Process & Decision Documentation

Project/Assignment Decisions

Side Quests and A4 (Individual Work)

Key Decision:

- I decided to transform Example 5 into a vertical underwater camera experience instead of a traditional platformer.
- A key decision was focusing on the camera as the main experience, keeping the diver centred while the world scrolls downward.
- I added bubbles as hazards and glowing stars as interactive discoveries to better meet the reflective camera requirement.

Role-Based Process Evidence

Name: Jenny South

Role(s): Designer + Developer (Individual Side Quest)

I designed the underwater concept, pacing, and interactive elements, and experimented with all systems, including camera movement, collision, lives, and game states.

Goal of Work Session

I wanted to prove that I could:

- Modify an existing modular camera system without breaking it
- Create a world larger than the screen with vertical scrolling
- Use motion and pacing to create a calm, reflective feeling
- Add interactive elements (bubbles + stars) while maintaining atmosphere
- Implement start and game over states cleanly

Tools, Resources, or Inputs Used

- p5.js (movement/animation/interaction)
- Visual Studio Code
- ChatGPT Version 5.2
- Example 5 starter code (week 05)

Process Evidence:

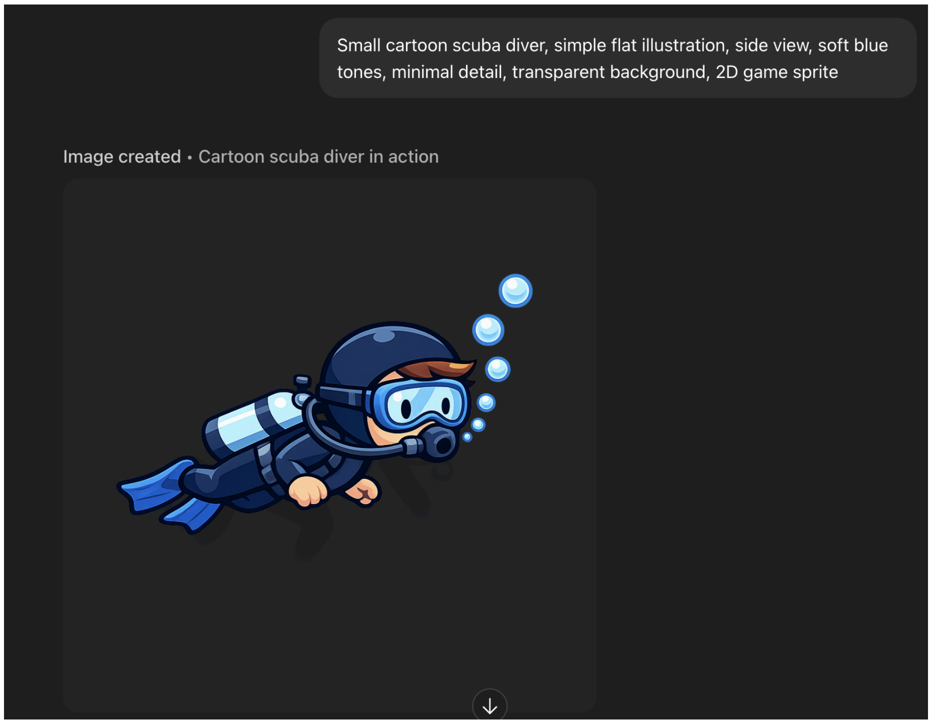
Throughout the creation of this sidequest, I kept a Google Doc open with all my ideas and prompts. This is the content of the doc and my process:

Things I want:

- Bubbles floating up
- Avoid hitting the bubbles
- I want to get a photo of a little diver
- Blue background
- Start with three lives, and when you touch a bubble, you lose one life lost

My Prompt:
Create a vertical underwater scrolling game in p5.js. The camera slowly scrolls downward through a world larger than the screen. The background should be a soft blue ocean gradient. Bubbles float upward from the bottom of the screen. A small diver image stays near the center and can move left and right. The player starts with 3 lives. If the diver touches a bubble, one life is lost, and the bubble disappears. The movement and pacing should feel calm and meditative.

Used AI to generate the diver image:



Small cartoon scuba diver, simple flat illustration, side view, soft blue tones, minimal detail, transparent background, 2D game sprite

Image created • Cartoon scuba diver in action

Saved as a PNG and used inside the game

Prompt:
I am building my Week 5 sidequest for GBDA302 using my existing modular camera system (Camera2D, WorldLevel, sketch.js).

