Technical Requirements – Fresh Start Settings

Challenge is accessible on Memberspot: https://pl-coding.mymemberspot.io/library/jx3b7Qik9ip5qpNI8IF2/BwAxp75QNXGaDGF2ZJEG/gmth6S9s7pD2mBDgk7cP/details

🎭 Scenario

This mini-challenge simulates a simple New Year–themed settings screen where the user adjusts personal preferences. The screen represents a typical settings page and focuses on persisting user-defined values and restoring them correctly after app restarts.

🎨 Figma Mockups

https://www.figma.com/design/XKuvJCxCybCUpKZP0LelAs/New-Year-Fresh-Start?node-id=1-2

🔤 Font - Plus Jakarta Sans

🎯 Feature Goal

Implement a single settings screen using Jetpack Compose that demonstrates working with **DataStore Preferences**, including reading, writing, and reacting to preference changes, while keeping the UI clean, simple, and state-driven.

📌 Requirements

This screen is the only screen in this mini-challenge and simulates a typical user settings screen.

The main goal of the challenge is to demonstrate working with **DataStore Preferences**: saving, reading, and automatically restoring values.

All settings are saved automatically when their values change. All saved values must be restored after reopening the screen or restarting the app.

Top Bar

- Positioned at the top of the screen.

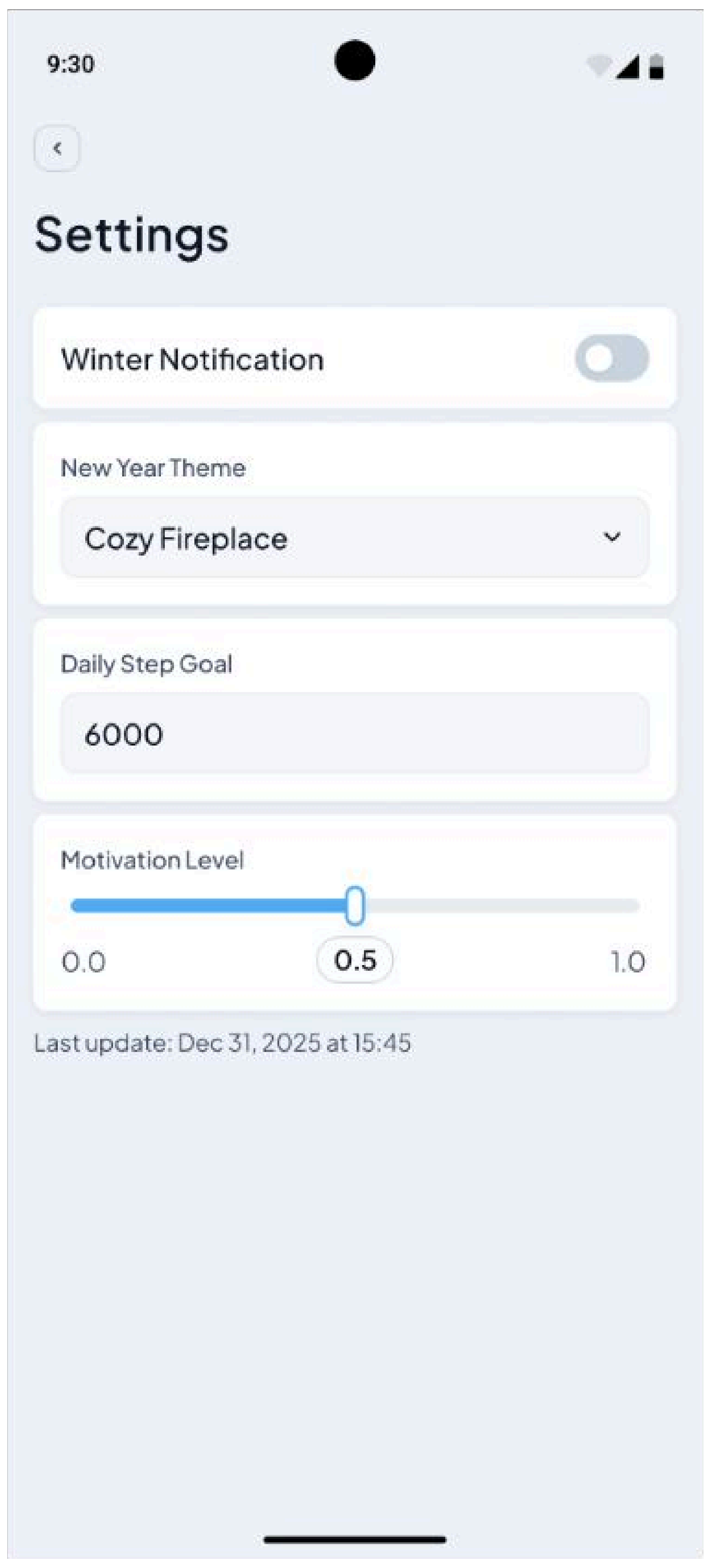
- Displays the title *“Settings”*.
- Contains a **Back button**.
- The Back button is a **decorative element** only.
- Pressing the Back button **does not** perform any action.

