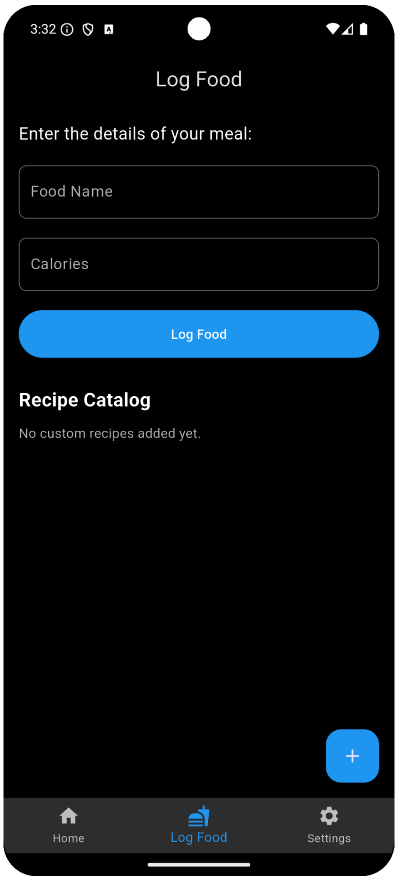
Your Goals

Remaining Calories
850 cal

Water Intake
5/8 glasses

Progress Graph

Home Log Food Settings


3:32

Log Food

Enter the details of your meal:

Food Name

Calories

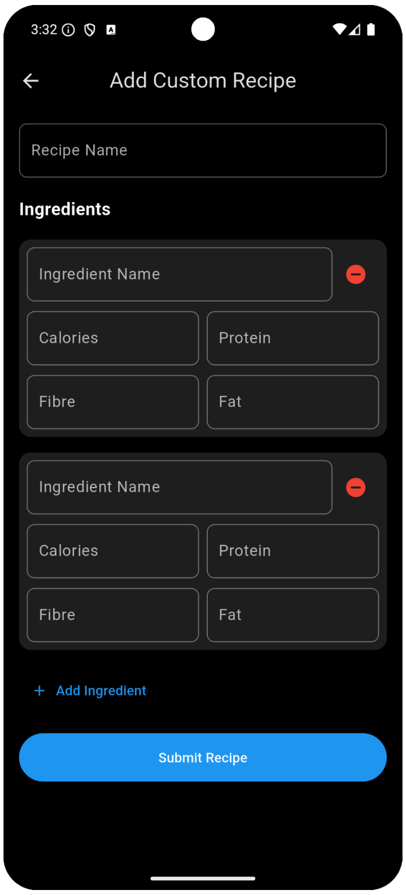
Log Food

Recipe Catalog

No custom recipes added yet.

+

Home Log Food Settings


3:32

Add Custom Recipe

Recipe Name

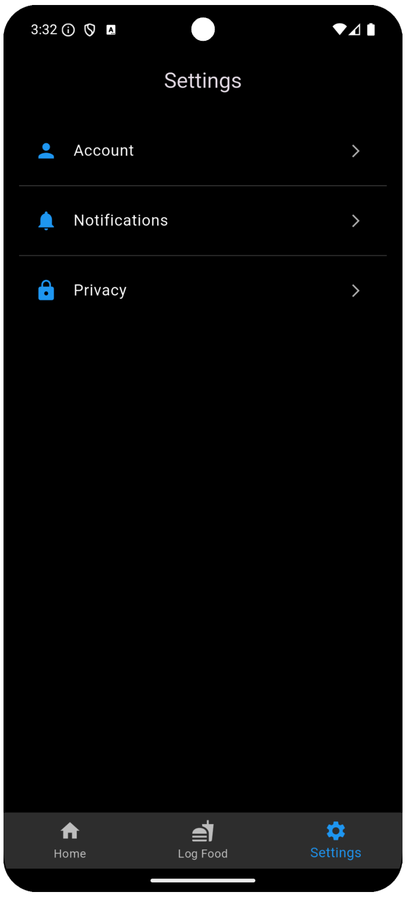
Ingredients

Ingredient Name
Calories Protein
Fibre Fat

Ingredient Name
Calories Protein
Fibre Fat

+ Add Ingredient

Submit Recipe


3:32

Settings

Account

Notifications

Privacy

Home Log Food Settings