1. Main App Structure & Models

main.dart

dart

import 'package:flutter/material.dart';

import 'screens/home\_screen.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'FitTrack Pro',

theme: ThemeData(

primarySwatch: Colors.blue,

visualDensity: VisualDensity.adaptivePlatformDensity,

),

home: HomeScreen(),

debugShowCheckedModeBanner: false,

);

}

}

models/exercise\_model.dart

dart

class Exercise {

final String id;

final String name;

final String type;

final String muscleGroup;

final String difficulty;

final String instructions;

final String imageUrl;

final int duration; // in seconds

Exercise({

required this.id,

required this.name,

required this.type,

required this.muscleGroup,

required this.difficulty,

required this.instructions,

required this.imageUrl,

required this.duration,

});

}

class Workout {

final String id;

final String name;

final List<Exercise> exercises;

final int totalDuration;

final DateTime created;

int caloriesBurned;

Workout({

required this.id,

required this.name,

required this.exercises,

required this.totalDuration,

required this.created,

this.caloriesBurned = 0,

});

}

class FoodEntry {

final String id;

final String name;

final int calories;

final DateTime timestamp;

FoodEntry({

required this.id,

required this.name,

required this.calories,

required this.timestamp,

});

}

2. Home Screen

screens/home\_screen.dart

dart

import 'package:flutter/material.dart';

import 'workout\_tracker\_screen.dart';

import 'calorie\_tracker\_screen.dart';

import 'exercise\_library\_screen.dart';

import 'progress\_screen.dart';

class HomeScreen extends StatelessWidget {

final List<Map<String, dynamic>> features = [

{

'title': 'Workout Tracker',

'icon': Icons.fitness\_center,

'color': Colors.blue,

'screen': WorkoutTrackerScreen(),

},

{

'title': 'Calorie Tracker',

'icon': Icons.restaurant,

'color': Colors.green,

'screen': CalorieTrackerScreen(),

},

{

'title': 'Exercise Library',

'icon': Icons.video\_library,

'color': Colors.orange,

'screen': ExerciseLibraryScreen(),

},

{

'title': 'Progress',

'icon': Icons.trending\_up,

'color': Colors.purple,

'screen': ProgressScreen(),

},

];

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('FitTrack Pro'),

backgroundColor: Colors.blue,

foregroundColor: Colors.white,

elevation: 0,

),

body: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

// Header Section

Container(

width: double.infinity,

padding: EdgeInsets.all(20),

decoration: BoxDecoration(

color: Colors.blue,

borderRadius: BorderRadius.only(

bottomLeft: Radius.circular(20),

bottomRight: Radius.circular(20),

),

),

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

Text(

'Welcome Back!',

style: TextStyle(

color: Colors.white,

fontSize: 24,

fontWeight: FontWeight.bold,

),

),

SizedBox(height: 8),

Text(

'Ready for your next workout?',

style: TextStyle(

color: Colors.white70,

fontSize: 16,

),

),

],

),

),

// Features Grid

Expanded(

child: Padding(

padding: EdgeInsets.all(16),

child: GridView.builder(

gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(

crossAxisCount: 2,

crossAxisSpacing: 16,

mainAxisSpacing: 16,

childAspectRatio: 1.2,

),

itemCount: features.length,

itemBuilder: (context, index) {

final feature = features[index];

return \_buildFeatureCard(

context,

feature['title'],

feature['icon'],

feature['color'],

feature['screen'],

);

},

),

),

),

],

),

);

}

Widget \_buildFeatureCard(

BuildContext context, String title, IconData icon, Color color, Widget screen) {

return Card(

elevation: 4,

shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(12)),

child: InkWell(

onTap: () {

Navigator.push(

context,

MaterialPageRoute(builder: (context) => screen),

);

},

borderRadius: BorderRadius.circular(12),

child: Container(

padding: EdgeInsets.all(16),

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

Container(

padding: EdgeInsets.all(12),

decoration: BoxDecoration(

color: color.withOpacity(0.1),

shape: BoxShape.circle,

),

child: Icon(icon, color: color, size: 30),

),

SizedBox(height: 12),

Text(

title,

style: TextStyle(

fontSize: 16,

fontWeight: FontWeight.bold,

),

textAlign: TextAlign.center,

),

],

),

),

),

);

}

}

3. Workout Tracker Screen

screens/workout\_tracker\_screen.dart

dart

import 'package:flutter/material.dart';

import '../models/exercise\_model.dart';

class WorkoutTrackerScreen extends StatefulWidget {

@override

\_WorkoutTrackerScreenState createState() => \_WorkoutTrackerScreenState();