Preferences Storage

- All settings must be stored using **DataStore Preferences**.
- Values are stored as simple key-value pairs.
- Data must persist and be restored even after the app process is killed.

Last Updated Behavior

- The **Last Updated** field displays the date and time of the most recent settings save.
- It is updated every time any setting is changed and saved.
- The value is stored in DataStore and restored when the screen is reopened.



Winter Notification

- Implemented as a toggle switch.
- Has two states: enabled and disabled.

New Year Theme

- Implemented as a dropdown menu.
- Contains three options:
  - Cozy Fireplace
  - Snowy Morning
  - Midnight Blue
- The currently selected value is always visible.

Daily Step Goal

- Implemented as a text field.
- Accepts numeric values only.
- Displays the numeric keyboard when focused.
- The keyboard includes a confirmation action (IME action, such as a “Done” button).
- The value is saved when either of the following occurs:
  - the user presses the keyboard confirmation action;
  - the text field loses focus after the value has been changed.

- When the value is saved:
  - the keyboard is dismissed;
  - focus is cleared from the text field;
  - the value is persisted to DataStore Preferences;
- Changes are not saved while the user is actively typing.

Motivation Level

- Implemented as a slider.
- The value range is from **0.0 to 1.0**.
- The slider is divided into **10 fixed steps** with an increment of 0.1 (e.g. 0.0, 0.1, 0.2, etc.).
- Only these fixed values can be selected; intermediate values are not allowed.

Value Display

- A numeric value is displayed below the slider.
- The displayed value updates when the slider value changes.
- The numeric value is positioned at a fixed, centered location.
- The numeric value **does NOT** move together with the slider thumb.
- The numeric value acts as a static indicator, not as a tooltip or thumb-following label.

Default State

- 0.5 is used as the default initial value.
- On first launch, the slider thumb is positioned at 0.5, and the numeric value displays 0.5.

Last Updated

- Implemented as a non-interactive text field.
- Displays the date and time of the last settings update.
- Before any setting has been changed and saved, the field displays a placeholder text, for example: *“No changes yet”*.
- The date and time are displayed in the following format: MMM dd, yyyy at HH:mm (example: Dec 31, 2025 at 15:45)

🤔 What's Allowed?

- Standard Android/Jetpack libraries
- No 3rd party libraries are allowed or would be required to complete this challenge
- DataStore Preferences.

⚠️ What's not important

- Responsiveness across every device size or orientation is not mandatory.
- Animations or visual effects.
- Light/Dark mode support.
- Pixel-perfect UI accuracy is not required, as this challenge focuses on working with DataStore; standard UI components are sufficient, though you may optionally match the mockup more closely for additional practice

🔗 Useful Links for This Challenge

- DataStore
- Preferences DataStore Codelab
- Switch
- Dropdown Menu
- Focus in Compose
- Text Fields - UX With Material3
- Slider
- Stateful vs. Stateless Composables
- State Hoisting in Compose
- Managing State in Jetpack Compose (Codelab)

🏆 Submission & Rewards

- Successfully submitting this challenge via the **/submit-challenge** command on Discord grants you **100 XP**.
- Your submission must include:
  a. A **Gist link** with your implementation.
  b. A **screen recording** (max 20 seconds) showing:
    - Opening the settings screen with:
      - the initial Last Updated state (placeholder), or
      - a previously saved date and time.
    - Changing each available setting at least once.
    - Verifying that the Last Updated value updates to the current date and time after changes.
    - Closing and reopening the app.
    - Confirming that all settings and the Last Updated value are correctly restored.