

# Course project

CCSW 315 Software Process Models

## Daily Habit Tracker

Team members:

Name	ID
Yara Yasser Alshikhy	2211162
Jumana Muhammed AL-Jouhi	2210500
Galla Muhammed Alfaisal	2212048
Tamara Emran Hwsawi	2317148
Hitoon Adel Alasmarei	2210994

# **Table of contents:**

<b>Sprint 0: Project Initiation.....</b>	3
<b>Project overview:.....</b>	4
<b>Scope:.....</b>	5
<b>Functional and non-functional requirements.....</b>	6
<b>Initial Use case and Class Diagrams:.....</b>	7
<b>Sprint 1.....</b>	9
<b>Sprint #1 initial Meeting.....</b>	10
<b>Requirements Documentation.....</b>	11
<b>Sprint #1 Stand up Meeting.....</b>	14
<b>Sprint #1 Test Cases - [9/11/2025].....</b>	15
<b>Sprint 2.....</b>	24
<b>Sprint #2 initial Meeting.....</b>	25
<b>Requirements Documentation.....</b>	26
<b>Sprint #2 Stand-up Meeting.....</b>	30
<b>Sprint #2 Test Cases - [13/11/2025].....</b>	31
<b>Conclusion.....</b>	44
<b>Tools setup :</b> .....	44

# **Sprint 0: Project Initiation**

## **Project overview:**

### **Project Description:**

The Daily Habit Tracker is a mobile application designed to help individuals build, monitor, and maintain positive habits through simple and intuitive daily tracking. The app allows users to create personalized habits, set their desired tracking frequency, and receive timely reminders that support consistent routine-building. With features like streak tracking, progress visualization, calendar views, and motivational badges, the application encourages long-term commitment and personal growth. All data is stored locally on the device, ensuring fast performance and seamless user experience.

### **Problem Definition:**

Many individuals find it difficult to establish and maintain positive habits due to several common challenges, which often lead to inconsistency or early abandonment of habits:

- Lack of consistent reminders to perform tasks.
- Limited visibility into progress or improvement over time.
- Decreased motivation when results are not immediately noticeable.
- Feeling overwhelmed when managing multiple habits at once.

### **Proposed Solution:**

The proposed solution is to develop a Daily Habit Tracker mobile application that:

- Allows users to add, edit, and customize their habits based on personal goals.
- Supports flexible tracking (daily, weekly, monthly) depending on the habit type.
- Provides visual progress feedback through charts, streaks, and calendar views.
- Sends timely reminders and notifications to encourage consistency.
- Offers simple motivational features such as badges or streak rewards.
- Includes habit archiving and history management for better organization.

## **Scope:**

### **In Scope:**

- Habit creation, editing, and customization.
- Progress visualization (calendar view, charts, streaks).
- Notifications and reminders for habits.
- Basic motivational features (badges/rewards).
- Habit archiving and history management.

### **Out of Scope (for initial version):**

- User accounts / login system.
- Social sharing features.
- Advanced habit categories or templates.
- Custom push-notification sounds or tones.
- In-depth gamification systems.

## **Users:**

The main target users include:

- Students (to build study habits, time management, and self-discipline).
- Working Professionals (to manage productivity, exercise, or wellness routines).
- Fitness Enthusiasts (to track workouts, diet, or health-related habits).
- General Individuals (anyone seeking to adopt positive habits or reduce negative ones -drinking more water, limiting screen time)

## **Tools setup:**

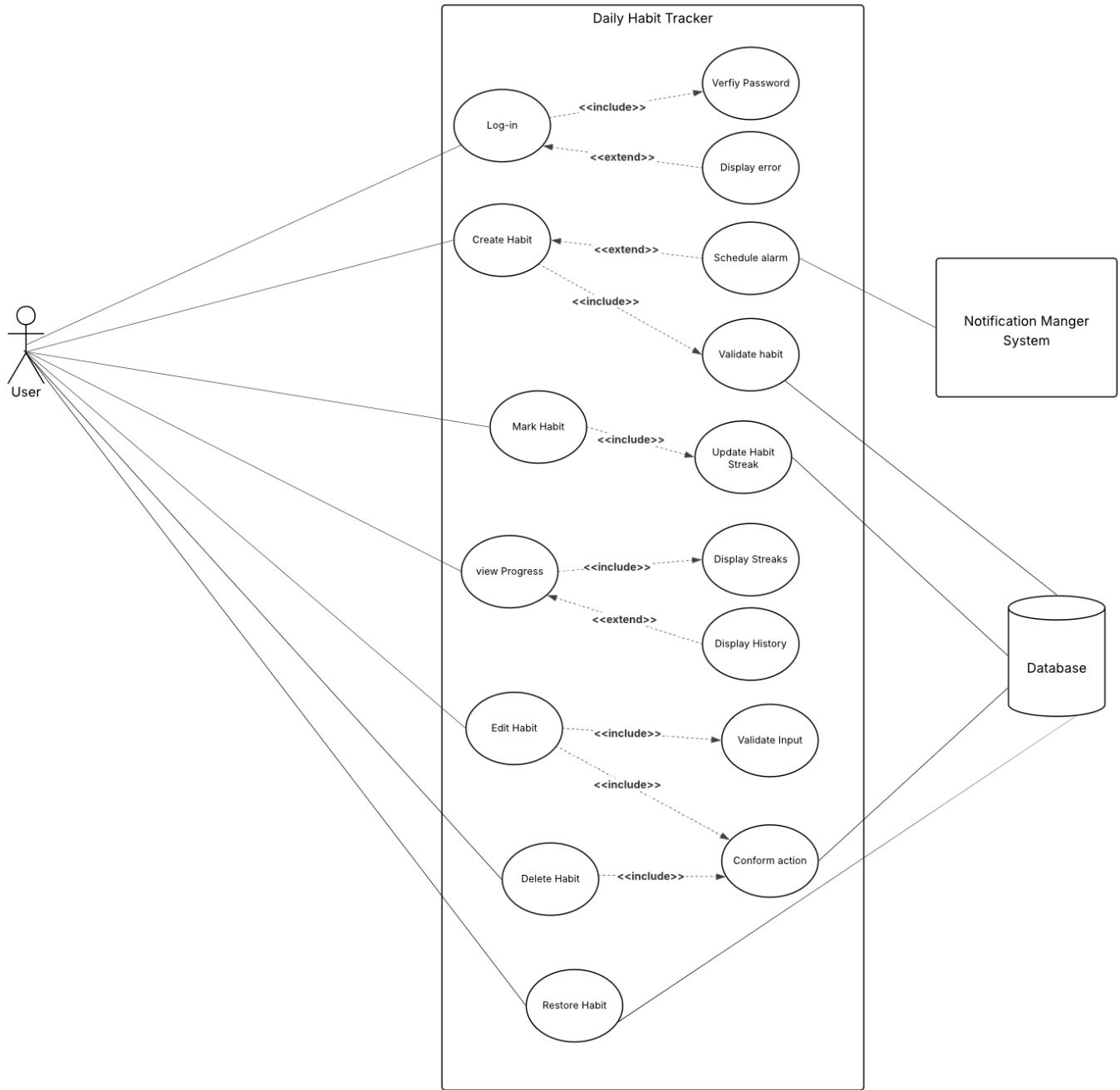
- Design tool: draw.io
- Communication tool: WhatsApp - TeamViewer
- Repository tool: Github
- Agile planning tool: JIRA
- Development tool: Visual Studio Code (VS Code)
- Documents: Microsoft Word

# Functional and non-functional requirements

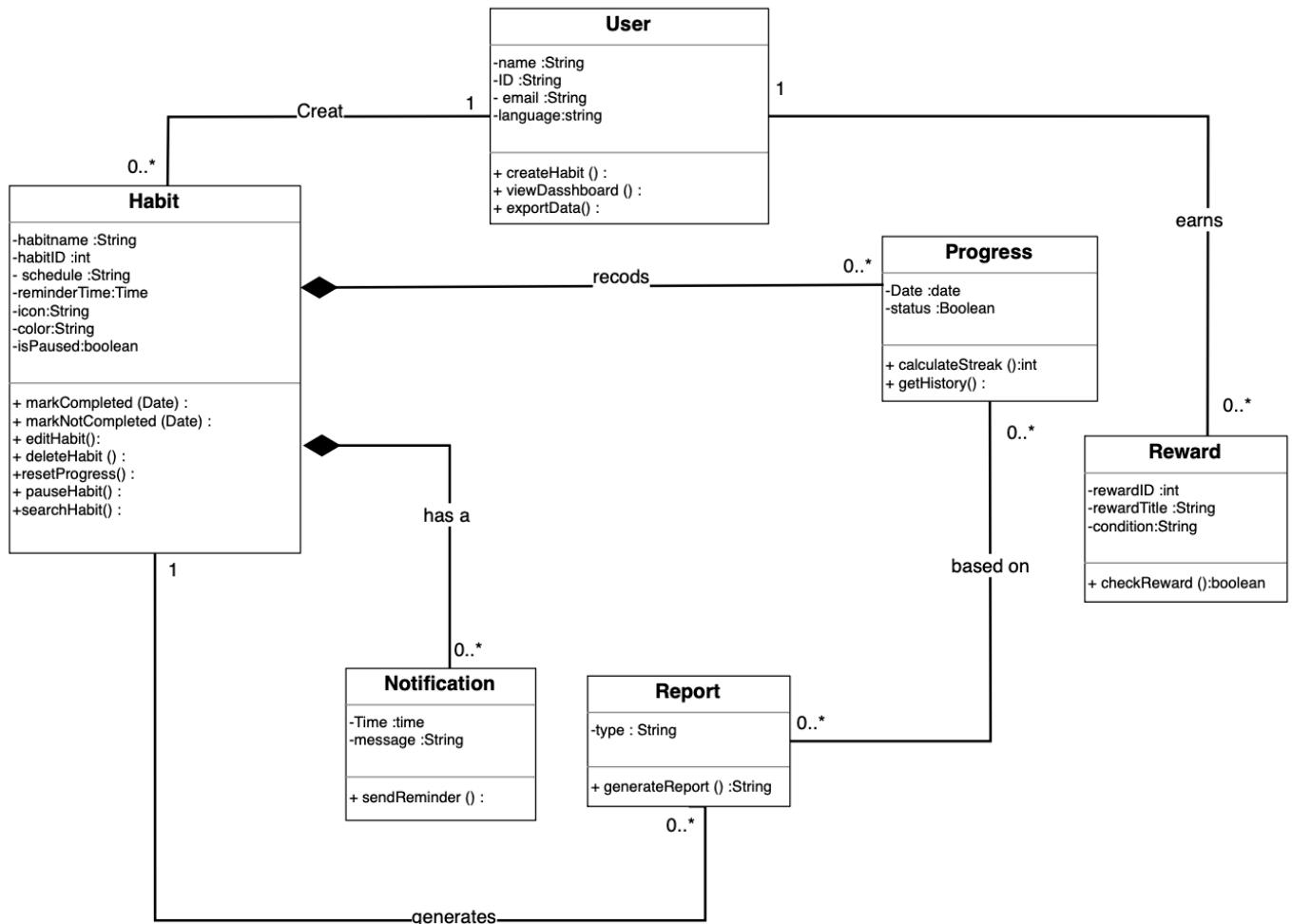
ID	Requirement Definition
FR1	The system shall allow the user to create a new habit by entering the habit name, schedule, and optional reminder time.
FR2	The system shall allow the user to mark a habit as completed or not completed for the current day.
FR4	The system shall send a notification to the user at the reminder time specified for a habit.
FR5	The system shall calculate and display streaks (consecutive days of completion) for each habit.
FR6	The system shall allow the user to reset or clear all progress for a habit if needed.
FR3	The system shall allow the user to view today's list of habits in a dashboard.
NFR1	The system shall provide a user-friendly and intuitive interface.
NFR4	The system shall securely store user habit data locally on the device.
NFR7	The system shall maintain data integrity and prevent accidental loss of records during updates.
FR7	The system shall allow the user to edit or delete an existing habit.
FR8	The system shall display daily, weekly, and monthly progress reports for each habit.
FR9	The system shall allow the user to view past history of completed and missed habits.
FR10	The system shall allow the user to search for a habit by name.
NFR2	The user interface shall be graphical (GUI) and responsive across mobile devices.
NFR3	The system shall load the home screen within 3 seconds under normal conditions.
NFR5	The system shall be easy to use by all users without requiring prior training.
NFR6	The system shall follow accessibility guidelines (e.g., readable font sizes, color contrast).
FR13	The system shall allow the user to manage archived habits (archive, restore, permanently delete).
FR14	The system shall provide extended report filters and visual charts (bar/line charts).
FR11	The system shall allow the user to customize habits with icons and colors.
FR12	The system shall allow the user to export habit data (e.g., CSV/JSON).
NFR8	The system shall support multiple languages, including English and Arabic.
NFR9	The system shall ensure low battery consumption while running background reminders

# Initial Use case and Class Diagrams:

## Use case diagram:



## Class Diagram:



# **Sprint 1**

# Sprint #1 initial Meeting

## Sprint Meeting(s)

**Project Name:** Daily Habit Tracker

**Project Members:** Galla Al-faisal , Jumana AL-Jouhi ,Yara Al-shikhy ,Tamara Hawsawi, Hutoon Alasmarei

**Sprint #1 initial Meeting - [27/9/2025]**

**Sprint Duration:** : 2 Weeks

**Scrum Master:** Galla

**Client:** : dr. Malak Al -Harbi

**Pair Programmers:** Galla, Jumana, Yara, Tamara, Hitoon

**Stories:**

Component Name	Story Sequence Number	Use Cases (e.g., functionalities)
Habit management	01	Create a new habit
Habit Tracking	02	Mark Habit Status

**meeting discussion:**

Discussed the main goals and how to make the app in a way that will help users monitor progress and track their habits

Decided on the tools and technologies Flutter Dart Visual Studio Code

Decided to work on coding together and divided the rest of the work equally among all team members

Planned to start with the user stories ,sequence diagrams, and setting up the needed apps

**Expected outcomes**

Completed user stories and sequence diagrams for the two functions

Initial app layout created

**Scrum's Master comments based on the above questions:**

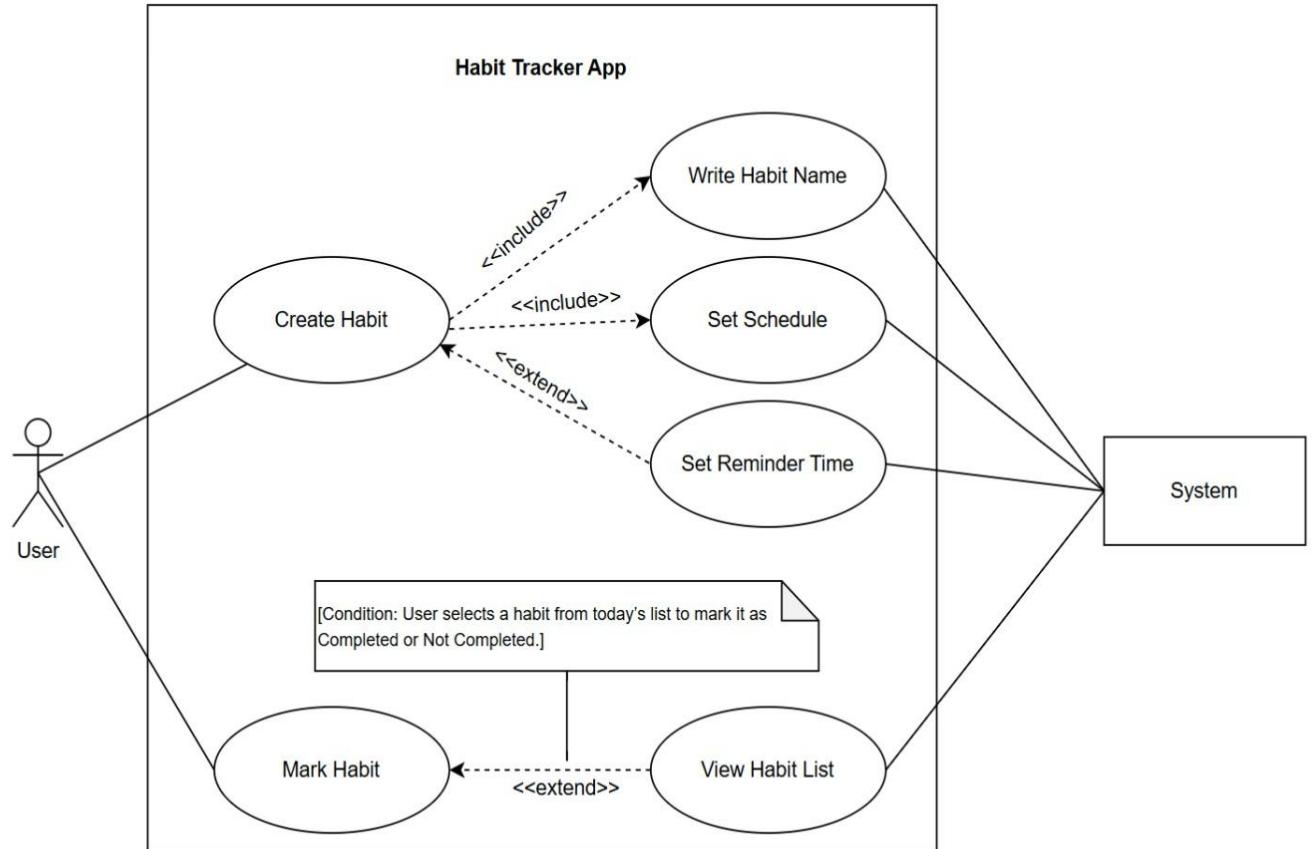
The team showed great task distribution and good organization

Everyone understood their roles and responsibilities

Next meeting will focus on implementing and testing the code

# Requirements Documentation

## Use case diagram:



## User Story Specification:

**Component Name:** Habit Management

**Story Name:** Create a New Habit

**Story Sequence No:** 01

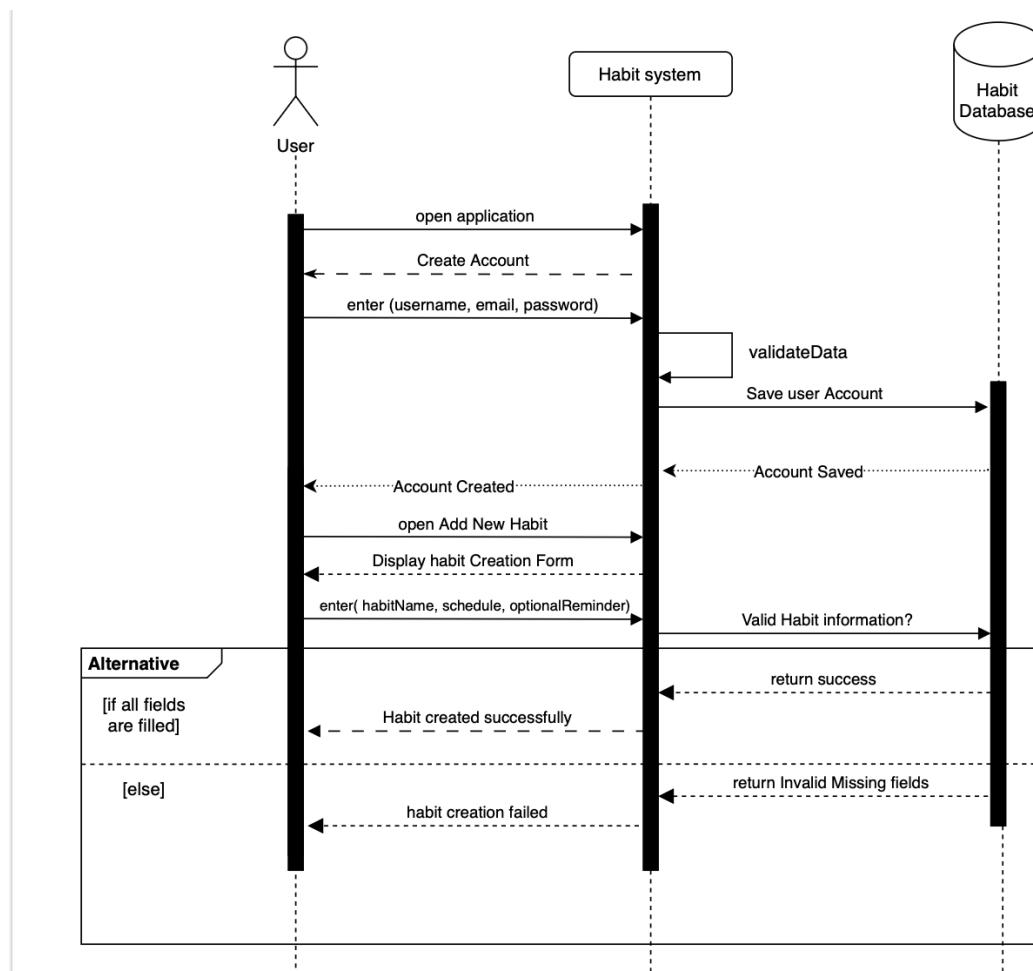
**Story short description:** As a user, I want to be able to create new habits which will allow me to track my progress and determination.

