جامعة جدة
University of Jeddah

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:

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:

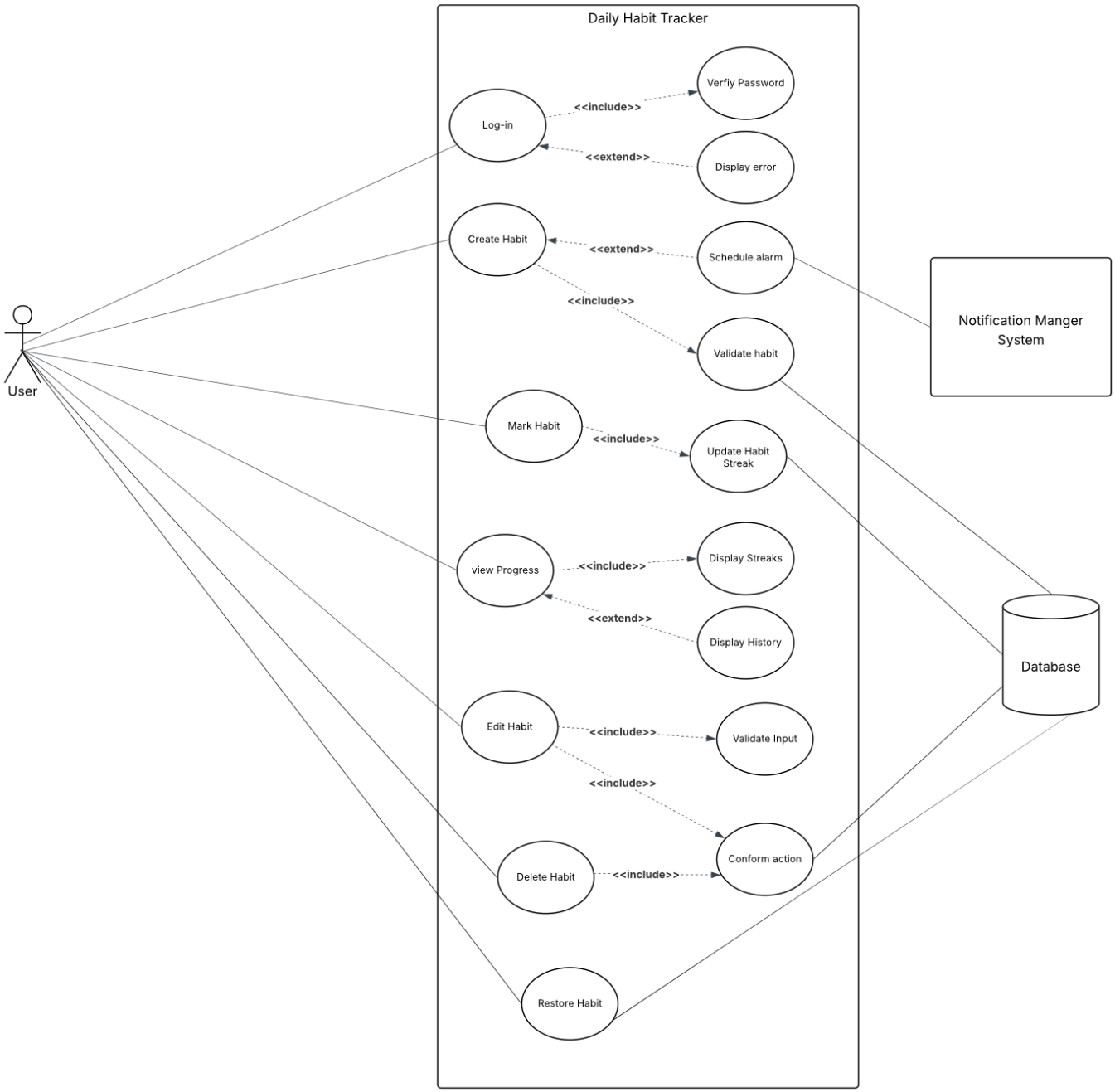
- Design tool: draw.io
- Communication tool: WhatsApp - TeamViewer
- Repository tool: Github
- Agile planning tool: JIRA
- Development tool: Visual Studio Code (VS Code)
- Documents: Microsoft Word

Functional and non-functional requirements

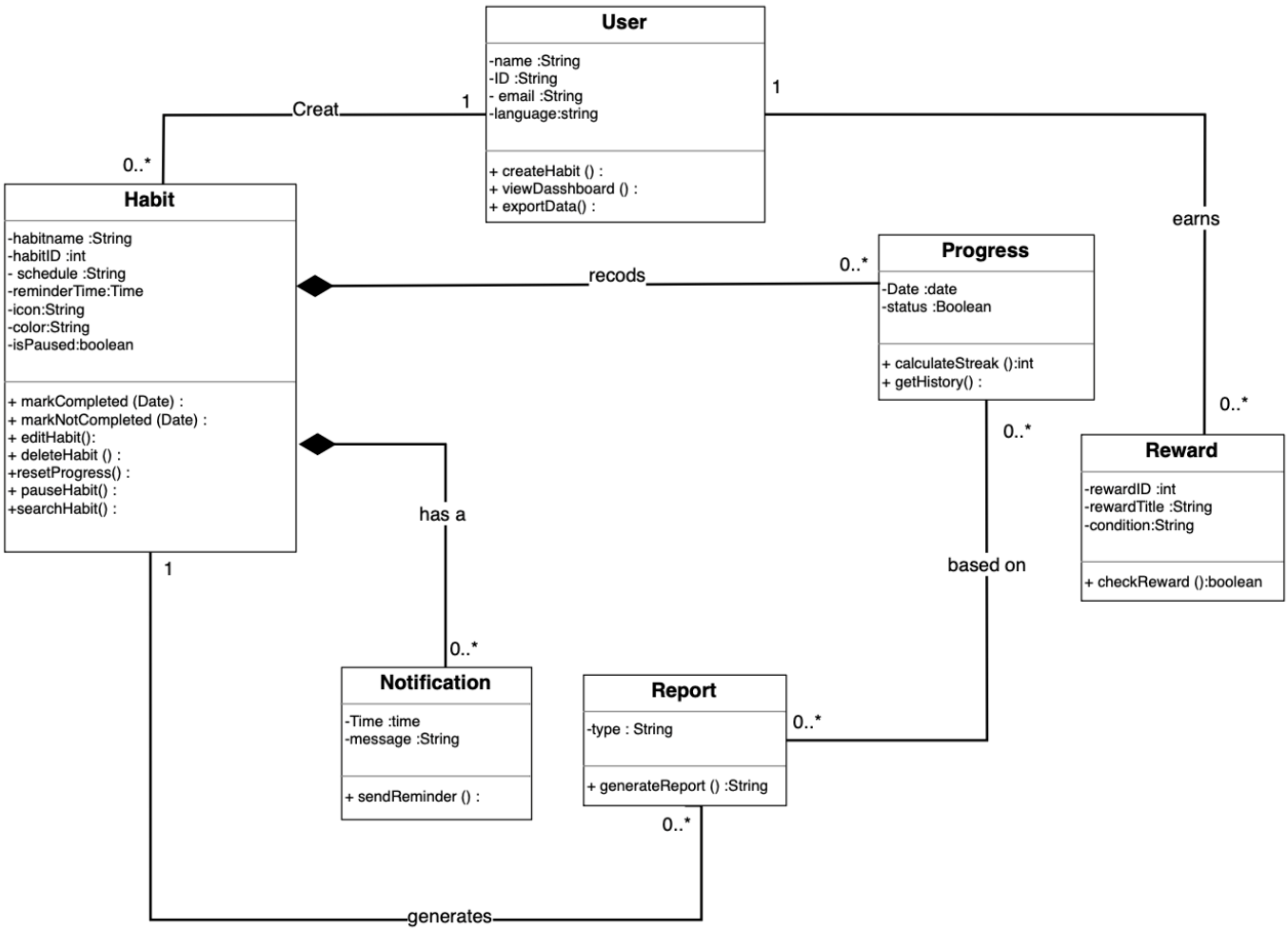
ID	Requirement Definition
FR1	The system shall allow the user to create a new habit by entering the habit name, schedule, and optional reminder time.
FR2	The system shall allow the user to mark a habit as completed or not completed for the current day.
FR4	The system shall send a notification to the 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:

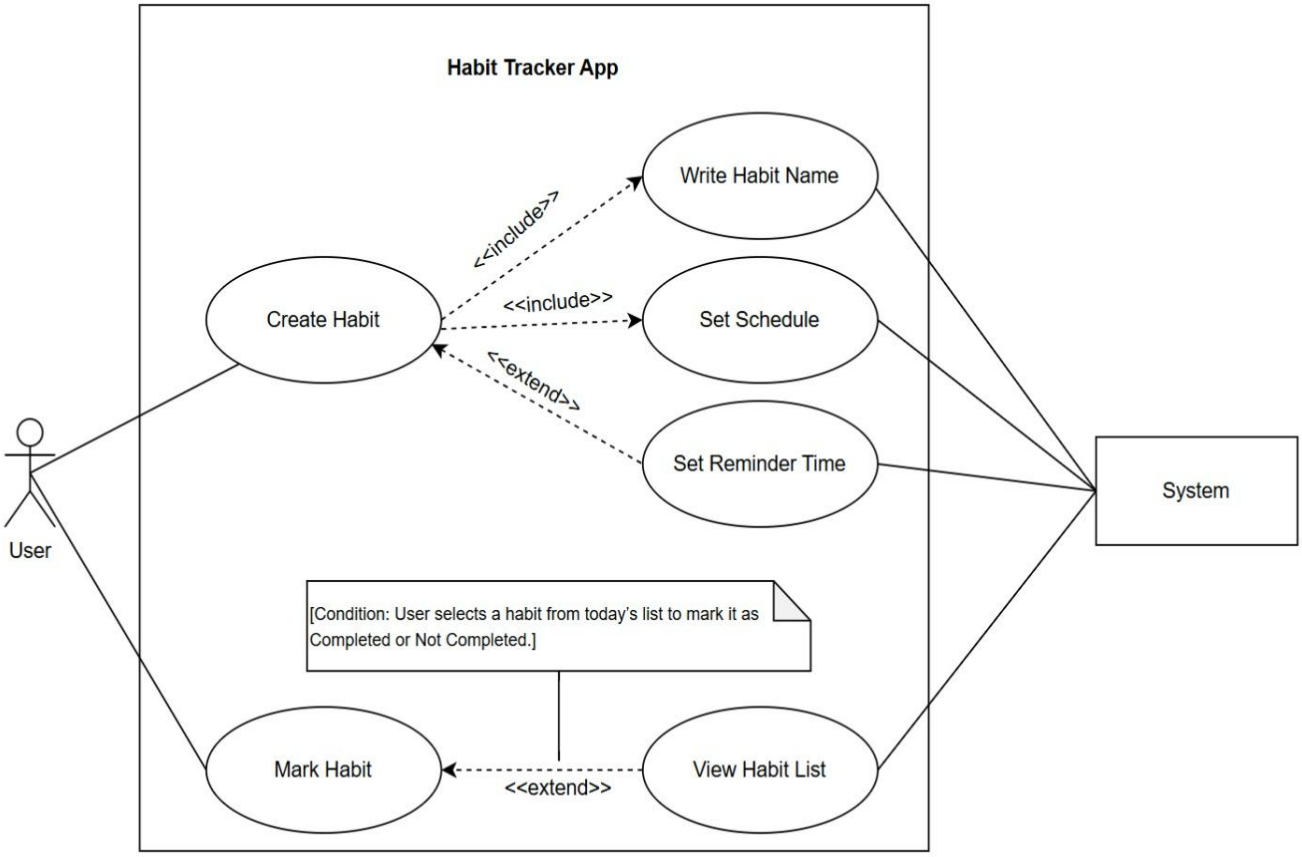
The team showed great task distribution and good organization

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

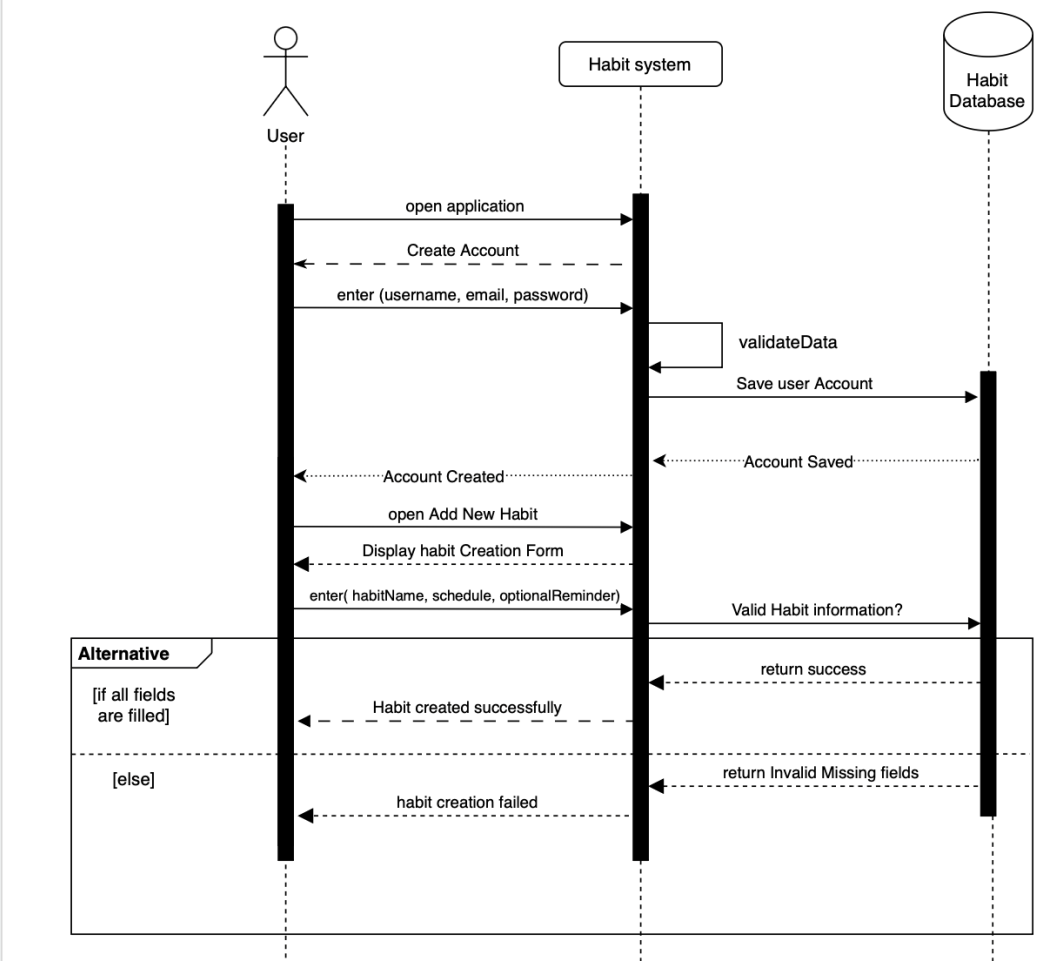
Story short description: As a user, I want to be able to create new habits which will allow me to track my progress and determination.

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

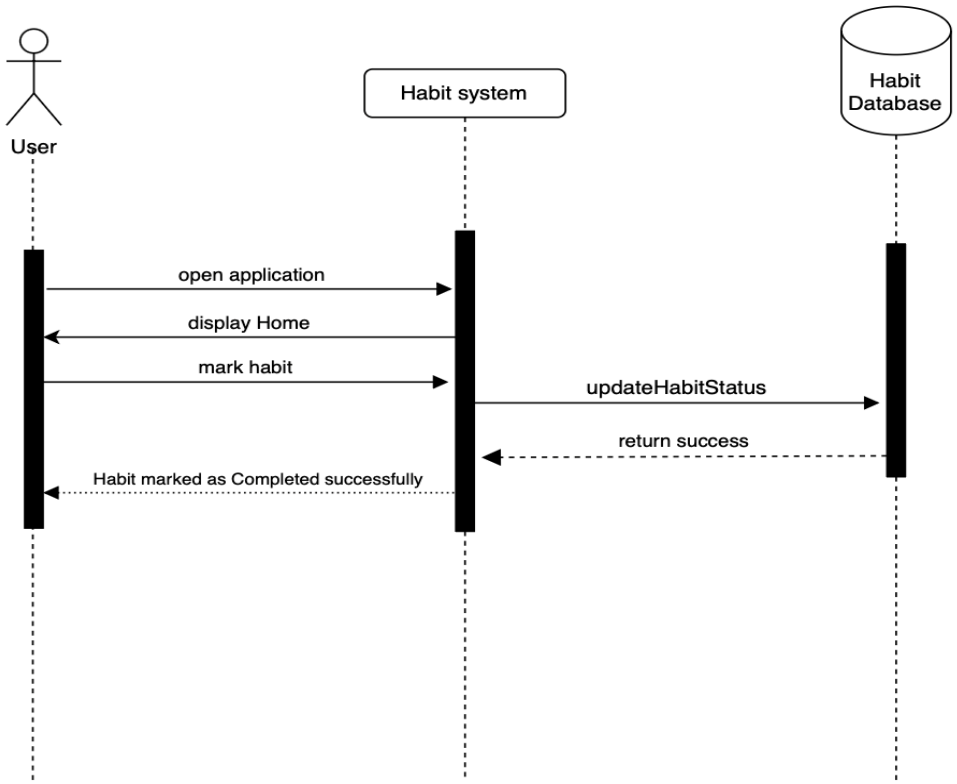
Story Sequence No: 02

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 proccess 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]