}

class \_WorkoutTrackerScreenState extends State<WorkoutTrackerScreen> {

List<Workout> workouts = [];

List<Exercise> availableExercises = [

Exercise(

id: '1',

name: 'Push-ups',

type: 'Strength',

muscleGroup: 'Chest',

difficulty: 'Beginner',

instructions: 'Place hands shoulder-width apart and lower your body until chest nearly touches floor.',

imageUrl: 'assets/pushups.jpg',

duration: 180,

),

Exercise(

id: '2',

name: 'Squats',

type: 'Strength',

muscleGroup: 'Legs',

difficulty: 'Beginner',

instructions: 'Stand with feet shoulder-width apart, lower hips back and down.',

imageUrl: 'assets/squats.jpg',

duration: 120,

),

Exercise(

id: '3',

name: 'Plank',

type: 'Core',

muscleGroup: 'Abs',

difficulty: 'Beginner',

instructions: 'Hold your body in a straight line supported on forearms and toes.',

imageUrl: 'assets/plank.jpg',

duration: 60,

),

];

TextEditingController workoutNameController = TextEditingController();

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Workout Tracker'),

actions: [

IconButton(

icon: Icon(Icons.add),

onPressed: \_showCreateWorkoutDialog,

),

],

),

body: workouts.isEmpty

? Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

Icon(Icons.fitness\_center, size: 64, color: Colors.grey),

SizedBox(height: 16),

Text(

'No workouts yet',

style: TextStyle(fontSize: 18, color: Colors.grey),

),

SizedBox(height: 8),

Text(

'Tap + to create your first workout',

style: TextStyle(color: Colors.grey),

),

],

),

)

: ListView.builder(

itemCount: workouts.length,

itemBuilder: (context, index) {

final workout = workouts[index];

return Card(

margin: EdgeInsets.symmetric(horizontal: 16, vertical: 8),

child: ListTile(

leading: CircleAvatar(

backgroundColor: Colors.blue.shade100,

child: Icon(Icons.fitness\_center, color: Colors.blue),

),

title: Text(workout.name),

subtitle: Text(

'${workout.exercises.length} exercises • ${workout.totalDuration ~/ 60} min'),

trailing: Text('${workout.caloriesBurned} cal'),

onTap: () {

\_showWorkoutDetails(workout);

},

),

);

},

),

);

}

void \_showCreateWorkoutDialog() {

showDialog(

context: context,

builder: (context) => AlertDialog(

title: Text('Create New Workout'),

content: Column(

mainAxisSize: MainAxisSize.min,

children: [

TextField(

controller: workoutNameController,

decoration: InputDecoration(

labelText: 'Workout Name',

border: OutlineInputBorder(),

),

),

SizedBox(height: 16),

Text('Select Exercises:'),

Expanded(

child: Container(

height: 200,

child: ListView.builder(

itemCount: availableExercises.length,

itemBuilder: (context, index) {

final exercise = availableExercises[index];

return CheckboxListTile(

title: Text(exercise.name),

subtitle: Text('${exercise.duration}s • ${exercise.muscleGroup}'),

value: false,

onChanged: (value) {

// Handle exercise selection

},

);

},

),

),

),

],

),

actions: [

TextButton(

onPressed: () => Navigator.pop(context),

child: Text('Cancel'),

),

ElevatedButton(

onPressed: () {

\_createWorkout();

Navigator.pop(context);

},

child: Text('Create'),

),

],

),

);

}

void \_createWorkout() {

if (workoutNameController.text.isNotEmpty) {

final newWorkout = Workout(

id: DateTime.now().millisecondsSinceEpoch.toString(),

name: workoutNameController.text,

exercises: availableExercises.sublist(0, 3), // Sample exercises

totalDuration: availableExercises.sublist(0, 3).fold(

0, (total, exercise) => total + exercise.duration),

created: DateTime.now(),

caloriesBurned: 150,

);

setState(() {

workouts.add(newWorkout);

});

workoutNameController.clear();

}

}

void \_showWorkoutDetails(Workout workout) {

showModalBottomSheet(

context: context,

builder: (context) => Container(

padding: EdgeInsets.all(16),

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

mainAxisSize: MainAxisSize.min,

children: [

Text(

workout.name,

style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),

),

SizedBox(height: 16),

Text('Exercises:'),

...workout.exercises.map((exercise) => ListTile(

leading: Icon(Icons.play\_arrow),

title: Text(exercise.name),

subtitle: Text('${exercise.duration}s'),

)),

SizedBox(height: 16),

Row(

mainAxisAlignment: MainAxisAlignment.spaceBetween,

children: [

Text('Total Duration: ${workout.totalDuration ~/ 60} min'),

Text('Calories: ${workout.caloriesBurned} cal'),

],

),

],

),

),

);