**Story long description:** As a user, I need the ability to create and then add a new habit to my tracker by defining the habit, giving it a unique name, setting a schedule for how often I want to do it (daily, weekly, etc), and have the option to set a specific reminder time to get motivational notifications. After providing the details, the habit should be saved and ready for tracking.

Precondition: User is logged and ready to add habits

Postcondition: new Habit is stored in the database with its details and optional reminder

## Sequence Diagram:



## User Story Specification:

**Component Name:** Habit Tracking

**Story Name:** Mark Habit Status

**Story Sequence No:** 02

**Story short description:** As a user, I want to mark any habit as "completed" or "not completed" to track my daily progress.

**Story long description:** As a user viewing my list of habits for the current day, I need something like a (checkbox or a button) to mark each habit as either done or missed.

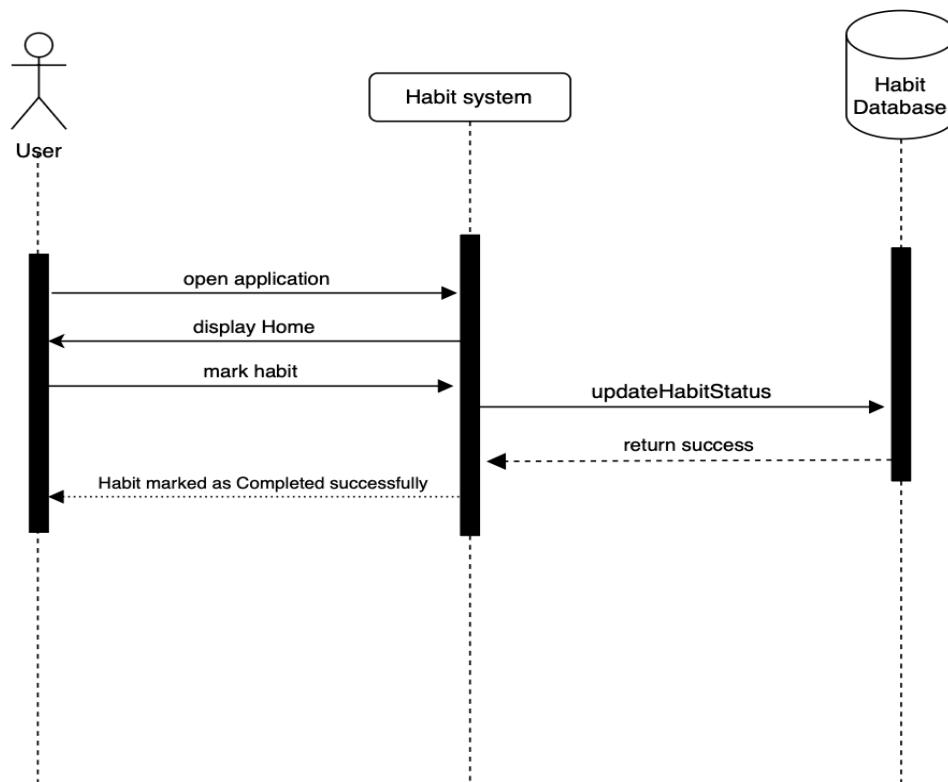
When I mark a habit as completed, the system should immediately reflect this change to me visually, perhaps by changing its color or showing a checkmark.

This allows me to easily record all my actions and helps the system calculate my progress and streaks over time.

**Precondition:** User is logged and has existing habits displayed

**Postcondition:** Habit status is updated in the database and visually reflected in the app

## Sequence Diagram:



# Sprint #1 Stand up Meeting

## Sprint Meeting(s)

**Project Name :** Daily Habit Tracker

**Project Members:** Galla Al-faisal , Jumana AL-Jouhi ,Yara Al-shikhy ,Tamara Hawsawi,  
Hutoon Alasmarei

### Sprint #1 Stand up Meeting - [7/10/2025]

**Sprint Duration:** : 2 Weeks

**Scrum Master:** Galla

**Client:** : dr. Malak Al -Harbi

**Pair Programmers:** Galla, Jumana, Yara, Tamara, Hitoon

**Stories:**

Component Name	Story Sequence Number	Use Cases (e.g., functionalities)
Habit management	01	Create a new habit
Habit Tracking	02	Mark Habit Status

**Follow-up meeting questions:**

From the last meeting the user stories and sequence diagrams for the first and second functions were completed.

After the design phase ,we implemented and tested the code for both functions successfully using visual studio code with flutter framework and dart language

We also had extra time so we created a simple login page

During this sprint we encounterd some difficulties while running and loading the application

**Scrum's Master comments based on the above questions:**

The scrum master(Galla) who is also one of the team members identfied some issues in the coding procces and helped the team fix them ,overall the team made great progress in completing the main features the next sprint will focus on adding new functions and refining the overall idea

# Sprint #1 Test Cases - [9/11/2025]

Test Case Name: [Sprint 1 – Create New Habit]

Test Case Id: [Daily Habit Tracker – Create New Habit]

Test Case No.	Test Case Description	Expected Results	Outcome (Pass / Fail / Other (Comments))
TC001	User leaves all fields blank, and taps “Save”.	System should display a message: “Habit name cannot be empty.”	Pass
TC002	User enters only habit name and taps “Save”.	System should create the habit successfully with default settings.	Pass
TC003	User enters habit name, schedule, and reminder time correctly and taps “Save”.	System should display: “Habit created successfully” and show it in the habit list.	Pass
TC004	Users try to create a habit with a duplicate name.	System should display: “Habit name already exists.”	Pass

**Test Case Name:** [Sprint 1 – Mark Habit Status]

**Test Case Id:** [Daily Habit Tracker – Mark Habit Status]

Test Case No.	Test Case Description	Expected Results	Outcome (Pass/Fail/Other (Comments))
TC001	User selects a habit and marks it as "Completed" for today.	System should display a check mark or success message: "Habit marked as completed."	Pass
TC002	The user can record the habit status as "Not Completed."	User selects "Not Completed" for a habit they did not do.	Pass
TC003	User does not mark a habit as completed	System should keep the habit in the list and show notifications the next day	Pass
TC004	User marks multiple habits as completed on the same day.	System should update progress for all selected habits correctly.	Pass

## Sign-Up Screen

The screenshot shows a Dart code editor on the left and a mobile application interface on the right. The code editor displays the file `sign_up_page.dart` with several methods and logic for handling sign-up operations. The mobile application is titled "Join Habitly Today" and features fields for Email and Password, a checkbox for agreeing to terms, and links for Google and Apple sign-in. It also includes a "Sign up" button.

```
sign_up_page.dart
lib > features > auth > presentation > pages > sign_up_page.dart > ...
16 class _SignUpPageState extends State<SignUpPage> {
23   @override
24   void dispose() {
25     _emailController.dispose();
26     _passwordController.dispose();
27     super.dispose();
28   }
29
30   void _handleSignUp() {
31     if (_formKey.currentState?.validate() ?? false) {
32       if (!_agreeToTerms) {
33         ScaffoldMessenger.of(context).showSnackBar(
34           const SnackBar(
35             content: Text('Please agree to the Terms & Conditions'),
36             behavior: SnackBarBehavior.floating,
37           ), // SnackBar
38         );
39       }
40     }
41
42     context.read<SignUpCubit>().signUpWithEmail(
43       email: _emailController.text.trim(),
44       password: _passwordController.text,
45     );
46   }
47
48   void _handleGoogleSignIn() {
49     context.read<SignUpCubit>().signUpWithGoogle();
50 }
```

Ln 1, Col 1 Spaces:2 UTF-8 CRLF {} Dart Windows (windows-x64)

## Test Case Name: Create New Habit

TC001

The screenshot shows a Dart code editor on the left and a mobile application interface on the right. The code editor displays the file `create_habit_page.dart` with logic for saving habits and checking for duplicates. The mobile application is titled "Create New Habit" and allows users to enter a habit name, choose an icon and color, set repeat frequency (Daily, Weekly, Monthly), and specify the day of the week and time.

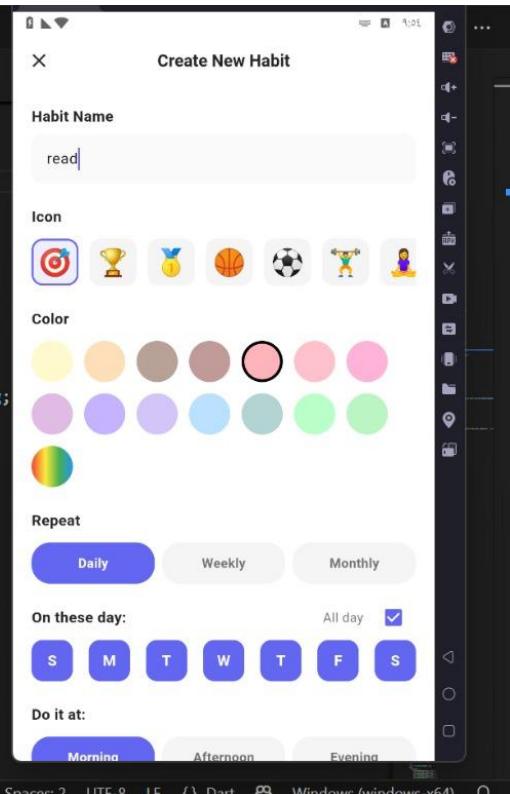
```
create_habit_page.dart
lib > features > habits > presentation > pages > create_habit_page.dart > _CreateHabitPageState > ...
15 class _CreateHabitPageState extends State<CreateHabitPage> {
144   domain.RepeatType _convertRepeatTypeToDomain(data.RepeatType dataRepeatType)
152   }
153 }
154
155 Future<void> _saveHabit() async {
156   if (_nameController.text.isEmpty) {
157     ScaffoldMessenger.of(context).showSnackBar(
158       const SnackBar(content: Text('Please enter a habit name')),
159     );
160   }
161   return;
162 }
163
164 // Check for duplicate name
165 final isDuplicate = await context.read<HabitsCubit>().isHabitNameDuplicate(
166   _nameController.text.trim(),
167 );
168 if (isDuplicate) {
169   ScaffoldMessenger.of(context).showSnackBar(
170     SnackBar(
171       content: Text(
172         'A habit with the name "${_nameController.text.trim()}" already exists',
173       ), // Text
174       backgroundColor: Colors.orange,
175     ), // SnackBar
176   );
177   return;
178 }
```

Ln 604, Col 44 Spaces:2 UTF-8 LF {} Dart Windows (windows-x64)

TC002

```
create_habit_page.dart X

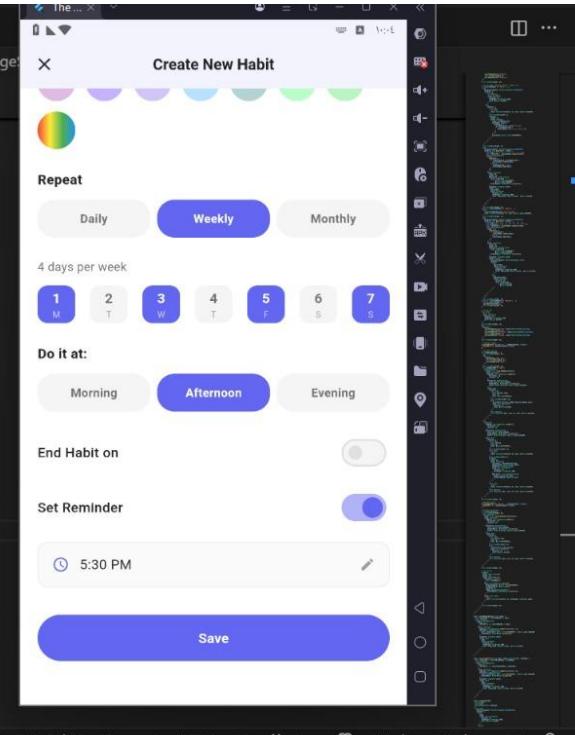
lib > features > habits > presentation > pages > create_habit_page.dart > ...
8   class CreateHabitPage extends StatefulWidget {
11     @override
12       State<CreateHabitPage> createState() => _CreateHabitPageState();
13   }
14
15   class _CreateHabitPageState extends State<CreateHabitPage> {
16     final _nameController = TextEditingController();
17     final _daysAfterController = TextEditingController(text: '365');
18     String _selectedEmoji = '🎯';
19     String _selectedColor = 'FFB3BA';
20     String _selectedRepeat = 'Daily';
21     Set<int> _selectedDays = {0, 1, 2, 3, 4, 5, 6}; // All days selected
22     final Set<int> _selectedMonthDates = {};// For monthly selection
23     domain.HabitTimeOfDay _selectedTimeOfDay = domain.HabitTimeOfDay.morning;
24     bool _endHabitEnabled = false;
25     bool _reminderEnabled = false;
26     bool _allDaySelected = true;
27     String _endHabitMode = 'Days'; // 'Date' or 'Days'
28     DateTime _endDate = DateTime.now().add(const Duration(days: 365));
29     TimeOfDay _reminderTime = const TimeOfDay(hour: 7, minute: 0);
30
31     final List<String> _emojiList = [
32       '🎯',
33       '🏋️',
34       '🏅',
35       '⚽',
36       '🏀',
37       '⚽',
38       '🏆',
39       '🌟'
40     ];
41   }
42 }
```



Ln 14, Col 1 Spaces: 2 UTF-8 LF {} Dart  Windows (windows-x64) 

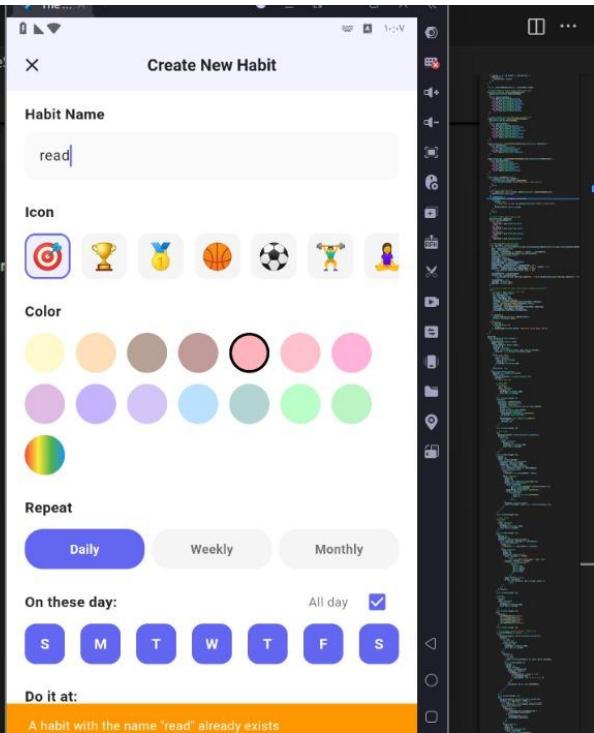
TC003

```
lib > features > habits > presentation > pages >  create_habit_page.dart >  _CreateHabitPage
15   createHabitPageState extends State<CreateHabitPage> {
252   build(BuildContext context) {
698     }),
699
700     // Reminder Time Picker
701     if (_reminderEnabled) ...[
702       const SizedBox(height: 16),
703       GestureDetector(
704         onTap: () => _selectReminderTime(context),
705         child: Container(
706           padding: const EdgeInsets.symmetric(
707             horizontal: 16,
708             vertical: 14,
709           ), // EdgeInsets.symmetric
710           decoration: BoxDecoration(
711             color: <Colors.grey.shade50,
712             borderRadius: BorderRadius.circular(12),
713             border: Border.all(color: <Colors.grey.shade200),
714           ), // BoxDecoration
715           child: Row(
716             children: [
717               const Icon(
718                 Icons.access_time,
719                 size: 20,
720                 color: <Color(0xFF6366F1),
721               ), // Icon
722               const SizedBox(width: 12),
723               Text(
724                 _reminderTime.format(context),
725               ),
726             ],
727           ),
728         ),
729       ),
730     ],
731   ),
732 
```



Ln 716, Col 32 Spaces: 2 UTF-8 LF {} Dart  Windows (windows-x64) 

