Smart Step Milestone #3 Requirements

This document describes the requirements for the **third development milestone** (Milestone #3) of the **Smart Step** application.

Milestone #3 focuses on introducing AI-powered user guidance and insights into the application. At this stage, the app is extended with an AI Coach that provides contextual feedback, motivational messages, and interactive recommendations based on the user's activity data, building on the tracking and analytics features implemented in Milestone #2.

The mockups define the overall appearance and expected behavior of the interface. Specific technical decisions (component structure, state handling, animation implementation) are left to the participants, as long as the final result follows the described logic and interaction patterns.

You can find the Smart Step **mockups here**:

> https://www.figma.com/design/aTiGCCavPLtgZNGXcV23H6/Smart-Step?node-id=57-1002

Milestone #3 Goal

- Introduce the **AI Coach** block on the Main Screen with contextual activity insights
- Implement the **AI Coach Chat Screen** for conversational interaction with the AI
- Integrate a real **AI API** to generate dynamic, user-specific responses
- Handle **online and offline states** for AI-related features
- Define adaptive UI behavior for AI messages on **mobile and wider screens**

Icons

You may use Material Design icons where appropriate. If a suitable Material icon is not available, use the custom icons provided in the Figma mockups.

In Figma, any icon or image can be **exported** by selecting the element and clicking "Export" in the **right-hand panel**. In this panel, you can also choose the desired format (PNG, SVG, etc.).

Adaptive Layouts

The app must support two breakpoints:

- **Up to 840 dp** → mobile layout.
- **From 840 dp and above** → wide-screen layout.

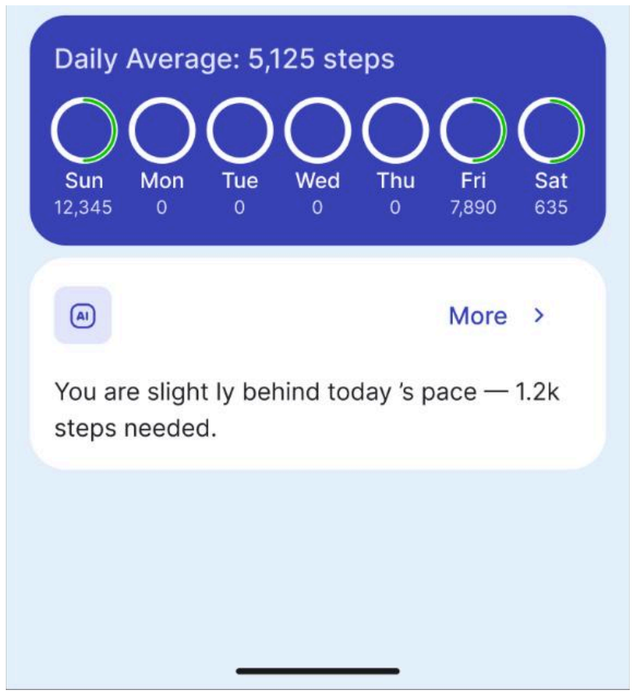
Technical Requirements

🤖 AI Insights Block (New)

A new AI-powered insights block is added to the **Main Screen**, positioned below the Daily Average section.

The block displays a short analytical or motivational message generated based on the user's current activity data (step count, daily goal, progress).

Behavior



- The AI insight is refreshed only when the Main Screen becomes active under **specific conditions**:
  - on the **first app launch** of the current session;
  - when the app **returns from the background**;
  - when the user **reaches the daily step goal**;
  - when the user **changes the daily step goa**l.
- The AI insight must not be regenerated for minor step count changes and must not be updated on a timer.
- If none of the conditions above are met, the previously generated insight may be reused.

Purpose

- The goal of the AI block is to:
  - interpret the current activity state,
  - provide a concise, human-readable insight,
  - increase user engagement without interrupting the main flow.
- Example message: *"You are slightly behind today's pace — 1.2k steps needed."*

Interaction

- The block itself is informational.
- The More button opens a dedicated AI Chat Screen, where the user can interact with the AI assistant in more detail.

