Tech Art Test – Design & Implementation Notes

Author: Anna Straister
Project: Tech Art Test (Home Screen, Bottom Bar, Settings, Level Completed)

1. Resolution and Scaling Approach

The original PSD layout had a resolution of 2484x5376, which equals approximately 1.93x the target resolution (1290x2796). In production, layouts are usually prepared at exact multiples of the target (2x, 3x, etc.), which allows for whole Pixels Per Unit (PPU) values — for example, 200 or 300 — and avoids fractional scaling. For this assignment, I decided to follow a standard 2x asset workflow with PPU = 200. This keeps the project technically clean and eliminates any floating-point rounding errors or desynchronization during scaling. As a result, the proportions in Unity fully match the PSD, even if the visual size slightly differs (since the PSD is not exactly 2x). In a real production environment, I would recommend preparing layouts strictly at 2x or 3x to maintain a predictable and unified pipeline.

2. Font Assets and Localization Readiness

For localization, I created a Font Asset based on Lato-Black, which includes:

- Latin and Cyrillic alphabets,
- numbers and standard punctuation,
- extended character sets for major European languages (German, French, Spanish, Italian).

Some exotic symbols are not present in the base typeface, so in production I would add fallback fonts. Font assets were generated for TextMeshPro using the provided fonts Yorkie-SemiBold and Lato-Black. I imported .ttf versions because they have the best compatibility with Unity, although .otf formats are equally valid. This decision ensures the interface is ready for future localization and avoids layout breaks when text expands in other languages.

3. Bottom Bar Architecture and Animation Logic

I implemented the Bottom Bar using Toggles and a ToggleGroup so that only one tab can be active at a time. All visual behavior — expanding the active tab, shifting neighbors, bouncing icons, revealing labels, and highlights — is done entirely through Animator, without extra C# logic. For tab expansion, I animated LayoutElement.preferredWidth. Highlights and label fades were handled per tab, creating a smooth, continuous transition similar to the reference animation.
This approach provides:

- a clean and modular hierarchy,
- simple visual tuning directly in Unity,
- and independence from code for visual designers.

4. Project Structure and Prefab System

All folders and assets are logically organized for quick navigation. Core UI elements (buttons, panels, icons) were converted into reusable prefabs, so updates to base components automatically propagate to all scenes. This makes the project scalable and easy to maintain, reducing duplication and minimizing the risk of inconsistencies. The structure reflects a production-oriented workflow and prepares the project for team collaboration.

5. Texture Optimization and Asset Quality

I always focus on efficient texture usage:

- packing sprite atlases where possible,
- using 9-slice sprites for scalable panels,
- employing universal white textures recolored in Unity for gradients, overlays, and shadows.

A simple 4×4 white pixel is often enough to build backgrounds or subtle transitions while saving texture memory.
All assets were imported in 2× resolution, then compressed via Unity's texture settings.
Before finalizing, I test on a real device to ensure the best balance between sharpness and memory footprint.

6. UI Interactivity and Screen Flow

The interface is fully interactive:

- Bottom Bar buttons are toggles controlled by a shared ToggleGroup;
- Settings Window opens via the gear icon and closes via the “X” button;
- Level Completed Screen can be opened from the Home screen and closed via the Home button.

All opening/closing logic is animated for smooth transitions and handled through clean controller scripts. The result is a responsive, consistent, and predictable UI flow — simple to expand with additional screens later.

7. UX Validation and Visual QA

Before final delivery, I always review:

- text containers (to ensure localizations fit properly),
- button click zones (for comfortable interaction),
- canvas hierarchy (to prevent unwanted blocking or overlap),
- Safe Area support, general UI behavior across different screens
- draw call count

These checks guarantee that the interface behaves correctly on various devices and aspect ratios, both visually and technically.

8. Visual Effects and Animation

In the project, I implemented basic Particle Systems and animations to bring the interface to life — adding reward effects, smooth button transitions, and panel hiding animations. Everything is made in a way that it can be easily extended: changing particle parameters, adding extra effects, or making animations more complex. The main goal was to show my general approach and understanding of visual effect principles in UI.

9. Production Thinking and Collaboration

There are always multiple ways to implement UI and visual effects. I choose approaches based on context — project goals, deadlines, quality expectations, and optimization needs. Sometimes speed and simplicity matter most, other times precision and visual detail take priority. In a production team, I usually discuss priorities with producers, artists, and engineers to find the most effective, maintainable, and scalable solution. For me, the balance between creativity, structure, and collaboration defines a strong tech art workflow.