## TC004

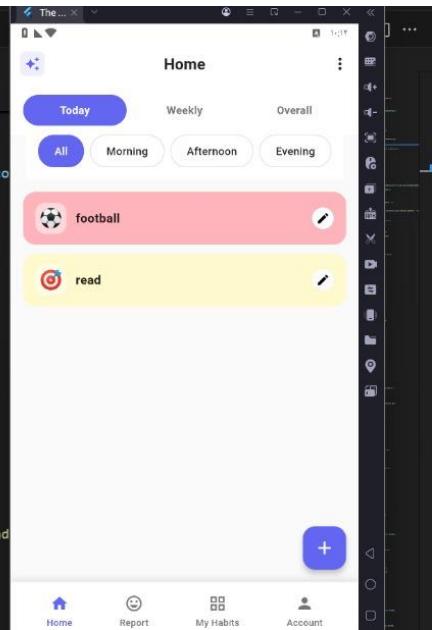


create\_habit\_page.dart

```
lib > features > habits > presentation > pages > create_habit_page.dart > _CreateHabitPageState > _saveHabit
```

```
15 class _CreateHabitPageState extends State<CreateHabitPage> {
155   Future<void> _saveHabit() async {
156     if (_nameController.text.isEmpty) {
157       ScaffoldMessenger.of(context).showSnackBar(
158         SnackBar(
159           content: Text('Please enter a habit name'),
160         ),
161       );
162     }
163     // Check for duplicate name
164     final isDuplicate = await context.read<HabitsCubit>().isHabitNameDuplicate(_nameController.text.trim());
165     if (isDuplicate) {
166       ScaffoldMessenger.of(context).showSnackBar(
167         SnackBar(
168           content: Text('A habit with the name "${_nameController.text.trim()}" already exists'),
169           backgroundColor: Colors.orange,
170         ),
171       );
172     }
173     return;
174   }
175
176   // Convert repeat type string to enum
177   data.RepeatType repeatType;
178   switch (_selectedRepeat) {
179     case 'Daily':
180       repeatType = data.RepeatType.daily;
181       break;
182     case 'Weekly':
183       repeatType = data.RepeatType.weekly;
184     case 'Monthly':
185       repeatType = data.RepeatType.monthly;
186   }
187
188   // Create HabitModel with all data
189   final habitModel = data.HabitModel(
190     id: '${const Uuid().v4()}_${DateTime.now().millisecondsSinceEpoch}_${DateTime.now().microseconds}',
191     name: _nameController.text,
192     emoji: _selectedEmoji,
193     colorHex: _selectedColor == 'RAINBOW' ? '#FFB3BA' : _selectedColor,
194     timeOfDay: _convertTimeOfDayToDate(_selectedTimeOfDay),
195     status: data.HabitStatus.active,
196     isCompleted: false,
197     repeatType: repeatType,
198     selectedDays: _selectedDays,
199     selectedMonthDates: _selectedMonthDates,
200     endHabitEnabled: _endHabitEnabled,
201     endHabitMode: _endHabitEnabled ? _endHabitMode : null,
202     endDate: _endHabitEnabled && _endHabitMode == 'Date' ? _endDate : null,
203     daysAfter: _endHabitEnabled && _endHabitMode == 'Days'
204       ? int.tryParse(_daysAfterController.text) ?? 365
205       : null,
206     reminderEnabled: _reminderEnabled,
207     reminderTime: _reminderEnabled
208       ? '${_reminderTime.hour.toString().padLeft(2, '0')}:${_reminderTime.minute.toString().padLeft(2, '0')}' : null,
209     isSkippedToday: false,
210     createdAt: DateTime.now(),
211     updatedAt: DateTime.now(),
212   ); // data.HabitModel
```

Ln 714, Col 21 Spaces: 2 UTF-8 LF {} Dart Windows (windows-x64)



create\_habit\_page.dart

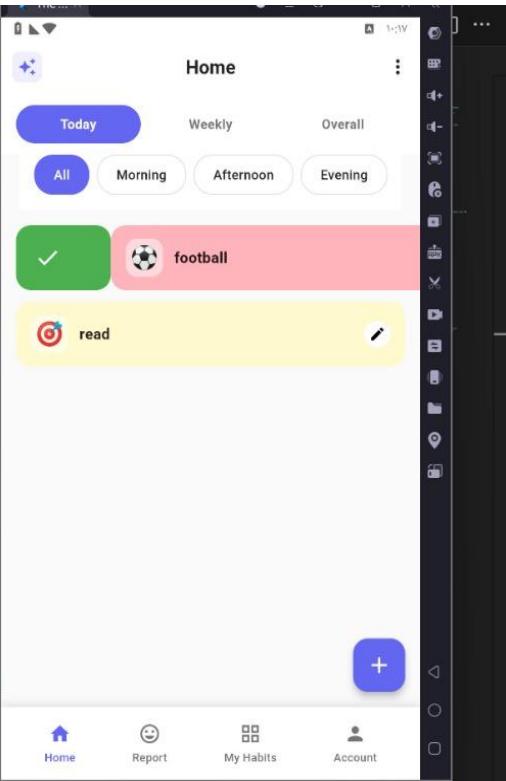
```
lib > features > habits > presentation > pages > create_habit_page.dart > _CreateHabitPageState > _saveHabit
```

```
15 class _CreateHabitPageState extends State<CreateHabitPage> {
155   Future<void> _saveHabit() async {
156     if (_nameController.text.isEmpty) {
157       ScaffoldMessenger.of(context).showSnackBar(
158         SnackBar(
159           content: Text('Please enter a habit name'),
160         ),
161       );
162     }
163     // Create HabitModel with all data
164     final habitModel = data.HabitModel(
165       id: '${const Uuid().v4()}_${DateTime.now().millisecondsSinceEpoch}_${DateTime.now().microseconds}',
166       name: _nameController.text,
167       emoji: _selectedEmoji,
168       colorHex: _selectedColor == 'RAINBOW' ? '#FFB3BA' : _selectedColor,
169       timeOfDay: _convertTimeOfDayToDate(_selectedTimeOfDay),
170       status: data.HabitStatus.active,
171       isCompleted: false,
172       repeatType: repeatType,
173       selectedDays: _selectedDays,
174       selectedMonthDates: _selectedMonthDates,
175       endHabitEnabled: _endHabitEnabled,
176       endHabitMode: _endHabitEnabled ? _endHabitMode : null,
177       endDate: _endHabitEnabled && _endHabitMode == 'Date' ? _endDate : null,
178       daysAfter: _endHabitEnabled && _endHabitMode == 'Days'
179         ? int.tryParse(_daysAfterController.text) ?? 365
180         : null,
181       reminderEnabled: _reminderEnabled,
182       reminderTime: _reminderEnabled
183         ? '${_reminderTime.hour.toString().padLeft(2, '0')}:${_reminderTime.minute.toString().padLeft(2, '0')}' : null,
184       isSkippedToday: false,
185       createdAt: DateTime.now(),
186       updatedAt: DateTime.now(),
187     ); // data.HabitModel
```

Ln 173, Col 42 Spaces: 2 UTF-8 LF {} Dart Windows (windows-x64)

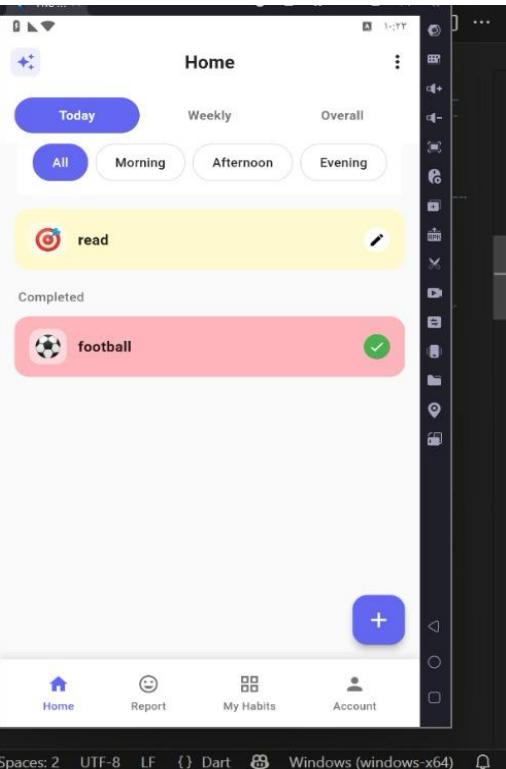
## **Test Case Name:** Mark Habit Status

**TC001**



The screenshot shows the app's main screen titled "Home". At the top, there are three tabs: "Today" (highlighted in blue), "Weekly", and "Overall". Below the tabs are four buttons: "All" (blue), "Morning", "Afternoon", and "Evening". The main area displays two habit cards. The first card, for "football", has a green checkmark icon and a soccer ball icon. The second card, for "read", has a red target icon and a book icon. At the bottom of the screen are five navigation icons: Home, Report, My Habits, and Account.

```
lib > features > habits > presentation > widgets > swipeable_habit_card.dart > _SwipeableHabitCardState.dart
22 class _SwipeableHabitCardState extends State<SwipeableHabitCard> {
36   Widget build(BuildContext context) {
57     ),
58     child: Container(
59       margin: const EdgeInsets.only(bottom: 12),
60       child: Stack(
61         children: [
62           // Background actions
63           if (!widget.habit.isCompleted) ...[
64             // Complete action (swipe right)
65             if (_dragExtent > 0)
66               Positioned(
67                 left: 0,
68                 top: 0,
69                 bottom: 0,
70                 child: Container(
71                   width: _dragExtent,
72                   decoration: BoxDecoration(
73                     color: const Color(0xFF4CAF50),
74                     borderRadius: BorderRadius.circular(16),
75                   ), // BoxDecoration
76                   alignment: Alignment.centerLeft,
77                   padding: const EdgeInsets.only(left: 20),
78                   child: _dragExtent > 40
79                     ? const Icon(Icons.check, color: Colors.white, size: null,
80                     ),
81                   ), // Container
82                 ), // Positioned
83               ),
84             // Skip action (swipe left)
85             if (_dragExtent < 0)
86               Positioned(
87                 right: 0,
```



The screenshot shows the app's main screen titled "Home". At the top, there are three tabs: "Today" (highlighted in blue), "Weekly", and "Overall". Below the tabs are four buttons: "All" (blue), "Morning", "Afternoon", and "Evening". The main area displays two habit cards. The first card, for "read", has a red target icon and a book icon. The second card, for "football", has a green checkmark icon and a soccer ball icon. A "Completed" status message is visible above the second card. At the bottom of the screen are five navigation icons: Home, Report, My Habits, and Account.

```
lib > features > habits > presentation > widgets > swipeable_habit_card.dart > _SwipeableHabitCardState.dart
22 class _SwipeableHabitCardState extends State<SwipeableHabitCard> {
36   Widget build(BuildContext context) {
61     ),
62     child: Container(
63       if (!widget.habit.isCompleted) ...[
64         // Complete action (swipe right)
65         if (_dragExtent > 0)
66           Positioned(
67             left: 0,
68             top: 0,
69             bottom: 0,
70             child: Container(
71               width: _dragExtent,
72               decoration: BoxDecoration(
73                 color: const Color(0xFF4CAF50),
74                 borderRadius: BorderRadius.circular(16),
75               ), // BoxDecoration
76               alignment: Alignment.centerLeft,
77               padding: const EdgeInsets.only(left: 20),
78               child: _dragExtent > 40
79                 ? const Icon(Icons.check, color: Colors.white, size: null,
80                 ),
81               ), // Container
82             ), // Positioned
83           ),
84         // Skip action (swipe left)
85         if (_dragExtent < 0)
86           Positioned(
```