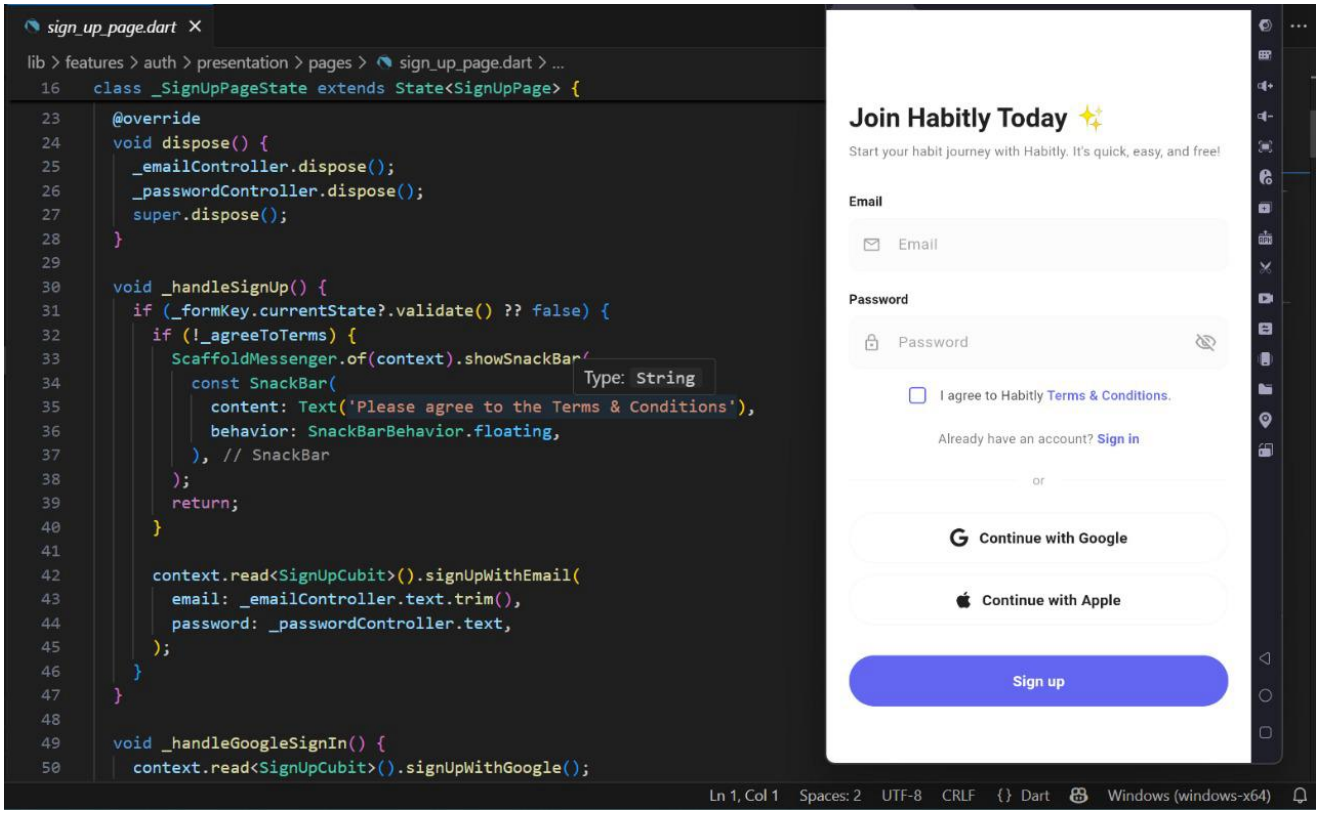
Test Case Name: [Sprint 1 – Create New Habit]
Test Case Id: [Daily Habit Tracker – Create New Habit]

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]

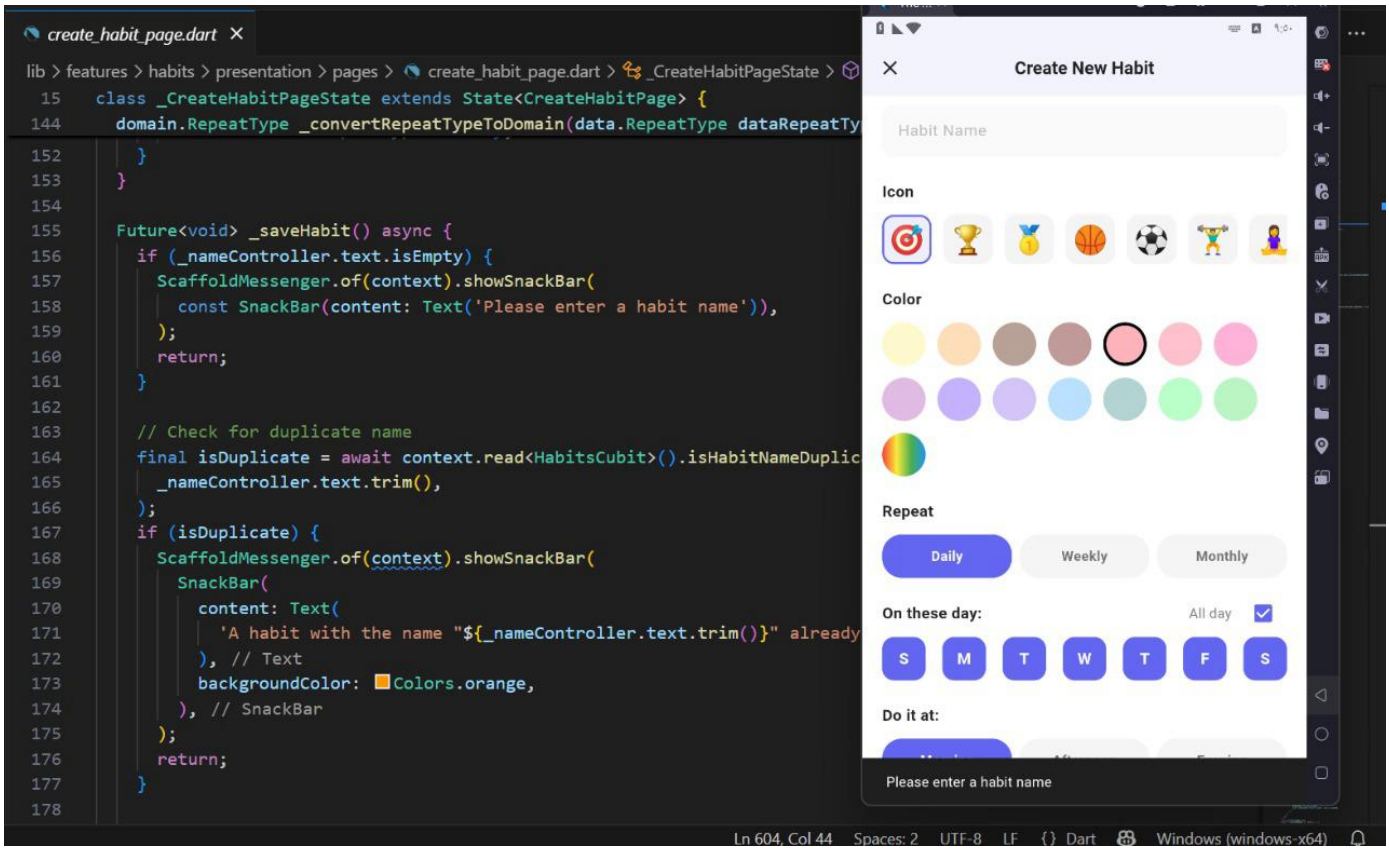
Test Case No.	Test Case Description	Expected Results	Outcome (Pass/Fail/Other (Comments)
TC001	User selects a habit and marks it as "Completed" for today.	System should display a check mark or success message: "Habit marked as completed."	Pass
TC002	The user can record the habit status as "Not Completed."	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

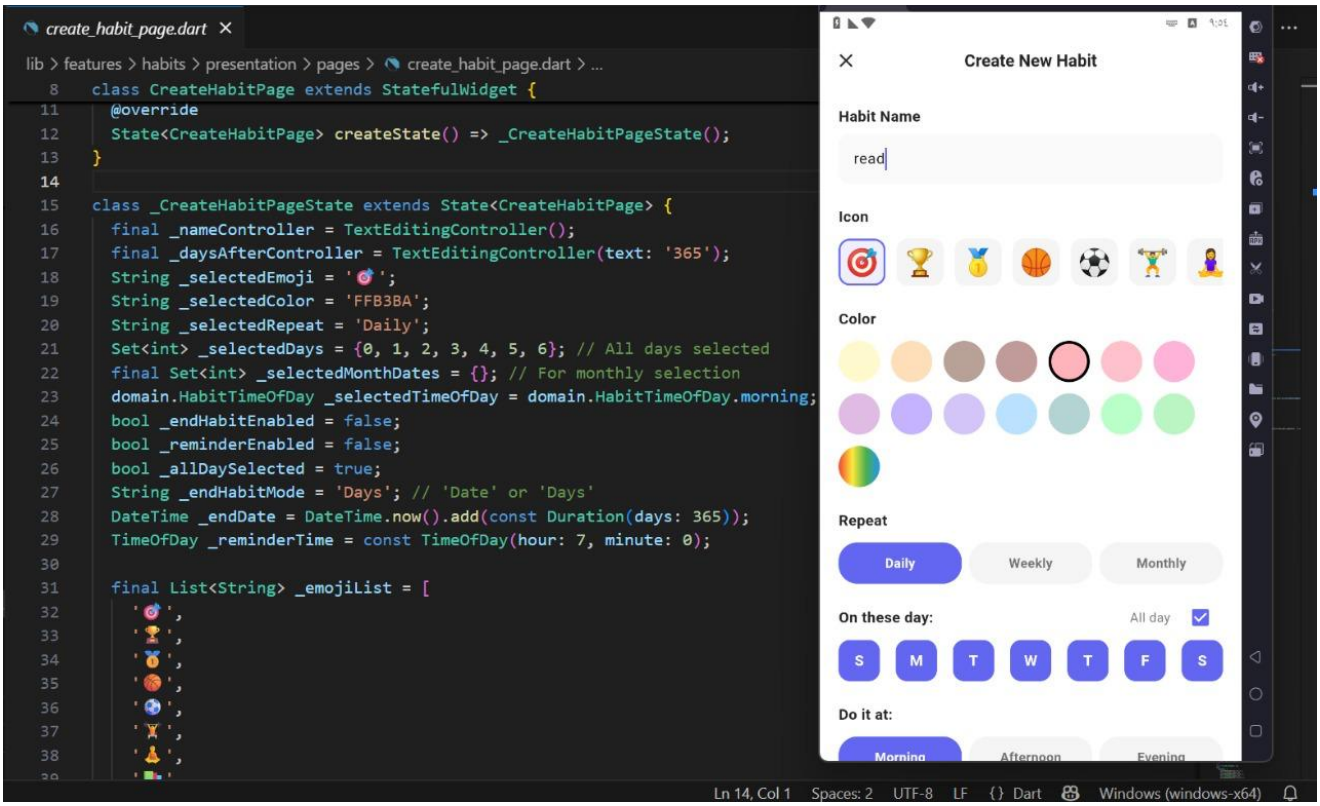


Test Case Name: Create New Habit

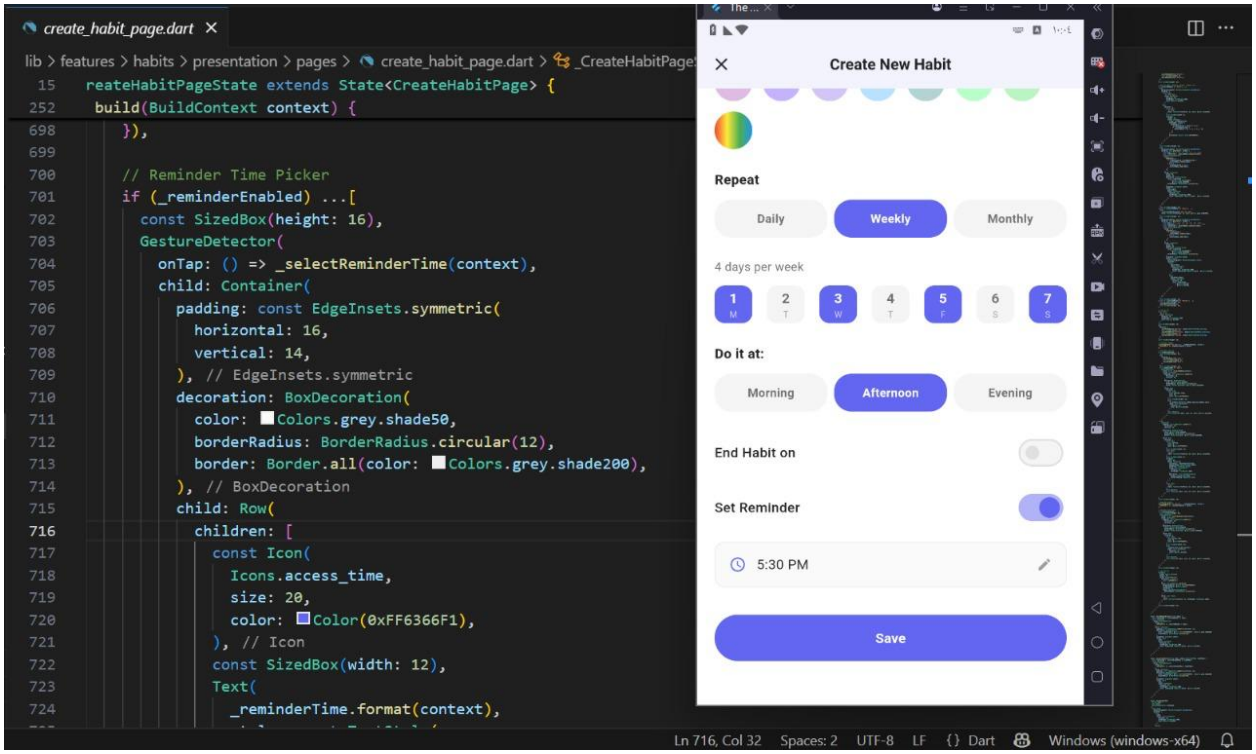
TC001



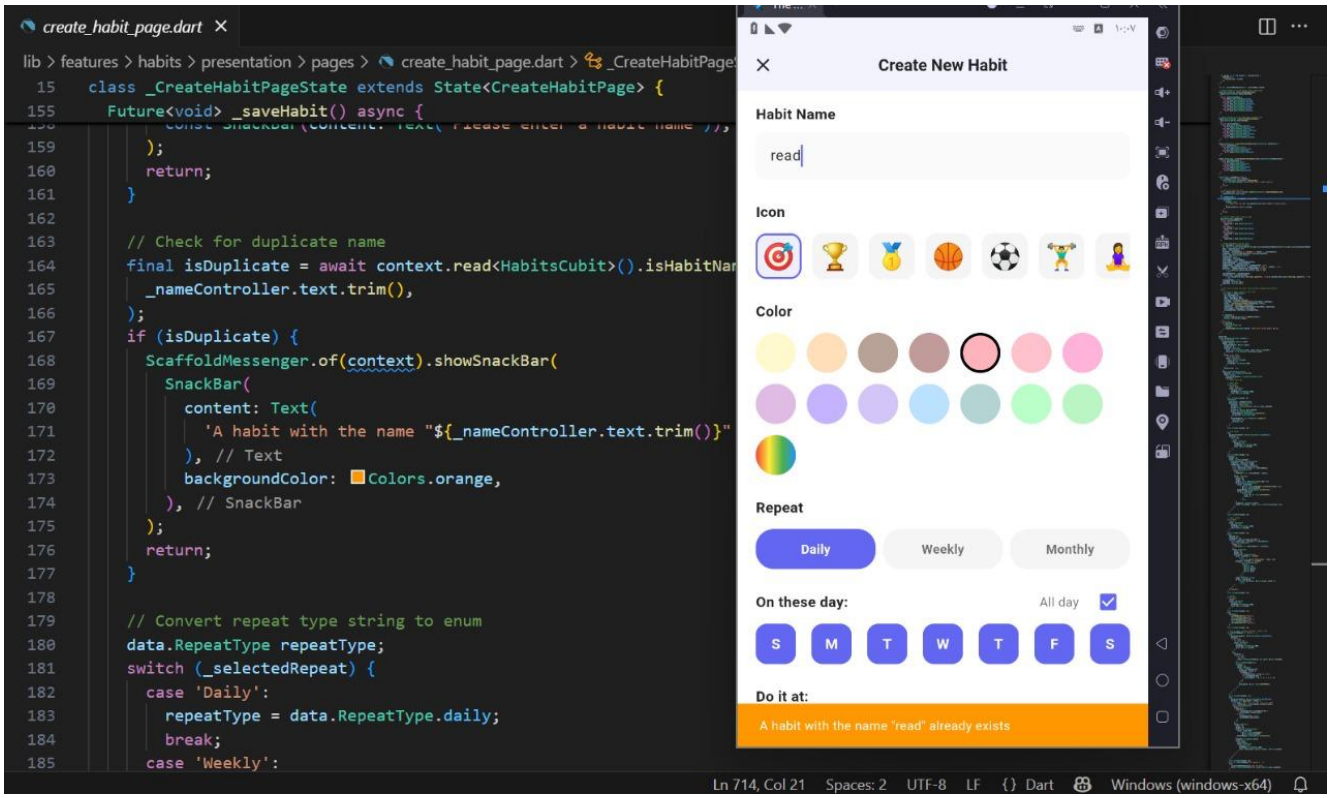
TC002

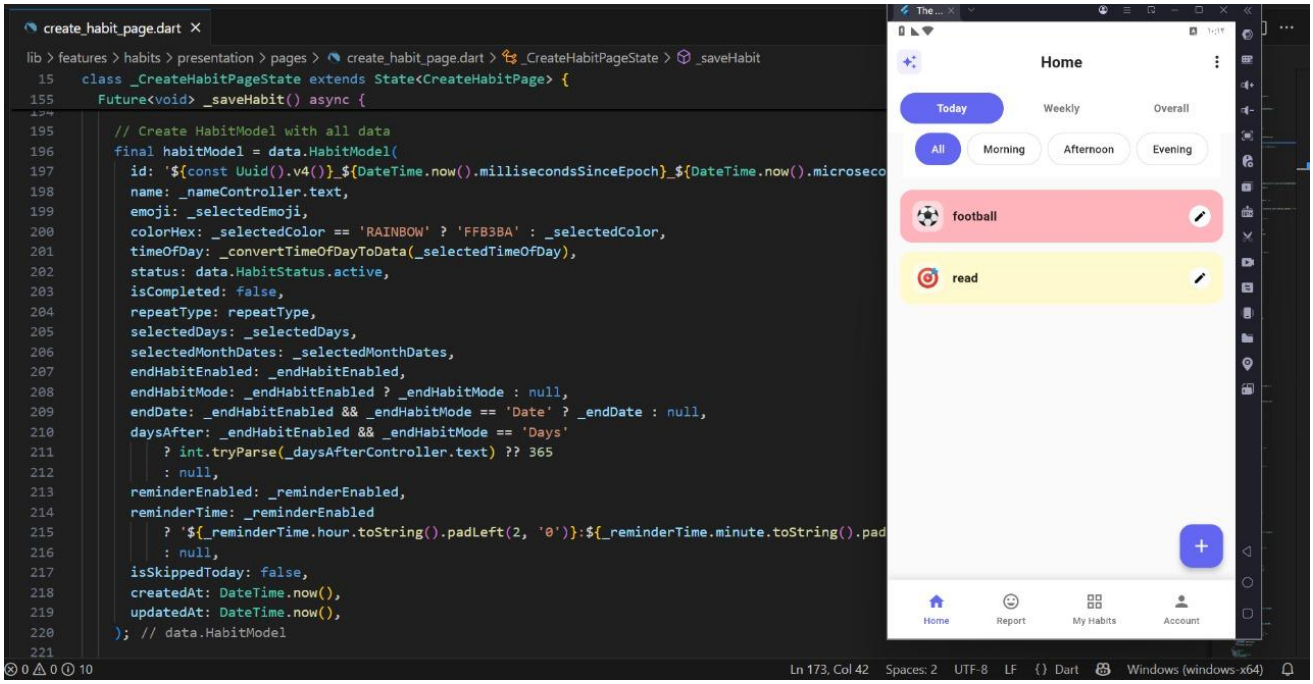

create_habit_page.dart
lib > features > habits > presentation > pages > create_habit_page.dart > ...
class CreateHabitPage extends StatefulWidget {
@override
State<CreateHabitPage> createState() => _CreateHabitPageState();
}
class _CreateHabitPageState extends State<CreateHabitPage> {
final _nameController = TextEditingController();
final _daysAfterController = TextEditingController(text: '365');
String _selectedEmoji = '🎯';
String _selectedColor = 'FFB3BA';
String _selectedRepeat = 'Daily';
Set<int> _selectedDays = {0, 1, 2, 3, 4, 5, 6}; // All days selected
final Set<int> _selectedMonthDates = {}; // For monthly selection
domain.HabitTimeOfDay _selectedTimeOfDay = domain.HabitTimeOfDay.morning;
bool _endHabitEnabled = false;
bool _reminderEnabled = false;
bool _allDaySelected = true;
String _endHabitMode = 'Days'; // 'Date' or 'Days'
DateTime _endDate = DateTime.now().add(const Duration(days: 365));
TimeOfDay _reminderTime = const TimeOfDay(hour: 7, minute: 0);
final List<String> _emojiList = [
Create New Habit
Habit Name
read
Icon
Color
Repeat
Daily
Weekly
Monthly
On these day:
All day
S
M
T
W
T
F
S
Do it at:
Morning
Afternoon
Evening
Ln 14, Col 1
Spaces: 2
UTF-8
LF
{} Dart
Windows (windows-x64)

