Technical Requirements – Halloween Skeleton Puzzle

Challenge is accessible on Memberspot: https://pl-coding.mymemberspot.io/library/jx3b7Qik9ip5qpNI8IF2/FgYR5e2HfPtserJY1udz/TGwuciFEC7ZdBuakazvH/details

🎭 Scenario

This mini-app allows the user to assemble a skeleton by dragging its parts into the correct positions. The challenge task is to implement drag-and-drop interaction, precise slot validation, and visual feedback. Once the skeleton is fully assembled, it comes to life and a themed message appears on screen.

🎨 Figma Mockups

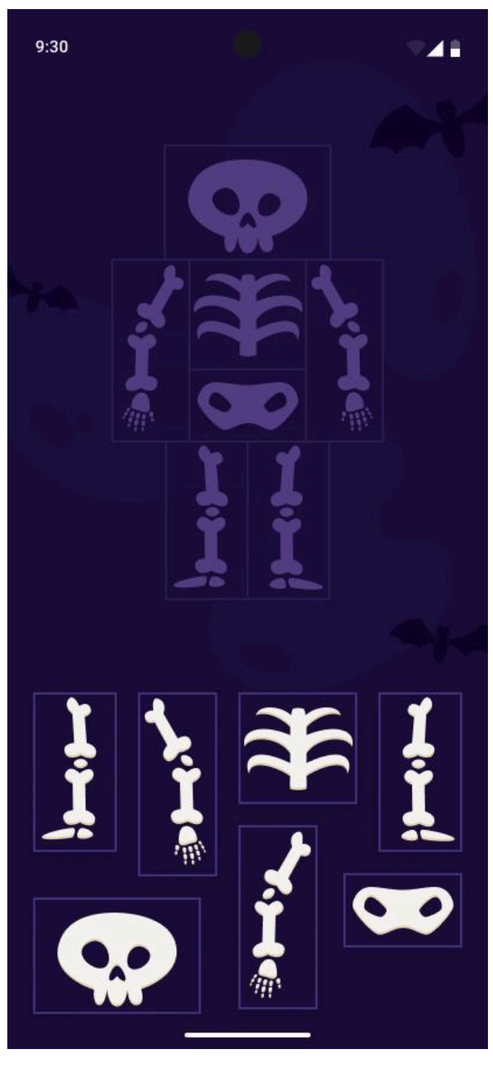
> https://www.figma.com/design/jBflfs7N6IphZE3Ue03Lot/Android-Halloween-Lab?node-id=0-1

🔤 Fonts - Road Rage

🎯 Feature Goal

Develop an Android mini-app where the user assembles a skeleton by placing all its parts into predefined slots.

📌 Requirements



UI Components

- Skeleton parts (skull, ribs, arms, legs, pelvis) are displayed at the bottom of the screen as separate cards placed in random order.
- Each part can be **dragged** into the puzzle area.