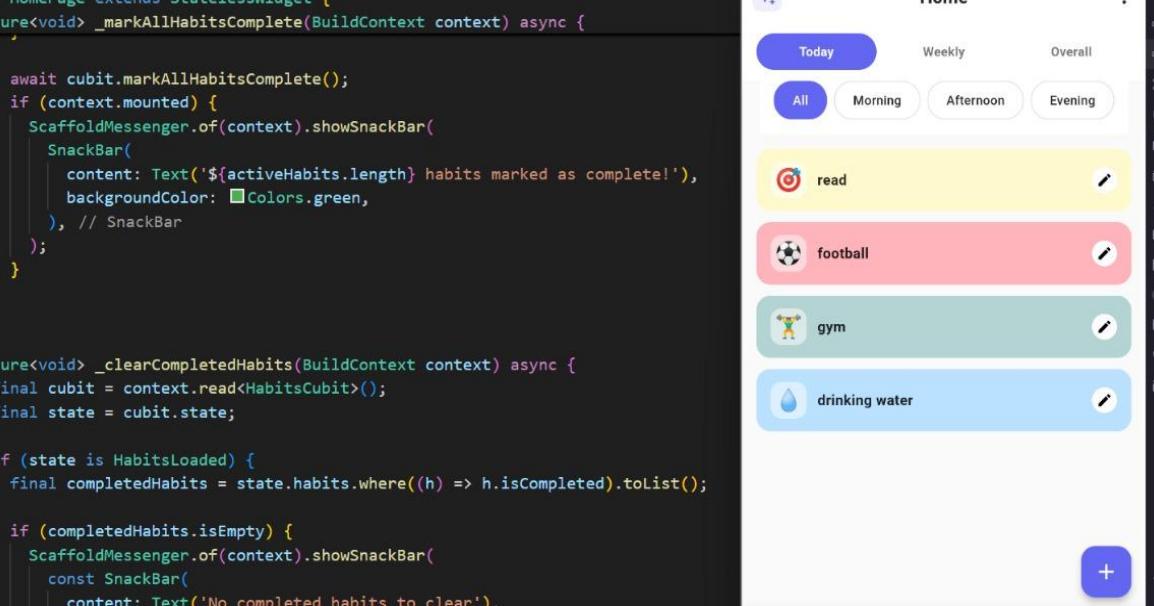
## TC002

The screenshot shows the Dart IDE with two files open: `create_habit_page.dart` and `swipeable_habit_card.dart`. The `swipeable_habit_card.dart` file contains the implementation of a swipeable habit card. The code includes logic for dragging, styling with BoxDecoration, and a skip button. The mobile application interface on the right shows a 'Home' screen with tabs for Today, Weekly, and Overall. It displays a completed habit card for 'football' with a checkmark and a skip button. Below it, there's a 'Completed' section and a 'Skipped Today' section with a habit for 'read'.

```
lib > features > habits > presentation > widgets > swipeable_habit_card.dart > _SwipeableHabitCardState.dart
22 class _SwipeableHabitCardState extends State<SwipeableHabitCard> {
36 Widget build(BuildContext context) {
-->
84     // Skip action (swipe left)
85     if (_dragExtent < 0)
86         Positioned(
87             right: 0,
88             top: 0,
89             bottom: 0,
90             child: Container(
91                 width: -_dragExtent,
92                 decoration: BoxDecoration(
93                     color: const Color(0xFFE444),
94                     borderRadius: BorderRadius.circular(16),
95                 ), // BoxDecoration
96                 alignment: Alignment.centerRight,
97                 padding: const EdgeInsets.only(right: 20),
98                 child: _dragExtent < -40
99                     ? ClipRect(
100                         child: Row(
101                             mainAxisAlignment: MainAxisAlignment.end,
102                             mainAxisSize: MainAxisSize.min,
103                             children: [
104                                 Text(
105                                     'Skip',
106                                     style: TextStyle(
107                                         color: Colors.white,
108                                         fontSize: 12,
109                                         fontWeight: FontWeight.w600,
110                                     ), // TextStyle
111                                 ), // Text
112                         ],
113                     ),
114                 ),
115             ),
116         );
117     ),
118     ),
119     ),
120     ),
121     ),
122     ),
123     ),
124     ),
125     ),
126     ),
127     ),
128     ),
129     ),
130     ),
131     ),
132     ),
133     ),
134     ),
135     ),
136     ),
137     ),
138     ),
139     ),
140     ),
141     ),
142     ),
143     ),
144     ),
145     ),
146     ),
147     ),
148     ),
149     ),
150     ),
151     ),
152     ),
153     ),
154     ),
155     ),
156     ),
157     ),
158     ),
159     ),
160     ),
161     ),
162     ),
163     ),
164     ),
165     ),
166     ),
167     ),
168     ),
169     ),
170     ),
171     ),
172     ),
173     ),
174     ),
175     ),
176     ),
177     ),
178     ),
179     ),
180     ),
181     ),
182     ),
183     ),
184     ),
185     ),
186     ),
187     ),
188     ),
189     ),
190     ),
191     ),
192     ),
193     ),
194     ),
195     ),
196     ),
197     ),
198     ),
199     ),
200     ),
201     ),
202     ),
203     ),
204     ),
205     ),
206     ),
207     ),
208     ),
209     ),
210     ),
211     ),
212     ),
213     ),
214     ),
215     ),
216     ),
217     ),
218     ),
219     ),
220     ),
221     ),
222     ),
223     ),
224     ),
225     ),
226     ),
227     ),
228     ),
229     ),
230     ),
231     ),
232     ),
233     ),
234     ),
235     ),
236     ),
237     ),
238     ),
239     ),
240     ),
241     ),
242     ),
243     ),
244     ),
245     ),
246     ),
247     ),
248     ),
249     ),
250     ),
251     ),
252     ),
253     ),
254     ),
255     ),
256     ),
257     ),
258     ),
259     ),
260     ),
261     ),
262     ),
263     ),
264     ),
265     ),
266     ),
267     ),
268     ),
269     ),
270     ),
271     ),
272     ),
273     ),
274     ),
275     ),
276     ),
277     ),
278     ),
279     ),
280     ),
281     ),
282     ),
283     ),
284     ),
285     ),
286     ),
287     ),
288     ),
289     ),
290     ),
291     ),
292     ),
293     ),
294     ),
295     ),
296     ),
297     ),
298     ),
299     ),
299     ),
300     ),
301     ),
302     ),
303     ),
304     ),
305     ),
306     ),
307     ),
308     ),
309     ),
310     ),
311     ),
312     ),
313     ),
314     ),
315     ),
316     ),
317     ),
318     ),
319     ),
320     ),
321     ),
322     ),
323     ),
324     ),
325     ),
326     ),
327     ),
328     ),
329     ),
330     ),
331     ),
332     ),
333     ),
334     ),
335     ),
336     ),
337     ),
338     ),
339     ),
339     ),
340     ),
341     ),
342     ),
343     ),
344     ),
345     ),
346     ),
347     ),
348     ),
349     ),
349     ),
350     ),
351     ),
352     ),
353     ),
354     ),
355     ),
356     ),
357     ),
358     ),
359     ),
359     ),
360     ),
361     ),
362     ),
363     ),
364     ),
365     ),
366     ),
367     ),
368     ),
369     ),
369     ),
370     ),
371     ),
372     ),
373     ),
374     ),
375     ),
376     ),
377     ),
378     ),
379     ),
379     ),
380     ),
381     ),
382     ),
383     ),
384     ),
385     ),
386     ),
387     ),
388     ),
389     ),
389     ),
390     ),
391     ),
392     ),
393     ),
394     ),
395     ),
396     ),
397     ),
398     ),
399     ),
399     ),
400     ),
401     ),
402     ),
403     ),
404     ),
405     ),
406     ),
407     ),
408     ),
409     ),
409     ),
410     ),
411     ),
412     ),
413     ),
414     ),
415     ),
416     ),
417     ),
418     ),
419     ),
419     ),
420     ),
421     ),
422     ),
423     ),
424     ),
425     ),
426     ),
427     ),
428     ),
429     ),
429     ),
430     ),
431     ),
432     ),
433     ),
434     ),
435     ),
436     ),
437     ),
438     ),
439     ),
439     ),
440     ),
441     ),
442     ),
443     ),
444     ),
445     ),
446     ),
447     ),
448     ),
449     ),
449     ),
450     ),
451     ),
452     ),
453     ),
454     ),
455     ),
456     ),
457     ),
458     ),
459     ),
459     ),
460     ),
461     ),
462     ),
463     ),
464     ),
465     ),
466     ),
467     ),
468     ),
469     ),
469     ),
470     ),
471     ),
472     ),
473     ),
474     ),
475     ),
476     ),
477     ),
478     ),
479     ),
479     ),
480     ),
481     ),
482     ),
483     ),
484     ),
485     ),
486     ),
487     ),
488     ),
489     ),
489     ),
490     ),
491     ),
492     ),
493     ),
494     ),
495     ),
496     ),
497     ),
498     ),
499     ),
499     ),
500     ),
501     ),
502     ),
503     ),
504     ),
505     ),
506     ),
507     ),
508     ),
509     ),
509     ),
510     ),
511     ),
512     ),
513     ),
514     ),
515     ),
516     ),
517     ),
518     ),
519     ),
519     ),
520     ),
521     ),
522     ),
523     ),
524     ),
525     ),
526     ),
527     ),
528     ),
529     ),
529     ),
530     ),
531     ),
532     ),
533     ),
534     ),
535     ),
536     ),
537     ),
538     ),
539     ),
539     ),
540     ),
541     ),
542     ),
543     ),
544     ),
545     ),
546     ),
547     ),
548     ),
549     ),
549     ),
550     ),
551     ),
552     ),
553     ),
554     ),
555     ),
556     ),
557     ),
558     ),
559     ),
559     ),
560     ),
561     ),
562     ),
563     ),
564     ),
565     ),
566     ),
567     ),
568     ),
569     ),
569     ),
570     ),
571     ),
572     ),
573     ),
574     ),
575     ),
576     ),
577     ),
578     ),
579     ),
579     ),
580     ),
581     ),
582     ),
583     ),
584     ),
585     ),
586     ),
587     ),
588     ),
589     ),
589     ),
590     ),
591     ),
592     ),
593     ),
594     ),
595     ),
596     ),
597     ),
598     ),
599     ),
599     ),
600     ),
601     ),
602     ),
603     ),
604     ),
605     ),
606     ),
607     ),
608     ),
609     ),
609     ),
610     ),
611     ),
612     ),
613     ),
614     ),
615     ),
616     ),
617     ),
618     ),
619     ),
619     ),
620     ),
621     ),
622     ),
623     ),
624     ),
625     ),
626     ),
627     ),
628     ),
629     ),
629     ),
630     ),
631     ),
632     ),
633     ),
634     ),
635     ),
636     ),
637     ),
638     ),
639     ),
639     ),
640     ),
641     ),
642     ),
643     ),
644     ),
645     ),
646     ),
647     ),
648     ),
649     ),
649     ),
650     ),
651     ),
652     ),
653     ),
654     ),
655     ),
656     ),
657     ),
658     ),
659     ),
659     ),
660     ),
661     ),
662     ),
663     ),
664     ),
665     ),
666     ),
667     ),
668     ),
669     ),
669     ),
670     ),
671     ),
672     ),
673     ),
674     ),
675     ),
676     ),
677     ),
678     ),
679     ),
679     ),
680     ),
681     ),
682     ),
683     ),
684     ),
685     ),
686     ),
687     ),
688     ),
689     ),
689     ),
690     ),
691     ),
692     ),
693     ),
694     ),
695     ),
696     ),
697     ),
698     ),
699     ),
699     ),
700     ),
701     ),
702     ),
703     ),
704     ),
705     ),
706     ),
707     ),
708     ),
709     ),
709     ),
710     ),
711     ),
712     ),
713     ),
714     ),
715     ),
716     ),
717     ),
718     ),
719     ),
719     ),
720     ),
721     ),
722     ),
723     ),
724     ),
725     ),
726     ),
727     ),
728     ),
729     ),
729     ),
730     ),
731     ),
732     ),
733     ),
734     ),
735     ),
736     ),
737     ),
738     ),
739     ),
739     ),
740     ),
741     ),
742     ),
743     ),
744     ),
745     ),
746     ),
747     ),
748     ),
749     ),
749     ),
750     ),
751     ),
752     ),
753     ),
754     ),
755     ),
756     ),
757     ),
758     ),
759     ),
759     ),
760     ),
761     ),
762     ),
763     ),
764     ),
765     ),
766     ),
767     ),
768     ),
769     ),
769     ),
770     ),
771     ),
772     ),
773     ),
774     ),
775     ),
776     ),
777     ),
778     ),
779     ),
779     ),
780     ),
781     ),
782     ),
783     ),
784     ),
785     ),
786     ),
787     ),
788     ),
789     ),
789     ),
790     ),
791     ),
792     ),
793     ),
794     ),
795     ),
796     ),
797     ),
798     ),
799     ),
799     ),
800     ),
801     ),
802     ),
803     ),
804     ),
805     ),
806     ),
807     ),
808     ),
809     ),
809     ),
810     ),
811     ),
812     ),
813     ),
814     ),
815     ),
816     ),
817     ),
818     ),
819     ),
819     ),
820     ),
821     ),
822     ),
823     ),
824     ),
825     ),
826     ),
827     ),
828     ),
829     ),
829     ),
830     ),
831     ),
832     ),
833     ),
834     ),
835     ),
836     ),
837     ),
838     ),
839     ),
839     ),
840     ),
841     ),
842     ),
843     ),
844     ),
845     ),
846     ),
847     ),
848     ),
849     ),
849     ),
850     ),
851     ),
852     ),
853     ),
854     ),
855     ),
856     ),
857     ),
858     ),
859     ),
859     ),
860     ),
861     ),
862     ),
863     ),
864     ),
865     ),
866     ),
867     ),
868     ),
869     ),
869     ),
870     ),
871     ),
872     ),
873     ),
874     ),
875     ),
876     ),
877     ),
878     ),
879     ),
879     ),
880     ),
881     ),
882     ),
883     ),
884     ),
885     ),
886     ),
887     ),
888     ),
889     ),
889     ),
890     ),
891     ),
892     ),
893     ),
894     ),
895     ),
896     ),
897     ),
898     ),
899     ),
899     ),
900     ),
901     ),
902     ),
903     ),
904     ),
905     ),
906     ),
907     ),
908     ),
909     ),
909     ),
910     ),
911     ),
912     ),
913     ),
914     ),
915     ),
916     ),
917     ),
918     ),
919     ),
919     ),
920     ),
921     ),
922     ),
923     ),
924     ),
925     ),
926     ),
927     ),
928     ),
929     ),
929     ),
930     ),
931     ),
932     ),
933     ),
934     ),
935     ),
936     ),
937     ),
938     ),
939     ),
939     ),
940     ),
941     ),
942     ),
943     ),
944     ),
945     ),
946     ),
947     ),
948     ),
949     ),
949     ),
950     ),
951     ),
952     ),
953     ),
954     ),
955     ),
956     ),
957     ),
958     ),
959     ),
959     ),
960     ),
961     ),
962     ),
963     ),
964     ),
965     ),
966     ),
967     ),
968     ),
969     ),
969     ),
970     ),
971     ),
972     ),
973     ),
974     ),
975     ),
976     ),
977     ),
978     ),
979     ),
979     ),
980     ),
981     ),
982     ),
983     ),
984     ),
985     ),
986     ),
987     ),
988     ),
989     ),
989     ),
990     ),
991     ),
992     ),
993     ),
994     ),
995     ),
996     ),
997     ),
998     ),
999     ),
999     ),
1000     ),
1001     ),
1002     ),
1003     ),
1004     ),
1005     ),
1006     ),
1007     ),
1008     ),
1009     ),
1009     ),
1010     ),
1011     ),
1012     ),
1013     ),
1014     ),
1015     ),
1016     ),
1017     ),
1018     ),
1019     ),
1019     ),
1020     ),
1021     ),
1022     ),
1023     ),
1024     ),
1025     ),
1026     ),
1027     ),
1028     ),
1029     ),
1029     ),
1030     ),
1031     ),
1032     ),
1033     ),
1034     ),
1035     ),
1036     ),
1037     ),
1038     ),
1039     ),
1039     ),
1040     ),
1041     ),
1042     ),
1043     ),
1044     ),
1045     ),
1046     ),
1047     ),
1048     ),
1049     ),
1049     ),
1050     ),
1051     ),
1052     ),
1053     ),
1054     ),
1055     ),
1056     ),
1057     ),
1058     ),
1059     ),
1059     ),
1060     ),
1061     ),
1062     ),
1063     ),
1064     ),
1065     ),
1066     ),
1067     ),
1068     ),
1069     ),
1069     ),
1070     ),
1071     ),
1072     ),
1073     ),
1074     ),
1075     ),
1076     ),
1077     ),
1078     ),
1079     ),
1079     ),
1080     ),
1081     ),
1082     ),
1083     ),
1084     ),
1085     ),
1086     ),
1087     ),
1088     ),
1088     ),
1089     ),
1090     ),
1091     ),
1092     ),
1093     ),
1094     ),
1095     ),
1096     ),
1097     ),
1098     ),
1098     ),
1099     ),
1100     ),
1101     ),
1102     ),
1103     ),
1104     ),
1105     ),
1106     ),
1107     ),
1108     ),
1109     ),
1109     ),
1110     ),
1111     ),
1112     ),
1113     ),
1114     ),
1115     ),
1116     ),
1117     ),
1118     ),
1119     ),
1119     ),
1120     ),
1121     ),
1122     ),
1123     ),
1124     ),
1125     ),
1126     ),
1127     ),
1128     ),
1129     ),
1129     ),
1130     ),
1131     ),
1132     ),
1133     ),
1134     ),
1135     ),
1136     ),
1137     ),
1138     ),
1139     ),
1139     ),
1140     ),
1141     ),
1142     ),
1143     ),
1144     ),
1145     ),
1146     ),
1147     ),
1148     ),
1148     ),
1149     ),
1150     ),
1151     ),
1152     ),
1153     ),
1154     ),
1155     ),
1156     ),
1157     ),
1158     ),
1158     ),
1159     ),
1160     ),
1161     ),
1162     ),
1163     ),
1164     ),
1165     ),
1166     ),
1167     ),
1168     ),
1169     ),
1169     ),
1170     ),
1171     ),
1172     ),
1173     ),
1174     ),
1175     ),
1176     ),
1177     ),
1178     ),
1178     ),
1179     ),
1180     ),
1181     ),
1182     ),
1183     ),
1184     ),
1185     ),
1186     ),
1186     ),
1187     ),
1188     ),
1189     ),
1190     ),
1191     ),
1192     ),
1193     ),
1194     ),
1195     ),
1196     ),
1197     ),
1198     ),
1198     ),
1199     ),
1200     ),
1201     ),
1202     ),
1203     ),
1204     ),
1205     ),
1206     ),
1207     ),
1208     ),
1209     ),
1209     ),
1210     ),
1211     ),
1212     ),
1213     ),
1214     ),
1215     ),
1216     ),
1217     ),
1218     ),
1219     ),
1219     ),
1220     ),
1221     ),
1222     ),
1223     ),
1224     ),
1225     ),
1226     ),
1227     ),
1228     ),
1229     ),
1229     ),
1230     ),
1231     ),
1232     ),
1233     ),
1234     ),
1235     ),
1236     ),
1237     ),
1238     ),
1239     ),
1239     ),
1240     ),
1241     ),
1242     ),
1243     ),
1244     ),
1245     ),
1246     ),
1247     ),
1248     ),
1248     ),
1249     ),
1250     ),
1251     ),
1252     ),
1253     ),
1254     ),
1255     ),
1256     ),
1257     ),
1258     ),
1258     ),
1259     ),
1260     ),
1261     ),
1262     ),
1263     ),
1264     ),
1265     ),
1266     ),
1267     ),
1268     ),
1269     ),
1269     ),
1270     ),
1271     ),
1272     ),
1273     ),
1274     ),
1275     ),
1276     ),
1277     ),
1278     ),
1278     ),
1279     ),
1280     ),
1281     ),
1282     ),
1283     ),
1284     ),
1285     ),
1286     ),
1286     ),
1287     ),
1288     ),
1289     ),
1290     ),
1291     ),
1292     ),
1293     ),
1294     ),
1295     ),
1296     ),
1297     ),
1297     ),
1298     ),
1299     ),
1300     ),
1301     ),
1302     ),
1303     ),
1304     ),
1305     ),
1306     ),
1307     ),
1308     ),
1309     ),
1309     ),
1310     ),
1311     ),
1312     ),
1313     ),
1314     ),
1315     ),
1316     ),
1317     ),
1318     ),
1319     ),
1319     ),
1320     ),
1321     ),
1322     ),
1323     ),
1324     ),
1325     ),
1326     ),
1327     ),
1328     ),
1328     ),
1329     ),
1330     ),
1331     ),
1332     ),
1333     ),
1334     ),
1335     ),
1336     ),
1337     ),
1337     ),
1338     ),
1339     ),
1340     ),
1341     ),
1342     ),
1343     ),
1344     ),
1345     ),
1346     ),
1347     ),
1347     ),
1348     ),
1349     ),
1350     ),
1351     ),
1352     ),
1353     ),
1354     ),
1355     ),
1356     ),
1357     ),
1358     ),
1358     ),
1359     ),
1360     ),
1361     ),
1362     ),
1363     ),
1364     ),
1365     ),
1366     ),
1367     ),
1368     ),
1368     ),
1369     ),
1370     ),
1371     ),
1372     ),
1373     ),
1374     ),
1375     ),
1376     ),
1377     ),
1377     ),
1378     ),
1379     ),
1380     ),
1381     ),
1382     ),
1383     ),
1384     ),
1385     ),
1386     ),
1386     ),
1387     ),
1388     ),
1389     ),
1390     ),
1391     ),
1392     ),
1393     ),
1394     ),
1395     ),
1396     ),
1397     ),
1397     ),
1398     ),
1399     ),
1400     ),
1401     ),
1402     ),
1403     ),
1404     ),
1405     ),
1406     ),
1407     ),
1408     ),
1409     ),
1409     ),
1410     ),
1411     ),
1412     ),
1413     ),
1414     ),
1415     ),
1416     ),
1417     ),
1418     ),
1419     ),
1419     ),
1420     ),
1421     ),
1422     ),
1423     ),
1424     ),
1425     ),
1426     ),
1427     ),
1427     ),
1428     ),
1429     ),
1430     ),
1431     ),
1432     ),
1433     ),
1434     ),
1435     ),
1436     ),
1437     ),
1437     ),
1438     ),
1439     ),
1440     ),
1441     ),
1442     ),
1443     ),
1444     ),
1445     ),
1446     ),
1447     ),
1447     ),
1448     ),
1449     ),
1450     ),
1451     ),
1452     ),
1453     ),
1454     ),
1455     ),
1456     ),
1457     ),
1458     ),
1458     ),
1459     ),
1460     ),
1461     ),
1462     ),
1463     ),
1464     ),
1465     ),
1466     ),
1467     ),
1468     ),
1468     ),
1469     ),
1470     ),
1471     ),
1472     ),
1473     ),
1474     ),
1475     ),
1476     ),
1477     ),
1477     ),
1478     ),
1479     ),
1480     ),
1481     ),
1482     ),
1483     ),
1484     ),
1485     ),
1486     ),
1486     ),
1487     ),
1488     ),
1489     ),
1490     ),
1491     ),
1492     ),
1493     ),
1494     ),
1495     ),
1496     ),
1497     ),
1497     ),
1498     ),
1499     ),
1500     ),
1501     ),
1502     ),
1503     ),
1504     ),
1505     ),
1506     ),
1507     ),
1508     ),
1509     ),
1509     ),
1510     ),
1511     ),
1512     ),
1513     ),
1514     ),
1515     ),
1516     ),
1517     ),
1518     ),
1519     ),
1519     ),
1520     ),
1521     ),
1522     ),
1523     ),
1524     ),
1525     ),
1526     ),
1527     ),
1527     ),
1528     ),
1529     ),
1530     ),
1531     ),
1532     ),
1533     ),
1534     ),
1535     ),
1536     ),
1537     ),
1537     ),
1538     ),
1539     ),
1540     ),
1541     ),
1542     ),
1543     ),
1544     ),
1545     ),
1546     ),
1547     ),
1547     ),
1548     ),
1549     ),
1550     ),
1551     ),
1552     ),
1553     ),
1554     ),
1555     ),
1556     ),
1557     ),
1558     ),
1558     ),
1559     ),
1560     ),
1561     ),
1562     ),
1563     ),
1564     ),
1565     ),
1566     ),
1567     ),
1568     ),
1568     ),
1569     ),
1570     ),
1571     ),
1572     ),
1573     ),
1574     ),
1575     ),
1576     ),
1577     ),
1577     ),
1578     ),
1579     ),
1580     ),
1581     ),
1582     ),
1583     ),
1584     ),
1585     ),
1586     ),
1586     ),
1587     ),
1588     ),
1589     ),
1590     ),
1591     ),
1592     ),
1593     ),
1594     ),
1595     ),
1596     ),
1597     ),
1597     ),
1598     ),
1599     ),
1600     ),
1601     ),
1602     ),
1603     ),
1604     ),
1605     ),
1606     ),
1607     ),
1608     ),
1609     ),
1609     ),
1610     ),
1611     ),
1612     ),
1613     ),
1614     ),
1615     ),
1616     ),
1617     ),
1618     ),
1619     ),
1619     ),
1620     ),
1621     ),
1622     ),
1623     ),
1624     ),
1625     ),
1626     ),
1627     ),
1627     ),
1628     ),
1629     ),
1630     ),
1631     ),
1632     ),
1633     ),
1634     ),
1635     ),
1636     ),
1637     ),
1637     ),
1638     ),
1639     ),
1640     ),
1641     ),
1642     ),
1643     ),
1644     ),
1645     ),
1646     ),
1647     ),
1647     ),
1648     ),
1649     ),
1650     ),
1651     ),
1652     ),
1653     ),
1654     ),
1655     ),
1656     ),
1657     ),
1658     ),
1658     ),
1659     ),
1660     ),
1661     ),
1662     ),
1663     ),
1664     ),
1665     ),
1666     ),
1667     ),
1668     ),
1668     ),
1669     ),
1670     ),
1671     ),
1672     ),
1673     ),
1674     ),
1675     ),
1676     ),
1677     ),
1677     ),
1678     ),
1679     ),
1680     ),
1681     ),
1682     ),
1683     ),
1684     ),
1685     ),
1686     ),
1686     ),
1687     ),
1688     ),
1689     ),
1690     ),
1691     ),
1692     ),
1693     ),
1694     ),
1695     ),
1696     ),
1697     ),
1697     ),
1698     ),
1699     ),
1700     ),
1701     ),
1702     ),
1703     ),
1704     ),
1705     ),
1706     ),
1707     ),
1708     ),
1709     ),
1709     ),
1710     ),
1711     ),
1712     ),
1713     ),
1714     ),
1715     ),
1716     ),
1717     ),
1718     ),
1719     ),
1719     ),
1720     ),
1721     ),
1722     ),
1723     ),
1724     ),
1725     ),
1726     ),
1727     ),
1727     ),
1728     ),
1729     ),
1730     ),
1731     ),
1732     ),
1733     ),
1734     ),
1735     ),
1736     ),
1737     ),
1737     ),
1738     ),
1739     ),
1740     ),
1741     ),
1742     ),
1743     ),
1744     ),
1745     ),
1746     ),
1747     ),
1747     ),
1748     ),
1749     ),
1750     ),
1751     ),
1752     ),
1753     ),
1754     ),
1755     ),
1756     ),
1757     ),
1758     ),
1758     ),
1759     ),
1760     ),
1761     ),
1762     ),
1763     ),
1764     ),
1765     ),
1766     ),
1767     ),
1768     ),
1768     ),
1769     ),
1770     ),
1771     ),
1772     ),
1773     ),
1774     ),
1775     ),
1776     ),
1777     ),
1777     ),
1778     ),
1779     ),
1780     ),
1781     ),
1782     ),
1783     ),
1784     ),
1785     ),
1786     ),
1786     ),
1787     ),
1788     ),
1789     ),
1790     ),
1791     ),
1792     ),
1793     ),
1794     ),
1795     ),
1796     ),
1797     ),
1797     ),
1798     ),
1799     ),
1800     ),
1801     ),
1802     ),
1803     ),
1804     ),
1805     ),
1806     ),
1807     ),
1808     ),
1809     ),
1809     ),
1810     ),
1811     ),
1812     ),
1813     ),
1814     ),
1815     ),
1816     ),
1817     ),
1818     ),
1819     ),
1819     ),
1820     ),
1821     ),
1822     ),
1823     ),
1824     ),
1825     ),
1826     ),
1827     ),
1827     ),
1828     ),
1829     ),
1830     ),
1831     ),
1832     ),
1833     ),
1834     ),
1835     ),
1836     ),
1837     ),
1837     ),
1838     ),
1839     ),
1840     ),
1841     ),
1842     ),
1843     ),
1844     ),
1845     ),
1846     ),
1847     ),
1847     ),
1848     ),
1849     ),
1850     ),
1851     ),
1852     ),
1853     ),
1854     ),
1855     ),
1856     ),
1857     ),
1858     ),
1858     ),
1859     ),
1860     ),
1861     ),
1862     ),
1863     ),
1864     ),
1865     ),
1866     ),
1867     ),
1868     ),
1868     ),
1869     ),
1870     ),
1871     ),
1872     ),
1873     ),
1874     ),
1875     ),
1876     ),
1877     ),
1878     ),
1879     ),
1879     ),
1880     ),
1881     ),
1882     ),
1883     ),
1884     ),
1885     ),
1886     ),
1886     ),
1887     ),
1888     ),
1889     ),
1890     ),
1891     ),
1892     ),
1893     ),
1894     ),
1895     ),
1896     ),
1897     ),
1897     ),
1898     ),
1899     ),
1900     ),
1901     ),
1902     ),
1903     ),
1904     ),
1905     ),
1906     ),
1907     ),
1908     ),
1909     ),
1909     ),
1910     ),
1911     ),
1912     ),
1913     ),
1914     ),
191
```