AI Integration Requirements

- Simulating AI behavior is **not allowed**.
- Insights must be generated using a **real AI API**.
- Any public LLM API with a free tier may be used.
- Recommended option: **Google Gemini API (free tier).**

The AI receives structured input (e.g. steps, goal, time of day) and returns a short textual insight.

AI Prompt Input

Each AI request must be built from the same fixed and predefined set of input data.

The following values must be prepared on the app side and passed to the AI as structured context:

- current step count for the day;
- daily step goal;
- goal completion percentage;
- current time context (e.g. morning / day / evening);

No additional or dynamic parameters are required.

Expected AI Output

The AI must generate **one short textual message** that:

- interprets the current activity state;
- **does not** contain medical advice;
- **does not** repeat raw numeric values;
- has a motivational or analytical tone.

Expected **response style examples**:

- *“You’re on track today. Keep the pace steady.”*
- *“You’re a bit behind your goal — a short walk could help.”*
- *“Great job! You’ve already reached today’s goal.”*

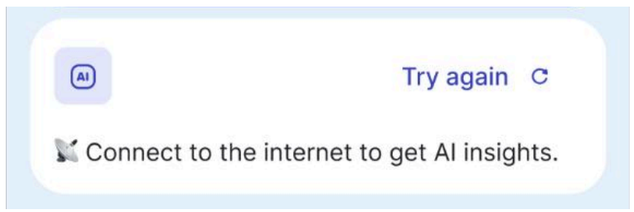
The AI **must not**:

- start or continue a conversation;
- ask follow-up questions;
- store or rely on conversation history for this block.

Each message is generated independently based on the current input only.

Offline State — AI Insights Unavailable

If there is no active internet connection when the Main Screen becomes active, the AI Insights block enters an offline state.



In this state:

- No request to the AI API is performed.
- The AI Insights block remains visible in the layout.
- A static informational message is displayed instead of an AI-generated insight.
- The **More button is not displayed** while the device is offline.
- A **Try Again** action is shown instead, accompanied by a refresh icon.

Displayed message example: 📡 *Connect to the internet to get AI insights*.

Try Again Behavior

- Tapping **Try Again** triggers a new check for internet connectivity.
- If an internet connection is available:
  - a request to the AI API is performed,
  - the offline message is replaced with a new AI-generated insight,
  - the **More button** becomes available again.
- If the device is still offline:
  - the offline state remains unchanged,
  - no error is shown.

🤖 AI Coach — Chat Screen

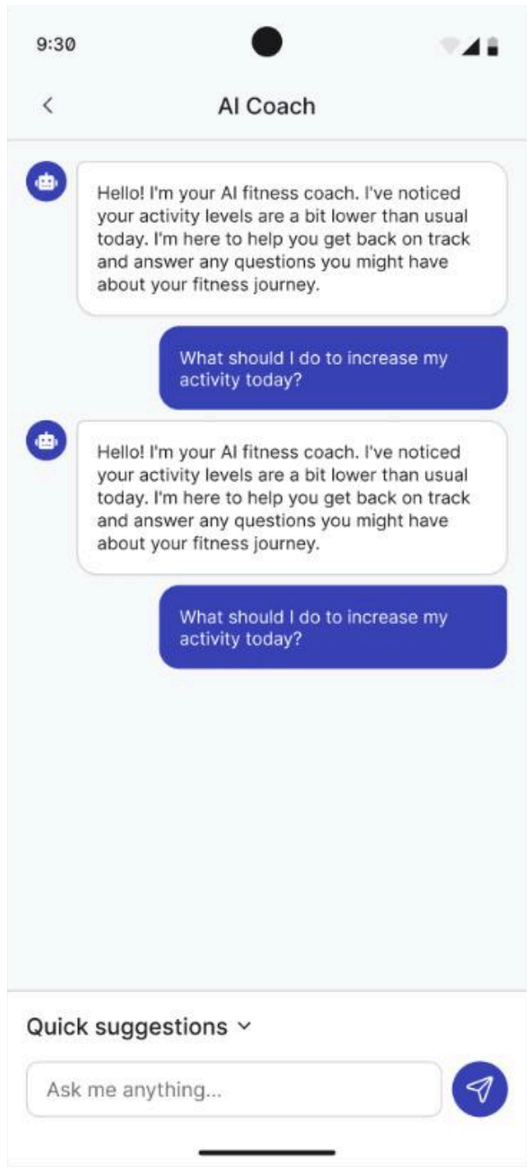
The AI Coach screen provides a short, focused conversational experience where the user can receive AI-driven fitness insights and recommendations based on their current activity.