Puzzle Slots

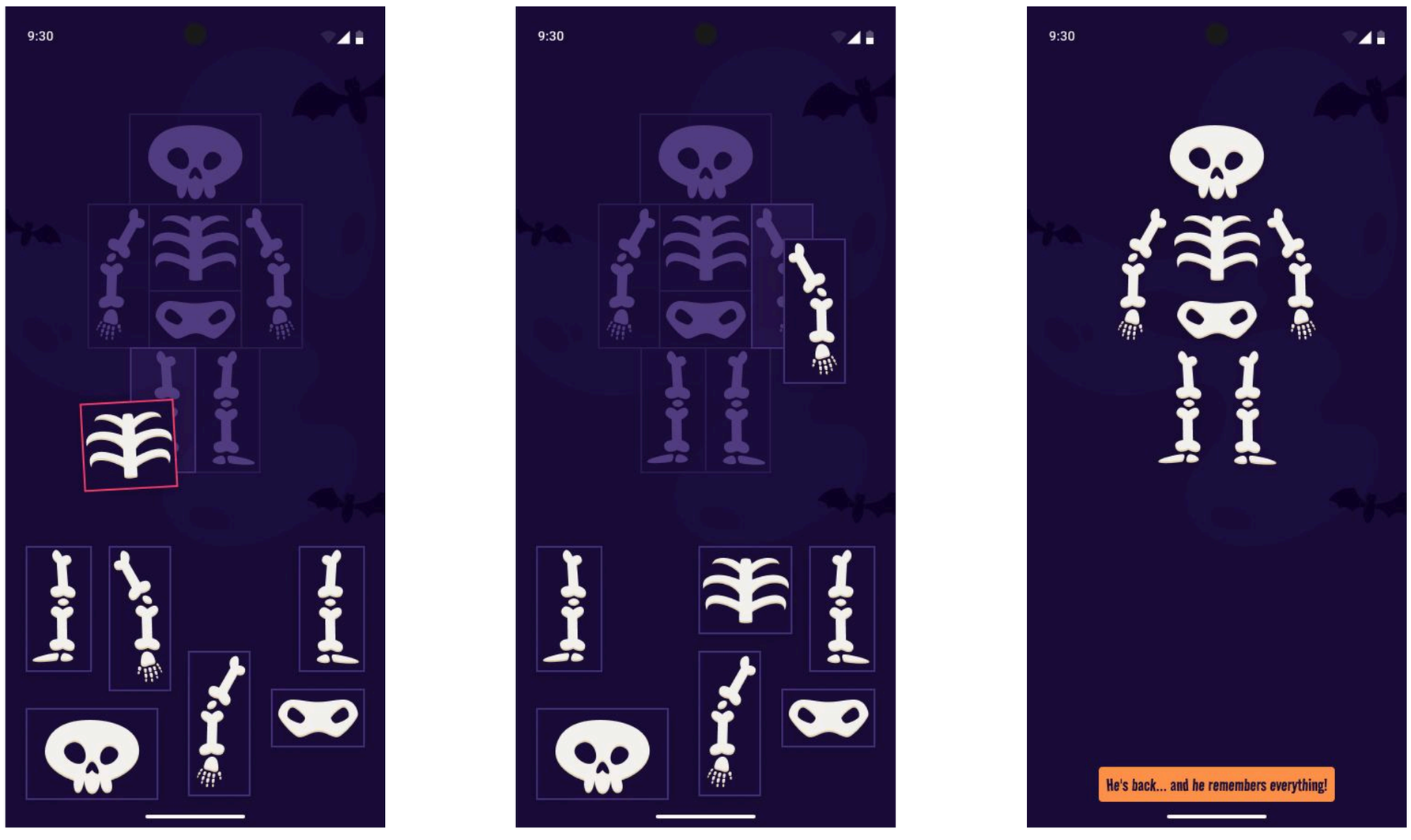
- Slot outlines are displayed in the center of the screen for each skeleton part.
- Each slot corresponds to a specific bone.
- Empty slots have a clear outline and remain visible even after being filled, but become more emphasized when completed.

Drag & Drop Logic

- The element size does not change while dragging.
- When an element is over a slot → the slot highlights.
- **Correct placement:**
  - The element snaps into the slot, fully fitting its shape.
  - The slot outline remains but becomes more emphasized.
- **Incorrect placement:**
  - The puzzle element flashes red.
  - The element returns to its original position.

Completion State

- Once all parts are correctly placed:
  - All slot outlines **disappear**, making the skeleton look complete.
  - The skeleton animates (moves head and arms).
  - A toast message appears: "*He's back... and he remembers everything!*"



🤔 What's Allowed?

- Standard Android/Jetpack libraries
- No 3rd party libraries are allowed or would be required to complete this challenge
- Use of Jetpack Compose (Modifier.dragGestureFilter or pointerInput for drag-and-drop).

⚠️ What's not important

- Responsiveness across all device sizes.
- Additional details of skeleton parts (basic shapes are enough).

🔗 Useful Links for This Challenge

- How to Implement Drag & Drop in Jetpack Compose
- Drag and drop
- Codelab: Drag and Drop in Compose
- Drag, swipe, and fling
- Understand gestures
- Advanced animation example: Gestures
- How You Use an AI Coding Agent

🏆 Submission & Rewards

- Successfully submitting this challenge via the /submit-challenge command on Discord grants you **300 XP**.
- Your submission must include:
  a. A link to a Gist with your implementation
  b. A screen recording (max 20 seconds) showing:
    - Dragging any part into an **incorrect** slot (showing red flash and reset).
    - Dragging several parts into the **correct** slots.
    - Completing the full skeleton assembly.
    - Displaying the final message and animation.