**TC003**

**TC004**



The screenshot shows a mobile application interface. At the top, there are two tabs: "create\_habit\_page.dart" and "home\_page.dart". The "home\_page.dart" tab is active, displaying the following code:

```
lib > features > habits > presentation > pages > home_page.dart > HomePage > build
9   class HomePage extends StatelessWidget {
467     Future<void> _markAllHabitsComplete(BuildContext context) async {
483       await cubit.markAllHabitsComplete();
484       if (context.mounted) {
485         ScaffoldMessenger.of(context).showSnackBar(
486           SnackBar(
487             content: Text('${activeHabits.length} habits marked as complete!'),
488             backgroundColor: Colors.green,
489           ), // SnackBar
490         );
491       }
492     }
493   }
494
495   Future<void> _clearCompletedHabits(BuildContext context) async {
496     final cubit = context.read<HabitsCubit>();
497     final state = cubit.state;
498
499     if (state is HabitsLoaded) {
500       final completedHabits = state.habits.where((h) => h.isCompleted).toList();
501
502       if (completedHabits.isEmpty) {
503         ScaffoldMessenger.of(context).showSnackBar(
504           const SnackBar(
505             content: Text('No completed habits to clear'),
506             backgroundColor: Colors.orange,
507           ), // SnackBar
508         );
509       }
510     }
511   }
512 }
```

The main screen displays a list of habits with icons and edit buttons:

- read (target icon)
- football (soccer ball icon)
- gym (person lifting weights icon)
- drinking water (water drop icon)

At the bottom, there are navigation icons for Home, Report, My Habits, and Account.

create\_habit\_page.dart    home\_page.dart

```

lib > features > habits > presentation > pages > home_page.dart > HomePage > build
  9   class HomePage extends StatelessWidget {
467     Future<void> _markAllHabitsComplete(BuildContext context) async {
  ...
483       await cubit.markAllHabitsComplete();
484       if (context.mounted) {
485         ScaffoldMessenger.of(context).showSnackBar(
486           SnackBar(
487             content: Text('${activeHabits.length} habits marked as complete!'),
488             backgroundColor: Colors.green,
489           ), // SnackBar
490         );
491       }
492     }
493   }

  Future<void> _clearCompletedHabits(BuildContext context) async {
    final cubit = context.read<HabitsCubit>();
    final state = cubit.state;

    if (state is HabitsLoaded) {
      final completedHabits = state.habits.where((h) => h.isCompleted).toList();

      if (completedHabits.isEmpty) {
        ScaffoldMessenger.of(context).showSnackBar(
          const SnackBar(
            content: Text('No completed habits to clear'),
            backgroundColor: Colors.orange,
          ), // SnackBar
        );
      }
    }
  }

```

Ln 241, Col 37   Spaces: 2   UTF-8   LF   {} Dart   Windows (windows-x64)

The screenshot shows a mobile application interface. At the top, there's a navigation bar with icons for back, forward, and search. Below it is a header with the text 'Mark All Complete' and three buttons: 'Today', 'Weekly', and 'All'. The 'All' button is highlighted. To the right of the buttons are 'Clear Completed' and 'Reset All Data' buttons. The main content area displays four habits in cards: 'read' (yellow), 'football' (pink), 'gym' (light green), and 'drinking water' (light blue). Each card has a edit icon on the right. At the bottom of the screen are navigation tabs for 'Home', 'Report', 'My Habits', and 'Account'.

create\_habit\_page.dart    home\_page.dart

```

lib > features > habits > presentation > pages > home_page.dart > HomePage > build
  9   class HomePage extends StatelessWidget {
467     Future<void> _markAllHabitsComplete(BuildContext context) async {
  ...
483       await cubit.markAllHabitsComplete();
484       if (context.mounted) {
485         ScaffoldMessenger.of(context).showSnackBar(
486           SnackBar(
487             content: Text('${activeHabits.length} habits marked as complete!'),
488             backgroundColor: Colors.green,
489           ), // SnackBar
490         );
491       }
492     }
493   }

  Future<void> _clearCompletedHabits(BuildContext context) async {
    final cubit = context.read<HabitsCubit>();
    final state = cubit.state;

    if (state is HabitsLoaded) {
      final completedHabits = state.habits.where((h) => h.isCompleted).toList();

      if (completedHabits.isEmpty) {
        ScaffoldMessenger.of(context).showSnackBar(
          const SnackBar(
            content: Text('No completed habits to clear'),
            backgroundColor: Colors.orange,
          ), // SnackBar
        );
      }
    }
  }

```

Ln 241, Col 37   Spaces: 2   UTF-8   LF   {} Dart   Windows (windows-x64)

This screenshot is identical to the one above, showing the same four habits ('read', 'football', 'gym', 'drinking water') all marked as completed with green checkmarks next to them. The bottom navigation bar is also visible.

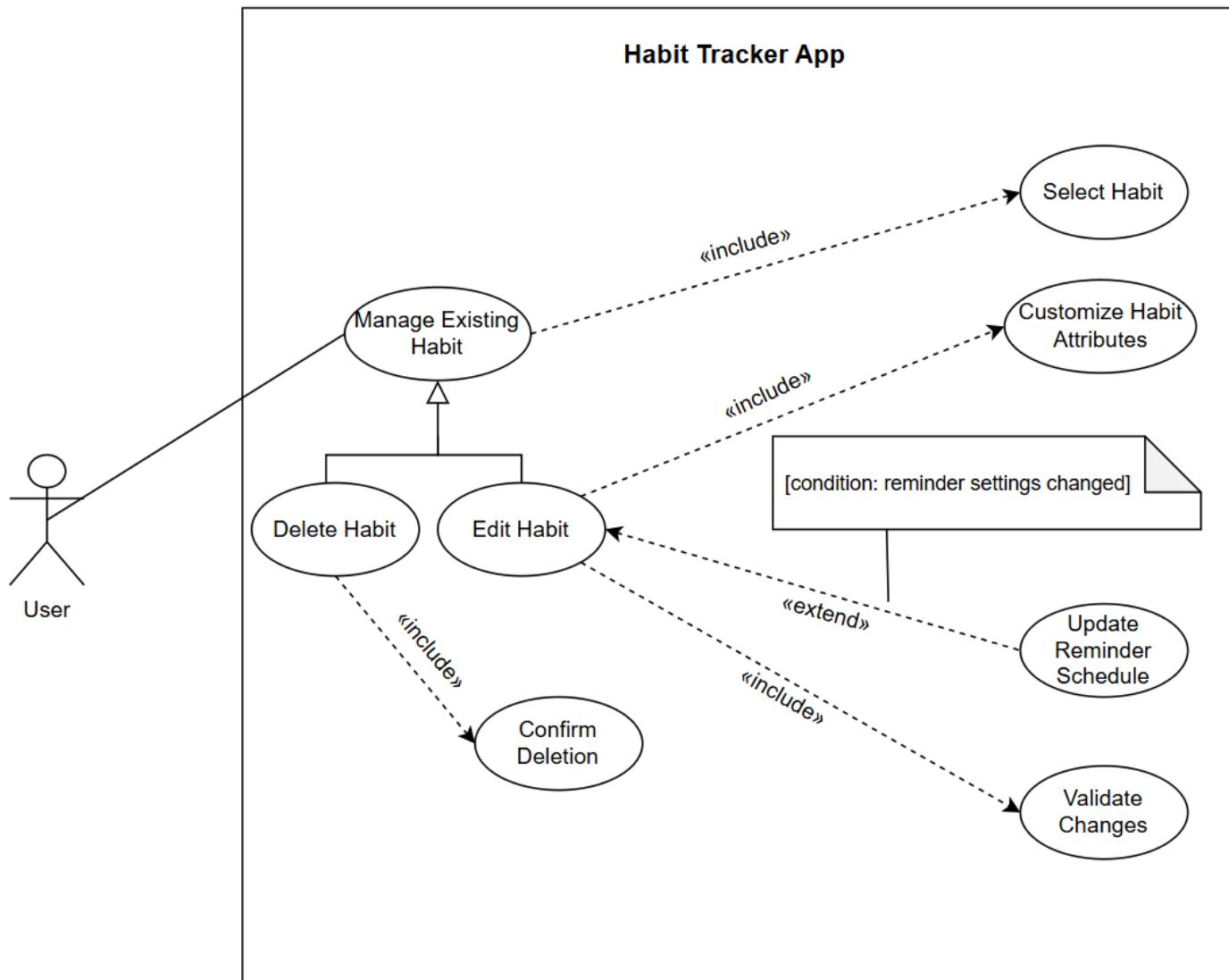
# **Sprint 2**

## Sprint #2 initial Meeting

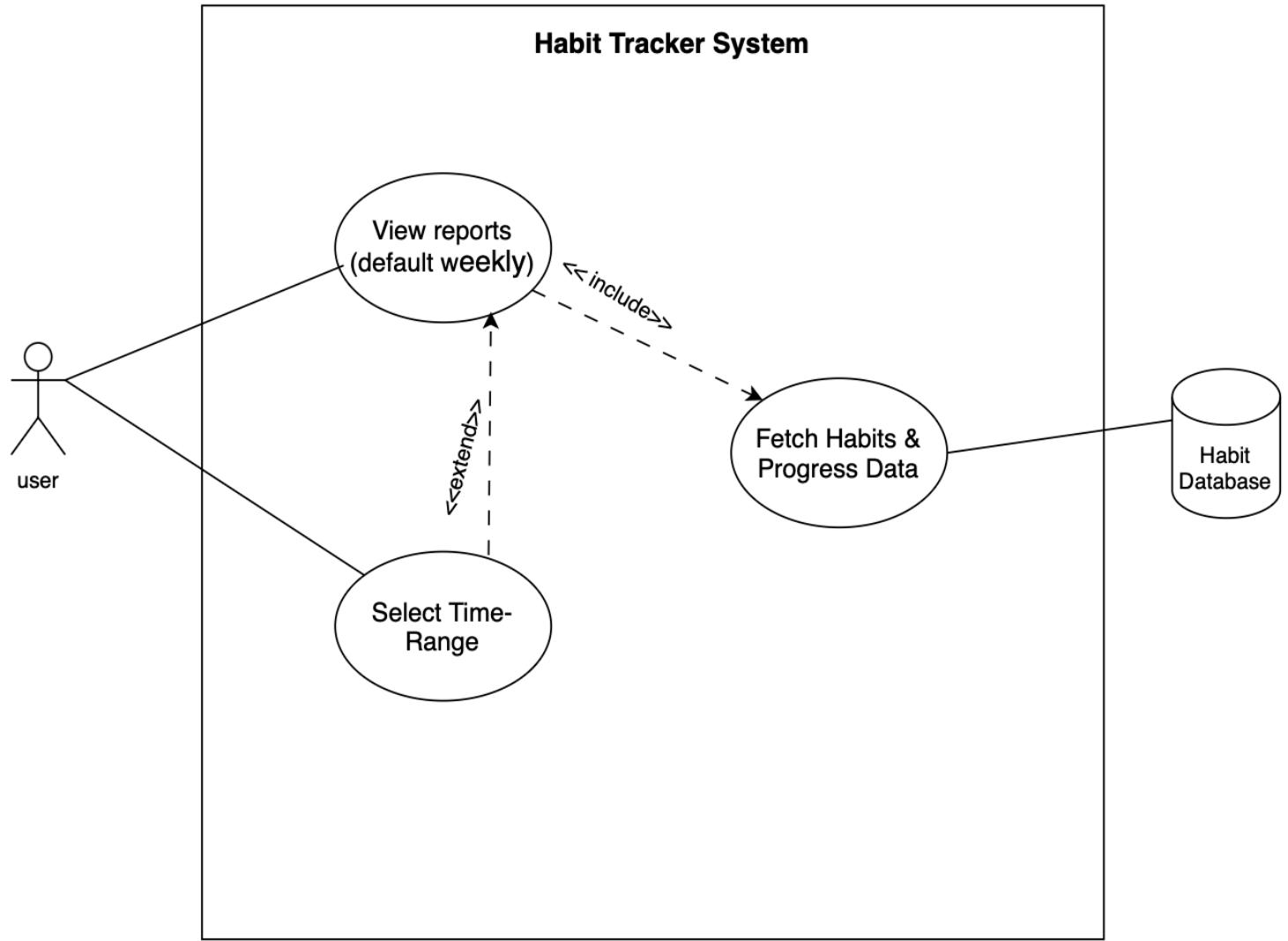
Sprint Meeting(s)									
<b>Project Name :</b> Daily Habit Tracker									
<b>Project Members:</b> Galla Al-faisal , Jumana AL-Jouhi ,Yara Al-shikhy ,Tamara Hawsawi, Hutoon Alasmarei									
<b>Sprint #2 initial Meeting - [5/11/2025]</b>									
<b><u>Sprint Duration:</u></b> : 2 Weeks									
<b><u>Scrum Master:</u></b> Galla									
<b><u>Client:</u></b> dr. Malak Al -Harbi									
<b><u>Pair Programmers:</u></b> Galla, Jumana, Yara, Tamara, Hitoon									
<b><u>Stories:</u></b>									
<table border="1"><thead><tr><th>Component Name</th><th>Story Sequence Number</th><th>Use Cases (e.g., functionalities)</th></tr></thead><tbody><tr><td>Edit Habit</td><td>03</td><td>Habit modification</td></tr><tr><td>Report Habit</td><td>04</td><td>View habit reports and overall progress</td></tr></tbody></table>	Component Name	Story Sequence Number	Use Cases (e.g., functionalities)	Edit Habit	03	Habit modification	Report Habit	04	View habit reports and overall progress
Component Name	Story Sequence Number	Use Cases (e.g., functionalities)							
Edit Habit	03	Habit modification							
Report Habit	04	View habit reports and overall progress							
<b><u>Follow-up meeting questions:</u></b>									
We reviewed the progress from the previous sprint and planned to add the ability to edit existing habits and create a full report page to track progress									
After that we discussed the goals of this sprint and how the new functions will improve the user experience									
Decided to continue using Flutter and divide the tasks equally among team members and communicate through WhatsApp and face to face meetings									
During this meeting we created user stories and use cases for both features as well as the Sequence diagrams to show how users will interact with the new components and prepared our test cases									
<b><u>Expected Outcomes</u></b>									
Define clear user stories and use cases for both new features									
Prepare test cases to validate the functions later									
Initial layout for the report page									
Basic editing functionality prepared									
<b><u>Scrum's Master comments based on the above questions:</u></b>									
The team demonstrated good planning and collaboration									
Our scrum master galla noted that the goals for this sprint were well defined and that the next step is to move forward with design and coding for both new features.									

# Requirements Documentation

## Use Case Diagram- Edit Habit



## Use Case Diagram – Habit Report



## User Story Specification:

**Component Name: Habit Editing**

**Story Name:** Edit Habit

**Story Sequence No:** 03

**Story short description:**

As a user, I want to be able to edit an existing habit so I can update their details whenever needed

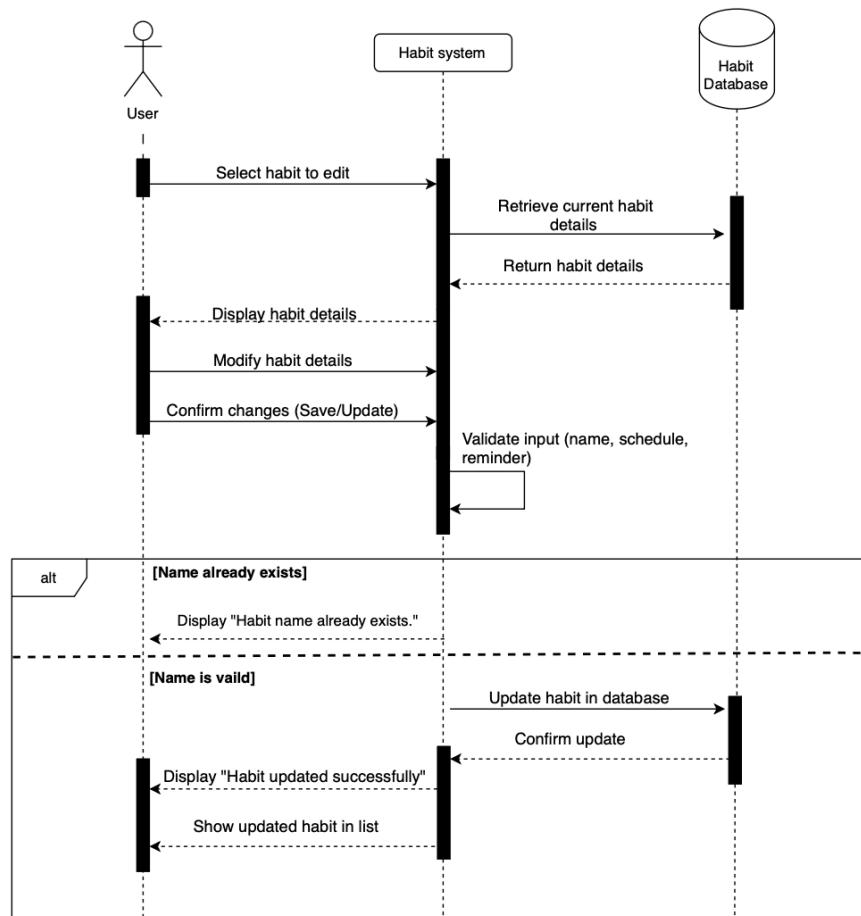
**Story long description:**

As a user, I need the ability to modify an existing habit by changing its name, icon, frequency, color or its reminder time after clicking the habit edit icon. I should be able to edit these details and save the changes. Then the updated habit should replace the old information and appear correctly in my habit list.

**Precondition:** User is logged and has at least one existing habit.

**Postcondition:** Habit details are updated and displayed with the new changes

## Sequence Diagram



## User Story Specification:

**Component Name:** Report Habit

**Story Name:** Edit Habit

**Story Sequence No:** 04

**Story short description:**

As a user, I want to view my habit report so I can track my progress .