The screen is designed as a lightweight, session-based chat rather than a persistent conversation

Top Bar

The top area of the screen contains a Top Bar with the following elements:

- A centered title: *AI Coach*.
- A **Back button** (arrow icon) on the left,
- Tapping the **Back button** navigates the user back to the **Main Screen** (Home).
- The AI Coach screen is removed from the navigation back stack.
- When the screen is opened again, a **new AI session** is started (previous conversation is not restored).



Initial AI Message

When the AI Coach screen is opened, the AI automatically sends an **initial starter message**.

This message is generated from a **predefined prompt** and serves as the starting point of the conversation. It should include:

- a brief greeting,
- a short introduction of the AI as a personal fitness coach,
- a concise explanation of the user's current activity context (e.g. today's activity level),
- a short question asking how the AI can help.

The message must be concise, contain no detailed recommendations, and is sent automatically without any user interaction.

Message History

- The central area of the screen displays the conversation history.
- Messages are shown in chronological order.
- User messages appear aligned on one side, AI responses on the opposite side.
- Messages are appended immediately after being sent or received.

AI Messages

- AI message bubbles are displayed on the left side.
- In the **mobile version,** AI message bubbles:
  - occupy the **full available width of the screen** (respecting standard horizontal paddings),
  - automatically wrap text to new lines.
- This behavior reflects the fact that AI responses are typically longer and more explanatory.

User Messages

- User message bubbles are displayed on the right side.
- In the **mobile version**, user message bubbles:
  - have a **maximum width of 75% of the screen width**,
  - **do not** have a fixed width in dp,
  - adapt to content:
    - short messages wrap tightly around the text,
    - longer messages expand up to the 75% width limit and then wrap to new lines.
- This approach ensures good readability across different screen sizes and devices.