TC003


create_habit_page.dart
lib > features > habits > presentation > pages > create_habit_page.dart > _CreateHabitPage
reateHabitPageState extends State<CreateHabitPage> {
build(BuildContext context) {
}),
// Reminder Time Picker
if (_reminderEnabled) ...[
const SizedBox(height: 16),
GestureDetector(
onTap: () => _selectReminderTime(context),
child: Container(
padding: const EdgeInsets.symmetric(
horizontal: 16,
vertical: 14,
), // EdgeInsets.symmetric
decoration: BoxDecoration(
color: Colors.grey.shade50,
borderRadius: BorderRadius.circular(12),
border: Border.all(color: Colors.grey.shade200),
), // BoxDecoration
child: Row(
children: [
const Icon(
Icons.access_time,
size: 20,
color: Color(0xFF6366F1),
), // Icon
const SizedBox(width: 12),
Text(
_reminderTime.format(context),
Create New Habit
Repeat
Daily
Weekly
Monthly
4 days per week
1
2
3
4
5
6
7
Do it at:
Morning
Afternoon
Evening
End Habit on
Set Reminder
5:30 PM
Save
Ln 716, Col 32
Spaces: 2
UTF-8
LF
{} Dart
Windows (windows-x64)

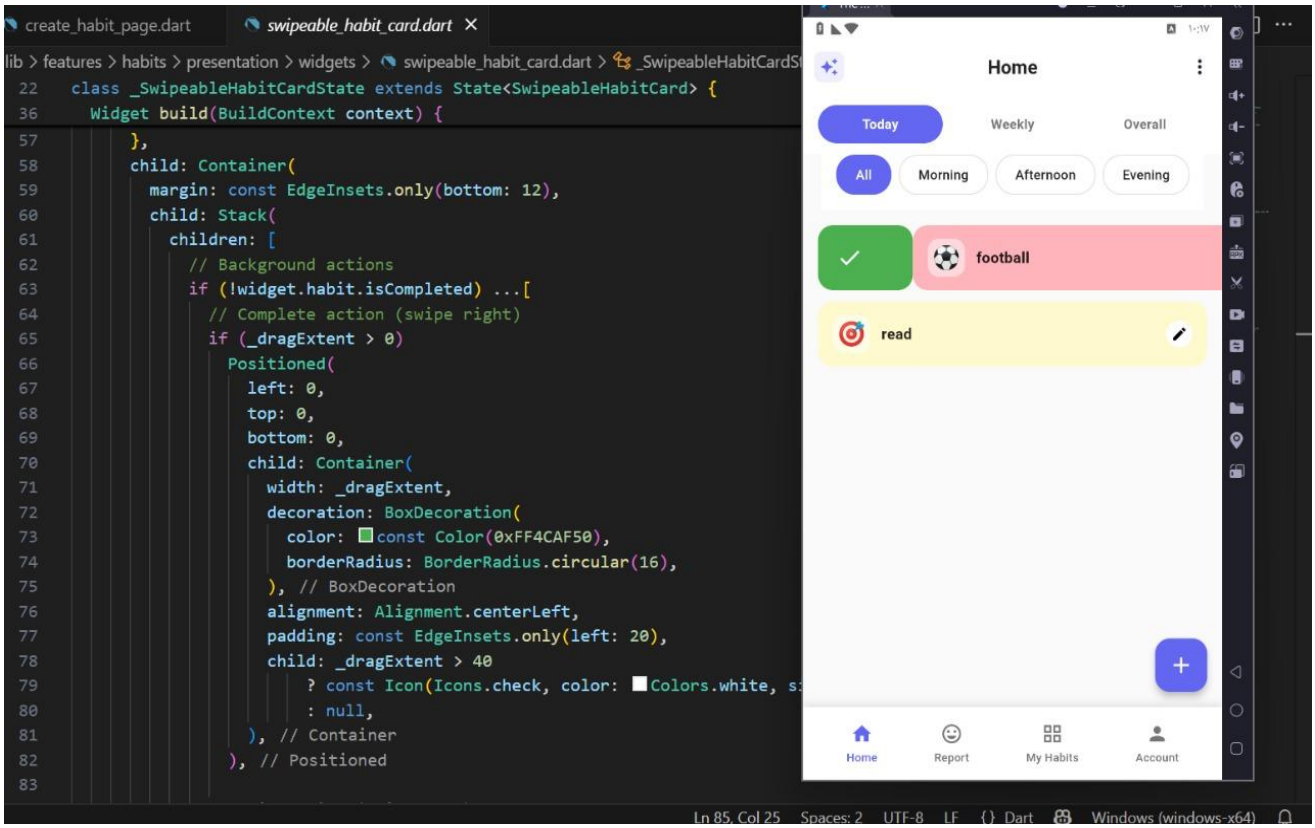
TC004

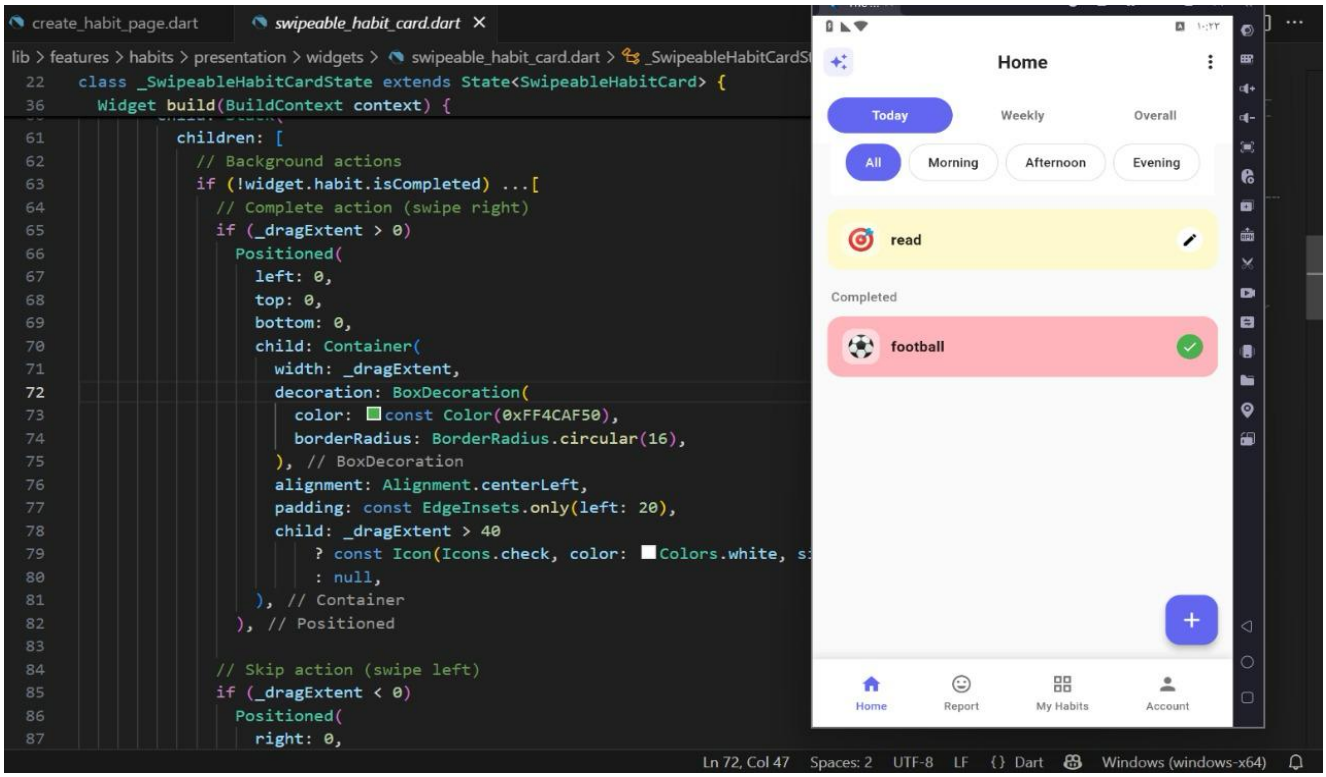

create_habit_page.dart
lib > features > habits > presentation > pages > create_habit_page.dart > _CreateHabitPage
class _CreateHabitPageState extends State<CreateHabitPage> {
Future<void> _saveHabit() async {
);
return;
}
// Check for duplicate name
final isDuplicate = await context.read<HabitsCubit>().isHabitNar
_nameController.text.trim(),
);
if (isDuplicate) {
ScaffoldMessenger.of(context).showSnackBar(
SnackBar(
content: Text(
'A habit with the name "${_nameController.text.trim()}"
), // Text
backgroundColor: Colors.orange,
), // SnackBar
);
return;
}
// Convert repeat type string to enum
data.RepeatType repeatType;
switch (_selectedRepeat) {
case 'Daily':
repeatType = data.RepeatType.daily;
break;
case 'Weekly':
Create New Habit
Habit Name
read
Icon
Color
Repeat
Daily
Weekly
Monthly
On these day:
All day
S
M
T
W
T
F
S
Do it at:
A habit with the name "read" already exists
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
class _CreateHabitPageState extends State<CreateHabitPage> {
Future<void> _saveHabit() async {
// Create HabitModel with all data
final habitModel = data.HabitModel(
id: '${const Uuid().v4()}_${DateTime.now().millisecondsSinceEpoch}_${DateTime.now().microseco
name: _nameController.text,
emoji: _selectedEmoji,
colorHex: _selectedColor == 'RAINBOW' ? 'FFB3BA' : _selectedColor,
timeOfDay: _convertTimeOfDayToData(_selectedTimeOfDay),
status: data.HabitStatus.active,
isCompleted: false,
repeatType: repeatType,
selectedDays: _selectedDays,
selectedMonthDates: _selectedMonthDates,
endHabitEnabled: _endHabitEnabled,
endHabitMode: _endHabitEnabled ? _endHabitMode : null,
endDate: _endHabitEnabled && _endHabitMode == 'Date' ? _endDate : null,
daysAfter: _endHabitEnabled && _endHabitMode == 'Days'
? int.tryParse(_daysAfterController.text) ?? 365
: null,
reminderEnabled: _reminderEnabled,
reminderTime: _reminderEnabled
? '${_reminderTime.hour.toString().padLeft(2, '0')}:${_reminderTime.minute.toString().pad
: null,
isSkippedToday: false,
createdAt: DateTime.now(),
updatedAt: DateTime.now(),
); // data.HabitModel
Home
Today
Weekly
Overall
All
Morning
Afternoon
Evening
football
read
Home
Report
My Habits
Account
Ln 173, Col 42
Spaces: 2
UTF-8
LF
{} Dart
Windows (windows-x64)