**Story long description:**

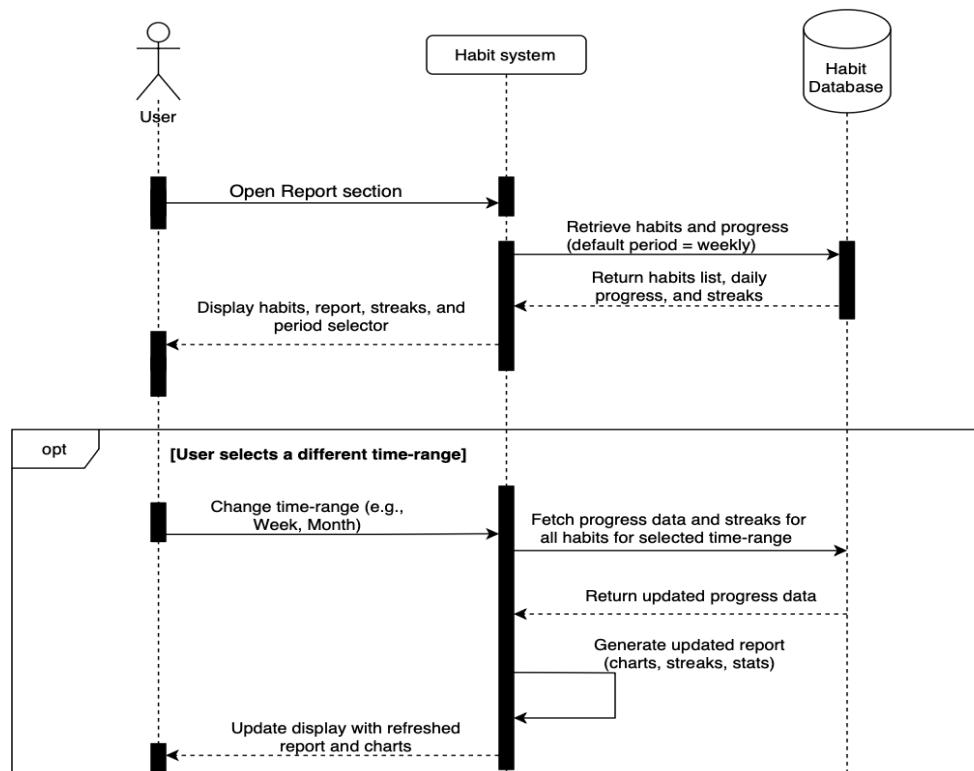
As a user, I need a report page that shows a clear summary of my habit activity

The report should display my current streak, completion rate, habits completed, and total perfect days. I should also be able to see a chart that shows how many habits I completed each day because this will help me understand my progress over time and stay motivated to complete my habits.

**Precondition:** user is logged and has existing habit completion data

**Postcondition:** a full updating report is displayed showing the streaks ,rates, statistics and weekly progress

## Sequence Diagram



## Sprint #2 Stand-up Meeting

### Sprint Meeting(s)

**Project Name :** Daily Habit Tracker

**Project Members:** Galla Al-faisal , Jumana AL-Jouhi ,Yara Al-shikhy ,Tamara Hawsawi, Hutoon Alasmarei

**Sprint #2 Stand-up Meeting - [12/11/2025]**

**Sprint Duration:** : 2 Weeks

**Scrum Master:** Galla

**Client:** : dr. Malak Al -Harbi

**Pair Programmers:** Galla, Jumana, Yara, Tamara, Hitoon

**Stories:**

Component Name	Story Sequence Number	Use Cases (e.g., functionalities)
Edit Habit	03	Habit modification
Report Habit	04	View habit reports and overall progress

#### **Follow-up meeting questions:**

During this sprint we finished coding both functions.

The edit feature now is completed and allows the user to change habit details while the report page now displays the streak completion rate, perfect days, and progress chart.

We tested the functions inside the app after completing the implementation and everything worked as expected We only faced a small issue with updating the report instantly after editing a habit but it was fixed

#### **Scrum's Master comments based on the above questions:**

All tasks were completed on time, and the team showed good problem-solving skills when fixing the issue between editing and reporting

The next sprint will focus on adding more refinements to the app Overall the team performed very well and achieved all the planned goals for Sprint 2

## Sprint #2 Test Cases - [13/11/2025]

**Test Case Name:** [Sprint 2 – Update Habit]

**Test Case Id:** [Daily Habit Tracker – Update Habit]

Test Case No.	Test Case Description	Expected Results	Outcome (Pass / Fail / Other (Comments))
TC001	User clears the Habit Name field and taps <b>Save Changes</b> .	System should show error: "Please enter a habit name".	Pass
TC002	Users try to Change a habit name to a duplicate name.	System should display: "Habit name already exists."	Pass
TC003	User changes the habit name only and taps <b>Save Changes</b>	Updated name should appear in main habit list	Pass
TC004	User selects a new <b>icon</b> and taps <b>Save Changes</b>	Habit list should show updated icon	Pass
TC005	User selects a new color and taps <b>Save Changes</b>	Habit color should be updated in habit list	Pass
TC006	User changes <b>Repeat</b> from Daily → Weekly and saves	Updated repeat rule should be reflected in habit details	Pass
TC007	User changes Do it at: from Morning → Evening	Habit should show updated time preference	Pass
TC008	User taps the Delete button	Confirmation dialog should appear	Pass
TC009	User selects Delete ( <b>permanently</b> ) in confirmation dialog (FROM ACTIVE HABIT LIST)	Habit should be removed completely from habit list	Pass
TC010	User selects Delete but keep data for archive	Habit disappears from list and appears in Archive	Pass
TC011	User goes to Archive and restores the habit (tap return icon)	Habit should appear back in active habit list with previous data	Pass
TC012	User taps "Delete" (FROM ARCHIVE PAGE)	Confirmation dialog should appear	Pass
TC013	User deletes archived habit	Habit removed permanently	Pass

**Test Case Name:** [Sprint 2 – Report]

**Test Case Id:** [Daily Habit Tracker – Report]

Test Case No.	Test Case Description	Expected Results	Outcome (Pass / Fail / Other (Comments ))
TC001	User opens the Report page with no completed habits	All values should show <b>0</b> (0 days streak, 0% completion, 0 habits completed, 0 perfect days)	Pass
TC002	User completes a habit at least once	“Habits completed” count increases based on actual completions	Pass
TC003	User completes habit for consecutive days	<b>Current streak</b> increases correctly	Pass
TC004	User completes all habits for a day	“Total perfect days” increases by 1	Pass
TC005	User switches date filter to <b>Today</b>	Charts and numbers update to show only today’s data	Pass
TC006	User switches date filter to <b>This Week</b>	Weekly data is displayed on all report widgets	Pass
TC007	User switches to <b>This Month</b>	Chart updates to show all completions for the month	Pass
TC008	User switches to <b>Last Month</b>	Last month’s statistics should be displayed accurately	Pass
TC009	User switches to <b>Last 6 Months</b>	Charts show aggregated 6-month data	Pass
TC010	User switches to <b>This Year</b>	Full yearly statistics are displayed	Pass
TC011	bar chart “Habits Completed” updates when habit is completed	The bar for the specific day increases	Pass
TC012	Line chart “Completion Rate” shows correct percentages per day/week	Percentage values match actual user behavior	Pass
TC013	Habit Streaks info updates when streak changes	“Active streak / total” updates correctly for each habit	Pass

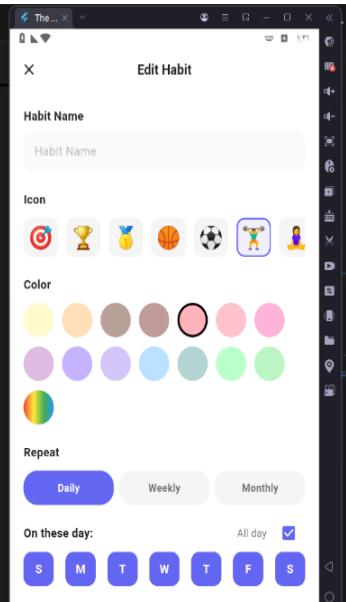
## Already existing habits

The screenshot shows a mobile application interface for managing habits. At the top, there is a navigation bar with icons for battery, signal, and notifications, followed by a font size selector and a three-dot menu icon. The main title "Home" is centered above a row of time filters: "Today", "Weekly", "Overall" (which is highlighted in blue), "All", "Morning", "Afternoon", and "Evening". Below this, four habit cards are displayed, each with an icon, the habit name, and three action buttons (edit, delete, add). A large blue "+" button is located at the bottom right of the main content area. At the very bottom, there is a navigation bar with four items: "Home" (selected), "Report", "My Habits", and "Account".

Habit	Icon	Action Buttons
grocery shopping	Target	Edit, Delete, Add
reading	Books	Edit, Delete, Add
drinking water	Water drop	Edit, Delete, Add
gym	Gym person	Edit, Delete, Add

## Test Case Name: Update Habit

### TC001

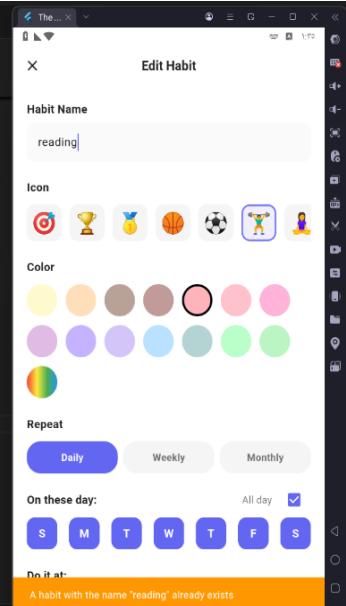


The screenshot shows the 'Edit Habit' dialog from a mobile application. The habit name is set to 'Habit Name'. An icon of a person is selected. The color palette shows a pink circle highlighted. The repeat frequency is set to 'Daily'. The 'On these day' section shows all days of the week selected. The 'Do it at:' field is empty.

```
edit_habit_page.dart
lib > features > habits > presentation > pages > edit_habit_page.dart > _EditHabitPageState extends State<EditHabitPage> {
  ...
  Future<void> _saveHabit() async {
    ...
    if (_nameController.text.trim().isEmpty) {
      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text('Please enter a habit name')),
      );
      return;
    }
    ...
    if (_nameController.text.trim().length > 100) {
      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(
          content: Text('Habit name must be 100 characters or less'),
          backgroundColor: Colors.orange,
        ),
      );
      return;
    }
    ...
    if (_originalHabit == null) return;

    // Validate weekly habit has selected days
    if (_selectedRepeat == 'Weekly' && _selectedDays.isEmpty) {
      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(
          content: Text('Please select at least one day for weekly habits'),
          backgroundColor: Colors.orange,
        ),
      );
      return;
    }
    ...
  }
}
```

### TC002

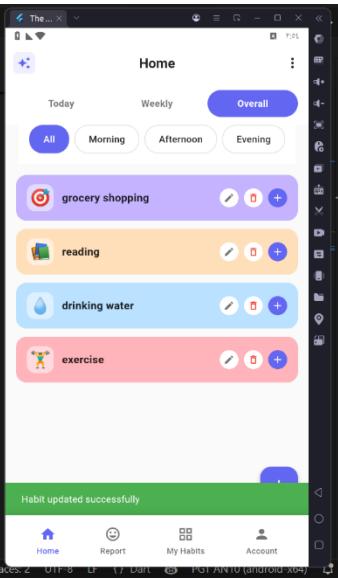


The screenshot shows the 'Edit Habit' dialog with the habit name 'reading'. An icon of a book is selected. The color palette shows a pink circle highlighted. The repeat frequency is set to 'Daily'. The 'On these day' section shows all days of the week selected. A message box at the bottom states 'A habit with the name "reading" already exists'.

```
edit_habit_page.dart
lib > features > habits > presentation > pages > edit_habit_page.dart > _EditHabitPageState extends State<EditHabitPage> {
  ...
  Future<void> _saveHabit() async {
    ...
    // Check for duplicate name (excluding current habit)
    final isDuplicate = await context.read().isHabitNameDuplicate(
      _nameController.text.trim(),
    );
    if (isDuplicate && _nameController.text.trim() != _originalHabit!.name) {
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(
          content: Text(
            'A habit with the name "${_nameController.text.trim()}" already exists',
          ),
          backgroundColor: Colors.orange,
        ),
      );
    }
    ...
  }

  ...
  // Convert repeat type string to enum
  data.RepeatType repeatType;
  switch (_selectedRepeat) {
    case 'Daily':
      repeatType = data.RepeatType.daily;
      break;
    case 'Weekly':
      repeatType = data.RepeatType.weekly;
      break;
    case 'Monthly':
      ...
  }
}
```

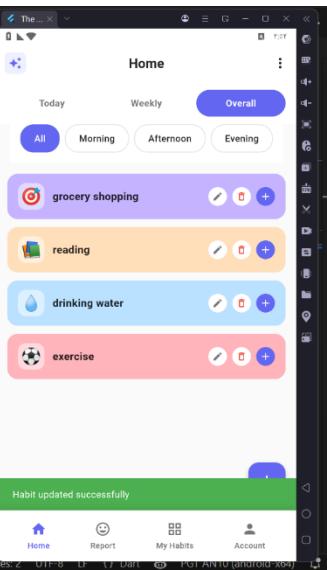
## TC003



The screenshot shows the Habitica mobile application's Home screen. It features a grid of four habits: "grocery shopping" (purple), "reading" (orange), "drinking water" (blue), and "exercise" (pink). Each habit card includes an edit icon, a delete icon, and a plus icon. Below the grid, a green banner displays the message "Habit updated successfully". At the bottom of the screen are navigation icons for Home, Report, My Habits, and Account.

```
edit_habit_page.dart
lib > features > habits > presentation > pages > edit_habit_page.dart > _EditHabitPageState > _saveHabit
31   class _EditHabitPageState extends State<EditHabitPage> {
219     Future<void> _saveHabit() async {
319       // Create Habit entity
320       final habit = domain.Habit(
321         id: _originalHabit!.id,
322         name: _nameController.text.trim(),
323         emoji: _selectedEmoji,
324         colorHex: _selectedColor == 'RAINBOW' ? 'FFB3BA' : _selectedColor,
325         timeOfDay: _selectedTimeOfDay,
326         status: _convertStatusToDomain(_originalHabit!.status),
327         isCompleted: _originalHabit!.isCompleted,
328         repeatType: _convertRepeatTypeToDomain(repeatType),
329         selectedDays: _selectedDays,
330         selectedMonthDates: _selectedMonthDates,
331         isSkippedToday: _originalHabit!.isSkippedToday,
332       );
333
334       // Update the full HabitModel with reminder and end habit settings
335       final updatedModel = _originalHabit!.copyWith(
336         name: _nameController.text.trim(),
337         emoji: _selectedEmoji,
338         colorHex: _selectedColor == 'RAINBOW' ? 'FFB3BA' : _selectedColor,
339         timeOfDay: _convertTimeOfDayToDate(_selectedTimeOfDay),
340         repeatType: repeatType,
341         selectedDays: _selectedDays,
342         selectedMonthDates: _selectedMonthDates,
343         endHabitEnabled: _endHabitEnabled,
344         endHabitMode: _endHabitEnabled ? _endHabitMode : null,
345         endDate: _endHabitEnabled && _endHabitMode == 'Date' ? endDate : null.
Ln 322, Col 7 (34 selected) Spaces: z OTF: 8 LF: 17 DFL: 63 FCL: A(110) (android-x64)
```

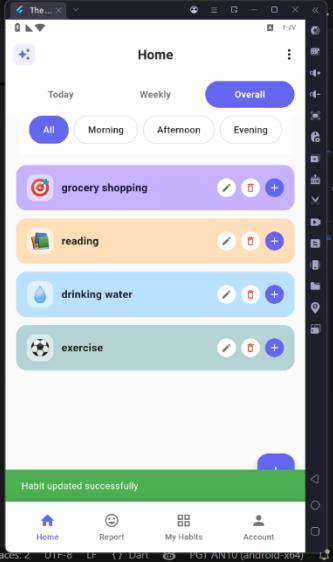
## TC004



The screenshot shows the Habitica mobile application's Home screen, identical to the one in TC003. It features a grid of four habits: "grocery shopping" (purple), "reading" (orange), "drinking water" (blue), and "exercise" (pink). Each habit card includes an edit icon, a delete icon, and a plus icon. Below the grid, a green banner displays the message "Habit updated successfully". At the bottom of the screen are navigation icons for Home, Report, My Habits, and Account.

```
edit_habit_page.dart
lib > features > habits > presentation > pages > edit_habit_page.dart > _EditHabitPageState > _saveHabit
31   class _EditHabitPageState extends State<EditHabitPage> {
219     Future<void> _saveHabit() async {
319       // Create Habit entity
320       final habit = domain.Habit(
321         id: _originalHabit!.id,
322         name: _nameController.text.trim(),
323         emoji: _selectedEmoji,
324         colorHex: _selectedColor == 'RAINBOW' ? 'FFB3BA' : _selectedColor,
325         timeOfDay: _selectedTimeOfDay,
326         status: _convertStatusToDomain(_originalHabit!.status),
327         isCompleted: _originalHabit!.isCompleted,
328         repeatType: _convertRepeatTypeToDomain(repeatType),
329         selectedDays: _selectedDays,
330         selectedMonthDates: _selectedMonthDates,
331         isSkippedToday: _originalHabit!.isSkippedToday,
332       );
333
334       // Update the full HabitModel with reminder and end habit settings
335       final updatedModel = _originalHabit!.copyWith(
336         name: _nameController.text.trim(),
337         emoji: _selectedEmoji,
338         colorHex: _selectedColor == 'RAINBOW' ? 'FFB3BA' : _selectedColor,
339         timeOfDay: _convertTimeOfDayToDate(_selectedTimeOfDay),
340         repeatType: repeatType,
341         selectedDays: _selectedDays,
342         selectedMonthDates: _selectedMonthDates,
343         endHabitEnabled: _endHabitEnabled,
344         endHabitMode: _endHabitEnabled ? _endHabitMode : null,
345         endDate: _endHabitEnabled && _endHabitMode == 'Date' ? endDate : null.
Ln 323, Col 5 (23 selected) Spaces: z OTF: 8 LF: 17 DFL: 63 FCL: A(110) (android-x64)
```

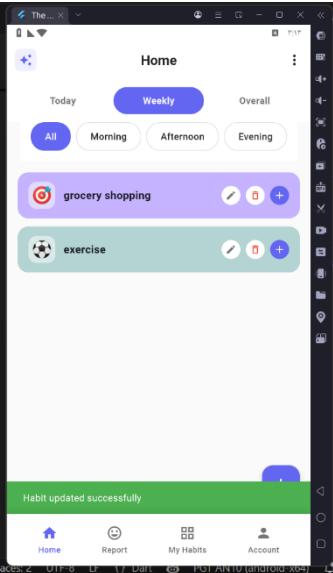
## TC005



edit\_habit\_page.dart

```
lib > features > habits > presentation > pages > edit_habit_page.dart > _EditHabitPageState > _saveHabit
31   class _EditHabitPageState extends State<EditHabitPage> {
32     Future<void> _saveHabit() async {
33
34       // Create Habit entity
35       final habit = domain.Habit(
36         id: _originalHabit!.id,
37         name: _nameController.text.trim(),
38         emoji: _selectedEmoji,
39         colorHex: _selectedColor == 'RAINBOW' ? '#FFB3BA' : _selectedColor,
40         timeOfDay: _selectedTimeOfDay,
41         status: _convertStatusToDomain(_originalHabit!.status),
42         isCompleted: _originalHabit!.isCompleted,
43         repeatType: _convertRepeatTypeToDomain(repeatType),
44         selectedDays: _selectedDays,
45         selectedMonthDates: _selectedMonthDates,
46         isSkippedToday: _originalHabit!.isSkippedToday,
47       );
48
49       // Update the full HabitModel with reminder and end habit settings
50       final updatedModel = _originalHabit!.copyWith(
51         name: _nameController.text.trim(),
52         emoji: _selectedEmoji,
53         colorHex: _selectedColor == 'RAINBOW' ? '#FFB3BA' : _selectedColor,
54         timeOfDay: _convertTimeOfDayToDate(_selectedTimeOfDay),
55         repeatType: repeatType,
56         selectedDays: _selectedDays,
57         selectedMonthDates: _selectedMonthDates,
58         endHabitEnabled: _endHabitEnabled,
59         endHabitMode: _endHabitEnabled ? _endHabitMode : null,
60         endDate: _endHabitEnabled && _endHabitMode == 'Date' ? _endDate : null,
61         daysAfter: _endHabitEnabled && _endHabitMode == 'Days'
62           ? int.tryParse(_daysAfterController.text) ?? 365
63           : null,
64         reminderEnabled: _reminderEnabled,
65       );
66
67       // Save habit
68       await habitRepository.updateHabit(updatedModel);
69
70       ScaffoldMessenger.of(context).showSnackBar(SnackBar(
71         content: Text('Habit updated successfully'),
72         duration: Duration(seconds: 2),
73       ));
74     }
75   }
```