}

}

4. Calorie Tracker Screen

screens/calorie\_tracker\_screen.dart

dart

import 'package:flutter/material.dart';

import '../models/exercise\_model.dart';

class CalorieTrackerScreen extends StatefulWidget {

@override

\_CalorieTrackerScreenState createState() => \_CalorieTrackerScreenState();

}

class \_CalorieTrackerScreenState extends State<CalorieTrackerScreen> {

int dailyCalorieLimit = 2000;

List<FoodEntry> foodEntries = [];

TextEditingController foodNameController = TextEditingController();

TextEditingController calorieController = TextEditingController();

int get totalCaloriesConsumed {

return foodEntries.fold(0, (total, entry) => total + entry.calories);

}

int get remainingCalories {

return dailyCalorieLimit - totalCaloriesConsumed;

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Calorie Tracker'),

actions: [

IconButton(

icon: Icon(Icons.settings),

onPressed: \_showCalorieSettings,

),

],

),

body: Column(

children: [

// Calorie Summary Card

Card(

margin: EdgeInsets.all(16),

child: Padding(

padding: EdgeInsets.all(16),

child: Column(

children: [

Text(

'Daily Calorie Summary',

style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),

),

SizedBox(height: 16),

Row(

mainAxisAlignment: MainAxisAlignment.spaceAround,

children: [

\_buildCalorieInfo('Limit', dailyCalorieLimit.toString()),

\_buildCalorieInfo('Consumed', totalCaloriesConsumed.toString()),

\_buildCalorieInfo('Remaining', remainingCalories.toString(),

color: remainingCalories >= 0 ? Colors.green : Colors.red),

],

),

SizedBox(height: 16),

LinearProgressIndicator(

value: totalCaloriesConsumed / dailyCalorieLimit,

backgroundColor: Colors.grey[200],

valueColor: AlwaysStoppedAnimation<Color>(

totalCaloriesConsumed <= dailyCalorieLimit

? Colors.green

: Colors.red,

),

),

],

),

),

),

// Add Food Entry

Padding(

padding: EdgeInsets.symmetric(horizontal: 16),

child: Row(

children: [

Expanded(

flex: 2,

child: TextField(

controller: foodNameController,

decoration: InputDecoration(

labelText: 'Food Name',

border: OutlineInputBorder(),

),

),

),

SizedBox(width: 8),

Expanded(

flex: 1,

child: TextField(

controller: calorieController,

decoration: InputDecoration(

labelText: 'Calories',

border: OutlineInputBorder(),

),

keyboardType: TextInputType.number,

),

),

SizedBox(width: 8),

IconButton(

icon: Icon(Icons.add\_circle, color: Colors.green),

onPressed: \_addFoodEntry,

),

],

),

),

// Food Entries List

Expanded(

child: foodEntries.isEmpty

? Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

Icon(Icons.restaurant, size: 64, color: Colors.grey),

SizedBox(height: 16),

Text(

'No food entries yet',

style: TextStyle(fontSize: 18, color: Colors.grey),

),

],

),

)

: ListView.builder(

itemCount: foodEntries.length,

itemBuilder: (context, index) {

final entry = foodEntries[index];

return Dismissible(

key: Key(entry.id),

background: Container(color: Colors.red),

onDismissed: (direction) {

setState(() {

foodEntries.removeAt(index);

});

},

child: Card(

margin: EdgeInsets.symmetric(horizontal: 16, vertical: 4),

child: ListTile(

leading: CircleAvatar(

backgroundColor: Colors.green.shade100,

child: Icon(Icons.restaurant, color: Colors.green),

),

title: Text(entry.name),

subtitle: Text('Added: ${\_formatTime(entry.timestamp)}'),

trailing: Text(

'${entry.calories} cal',

style: TextStyle(

fontWeight: FontWeight.bold,

color: Colors.red,

),

),

),

),

);

},

),

),

],

),

);

}

Widget \_buildCalorieInfo(String label, String value, {Color? color}) {

return Column(

children: [

Text(

label,

style: TextStyle(color: Colors.grey),

),

SizedBox(height: 4),

Text(

value,

style: TextStyle(

fontSize: 18,

fontWeight: FontWeight.bold,

color: color ?? Colors.blue,

),

),

],

);