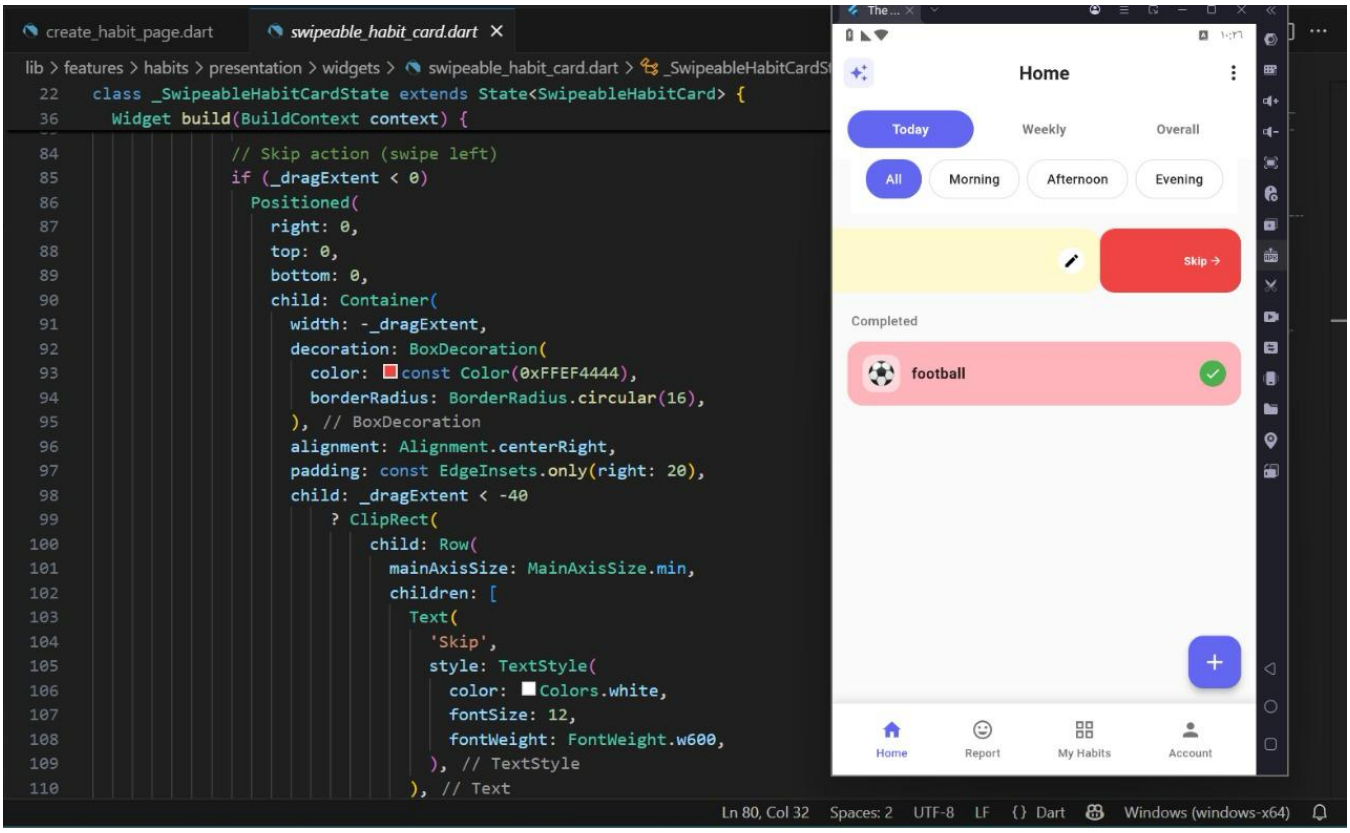
Test Case Name: Mark Habit Status

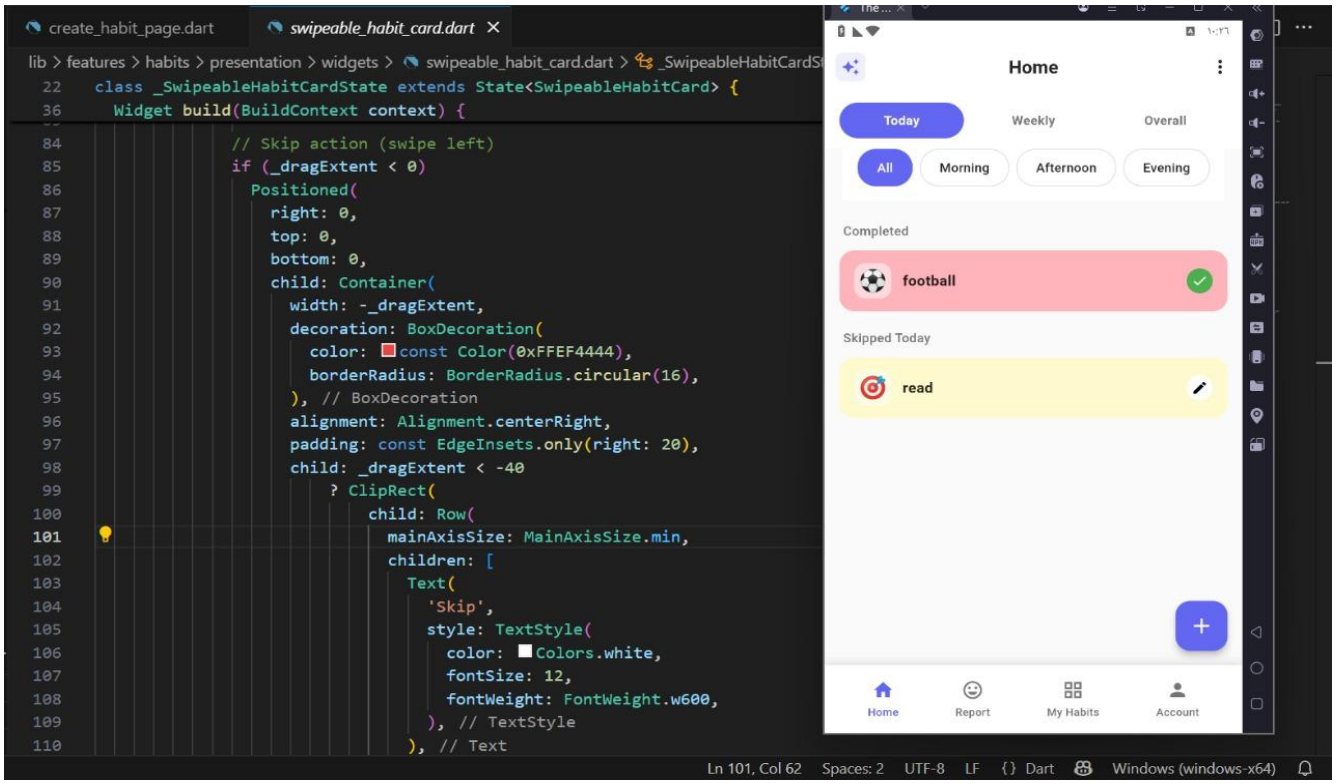
TC001



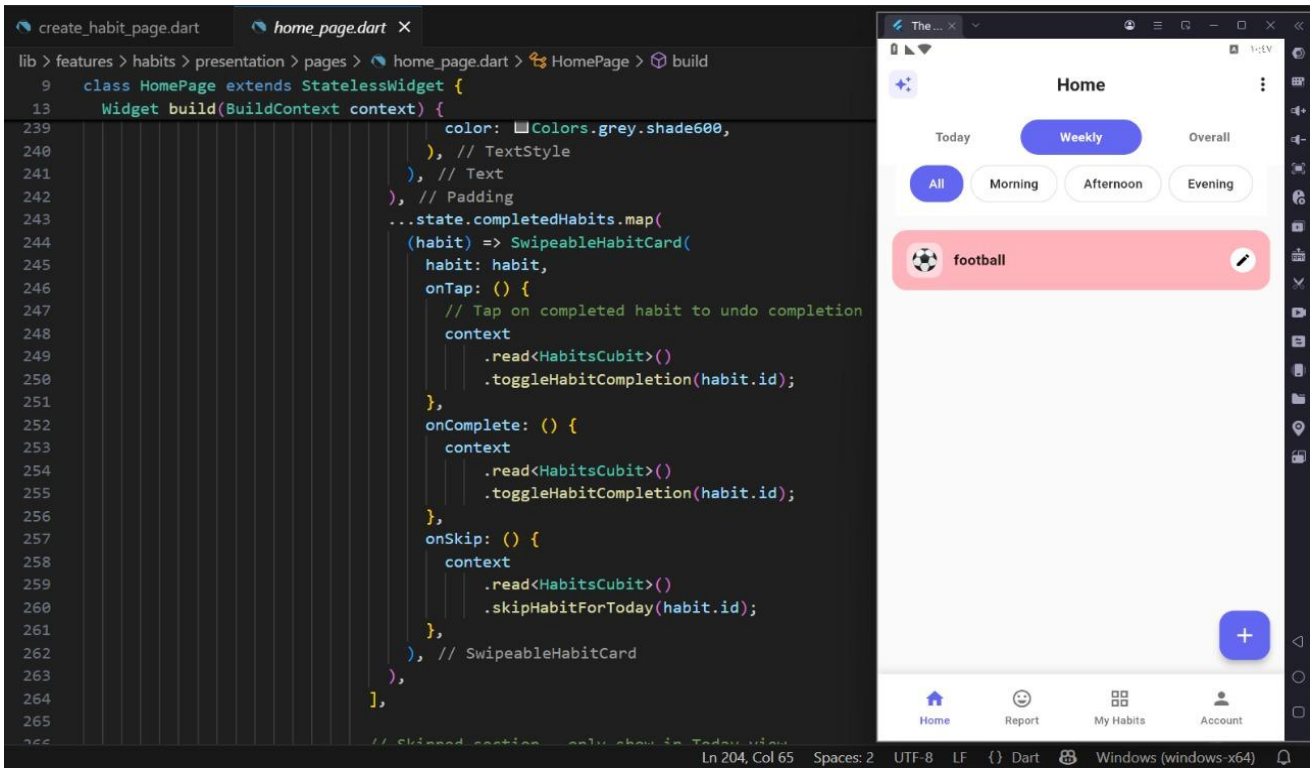


TC002

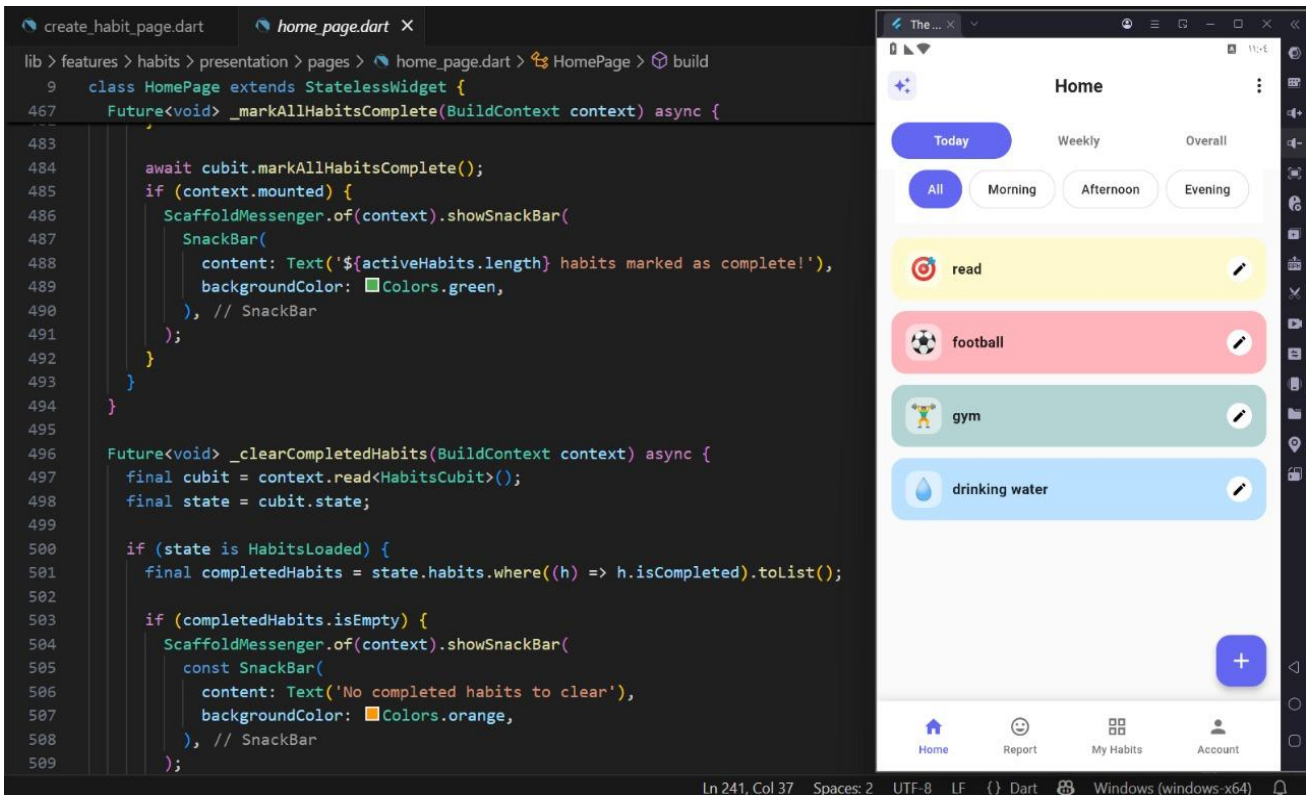

create_habit_page.dart
swipeable_habit_card.dart
lib > features > habits > presentation > widgets > swipeable_habit_card.dart > _SwipeableHabitCardSt
class _SwipeableHabitCardState extends State<SwipeableHabitCard> {
Widget build(BuildContext context) {
// Skip action (swipe left)
if (_dragExtent < 0)
Positioned(
right: 0,
top: 0,
bottom: 0,
child: Container(
width: -_dragExtent,
decoration: BoxDecoration(
color: const Color(0xFFEF4444),
borderRadius: BorderRadius.circular(16),
), // BoxDecoration
alignment: Alignment.centerRight,
padding: const EdgeInsets.only(right: 20),
child: _dragExtent < -40
? ClipRect(
child: Row(
mainAxisSize: MainAxisSize.min,
children: [
Text(
'Skip',
style: TextStyle(
color: Colors.white,
fontSize: 12,
fontWeight: FontWeight.w600,
), // TextStyle
), // Text
Home
Today
Weekly
Overall
All
Morning
Afternoon
Evening
Skip
Completed
football
Report
My Habits
Account
Ln 80, Col 32
Spaces: 2
UTF-8
LF
{} Dart
Windows (windows-x64)


create_habit_page.dart
swipeable_habit_card.dart
lib > features > habits > presentation > widgets > swipeable_habit_card.dart > _SwipeableHabitCardSt
class _SwipeableHabitCardState extends State<SwipeableHabitCard> {
Widget build(BuildContext context) {
// Skip action (swipe left)
if (_dragExtent < 0)
Positioned(
right: 0,
top: 0,
bottom: 0,
child: Container(
width: -_dragExtent,
decoration: BoxDecoration(
color: const Color(0xFFEF4444),
borderRadius: BorderRadius.circular(16),
), // BoxDecoration
alignment: Alignment.centerRight,
padding: const EdgeInsets.only(right: 20),
child: _dragExtent < -40
? ClipRect(
child: Row(
mainAxisSize: MainAxisSize.min,
children: [
Text(
'Skip',
style: TextStyle(
color: Colors.white,
fontSize: 12,
fontWeight: FontWeight.w600,
), // TextStyle
), // Text
Home
Today
Weekly
Overall
All
Morning
Afternoon
Evening
Completed
football
Skipped Today
read
Report
My Habits
Account
Ln 101, Col 62
Spaces: 2
UTF-8
LF
{} Dart
Windows (windows-x64)

TC003


create_habit_page.dart
home_page.dart
lib > features > habits > presentation > pages > home_page.dart > HomePage > build
class HomePage extends StatelessWidget {
Widget build(BuildContext context) {
color: Colors.grey.shade600,
), // TextStyle
), // Text
), // Padding
...state.completedHabits.map(
(habit) => SwipeableHabitCard(
habit: habit,
onTap: () {
// Tap on completed habit to undo completion
context
.read<HabitsCubit>()
.toggleHabitCompletion(habit.id);
},
onComplete: () {
context
.read<HabitsCubit>()
.toggleHabitCompletion(habit.id);
},
onSkip: () {
context
.read<HabitsCubit>()
.skipHabitForToday(habit.id);
},
), // SwipeableHabitCard
),
],
Home
Today
Weekly
Overall
All
Morning
Afternoon
Evening
football
Home
Report
My Habits
Account
Ln 204, Col 65
Spaces: 2
UTF-8
LF
{} Dart
Windows (windows-x64)

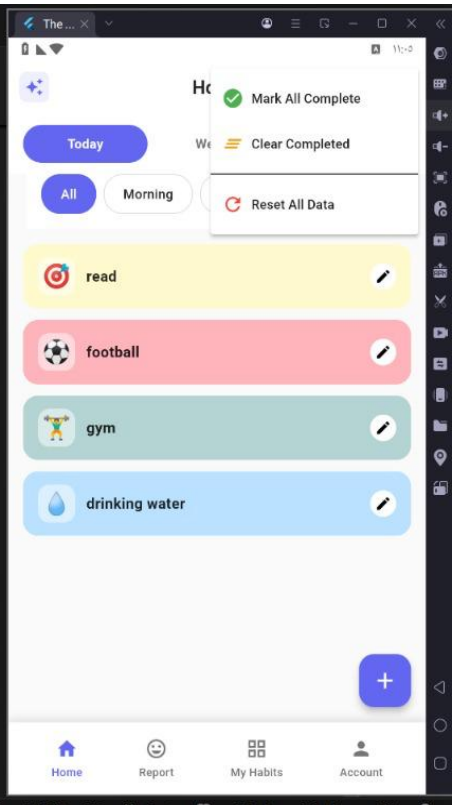
TC004


create_habit_page.dart
home_page.dart
lib > features > habits > presentation > pages > home_page.dart > HomePage > build
class HomePage extends StatelessWidget {
Future<void> _markAllHabitsComplete(BuildContext context) async {
await cubit.markAllHabitsComplete();
if (context.mounted) {
ScaffoldMessenger.of(context).showSnackBar(
SnackBar(
content: Text('${activeHabits.length} habits marked as complete!'),
backgroundColor: Colors.green,
), // SnackBar
);
}
}
}
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
);
Home
Today
Weekly
Overall
All
Morning
Afternoon
Evening
read
football
gym
drinking water
Home
Report
My Habits
Account
Ln 241, Col 37
Spaces: 2
UTF-8
LF
{} Dart
Windows (windows-x64)

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

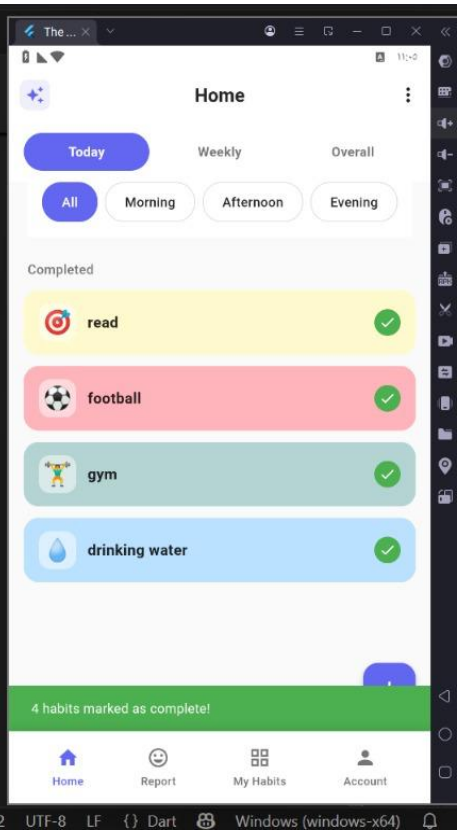


Ln 241, Col 37 Spaces: 2 UTF-8 LF {} Dart Windows (windows-x64)

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



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

Sprint 2

Sprint #2 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 #2 *initial* Meeting - [5/11/2025]

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:

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

Expected Outcomes

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

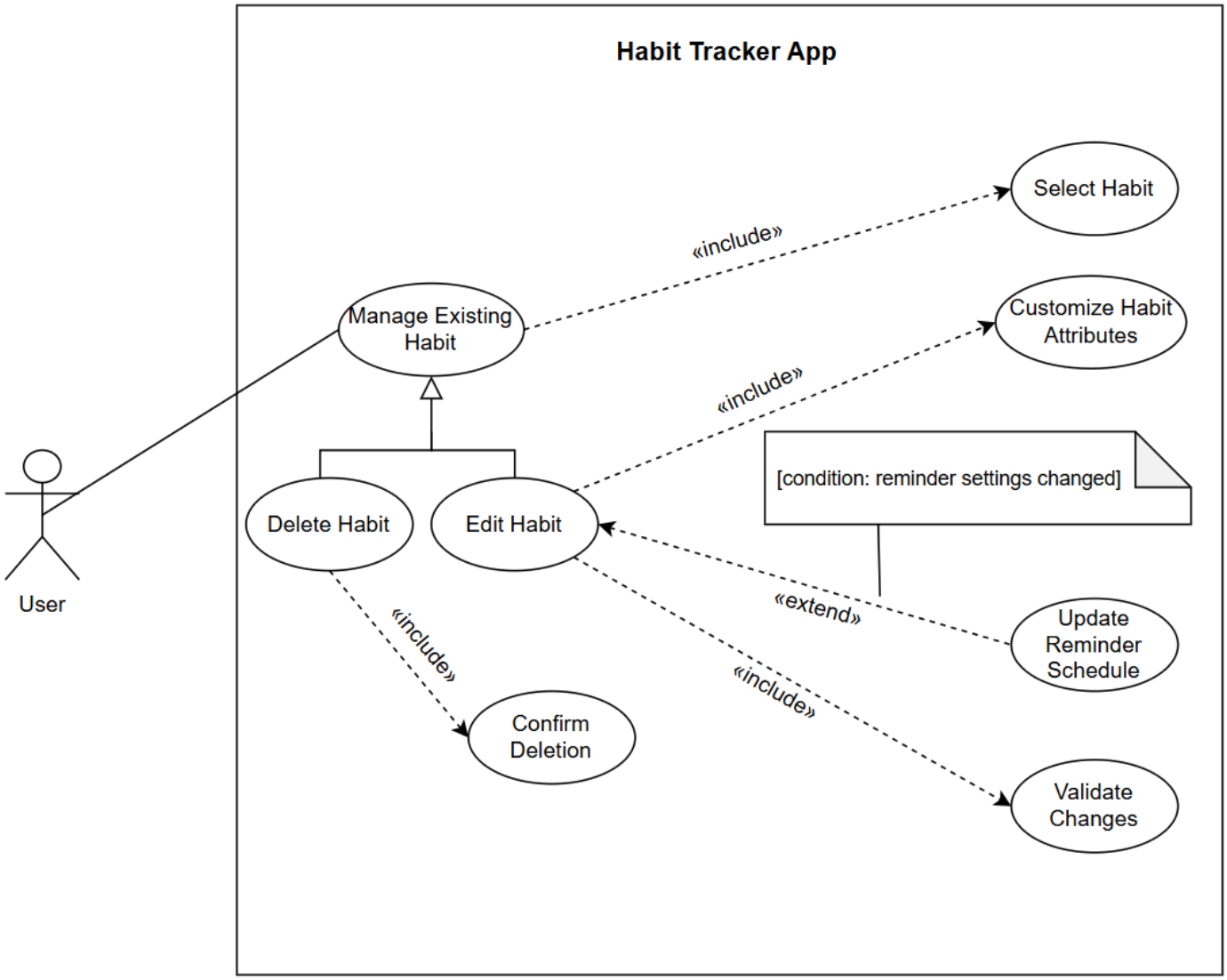
Scrum's Master comments based on the above questions:

The team demonstrated good planning and collaboration

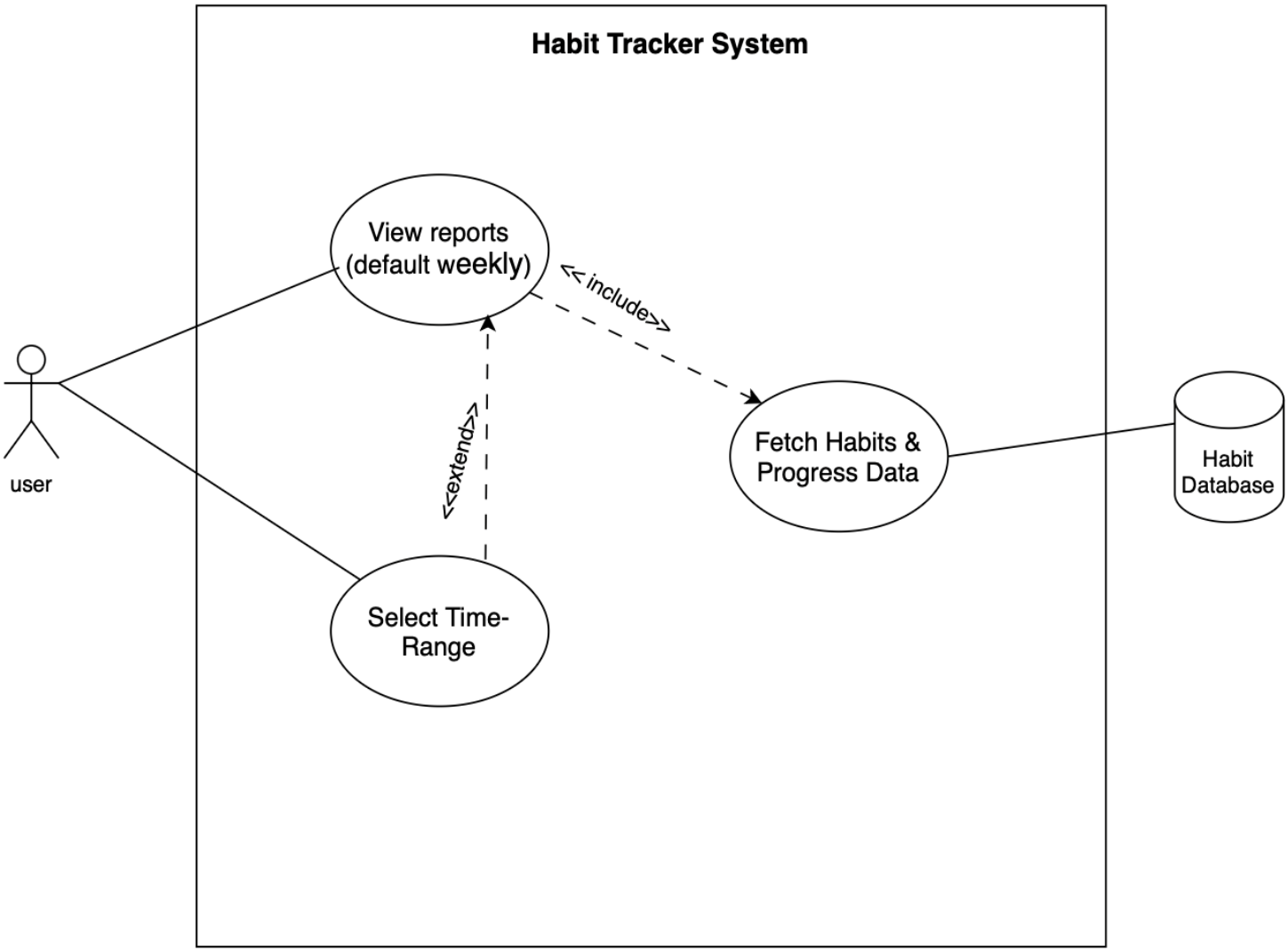
Our scrum master galla noted that the goals for this sprint were well defined and that the next step is to move forward with design and coding for both new features.

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

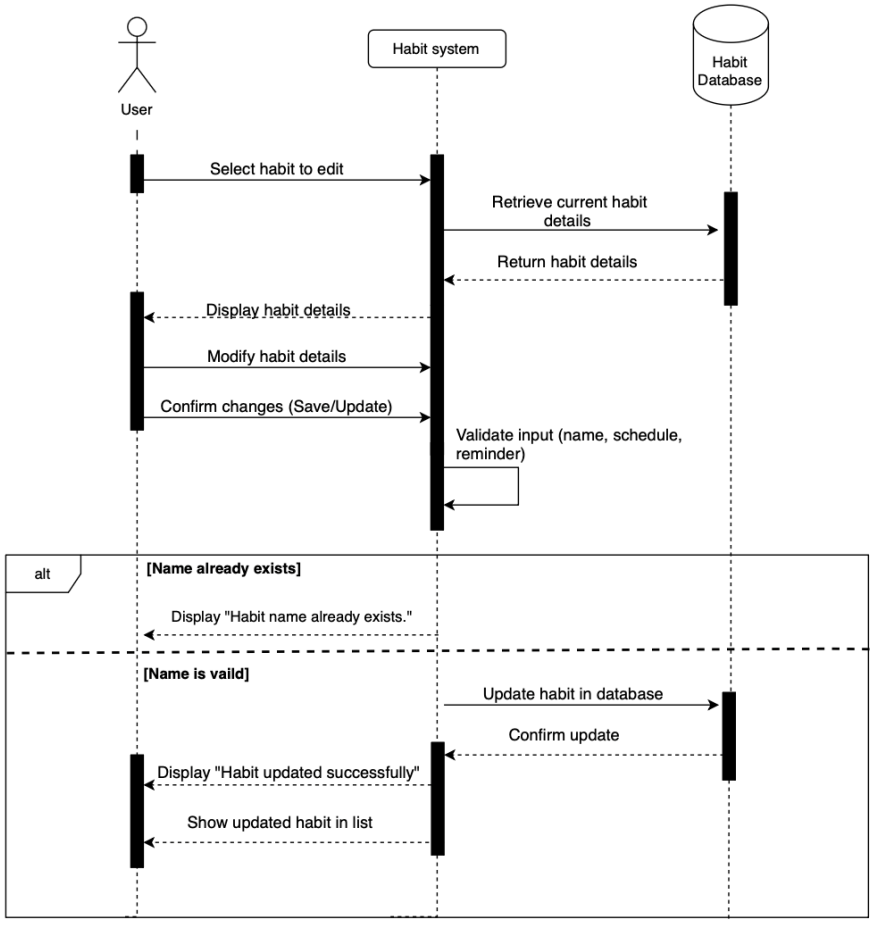
Story long description:

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 .

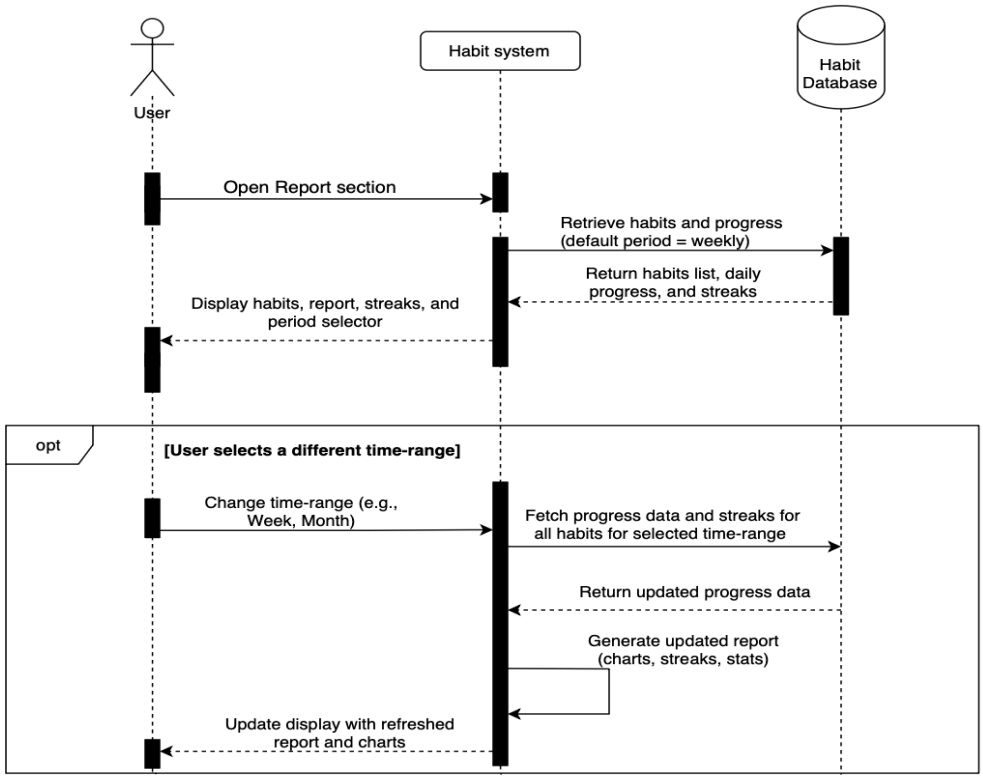
Story long description:
As a user, I need a report page that shows a clear summary of my habit activity

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]

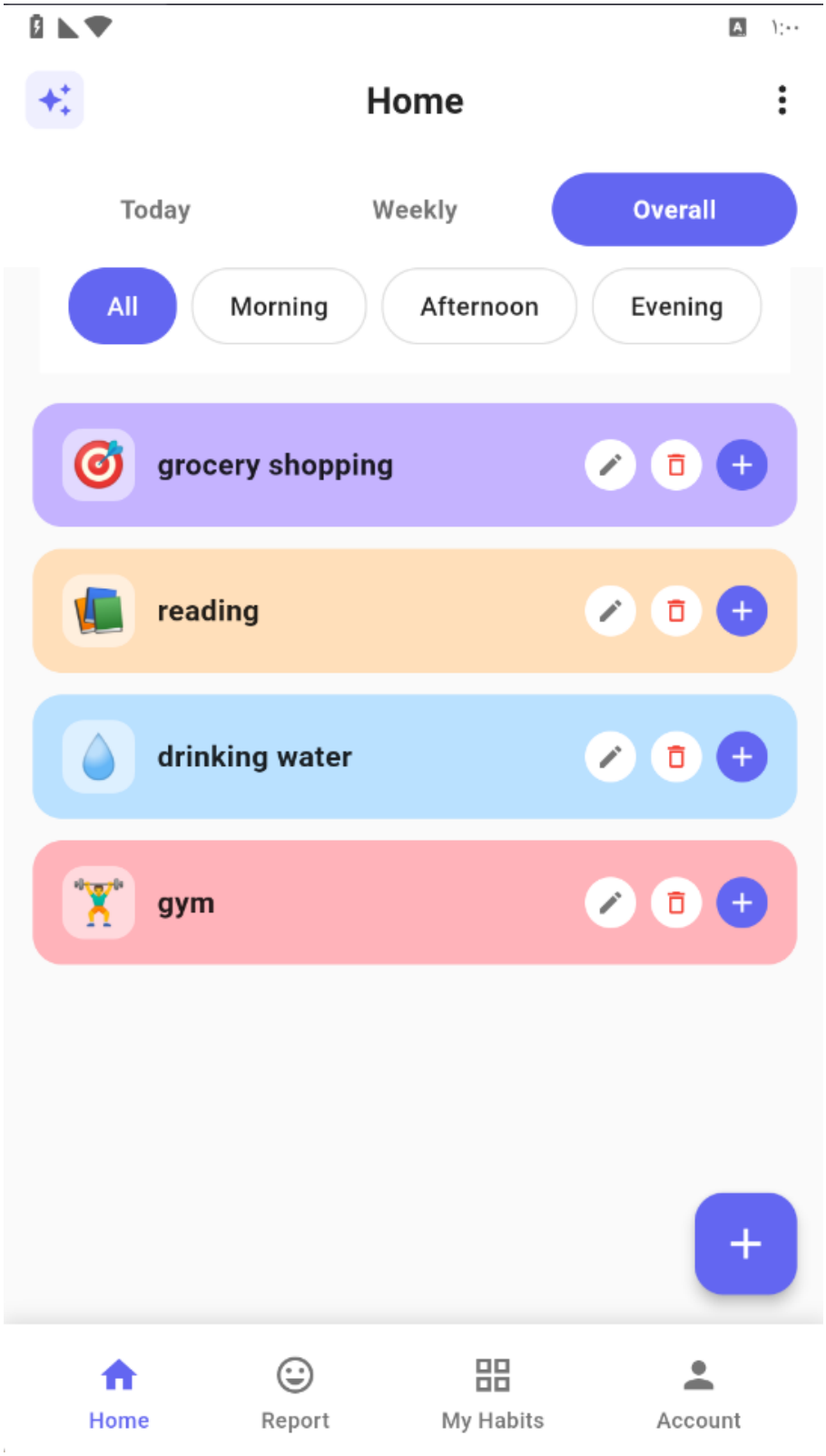
Test Case Name: [Sprint 2 – Update Habit]
Test Case Id: [Daily Habit Tracker – Update Habit]

Test Case No.	Test Case Description	Expected Results	Outcome (Pass / Fail / Other (Comments))
TC001	User clears the Habit Name field and taps **Save Changes**.	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 **Save Changes**	Updated name should appear in main habit list	Pass
TC004	User selects a new **icon** and taps **Save Changes**	Habit list should show updated icon	Pass
TC005	User selects a new color and taps **Save Changes**	Habit color should be updated in habit list	Pass
TC006	User changes **Repeat** 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 **(permanently)** 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]

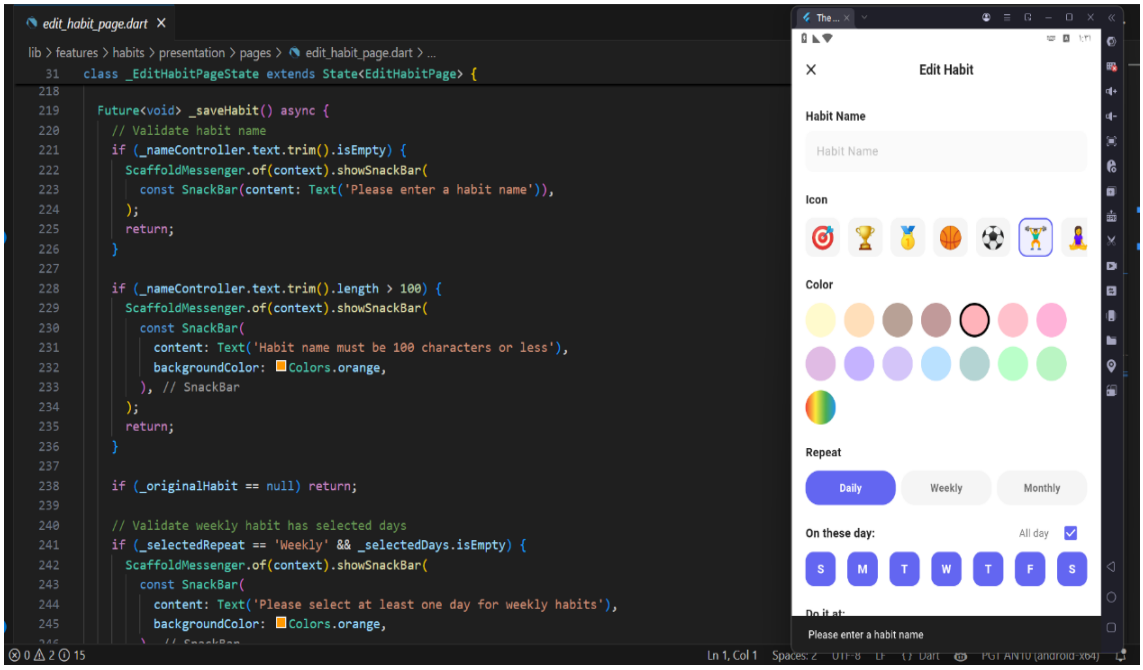
Test Case No.	Test Case Description	Expected Results	Outcome (Pass / Fail / Other (Comments))
TC001	User opens the Report page with no completed habits	All values should show **0** (0 days streak, 0% completion, 0 habits completed, 0 perfect days)	Pass
TC002	User completes a habit at least once	"Habits completed" count increases based on actual completions	Pass
TC003	User completes habit for consecutive days	**Current streak** increases correctly	Pass
TC004	User completes all habits for a day	"Total perfect days" increases by 1	Pass
TC005	User switches date filter to **Today**	Charts and numbers update to show only today's data	Pass
TC006	User switches date filter to **This Week**	Weekly data is displayed on all report widgets	Pass
TC007	User switches to **This Month**	Chart updates to show all completions for the month	Pass
TC008	User switches to **Last Month**	Last month's statistics should be displayed accurately	Pass
TC009	User switches to **Last 6 Months**	Charts show aggregated 6-month data	Pass
TC010	User switches to **This Year**	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