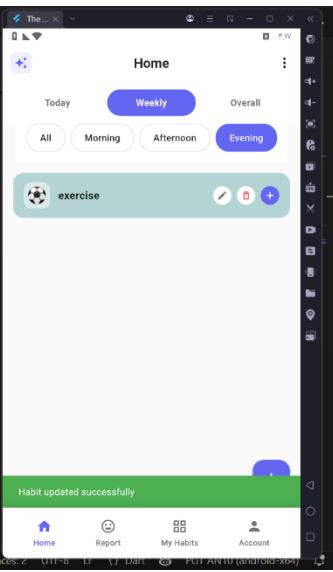
## TC006



edit\_habit\_page.dart

```
lib > features > habits > presentation > pages > edit_habit_page.dart > _EditHabitPageState > _saveHabit
219   Future<void> _saveHabit() async {
220     if (_nameController.text.isEmpty) {
221       ScaffoldMessenger.of(context).showSnackBar(SnackBar(
222         content: Text('Name cannot be empty'),
223         duration: Duration(seconds: 2),
224       ));
225       return;
226     }
227
228     // Convert repeat type string to enum
229     data.RepeatType repeatType;
230     switch (_selectedRepeat) {
231       case 'Daily':
232         repeatType = data.RepeatType.daily;
233         break;
234       case 'Weekly':
235         repeatType = data.RepeatType.weekly;
236         break;
237       case 'Monthly':
238         repeatType = data.RepeatType.monthly;
239         break;
240       default:
241         repeatType = data.RepeatType.daily;
242     }
243
244     // Create Habit entity
245     final habit = domain.Habit(
246       id: _originalHabit!.id,
247       name: _nameController.text.trim(),
248     );
249
250     // Save habit
251     await habitRepository.updateHabit(habit);
252
253     ScaffoldMessenger.of(context).showSnackBar(SnackBar(
254       content: Text('Habit updated successfully'),
255       duration: Duration(seconds: 2),
256     ));
257   }
258 }
```

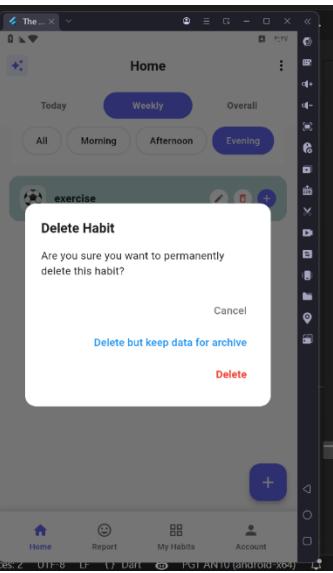
## TC007



edit\_habit\_page.dart

```
lib > features > habits > presentation > pages > edit_habit_page.dart > _EditHabitPageState > _saveHabit
31   class _EditHabitPageState extends State<EditHabitPage> {
32     Future<void> _saveHabit() async {
33
34       // Create Habit entity
35       final habit = domain.Habit(
36         id: _originalHabit!.id,
37         name: _nameController.text.trim(),
38         emoji: _selectedEmoji,
39         colorHex: _selectedColor == 'RAINBOW' ? '#FFB3BA' : _selectedColor,
40         timeOfDay: _selectedTimeOfDay,
41         status: _convertStatusToDomain(_originalHabit!.status),
42         isCompleted: _originalHabit!.isCompleted,
43         repeatType: _convertRepeatTypeToDomain(repeatType),
44         selectedDays: _selectedDays,
45         selectedMonthDates: _selectedMonthDates,
46         isSkippedToday: _originalHabit!.isSkippedToday,
47       );
48
49       // Update the full HabitModel with reminder and end habit settings
50       final updatedModel = _originalHabit!.copyWith(
51         name: _nameController.text.trim(),
52         emoji: _selectedEmoji,
53         colorHex: _selectedColor == 'RAINBOW' ? '#FFB3BA' : _selectedColor,
54         timeOfDay: _convertTimeOfDayToDate(_selectedTimeOfDay),
55         repeatType: repeatType,
56         selectedDays: _selectedDays,
57         selectedMonthDates: _selectedMonthDates,
58         endHabitEnabled: _endHabitEnabled,
59         endHabitMode: _endHabitEnabled ? _endHabitMode : null,
60         endDate: _endHabitEnabled && _endHabitMode == 'Date' ? _endDate : null,
61         daysAfter: _endHabitEnabled && _endHabitMode == 'Days'
62           ? int.tryParse(_daysAfterController.text) ?? 365
63           : null,
64         reminderEnabled: _reminderEnabled,
65       );
66
67       // Save habit
68       await habitRepository.updateHabit(updatedModel);
69
70       ScaffoldMessenger.of(context).showSnackBar(SnackBar(
71         content: Text('Habit updated successfully'),
72         duration: Duration(seconds: 2),
73       ));
74     }
75   }
```

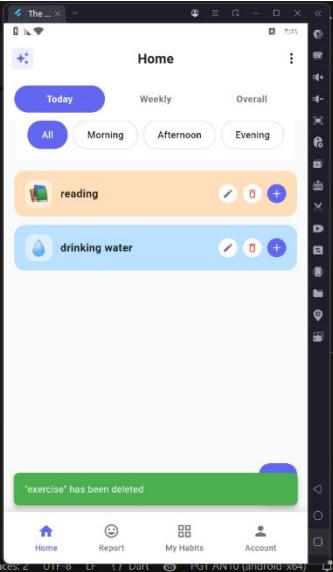
## TC008



The screenshot shows the 'Home' screen of the app. A modal dialog titled 'Delete Habit' is displayed, asking 'Are you sure you want to permanently delete this habit?'. It has 'Cancel' and 'Delete but keep data for archive' buttons. In the bottom right corner of the screen, there is a small floating action button.

```
home_page.dart
lib > features > habits > presentation > pages > home_page.dart > _showDeleteConfirmation
11 class HomePage extends StatelessWidget {
619 Future<void> _showDeleteConfirmation(
  <...> {
    return showDialog<void>(
      context: context,
      barrierDismissible: false,
      builder: (BuildContext context) {
        return AlertDialog(
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(16),
          ),
          title: const Text(
            'Delete Habit',
            style: TextStyle(fontSize: 20, fontWeight: FontWeight.w600),
          ),
          content: const Text(
            'Are you sure you want to permanently delete this habit?',
            style: TextStyle(fontSize: 16),
          ),
          actions: <Widget>[
            TextButton(
              onPressed: () {
                Navigator.of(context).pop();
              },
              child: const Text(
                'Cancel',
                style: TextStyle(fontSize: 16, color: Colors.black54),
              ),
            ),
            TextButton(
              onPressed: () {
                Navigator.of(context).pop();
              },
              child: const Text(
                'Delete but keep data for archive',
                style: TextStyle(fontSize: 16, color: Colors.red),
              ),
            ),
          ],
        );
      }
    );
  }
}
```

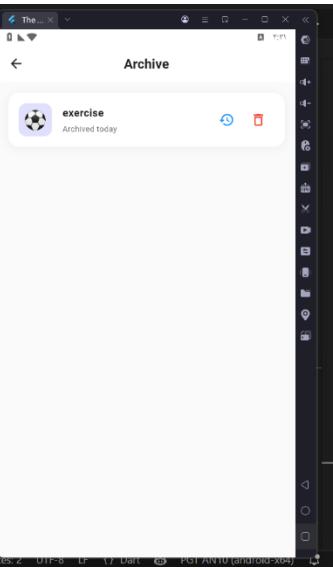
## TC009



The screenshot shows the 'Home' screen after a habit has been deleted. A green snack bar at the bottom displays the message "'exercise' has been deleted'. The habit card for 'drinking water' is still visible.

```
archive_page.dart
home_page.dart
lib > features > habits > presentation > pages > home_page.dart > _showDeleteConfirmation
11 class HomePage extends StatelessWidget {
619 Future<void> _showDeleteConfirmation(
  <...>
    // Delete from database and update UI instantly
    await context.read
```

## TC010

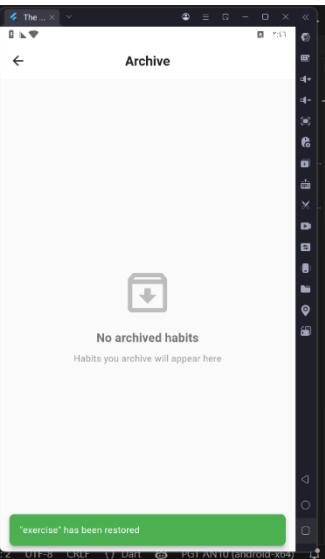


The screenshot shows the 'Archive' screen. It displays a single habit card for 'exercise' with the status 'Archived today'. There is a floating action button in the bottom right corner.

```
home_page.dart
lib > features > habits > presentation > pages > home_page.dart > _showDeleteConfirmation
11 class HomePage extends StatelessWidget {
619 Future<void> _showDeleteConfirmation(
  <...>
    TextButton(
      onPressed: () async {
        Navigator.of(context).pop();

        // Archive habit (keep data but remove from active list)
        await context.read
```

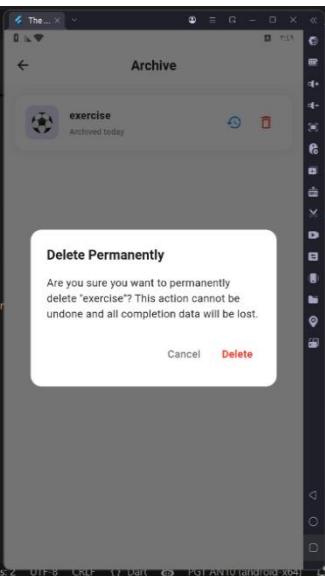
## TC011



The screenshot shows the 'Archive' screen of the Habit Tracker app. At the top, there's a large download icon. Below it, a message says 'No archived habits' with a note 'Habits you archive will appear here'. A green rectangular overlay at the bottom displays the text "'exercise' has been restored'". On the left, the code for the `_restoreHabit` method is visible.

```
lib > features > habits > presentation > pages > archive_page.dart > _ArchivePageState > _loadArchivedHabits
15 class _ArchivePageState extends State<ArchivePage> {
16   ...
17
18   Future<void> _restoreHabit(String habitId, String habitName) async {
19     try {
20       await context.read<HabitsCubit>().restoreHabit(habitId);
21       await _loadArchivedHabits();
22
23       if (mounted) {
24         ScaffoldMessenger.of(context).showSnackBar(
25           SnackBar(
26             content: Text("$habitName has been restored"),
27             backgroundColor: Colors.green,
28             behavior: SnackBarBehavior.floating,
29             shape: RoundedRectangleBorder(
30               borderRadius: BorderRadius.circular(8),
31             ), // RoundedRectangleBorder
32           ), // SnackBar
33         );
34       }
35     } catch (e) {
36       if (mounted) {
37         ScaffoldMessenger.of(context).showSnackBar(
38           SnackBar(
39             content: Text('Error restoring habit: $e'),
40             backgroundColor: Colors.red,
41           ), // SnackBar
42         );
43       }
44     }
45   }
46
47 }
```

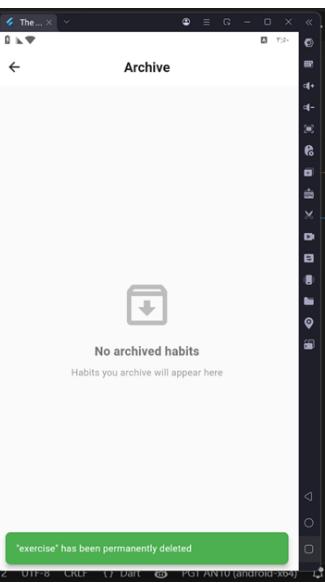
## TC012



The screenshot shows the 'Archive' screen with a single habit entry for 'exercise' (Archived today). A modal dialog titled 'Delete Permanently' is displayed, asking if the user wants to permanently delete the habit. It includes 'Cancel' and 'Delete' buttons. On the left, the code for the `_deletePermanently` method is shown.

```
lib > features > habits > presentation > pages > archive_page.dart > _ArchivePageState > _restoreHabit
15 class _ArchivePageState extends State<ArchivePage> {
16   ...
17
18   Future<void> _deletePermanently(String habitId, String habitName) async {
19     return AlertDialog(
20       shape: RoundedRectangleBorder(
21         borderRadius: BorderRadius.circular(16),
22       ), // RoundedRectangleBorder
23       title: const Text(
24         'Delete Permanently',
25         style: TextStyle(
26           fontSize: 20,
27           fontWeight: FontWeight.w600,
28         ), // TextStyle
29       ), // Text
30       content: Text(
31         'Are you sure you want to permanently delete "$habitName"? This action cannot be undone and all completion data will be lost.',
32         style: const TextStyle(
33           fontSize: 16,
34         ), // TextStyle
35       ), // Text
36       actions: <Widget>[
37         TextButton(
38           onPressed: () {
39             Navigator.of(context).pop(false);
40           },
41         ),
42         child: const Text(
43           'Cancel',
44           style: TextStyle(
45             fontSize: 16,
46             color: Colors.black54,
47           ), // TextStyle
48         ),
49       ],
50     );
51   }
52 }
```

## TC013



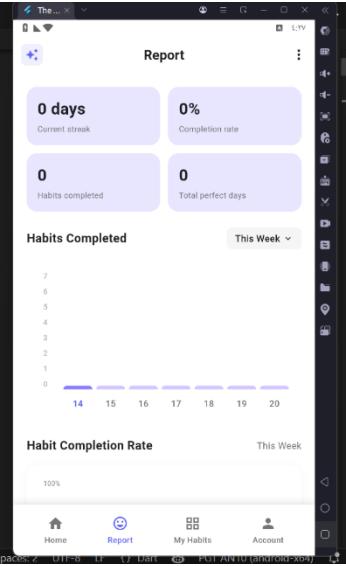
The screenshot shows the 'Archive' screen again. A green rectangular overlay at the bottom displays the text "'exercise' has been permanently deleted'. On the left, the code for the `_deletePermanently` method is shown, including the logic to confirm the deletion before proceeding.

```
lib > features > habits > presentation > pages > archive_page.dart > _ArchivePageState > _restoreHabit
15 class _ArchivePageState extends State<ArchivePage> {
16   ...
17
18   Future<void> _deletePermanently(String habitId, String habitName) async {
19     if (confirmed == true) {
20       try {
21         await context.read<HabitsCubit>().deleteHabit(habitId);
22         await _loadArchivedHabits();
23
24         if (mounted) {
25           ScaffoldMessenger.of(context).showSnackBar(
26             SnackBar(
27               content: Text("$habitName has been permanently deleted"),
28               backgroundColor: Colors.green,
29               behavior: SnackBarBehavior.floating,
30               shape: RoundedRectangleBorder(
31                 borderRadius: BorderRadius.circular(8),
32               ), // RoundedRectangleBorder
33             ), // SnackBar
34           );
35         }
36       } catch (e) {
37         if (mounted) {
38           ScaffoldMessenger.of(context).showSnackBar(
39             SnackBar(
40               content: Text('Error deleting habit: $e'),
41               backgroundColor: Colors.red,
42             ), // SnackBar
43           );
44         }
45       }
46     }
47   }
48 }
```

## **Test Case Name:** Report

TC001

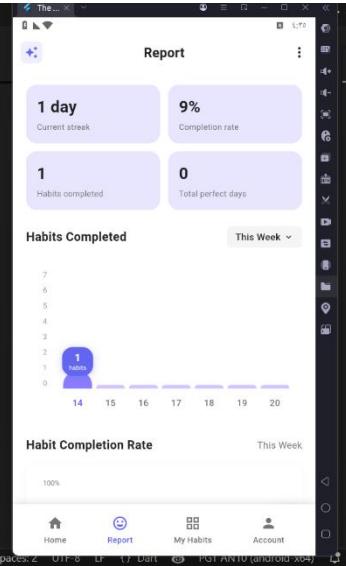
```
archive_page.dart   home_page.dart   report_page.dart X
lib/features/habits/presentation/pages/report_page.dart > ReportPageState > _loadStatistics
24
25 class _ReportPageState extends State<ReportPage> {
26   DateRangeOption _selectedDateRange = DateRangeOption.thisWeek;
27   bool _isLoading = false;
28   int _totalDays = 0;
29   int _habitsCompleted = 0;
30   int _totalPerfectDays = 0;
31   double _completionRate = 0.0;
32   int _currentStreak = 0;
33   List<Map<String, dynamic>>? _weeklyStats;
34   List<Map<String, dynamic>>? _monthlyCompletionRate;
35   List<Map<String, dynamic>>? _habitsStreaks;
36
37 @override
38 void initState() {
39   super.initState();
40   _loadStatistics();
41 }
42
43 Future<void> _loadStatistics() async {
44   setState(() => _isLoading = true);
45   try {
46     final repository = getIt<HabitsRepository>();
47     final today = DateTime.now();
48
49     DateTime startDate;
50     DateTime endDate = today;
51
52     // Calculate date range based on selection
53     switch (_selectedDateRange) {
```



TC002

```
archive_page.dart  home_page.dart  report_page.dart X

lib > features > habits > presentation > pages > report_page.dart > _ReportPageState > _loadStatistics
25   class _ReportPageState extends State<ReportPage> {
43     Future<void> _loadStatistics() async {
135   }
136   _weeklyStats = await repository.getWeeklyStatistics(startDateOfWeek);
137
138   // Calculate completion stats for the date range
139   final dailyStats = <Map<String, dynamic>>[];
140   for (int i = 0; i < _totalDays; i++) {
141     final currentDate = startDate.add(Duration(days: i));
142     final dayStats = await repository.getDailyStatistics(currentDate);
143     final completed = dayStats['completed'] as int;
144     final total = dayStats['total'] as int;
145
146     _habitsCompleted += completed;
147     totalPossibleHabits += total;
148
149     if (total > 0 && completed == total) {
150       _totalPerfectDays++;
151     }
152
153     dailyStats.add({
154       'date': currentDate,
155       'completed': completed,
156       'total': total,
157       'rate': total > 0 ? (completed / total * 100) : 0,
158     });
159   }
160
161   _completionRate = totalPossibleHabits > 0
```



TC003

**TC004**

The screenshot shows a Flutter application interface. At the top, there are three tabs: 'archive\_page.dart', 'home\_page.dart', and 'report\_page.dart'. The 'report\_page.dart' tab is active. Below the tabs, the code for the 'ReportPageState' class is displayed, showing methods for loading statistics and calculating completion rates. A yellow warning icon is present on line 149.

```
lib > features > habits > presentation > pages > report_page.dart > _ReportPageState > _loadStatistics
25 class _ReportPageState extends State<ReportPage> {
43 Future<void> _loadStatistics() async {
135     }
136     _weeklyStats = await repository.getWeeklyStatistics(startOfWeek);
137
138     // Calculate completion stats for the date range
139     final dailyStats = <Map<String, dynamic>>[];
140     for (int i = 0; i < _totalDays; i++) {
141         final currentDate = startDate.add(Duration(days: i));
142         final dayStats = await repository.getDailyStatistics(currentDate);
143         final completed = dayStats['completed'] as int;
144         final total = dayStats['total'] as int;

145         _habitsCompleted += completed;
146         totalPossibleHabits += total;
147
148         if (total > 0 && completed == total) {
149             _totalPerfectDays++;
150         }
151     }

152     dailyStats.add({
153         'date': currentDate,
154         'completed': completed,
155         'total': total,
156         'rate': total > 0 ? (completed / total * 100) : 0,
157     });
158 }
159
160
161
162     _completionRate = totalPossibleHabits > 0
```

The right side of the screen displays a 'Report' summary. It includes a 'Current streak' of 1 day with a 18% completion rate, 2 habits completed, and 1 total perfect days. Below this, a chart titled 'Habits Completed' shows a single bar reaching value 2 over a period from 14 to 20. Another chart titled 'Habit Completion Rate' shows a single point at 100% completion for the week.