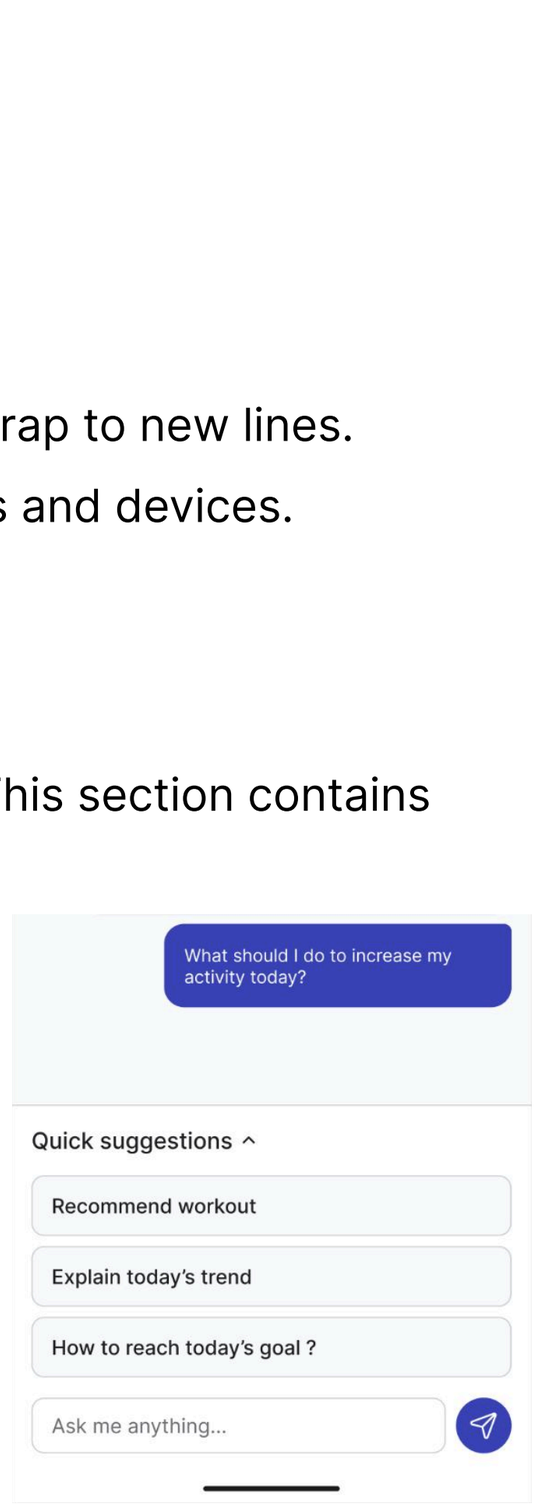
Quick Suggestions Section

Below the message history, a **Quick suggestions** section is displayed. This section contains exactly **three fixed action buttons**:

- *Recommend workout*
- *Explain today's trend*
- *How to reach today's goal?*

Behavior:

- The section can be expanded or collapsed by the user.
- Tapping any suggestion:
  - immediately sends the corresponding prompt to the AI,
  - adds it as a user message in the chat,
  - triggers an AI response.



Message Input Field

- A text input field is displayed at the bottom of the screen.
- The input starts as a single-line field.
- It automatically expands vertically as the user types, up to 5 lines maximum.
- After reaching the maximum height, the input becomes scrollable.
- A message can be sent only if the trimmed input contains at least one non-whitespace character.

When there is no internet connection:

- the message input field remains **visible but disabled**;
- the input field does **not accept focus** and does not open the keyboard;
- the send button is **disabled** and visually muted;
- the input placeholder text is replaced with: *Online connection required*;
- an inline **disabled-state icon** (e.g. crossed-out connection icon) is displayed inside the input field to reinforce the offline state;

When the internet connection is restored:

- the input field automatically becomes active;
- the placeholder text returns to its default value;
- the send button is re-enabled;
- the user can immediately start typing and send messages without additional actions.

Notes

- Internet availability must be observed via the system connectivity state.
- UI state must update **automatically** when connectivity changes, without requiring screen refresh or navigation.

Dialog Lifecycle & State Behavior

The chat does not persist between screen openings.

Behavior:

- When the AI Coach screen is opened:
  - the message history is empty,
  - the initial AI starter message is sent automatically.
- When the user navigates back to the main screen:
  - the AI Coach screen is removed from the back stack,
  - the conversation state is discarded.
- When the AI Coach screen is opened again:
  - the chat starts from scratch,
  - the initial AI message is sent again.

This approach intentionally treats each visit as a new consultation session, simplifying state management and avoiding outdated activity context.

📏 Adaptation for Larger Screens

On wider screens, the **same message layout logic** described above is preserved. The only difference lies in **container width constraints**.

Message History Area

- The entire message history is centered on the screen.
- Messages are displayed inside a container with a **maximum width of 600 dp**.