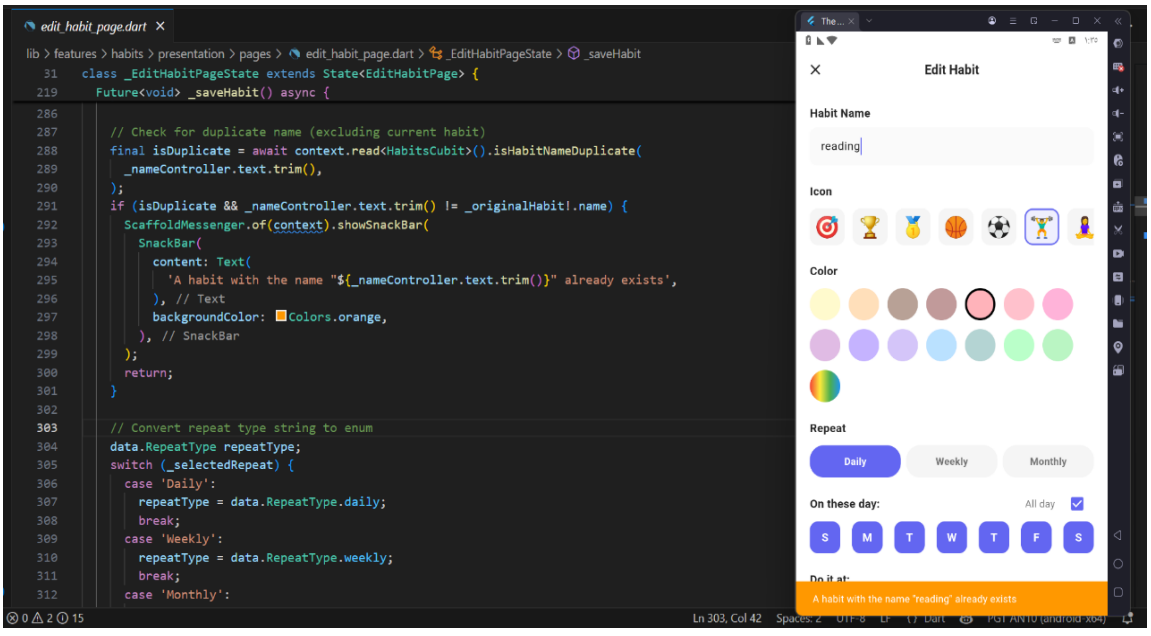


Test Case Name: Update Habit

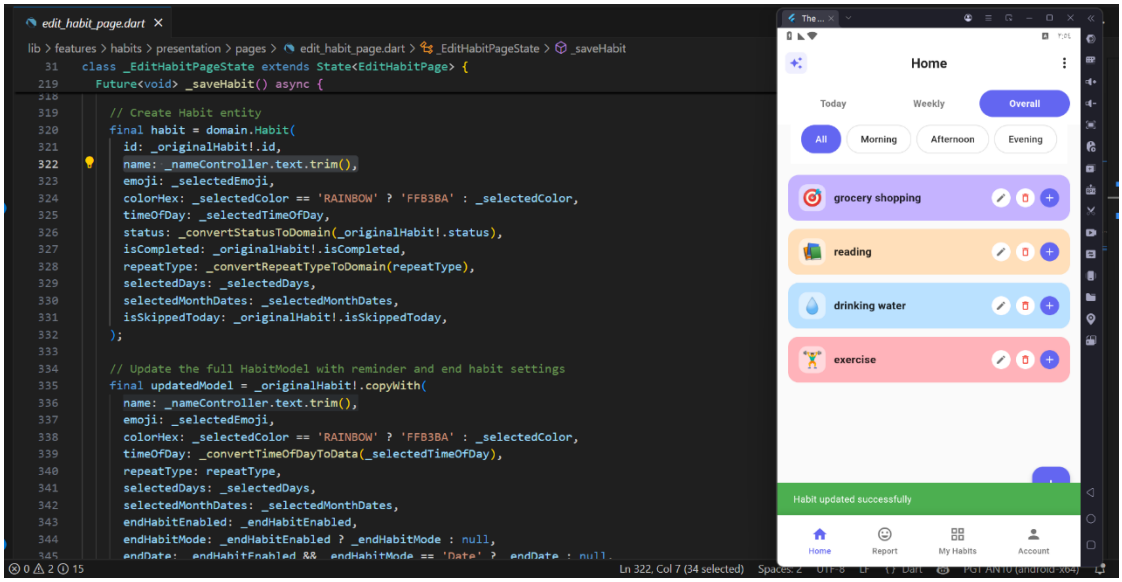
TC001



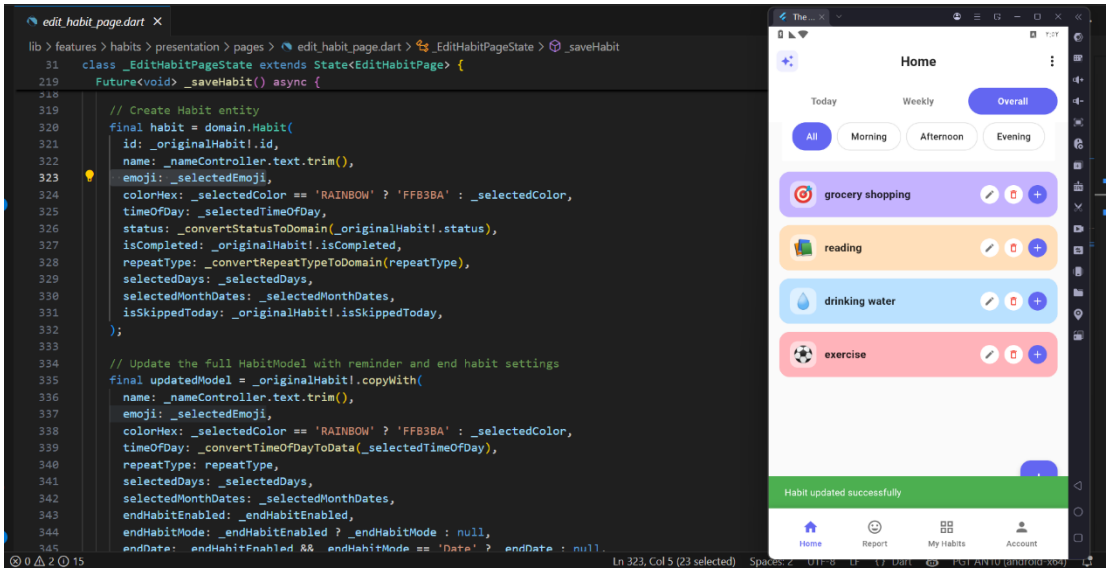
TC002



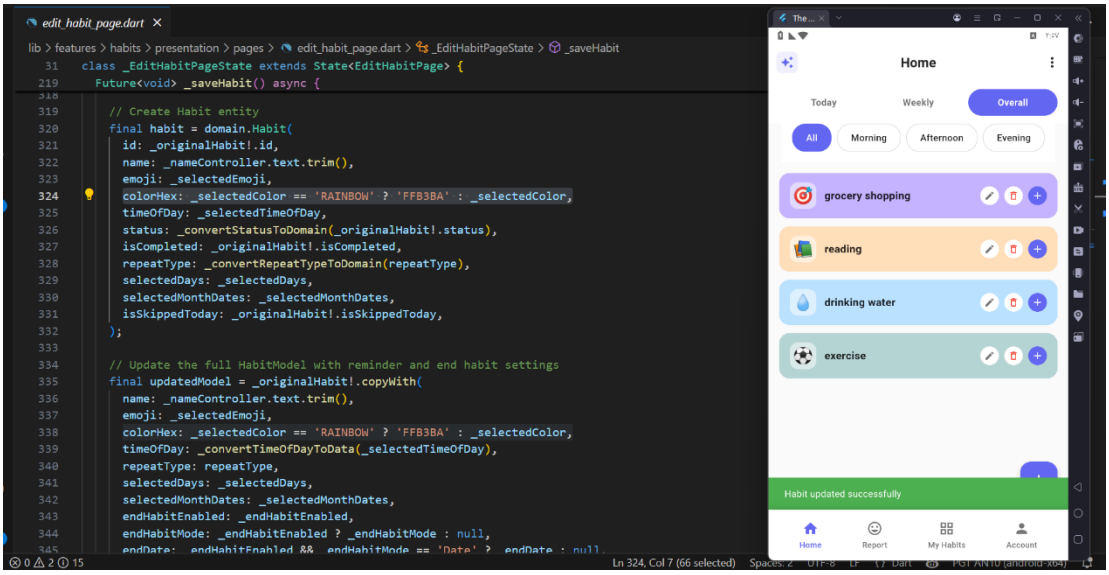
TC003


edit_habit_page.dart
lib > features > habits > presentation > pages > edit_habit_page.dart > _EditHabitPageState > _saveHabit
class _EditHabitPageState extends State<EditHabitPage> {
Future<void> _saveHabit() async {
// Create Habit entity
final habit = domain.Habit(
id: _originalHabit!.id,
name: _nameController.text.trim(),
emoji: _selectedEmoji,
colorHex: _selectedColor == 'RAINBOW' ? 'FFB3BA' : _selectedColor,
timeOfDay: _selectedTimeOfDay,
status: _convertStatusToDomain(_originalHabit!.status),
isCompleted: _originalHabit!.isCompleted,
repeatType: _convertRepeatTypeToDomain(repeatType),
selectedDays: _selectedDays,
selectedMonthDates: _selectedMonthDates,
isSkippedToday: _originalHabit!.isSkippedToday,
);
// Update the full HabitModel with reminder and end habit settings
final updatedModel = _originalHabit!.copyWith(
name: _nameController.text.trim(),
emoji: _selectedEmoji,
colorHex: _selectedColor == 'RAINBOW' ? 'FFB3BA' : _selectedColor,
timeOfDay: _convertTimeOfDayToData(_selectedTimeOfDay),
repeatType: repeatType,
selectedDays: _selectedDays,
selectedMonthDates: _selectedMonthDates,
endHabitEnabled: _endHabitEnabled,
endHabitMode: _endHabitEnabled ? _endHabitMode : null,
Ln 322, Col 7 (34 selected)
Home
Today
Weekly
Overall
All
Morning
Afternoon
Evening
grocery shopping
reading
drinking water
exercise
Habit updated successfully
Home
Report
My Habits
Account

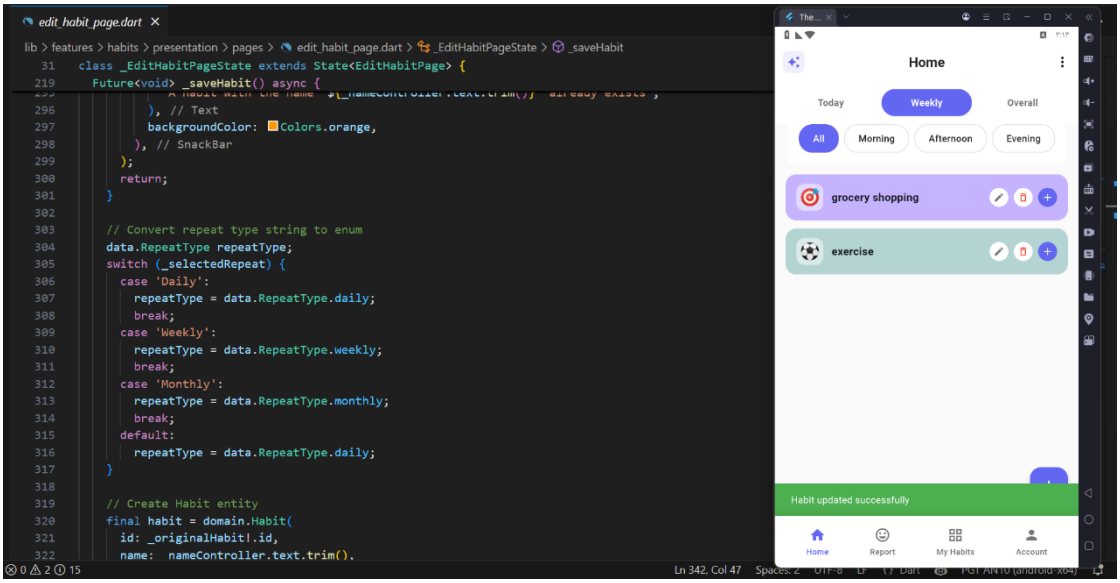
TC004


edit_habit_page.dart
lib > features > habits > presentation > pages > edit_habit_page.dart > _EditHabitPageState > _saveHabit
class _EditHabitPageState extends State<EditHabitPage> {
Future<void> _saveHabit() async {
// Create Habit entity
final habit = domain.Habit(
id: _originalHabit!.id,
name: _nameController.text.trim(),
emoji: _selectedEmoji,
colorHex: _selectedColor == 'RAINBOW' ? 'FFB3BA' : _selectedColor,
timeOfDay: _selectedTimeOfDay,
status: _convertStatusToDomain(_originalHabit!.status),
isCompleted: _originalHabit!.isCompleted,
repeatType: _convertRepeatTypeToDomain(repeatType),
selectedDays: _selectedDays,
selectedMonthDates: _selectedMonthDates,
isSkippedToday: _originalHabit!.isSkippedToday,
);
// Update the full HabitModel with reminder and end habit settings
final updatedModel = _originalHabit!.copyWith(
name: _nameController.text.trim(),
emoji: _selectedEmoji,
colorHex: _selectedColor == 'RAINBOW' ? 'FFB3BA' : _selectedColor,
timeOfDay: _convertTimeOfDayToData(_selectedTimeOfDay),
repeatType: repeatType,
selectedDays: _selectedDays,
selectedMonthDates: _selectedMonthDates,
endHabitEnabled: _endHabitEnabled,
endHabitMode: _endHabitEnabled ? _endHabitMode : null,
Ln 323, Col 5 (23 selected)
Home
Today
Weekly
Overall
All
Morning
Afternoon
Evening
grocery shopping
reading
drinking water
exercise
Habit updated successfully
Home
Report
My Habits
Account