TC005

The screenshot shows a Flutter application interface. On the left, there is a code editor with the file `report_page.dart` open. The code defines a `_ReportPageState` class that extends `State<ReportPage>`. It contains a method `_loadStatistics` which uses `FutureBuilder` to get the current date and calculate a date range based on the selected option (today, this week, this month, last month, last 6 months, this year, last year). The right side of the screen displays a "Report" page with three cards: "1 day" (Current streak: 100%), "2" (Habits completed: 1), and "Habits Completed" (a chart showing 2 habits completed out of 14 total). Below these are two circular progress indicators: "Habit Completion Rate" (100%) and "My Habits" (100%).

```
archive.page.dart home.page.dart report.page.dart
lib > features > habits > presentation > pages > report_page.dart > _ReportPageState > _loadStatistics
25 class _ReportPageState extends State<ReportPage> {
43 Future<void> _loadStatistics() async {
50   DateTime endDate = today,
51
52   // Calculate date range based on selection
53   switch (_selectedDateRange) {
54     case DateRangeOption.today:
55       startDate = DateTime(today.year, today.month, today.day);
56       break;
57     case DateRangeOption.thisWeek:
58       startDate = today.subtract(Duration(days: today.weekday - 1));
59       break;
60     case DateRangeOption.thisMonth:
61       startDate = DateTime(today.year, today.month, 1);
62       break;
63     case DateRangeOption.lastMonth:
64       startDate = DateTime(today.year, today.month - 1, 1);
65       endDate = DateTime(
66         today.year,
67         today.month,
68         1,
69       ).subtract(const Duration(days: 1));
70       break;
71     case DateRangeOption.last6Months:
72       startDate = DateTime(today.year, today.month - 6, today.day);
73       break;
74     case DateRangeOption.thisYear:
75       startDate = DateTime(today.year, 1, 1);
76       break;
77     case DateRangeOption.lastYear:
```

## TC006

The screenshot shows a mobile application interface titled "Report". At the top, there are two purple boxes: "1 day" (Current streak) with a value of 18% and "2" (Habits completed) with a value of 1 Total perfect days. Below this is a section titled "Habits Completed" with a bar chart showing a count of 2 habits from 14 to 20. A circular progress bar at the bottom indicates a Habit Completion Rate of 100%. Navigation tabs at the bottom include Home, Report (selected), My Habits, and Account.

```
archive_page.dart home_page.dart report_page.dart
lib/features/habits/presentation/pages/report_page.dart > _ReportPageState > _loadStatistics
25 class _ReportPageState extends State<ReportPage> {
43 Future<void> _loadStatistics() async {
50   DateTime endDate = today;
51
52   // Calculate date range based on selection
53   switch (_selectedDateRange) {
54     case DateRangeOption.today:
55       startDate = DateTime(today.year, today.month, today.day);
56       break;
57     case DateRangeOption.thisWeek:
58       startDate = today.subtract(Duration(days: today.weekday - 1));
59       break;
59
60     case DateRangeOption.thisMonth:
61       startDate = DateTime(today.year, today.month, 1);
62       break;
63     case DateRangeOption.lastMonth:
64       startDate = DateTime(today.year, today.month - 1, 1);
65       endDate = DateTime(
66         today.year,
67         today.month,
68         1,
69       ).subtract(const Duration(days: 1));
70       break;
71     case DateRangeOption.last6Months:
72       startDate = DateTime(today.year, today.month - 6, today.day);
73       break;
74     case DateRangeOption.thisYear:
75       startDate = DateTime(today.year, 1, 1);
76       break;
77     case DateRangeOption.lastYear:
77
In 57, Col 7 (122 selected) Spaces: 2 0TF-6 L7 Dart PGT ANTU (android XDP)
```

## TC007

The screenshot shows a mobile application interface titled "Report". At the top, there are two purple boxes: "1 day" (Current streak) with a value of 7% and "2" (Habits completed) with a value of 1 Total perfect days. Below this is a section titled "Habits Completed" with a bar chart showing a count of 2 habits from 14 to 20. A circular progress bar at the bottom indicates a Habit Completion Rate of 100%. Navigation tabs at the bottom include Home, Report (selected), My Habits, and Account.

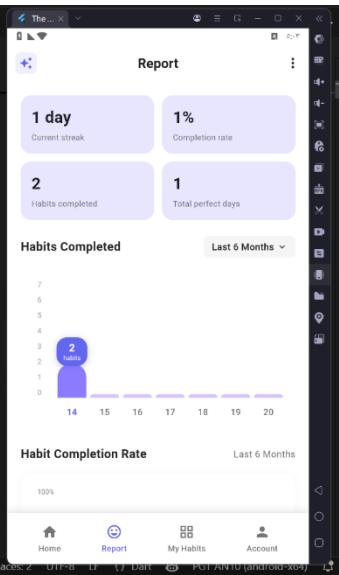
```
archive_page.dart home_page.dart report_page.dart
lib/features/habits/presentation/pages/report_page.dart > _ReportPageState > _loadStatistics
25 class _ReportPageState extends State<ReportPage> {
43 Future<void> _loadStatistics() async {
50   DateTime endDate = today;
51
52   // Calculate date range based on selection
53   switch (_selectedDateRange) {
54     case DateRangeOption.today:
55       startDate = DateTime(today.year, today.month, today.day);
56       break;
57     case DateRangeOption.thisWeek:
58       startDate = today.subtract(Duration(days: today.weekday - 1));
59       break;
59
60     case DateRangeOption.thisMonth:
61       startDate = DateTime(today.year, today.month, 1);
62       break;
63     case DateRangeOption.lastMonth:
64       startDate = DateTime(today.year, today.month - 1, 1);
65       endDate = DateTime(
66         today.year,
67         today.month,
68         1,
69       ).subtract(const Duration(days: 1));
70       break;
71     case DateRangeOption.last6Months:
72       startDate = DateTime(today.year, today.month - 6, today.day);
73       break;
74     case DateRangeOption.thisYear:
75       startDate = DateTime(today.year, 1, 1);
76       break;
77     case DateRangeOption.lastYear:
77
In 60, Col 8 (109 selected) Spaces: 2 0TF-6 L7 Dart PGT ANTU (android XDP)
```

## TC008

The screenshot shows a mobile application interface titled "Report". At the top, there are two purple boxes: "1 day" (Current streak) with a value of 0% and "0" (Habits completed) with a value of 0 Total perfect days. Below this is a section titled "Habits Completed" with a bar chart showing a count of 2 habits from 14 to 20. A circular progress bar at the bottom indicates a Habit Completion Rate of 100%. Navigation tabs at the bottom include Home, Report (selected), My Habits, and Account.

```
archive_page.dart home_page.dart report_page.dart
lib/features/habits/presentation/pages/report_page.dart > _ReportPageState > _loadStatistics
25 class _ReportPageState extends State<ReportPage> {
43 Future<void> _loadStatistics() async {
50   DateTime endDate = today;
51
52   // Calculate date range based on selection
53   switch (_selectedDateRange) {
54     case DateRangeOption.today:
55       startDate = DateTime(today.year, today.month, today.day);
56       break;
57     case DateRangeOption.thisWeek:
58       startDate = today.subtract(Duration(days: today.weekday - 1));
59       break;
59
60     case DateRangeOption.thisMonth:
61       startDate = DateTime(today.year, today.month, 1);
62       break;
63     case DateRangeOption.lastMonth:
64       startDate = DateTime(today.year, today.month - 1, 1);
65       endDate = DateTime(
66         today.year,
67         today.month,
68         1,
69       ).subtract(const Duration(days: 1));
70       break;
71     case DateRangeOption.last6Months:
72       startDate = DateTime(today.year, today.month - 6, today.day);
73       break;
74     case DateRangeOption.thisYear:
75       startDate = DateTime(today.year, 1, 1);
76       break;
77     case DateRangeOption.lastYear:
77
In 63, Col 4 (258 selected) Spaces: 2 0TF-6 L7 Dart PGT ANTU (android XDP)
```

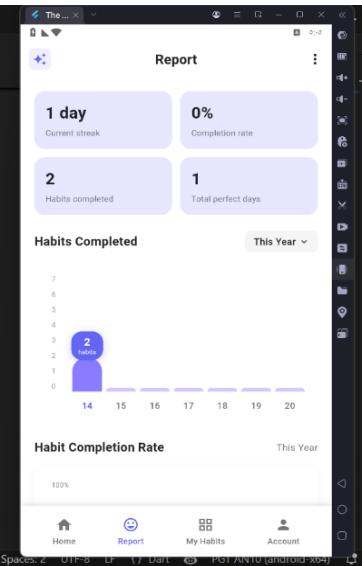
## TC009



The screenshot shows the 'Report' page with the following data:

- Current streak:** 1 day (1% completion rate)
- Habits completed:** 2 (1 total perfect days)
- Habits Completed chart:** A bar chart for the last 6 months showing 2 habits completed.
- Habit Completion Rate chart:** A line chart for the last 6 months showing a completion rate of 100%.

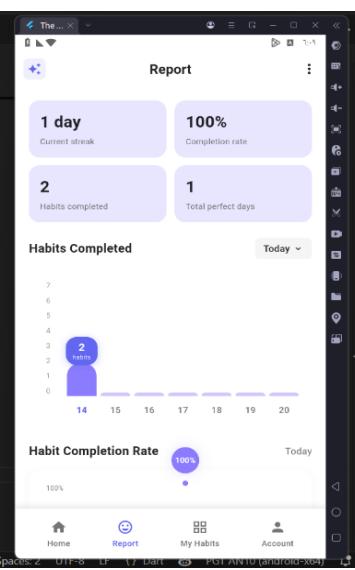
## TC010



The screenshot shows the 'Report' page with the following data:

- Current streak:** 1 day (0% completion rate)
- Habits completed:** 2 (1 total perfect days)
- Habits Completed chart:** A bar chart for this year showing 2 habits completed.
- Habit Completion Rate chart:** A line chart for this year showing a completion rate of 100%.

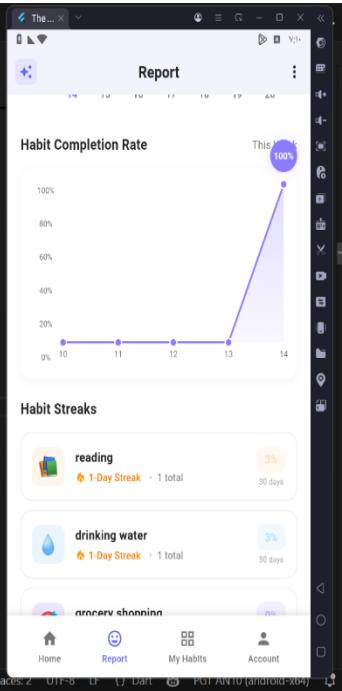
## TC011



The screenshot shows the 'Report' page with the following data:

- Current streak:** 1 day (100% completion rate)
- Habits completed:** 2 (1 total perfect days)
- Habits Completed chart:** A bar chart for today showing 2 habits completed.
- Habit Completion Rate chart:** A line chart for today showing a completion rate of 100%.

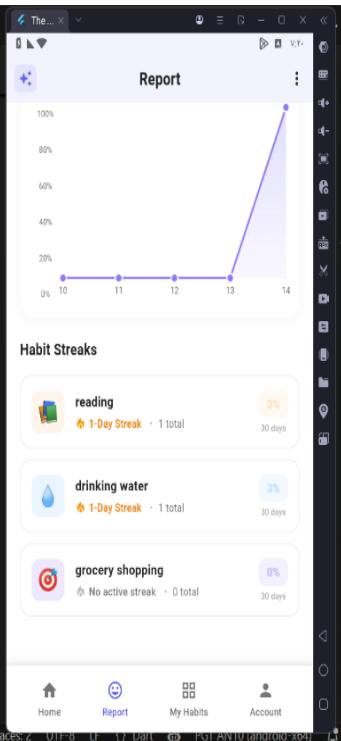
## TC012



The screenshot shows a mobile application interface for a habit tracking app. At the top, there are three tabs: 'archive\_page.dart', 'home\_page.dart', and 'report\_page.dart' (which is currently active). Below the tabs is a code editor window displaying Dart code for calculating completion rates and streaks. The code uses DateTime objects to determine the current month and day, then iterates through days to calculate completed and total counts. On the right side of the screen is the app's user interface. It features a header with a '+' button and the word 'Report'. Below the header is a chart titled 'Habit Completion Rate' showing a sharp increase from 0% at day 10 to 100% at day 14. A callout bubble points to the final data point at day 14. Below the chart is a section titled 'Habit Streaks' which lists three habits: 'reading', 'drinking water', and 'grocery shopping', each with a 1-day streak and a 30-day total. At the bottom of the screen are navigation icons for Home, Report (which is highlighted), My Habits, and Account.

```
lib > features > habits > presentation > pages > report_page.dart > _ReportPageState > _calculateCompletionRateChartData
25 class _ReportPageState extends State<ReportPage> {
357   Future<List<Map<String, dynamic>>> _calculateCompletionRateChartData(
406     // For longer ranges, show monthly data
407     var currentMonth = DateTime(startDate.year, startDate.month, 1);
408     final endMonth = DateTime(endDate.year, endDate.month, 1);
409
410     while (currentMonth.isBefore(endMonth) ||
411           currentMonth.isAtSameMomentAs(endMonth)) {
412       final nextMonth = DateTime(
413         currentMonth.year,
414         currentMonth.month + 1,
415         1,
416       );
417       final monthEnd = nextMonth.isBefore(today)
418         ? nextMonth
419         : DateTime.now();
420
421       int monthCompleted = 0;
422       int monthTotal = 0;
423       var currentDate = currentMonth;
424
425       while (currentDate.isBefore(monthEnd) &&
426             !currentDate.isAfter(endDate)) {
427         final dayStats = await repository.getDailyStatistics(currentDate);
428         monthCompleted += dayStats['completed'] as int;
429         monthTotal += dayStats['total'] as int;
430         currentDate = currentDate.add(const Duration(days: 1));
431       }
432     }
433   }
434 }
```

## TC013



The screenshot shows a mobile application interface for a habit tracking app. At the top, there are four tabs: 'archive\_page.dart', 'home\_page.dart', 'report\_page.dart' (which is currently active), and 'edit\_habit\_page.dart'. Below the tabs is a code editor window displaying Dart code for counting consecutive scheduled completions. The code defines a method '\_countConsecutiveScheduledCompletions' that takes a HabitModel, a start date, and a set of completion dates. It initializes a count to 0 and a check date to the start date. It then enters a loop that continues until it reaches a maximum iteration limit of 10,000. Inside the loop, it checks if the current date is scheduled for the habit. If it is, it increments the count. If it's not, it breaks the loop. If the habit has a daily repeat type, it subtracts one day from the check date and continues. If it's weekly/monthly, it skips non-scheduled dates and continues. After the loop, it checks if the date is scheduled and was completed. The right side of the screen shows the app's UI. It has a header with a '+' button and the word 'Report'. Below the header is a chart titled 'Habit Completion Rate' showing a sharp increase from 0% at day 10 to 100% at day 14. A callout bubble points to the final data point at day 14. Below the chart is a section titled 'Habit Streaks' which lists three habits: 'reading', 'drinking water', and 'grocery shopping'. 'reading' and 'drinking water' both have a 1-day streak and a 30-day total. 'grocery shopping' has no active streak and 0 total. At the bottom are navigation icons for Home, Report (highlighted), My Habits, and Account.

```
lib > features > habits > presentation > pages > report_page.dart > _ReportPageState > _countConsecutiveScheduledCompletions
25 class _ReportPageState extends State<ReportPage> {
316
317   /// Count consecutive scheduled completions going backwards from a start date
318   int _countConsecutiveScheduledCompletions(
319     HabitModel habit,
320     DateTime startDate,
321     Set<DateTime> completionsByDay,
322   ) {
323     int count = 0;
324     var checkDate = startDate;
325     int maxIterations = 10000; // Safety limit
326     int iterations = 0;
327
328     while (iterations < maxIterations) {
329       // Check if this date is scheduled for the habit
330       if (!_isDateScheduled(habit, checkDate)) {
331         // This date is not scheduled
332         // For daily habits, any gap breaks the streak
333         // For weekly/monthly, skip non-scheduled dates and continue
334         if (habit.repeatType == RepeatType.daily) {
335           break;
336         } else {
337           checkDate = checkDate.subtract(const Duration(days: 1));
338           iterations++;
339           continue;
340         }
341       }
342
343       // This date is scheduled, check if it was completed
344       if (completionsByDay.contains(checkDate)) {
345         count++;
346       }
347     }
348   }
349 }
```

# **Conclusion**

The Daily Habit Tracker project successfully demonstrated how structured planning, teamwork, and iterative development can lead to the creation of a practical and user-centered application. Throughout Sprint 0, Sprint 1, and Sprint 2, the team applied Agile principles to define requirements, design system components, implement core functionalities, and validate the system through comprehensive testing.

By completing features such as habit creation, habit tracking, habit editing, and detailed progress reporting, the project provides a solid foundation for users seeking to build and maintain positive habits. The collaborative efforts of all team members contributed to smooth progress and ensured that each sprint delivered meaningful improvements.

Overall, the project achieved its planned goals for the early development stages and established a clear roadmap for future enhancements. With additional refinements, the Daily Habit Tracker has strong potential to evolve into a fully functional wellness tool that supports long-term behavior change and encourages personal growth.

## **Tools setup :**

- Design tool: draw.io
- Communication tool: WhatsApp - TeamViewer
- Repository tool: Github
- Agile planning tool: JIRA
- development tool: Visual Studio Code (VS Code)
- documents: Microsoft Word

## **Github:**

<https://github.com/Haalasmarei17/Daily-Habit-Tracker-Application.git>