AI messages

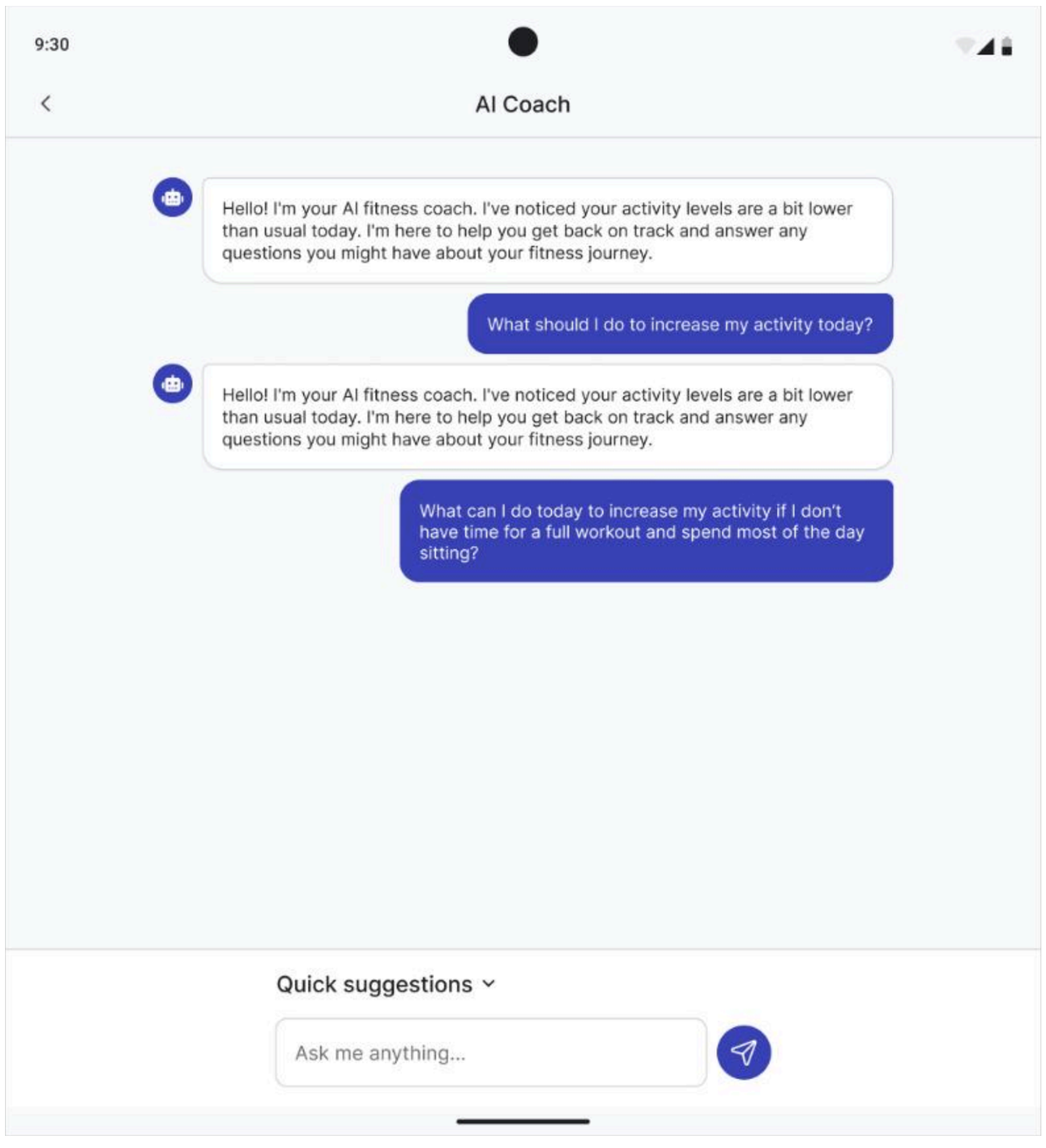
- Can expand to the full width of the message container (up to 600 dp).
- Follow the same wrapping and alignment rules as on mobile.

User messages

- Have a **maximum width of 400 dp**.
- Behavior remains consistent:
- short messages wrap tightly around text,
- longer messages expand up to the maximum width and then wrap.

Input & Suggestions Area

- The bottom section (Quick Suggestions, input field, send button):
- is centered horizontally,
- uses a **fixed width of 400 dp** on wider screens,
- follows the same interaction and resizing behavior as in the mobile version.



🔗 Useful Links for This Challenge

- Gemini API quickstart
- Gemini API
- Monitor connectivity status
- Read network state
- Foreground services overview
- Foreground Services video
- About notifications
- Guide to app architecture
- Create a Splash Screen
- UX With Material3
- Full Guide to Material3 Theming
- Request runtime permissions
- Bottom sheets
- DataStore
- Physical Activity Permission
- Ignore Battery Optimization
- Save data in a local database using Room
- The Full Jetpack Compose Responsive UI Crash Course
- How to Save & Restore the Scroll Position of a LazyColumn Persistently
- Stateful vs. Stateless Composables
- State Hoisting in Compose
- Managing State in Jetpack Compose (Codelab)