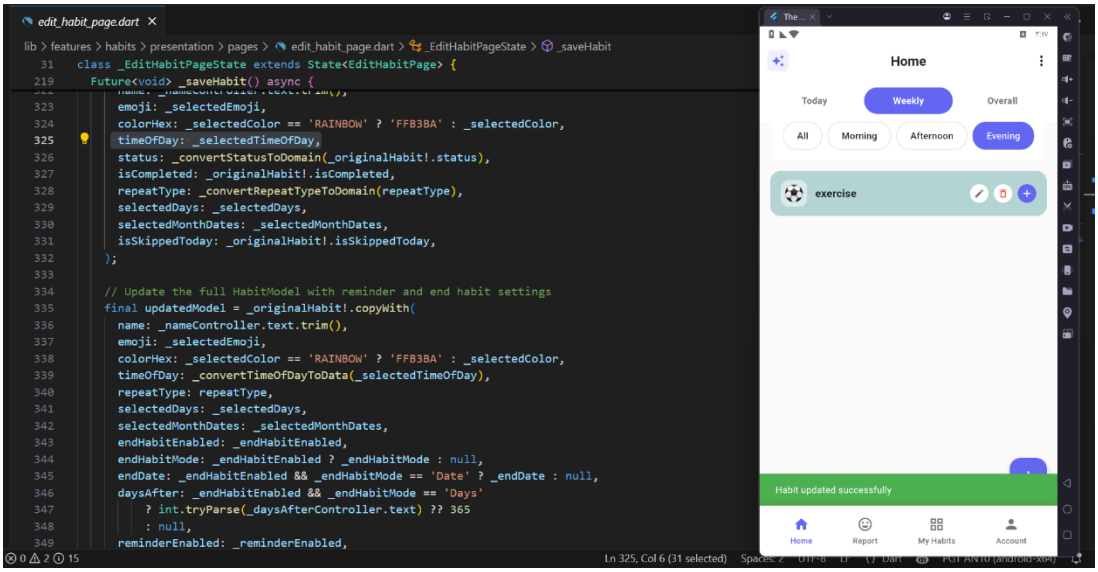
TC005



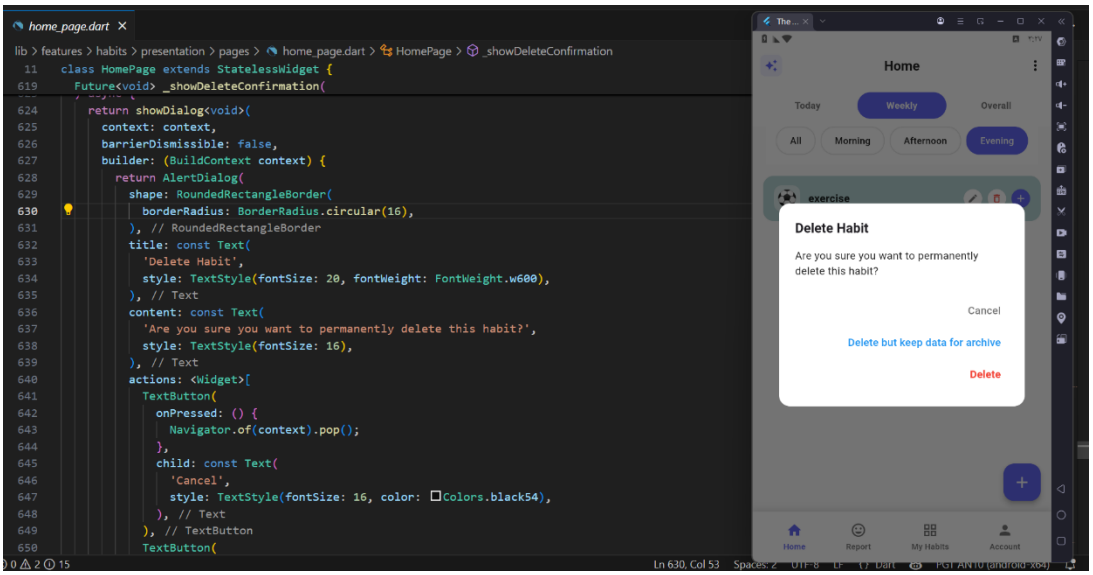
TC006



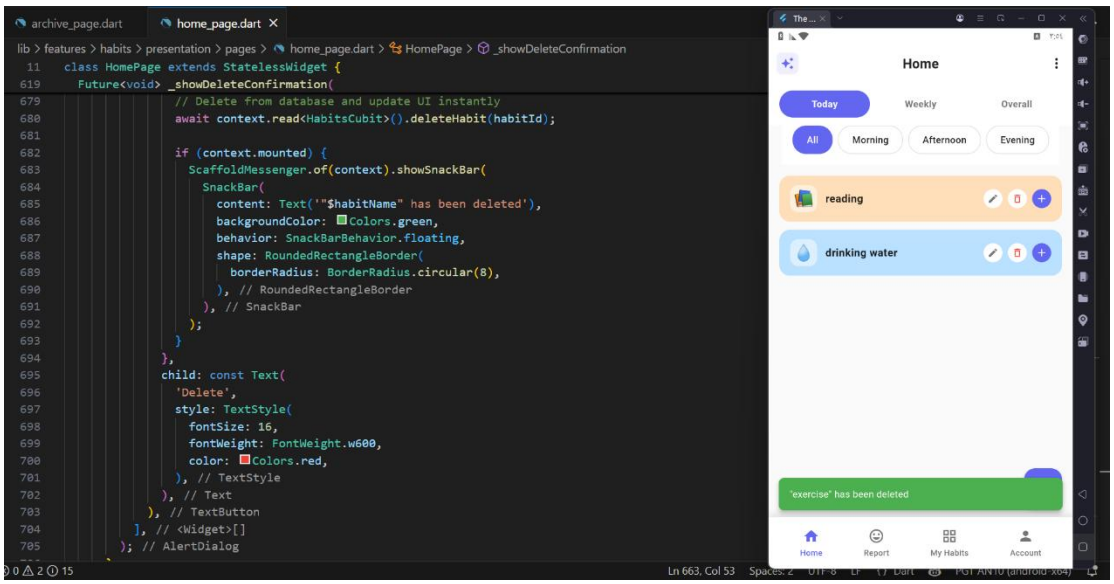
TC007



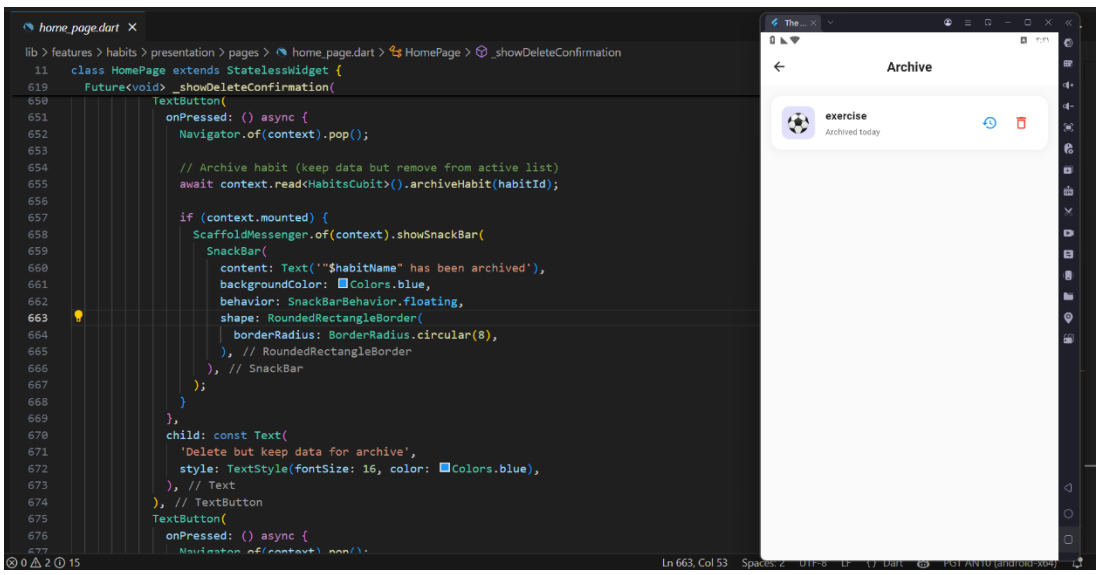
TC008



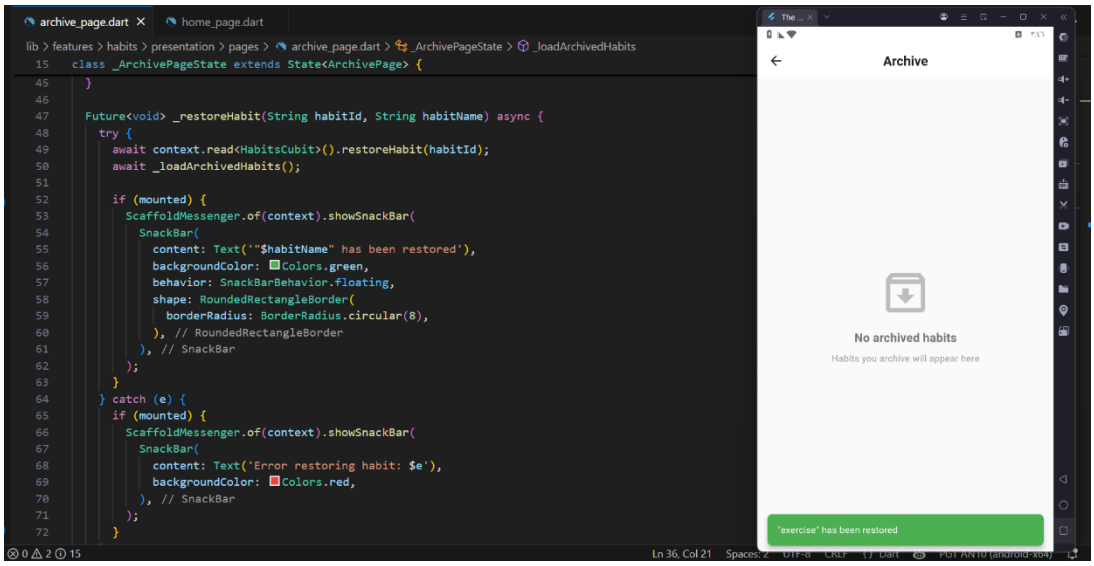
TC009



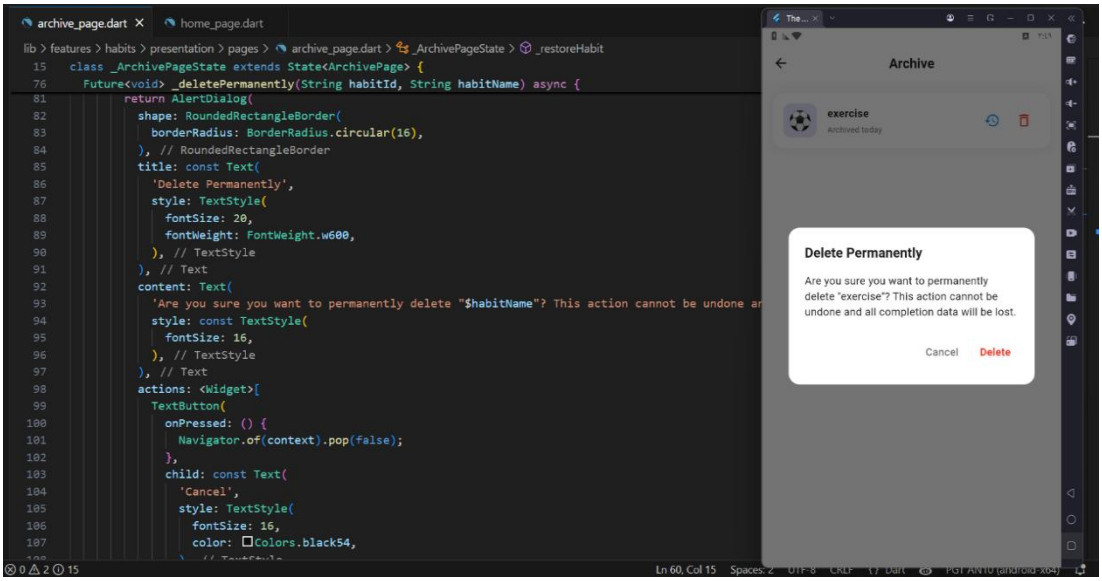
TC010



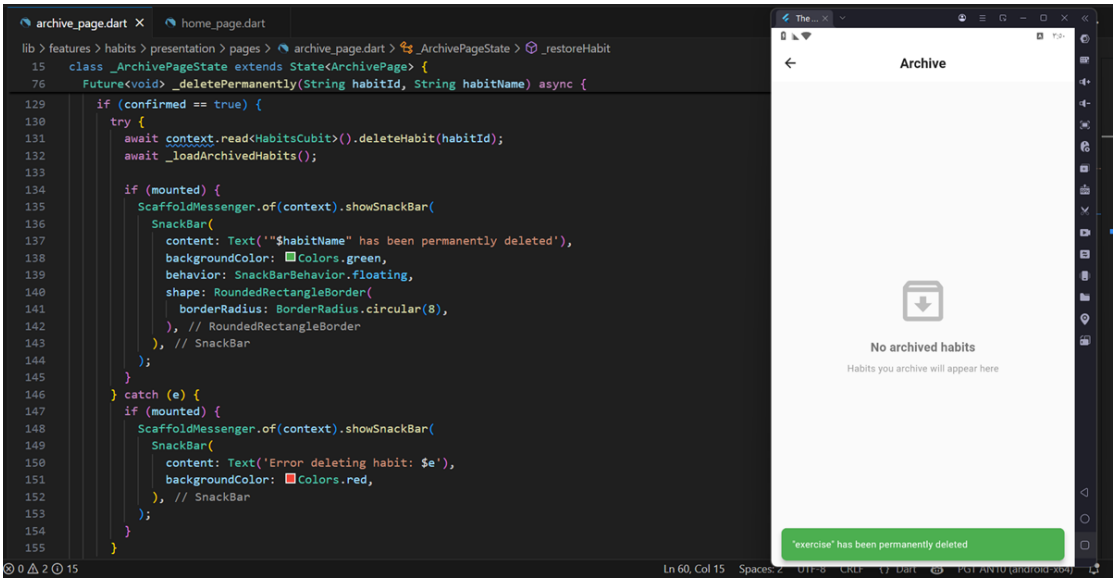
TC011



TC012

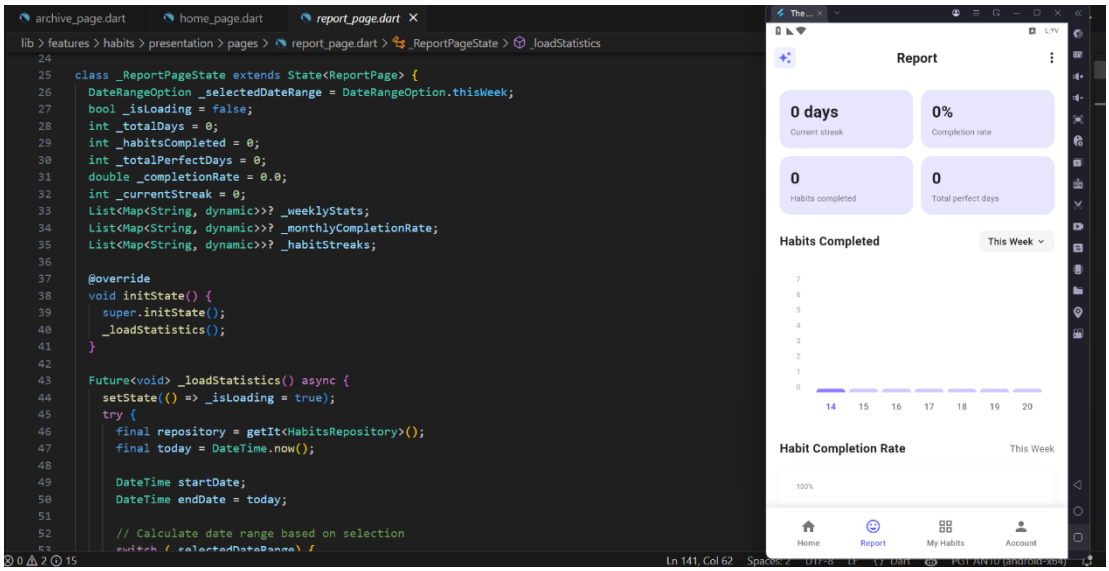


TC013

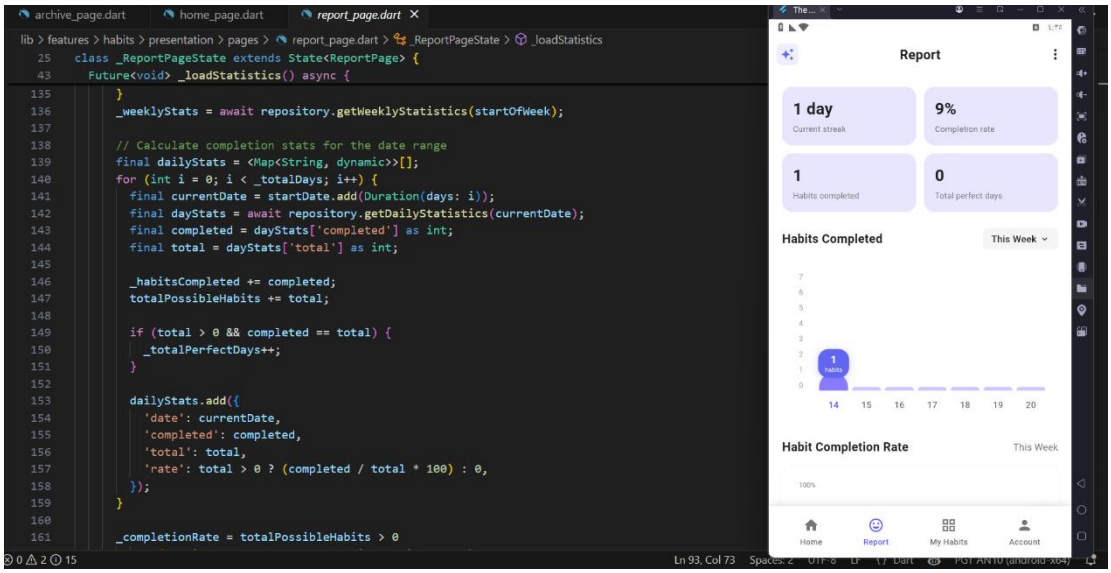


Test Case Name: Report

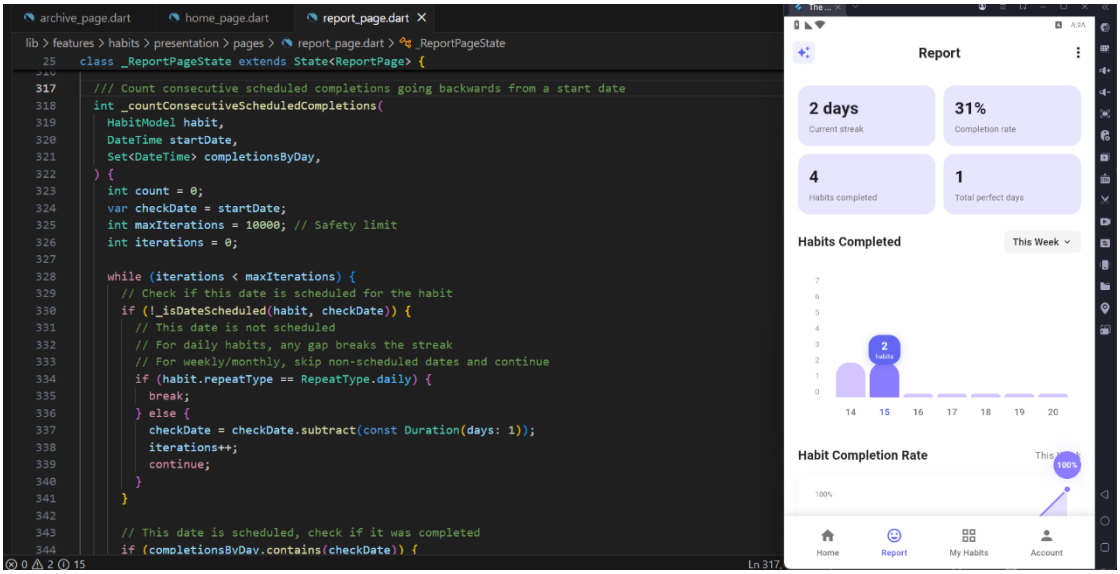
TC001



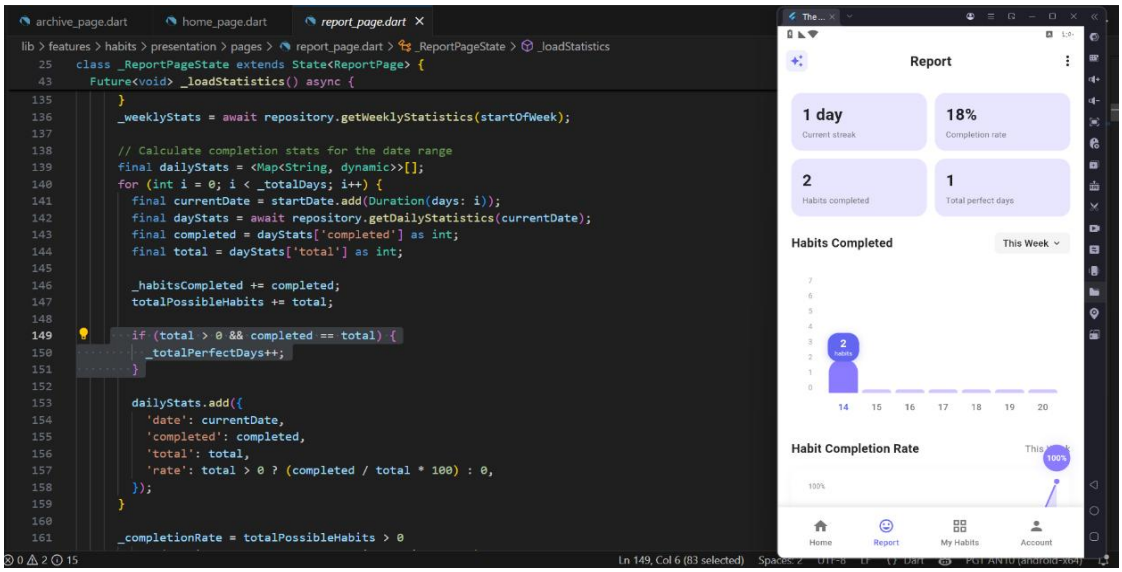
TC002



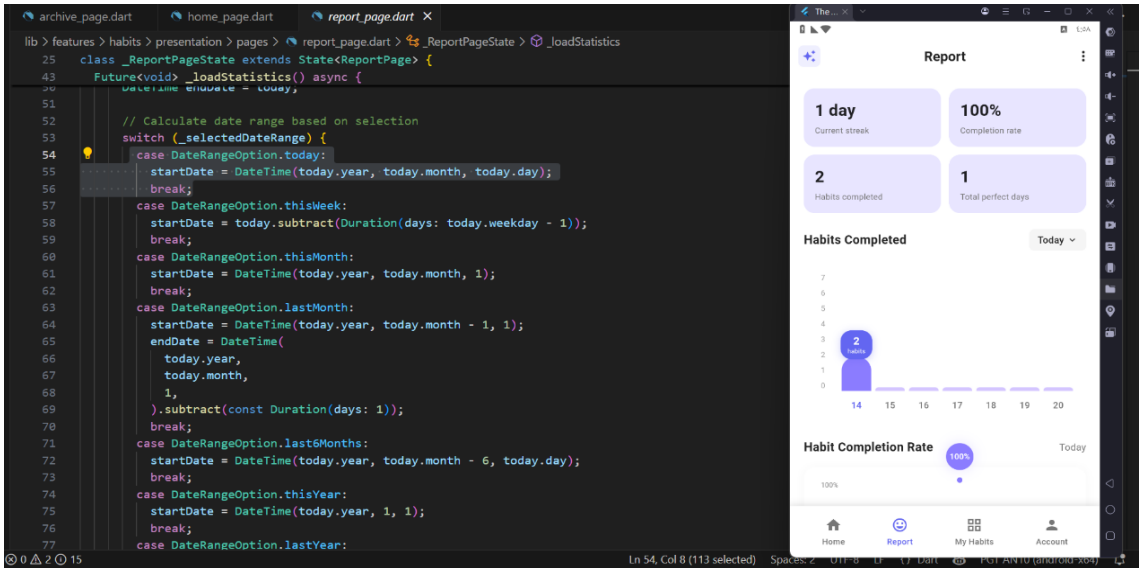
TC003



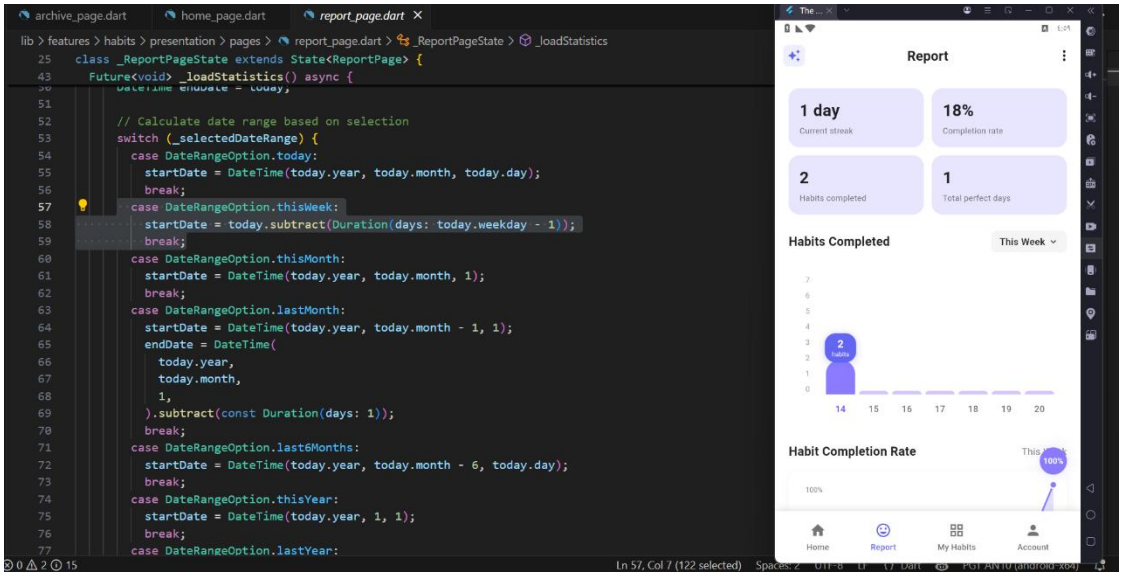
TC004



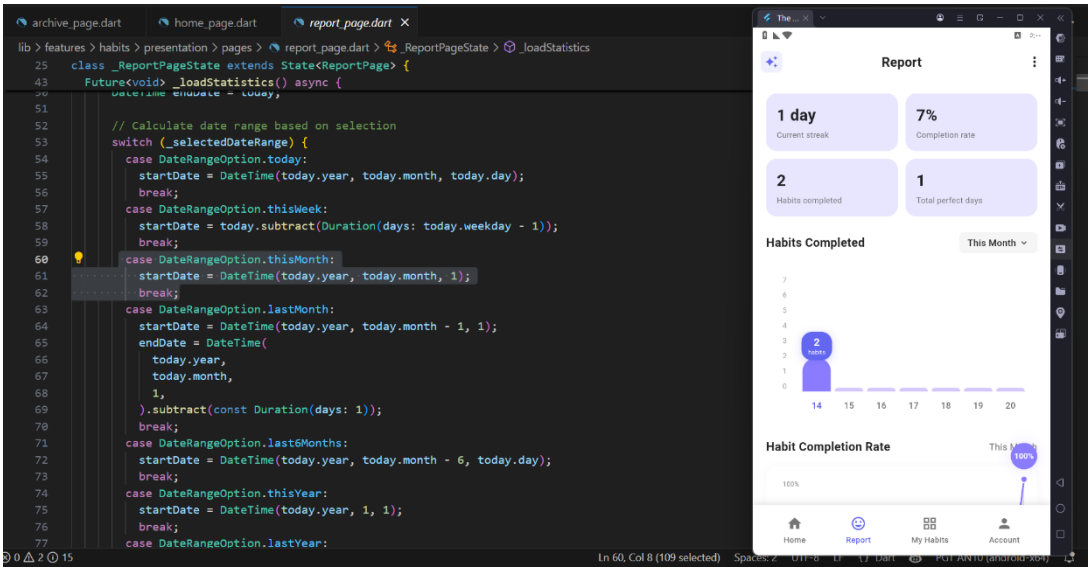
TC005



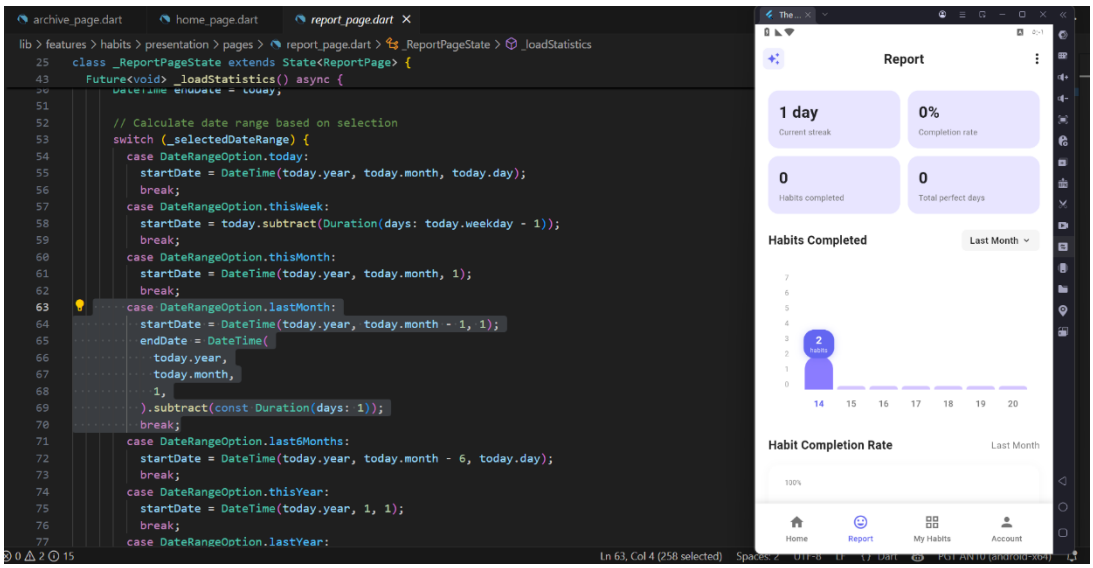
TC006



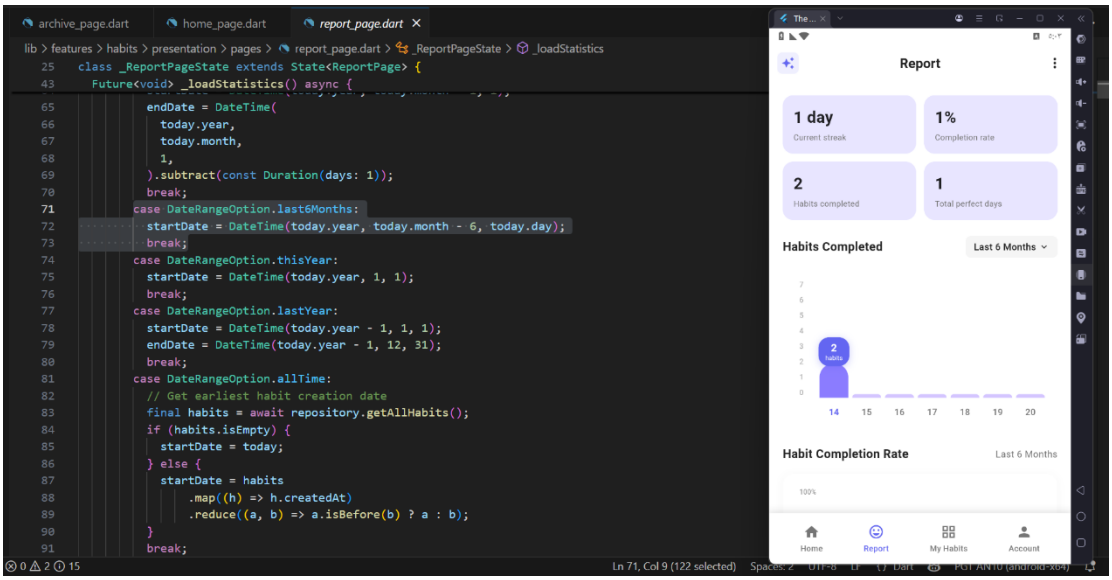
TC007



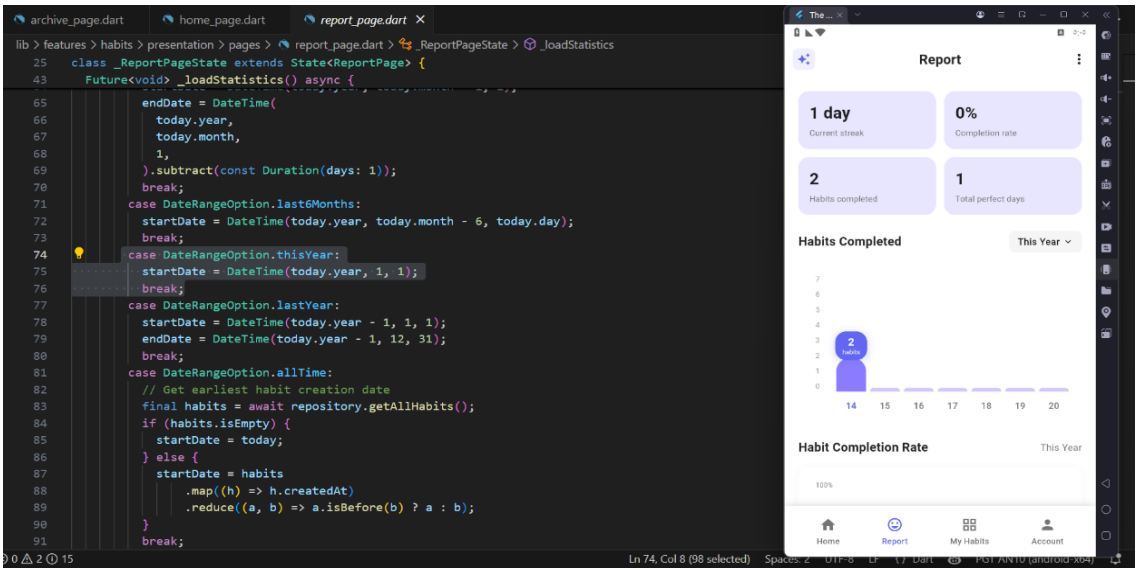
TC008



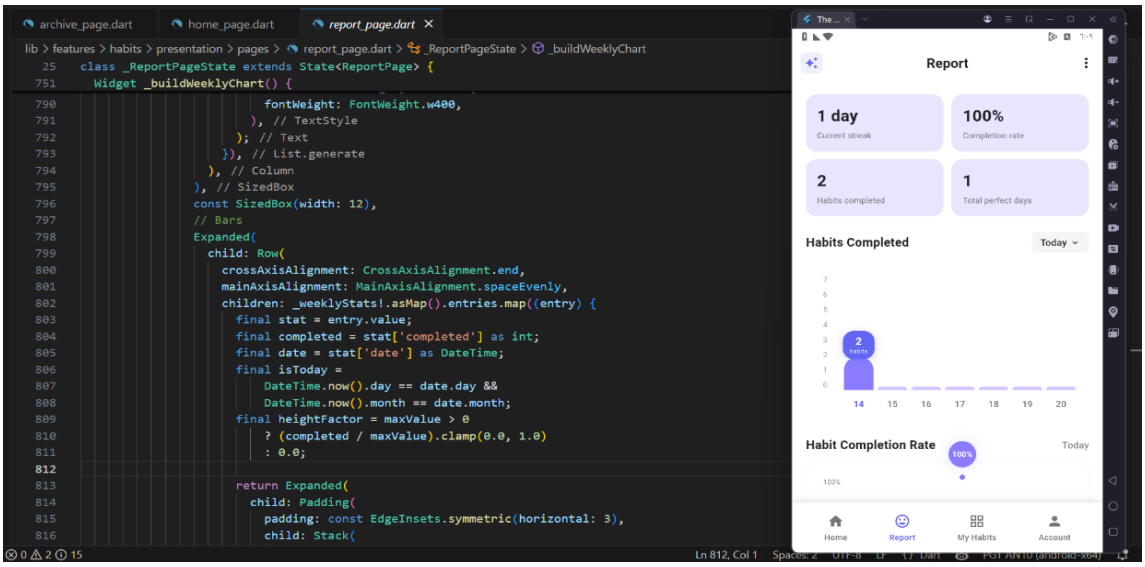
TC009



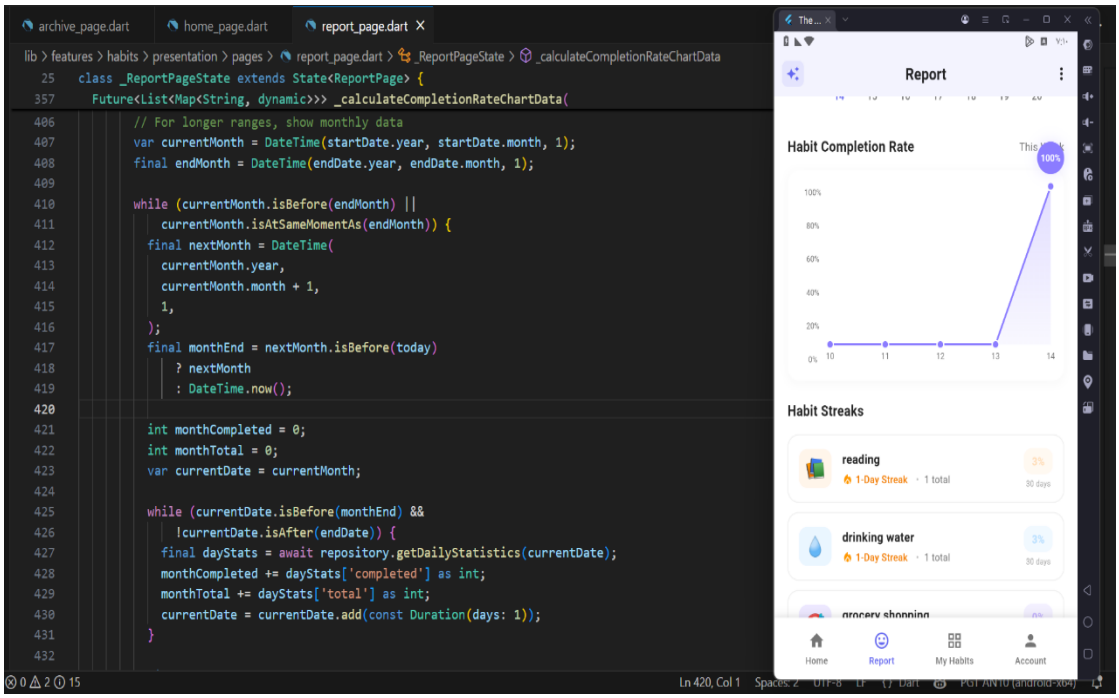
TC010



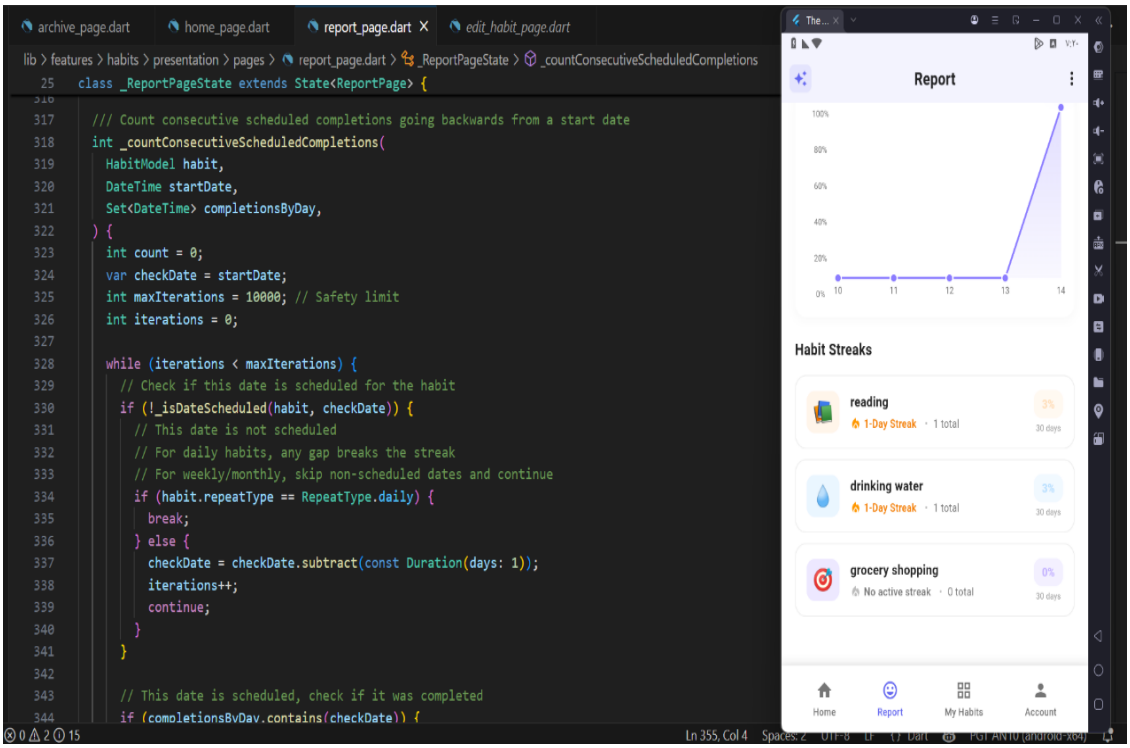
TC011



TC012


archive_page.dart
home_page.dart
report_page.dart
lib > features > habits > presentation > pages > report_page.dart > _ReportPageState > _calculateCompletionRateChartData
class _ReportPageState extends State<ReportPage> {
Future<List<Map<String, dynamic>>> _calculateCompletionRateChartData(
// For longer ranges, show monthly data
var currentMonth = DateTime(startDate.year, startDate.month, 1);
final endMonth = DateTime(endDate.year, endDate.month, 1);
while (currentMonth.isBefore(endMonth) ||
currentMonth.isAtSameMomentAs(endMonth)) {
final nextMonth = DateTime(
currentMonth.year,
currentMonth.month + 1,
1,
);
final monthEnd = nextMonth.isBefore(today)
? nextMonth
: DateTime.now();