- I want to create an Underwater Descent experience.
    - Requirements:
        - The world must be larger than the screen.
        - The camera slowly scrolls downward in a calm, meditative way.
        - Blue gradient ocean background.
        - Bubbles float upward continuously. (This is to meet this requirement - Hide small interactive symbols or objects for the camera to "discover.")
        - A small diver image is displayed near the center and can move left/right.
        - The player starts with 3 lives.
        - When the diver touches a bubble, one life is lost, and the bubble disappears.
        - The pacing should feel slow and atmospheric, not intense.

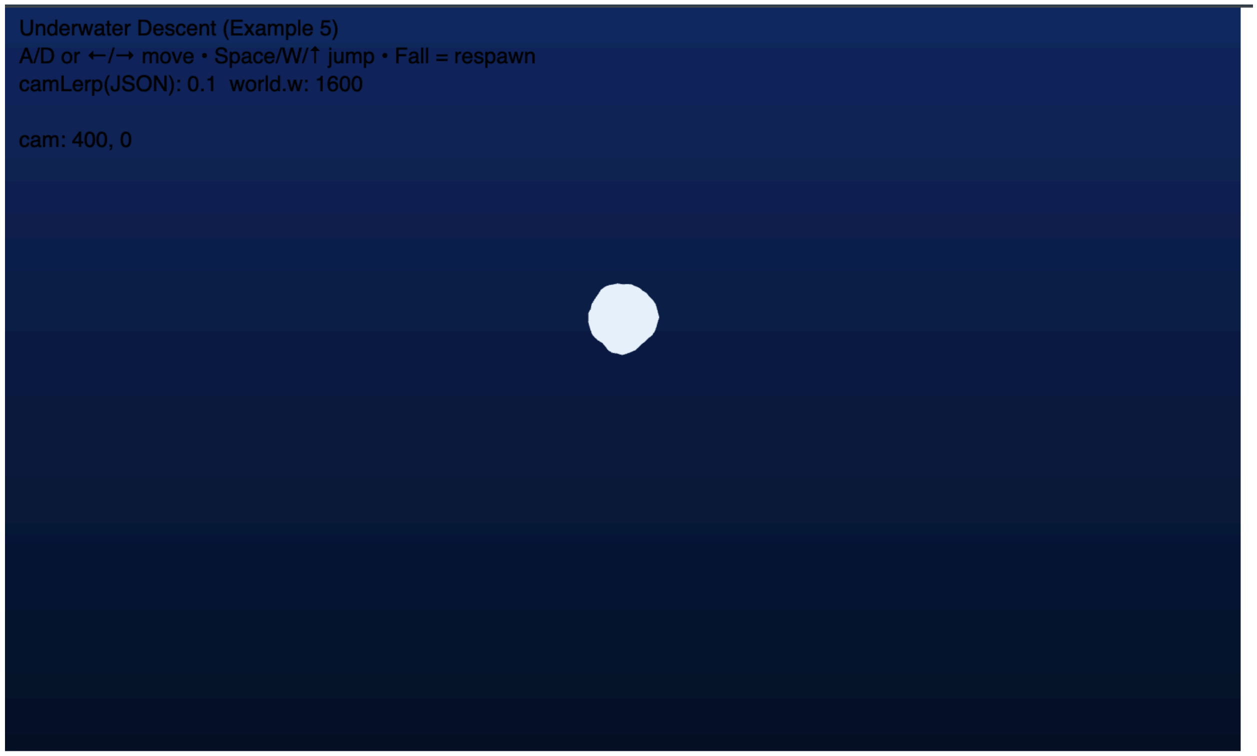
Please modify my existing modular structure instead of rewriting everything from scratch. Show me exactly what to change in each file, and explain as you go so I can learn how the process works and implement it myself next time.

After Completion:

New Ideas - Needs to be more immersive:

- Starting splash screen and end screen

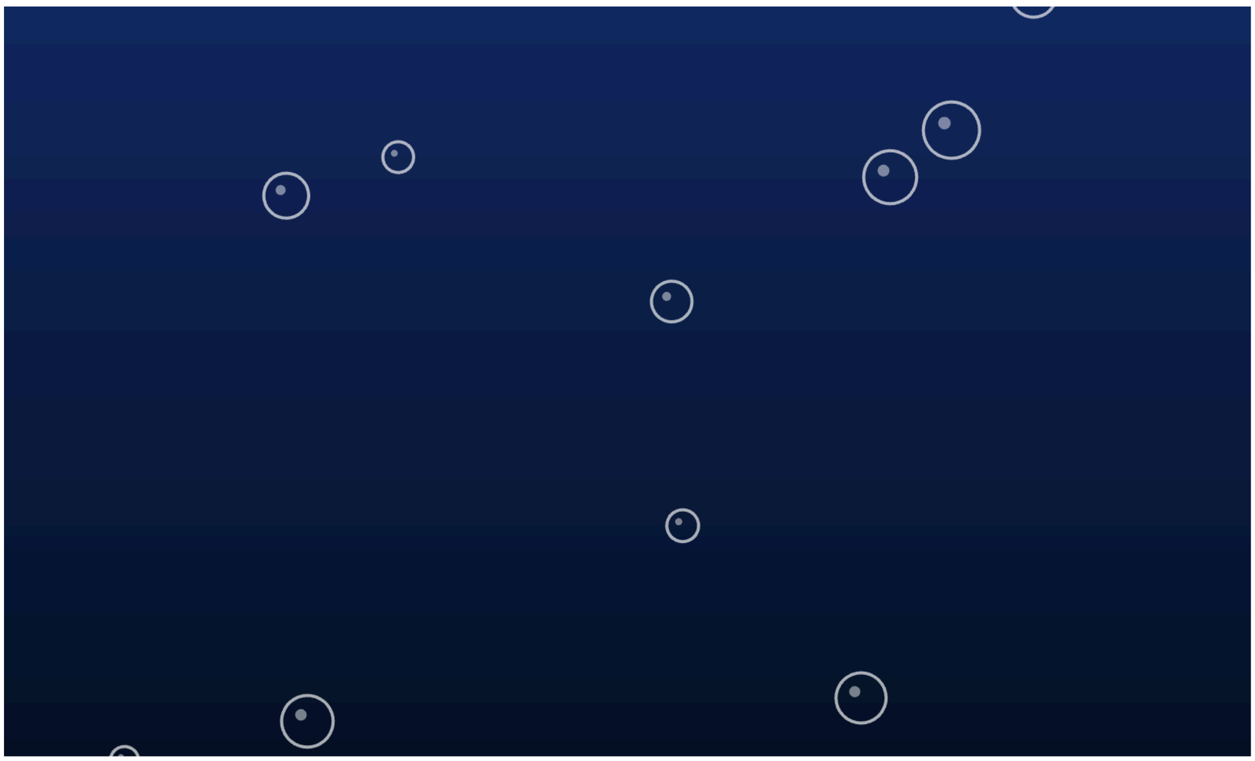
Beginnings:



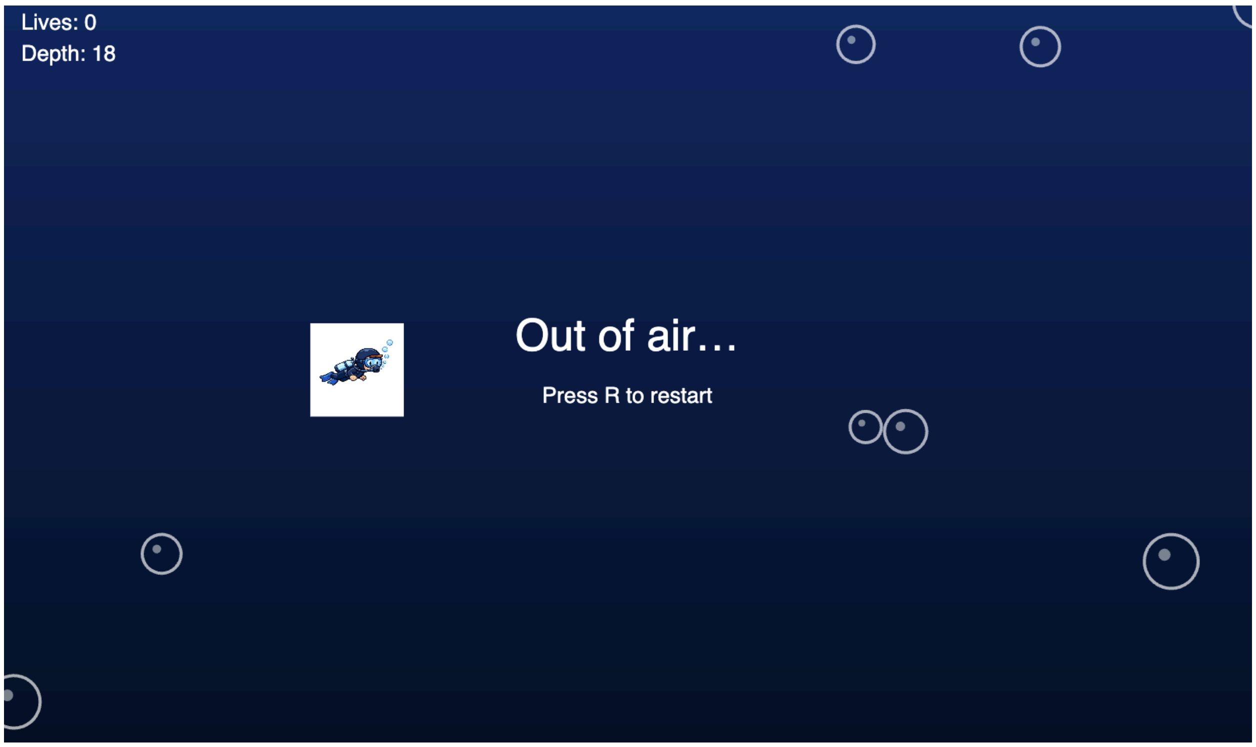
```
Underwater Descent (Example 5)
A/D or ←/→ move • Space/W/↑ jump • Fall = respawn
camLerp(JSON): 0.1  world.w: 1600

cam: 400, 0
```