}

void \_addFoodEntry() {

if (foodNameController.text.isNotEmpty && calorieController.text.isNotEmpty) {

final newEntry = FoodEntry(

id: DateTime.now().millisecondsSinceEpoch.toString(),

name: foodNameController.text,

calories: int.parse(calorieController.text),

timestamp: DateTime.now(),

);

setState(() {

foodEntries.add(newEntry);

});

foodNameController.clear();

calorieController.clear();

}

}

void \_showCalorieSettings() {

showDialog(

context: context,

builder: (context) => AlertDialog(

title: Text('Daily Calorie Limit'),

content: TextField(

keyboardType: TextInputType.number,

decoration: InputDecoration(

labelText: 'Calories',

border: OutlineInputBorder(),

),

onChanged: (value) {

if (value.isNotEmpty) {

setState(() {

dailyCalorieLimit = int.parse(value);

});

}

},

),

actions: [

TextButton(

onPressed: () => Navigator.pop(context),

child: Text('Cancel'),

),

ElevatedButton(

onPressed: () => Navigator.pop(context),

child: Text('Save'),

),

],

),

);

}

String \_formatTime(DateTime timestamp) {

return '${timestamp.hour}:${timestamp.minute.toString().padLeft(2, '0')}';

}

}

5. Exercise Library Screen

screens/exercise\_library\_screen.dart

dart

import 'package:flutter/material.dart';

import '../models/exercise\_model.dart';

class ExerciseLibraryScreen extends StatefulWidget {

@override

\_ExerciseLibraryScreenState createState() => \_ExerciseLibraryScreenState();

}

class \_ExerciseLibraryScreenState extends State<ExerciseLibraryScreen> {

List<Exercise> exercises = [

Exercise(

id: '1',

name: 'Push-ups',

type: 'Strength',

muscleGroup: 'Chest',

difficulty: 'Beginner',

instructions: '1. Place your hands shoulder-width apart\n2. Keep your body in a straight line\n3. Lower your body until your chest nearly touches the floor\n4. Push back up to the starting position',

imageUrl: 'assets/pushups.jpg',

duration: 180,

),

Exercise(

id: '2',

name: 'Squats',

type: 'Strength',

muscleGroup: 'Legs',

difficulty: 'Beginner',

instructions: '1. Stand with feet shoulder-width apart\n2. Lower your hips back and down\n3. Keep your chest up and back straight\n4. Go as low as you can comfortably\n5. Push through your heels to return to start',

imageUrl: 'assets/squats.jpg',

duration: 120,

),

Exercise(

id: '3',

name: 'Plank',

type: 'Core',

muscleGroup: 'Abs',

difficulty: 'Beginner',

instructions: '1. Place forearms on the ground\n2. Keep your body in a straight line\n3. Engage your core muscles\n4. Hold for the desired time',

imageUrl: 'assets/plank.jpg',

duration: 60,

),

];

String selectedFilter = 'All';

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Exercise Library'),

),

body: Column(

children: [

// Filter Chips

Container(

height: 60,

child: ListView(

scrollDirection: Axis.horizontal,

padding: EdgeInsets.symmetric(horizontal: 16),

children: [

'All',

'Chest',

'Legs',

'Abs',

'Arms',

'Back',

].map((filter) {

return Padding(

padding: EdgeInsets.only(right: 8),

child: FilterChip(

label: Text(filter),

selected: selectedFilter == filter,

onSelected: (selected) {

setState(() {

selectedFilter = filter;

});

},

),

);

}).toList(),

),

),

// Exercises List

Expanded(

child: ListView.builder(

itemCount: exercises.length,

itemBuilder: (context, index) {

final exercise = exercises[index];

if (selectedFilter != 'All' && exercise.muscleGroup != selectedFilter) {

return SizedBox.shrink();

}

return Card(

margin: EdgeInsets.symmetric(horizontal: 16, vertical: 8),

child: ListTile(

leading: CircleAvatar(

backgroundColor: \_getMuscleGroupColor(exercise.muscleGroup),

child: Text(

exercise.muscleGroup[0],

style: TextStyle(color: Colors.white),

),

),

title: Text(exercise.name),

subtitle: Text('${exercise.type} • ${exercise.difficulty}'),

trailing: Icon(Icons.arrow\_forward\_ios, size: 16),

onTap: () {

\_showExerciseDetails(exercise);

},

),

);

},

),

),

],

),

);