int monthCompleted = 0;
int monthTotal = 0;
var currentDate = currentMonth;
while (currentDate.isBefore(monthEnd) &&
!currentDate.isAfter(endDate)) {
final dayStats = await repository.getDailyStatistics(currentDate);
monthCompleted += dayStats['completed'] as int;
monthTotal += dayStats['total'] as int;
currentDate = currentDate.add(const Duration(days: 1));
}
Report
Habit Completion Rate
Habit Streaks
reading
1-Day Streak · 1 total
drinking water
1-Day Streak · 1 total
Home
Report
My Habits
Account

TC013


archive_page.dart
home_page.dart
report_page.dart
edit_habit_page.dart
lib > features > habits > presentation > pages > report_page.dart > _ReportPageState > _countConsecutiveScheduledCompletions
class _ReportPageState extends State<ReportPage> {
/// Count consecutive scheduled completions going backwards from a start date
int _countConsecutiveScheduledCompletions(
HabitModel habit,
DateTime startDate,
Set<DateTime> completionsByDay,
) {
int count = 0;
var checkDate = startDate;
int maxIterations = 10000; // Safety limit
int iterations = 0;
while (iterations < maxIterations) {
// Check if this date is scheduled for the habit
if (!_isDateScheduled(habit, checkDate)) {
// This date is not scheduled
// For daily habits, any gap breaks the streak
// For weekly/monthly, skip non-scheduled dates and continue
if (habit.repeatType == RepeatType.daily) {
break;
} else {
checkDate = checkDate.subtract(const Duration(days: 1));
iterations++;
continue;
}
}
// This date is scheduled, check if it was completed
Report
Habit Streaks
reading
1-Day Streak · 1 total
drinking water
1-Day Streak · 1 total
grocery shopping
No active streak · 0 total
Home
Report
My Habits
Account

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