}

void \_showExerciseDetails(Exercise exercise) {

showModalBottomSheet(

context: context,

isScrollControlled: true,

builder: (context) => Container(

padding: EdgeInsets.all(16),

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

mainAxisSize: MainAxisSize.min,

children: [

Row(

children: [

CircleAvatar(

backgroundColor: \_getMuscleGroupColor(exercise.muscleGroup),

child: Text(

exercise.muscleGroup[0],

style: TextStyle(color: Colors.white),

),

),

SizedBox(width: 16),

Expanded(

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

Text(

exercise.name,

style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),

),

Text('${exercise.type} • ${exercise.difficulty}'),

],

),

),

],

),

SizedBox(height: 16),

Text(

'Instructions:',

style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),

),

SizedBox(height: 8),

Text(exercise.instructions),

SizedBox(height: 16),

Row(

children: [

Icon(Icons.timer, size: 16),

SizedBox(width: 4),

Text('Duration: ${exercise.duration} seconds'),

],

),

SizedBox(height: 16),

SizedBox(

width: double.infinity,

child: ElevatedButton(

onPressed: () {

// Start exercise timer

},

child: Text('Start Exercise'),

),

),

],

),

),

);

}

Color \_getMuscleGroupColor(String muscleGroup) {

switch (muscleGroup.toLowerCase()) {

case 'chest':

return Colors.red;

case 'legs':

return Colors.green;

case 'abs':

return Colors.blue;

case 'arms':

return Colors.orange;

case 'back':

return Colors.purple;

default:

return Colors.grey;

}

}

}

6. Progress Screen

screens/progress\_screen.dart

dart

import 'package:flutter/material.dart';

import 'package:syncfusion\_flutter\_charts/charts.dart';

class ProgressScreen extends StatefulWidget {

@override

\_ProgressScreenState createState() => \_ProgressScreenState();

}

class \_ProgressScreenState extends State<ProgressScreen> {

List<WorkoutData> weeklyData = [

WorkoutData('Mon', 3, 450),

WorkoutData('Tue', 2, 300),

WorkoutData('Wed', 4, 600),

WorkoutData('Thu', 1, 150),

WorkoutData('Fri', 3, 450),

WorkoutData('Sat', 2, 300),

WorkoutData('Sun', 0, 0),

];

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Progress Tracking'),

),

body: SingleChildScrollView(

padding: EdgeInsets.all(16),

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

// Weekly Summary Card

Card(

child: Padding(

padding: EdgeInsets.all(16),

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

Text(

'This Week',

style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),

),

SizedBox(height: 16),

Row(

mainAxisAlignment: MainAxisAlignment.spaceAround,

children: [

\_buildStatCard('Workouts', '12'),

\_buildStatCard('Total Time', '8h 30m'),

\_buildStatCard('Calories', '3,450'),

],

),

],

),

),

),

SizedBox(height: 16),

// Workouts Chart

Card(

child: Padding(

padding: EdgeInsets.all(16),

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

Text(

'Weekly Workouts',

style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),

),

SizedBox(height: 16),

Container(

height: 200,

child: SfCartesianChart(

primaryXAxis: CategoryAxis(),

series: <CartesianSeries>[

ColumnSeries<WorkoutData, String>(

dataSource: weeklyData,

xValueMapper: (WorkoutData data, \_) => data.day,

yValueMapper: (WorkoutData data, \_) => data.workouts,

color: Colors.blue,

),

],

),

),

],

),

),

),

SizedBox(height: 16),

// Calories Chart

Card(

child: Padding(

padding: EdgeInsets.all(16),

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

Text(

'Calories Burned',

style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),

),

SizedBox(height: 16),

Container(

height: 200,

child: SfCartesianChart(

primaryXAxis: CategoryAxis(),

series: <CartesianSeries>[

LineSeries<WorkoutData, String>(

dataSource: weeklyData,

xValueMapper: (WorkoutData data, \_) => data.day,

yValueMapper: (WorkoutData data, \_) => data.calories,

color: Colors.green,

markerSettings: MarkerSettings(isVisible: true),

),

],

),

),

],

),

),

),

],

),

),

);

}

Widget \_buildStatCard(String title, String value) {

return Column(

children: [

Text(

title,

style: TextStyle(color: Colors.grey, fontSize: 12),

),

SizedBox(height: 4),

Text(

value,

style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),

),

],

);

}

}

class WorkoutData {

final String day;

final int workouts;

final int calories;

WorkoutData(this.day, this.workouts, this.calories);

}

7. Pubspec.yaml Dependencies

Add these dependencies to your pubspec.yaml:

yaml

dependencies:

flutter:

sdk: flutter

syncfusion\_flutter\_charts: ^20.4.48

dev\_dependencies:

flutter\_test:

sdk: flutter

flutter\_lints: ^2.0.0