Implementing Bubbles:



The code it gave me had no movement, and I was struggling to load my image



Use AI to Help With Some De-Bugging:

- The diver only moves side to side and is very stiff. I want to make a change to that

Tweaks:

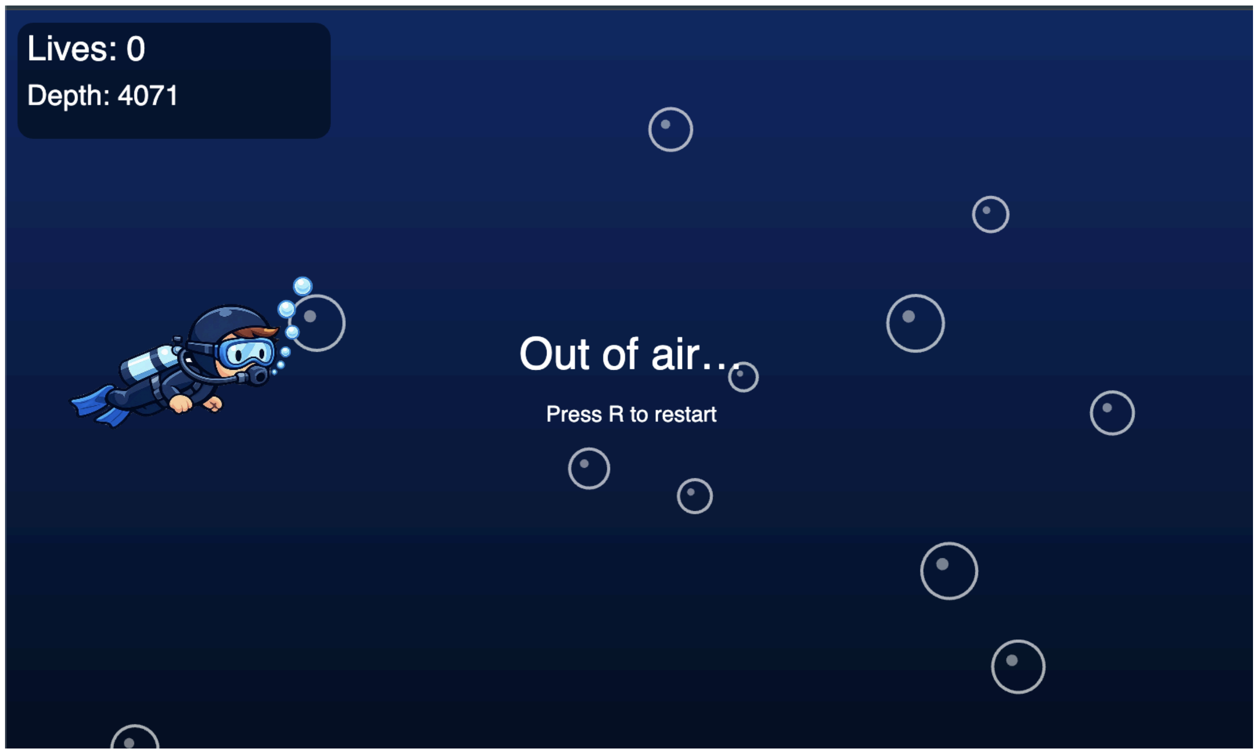
- Make the image bigger
- More populated bubbles
- Go down at a faster pace
- Show lives and depth more prominently in the corner
- Have the camera move down more clearly

Put into a prompt:

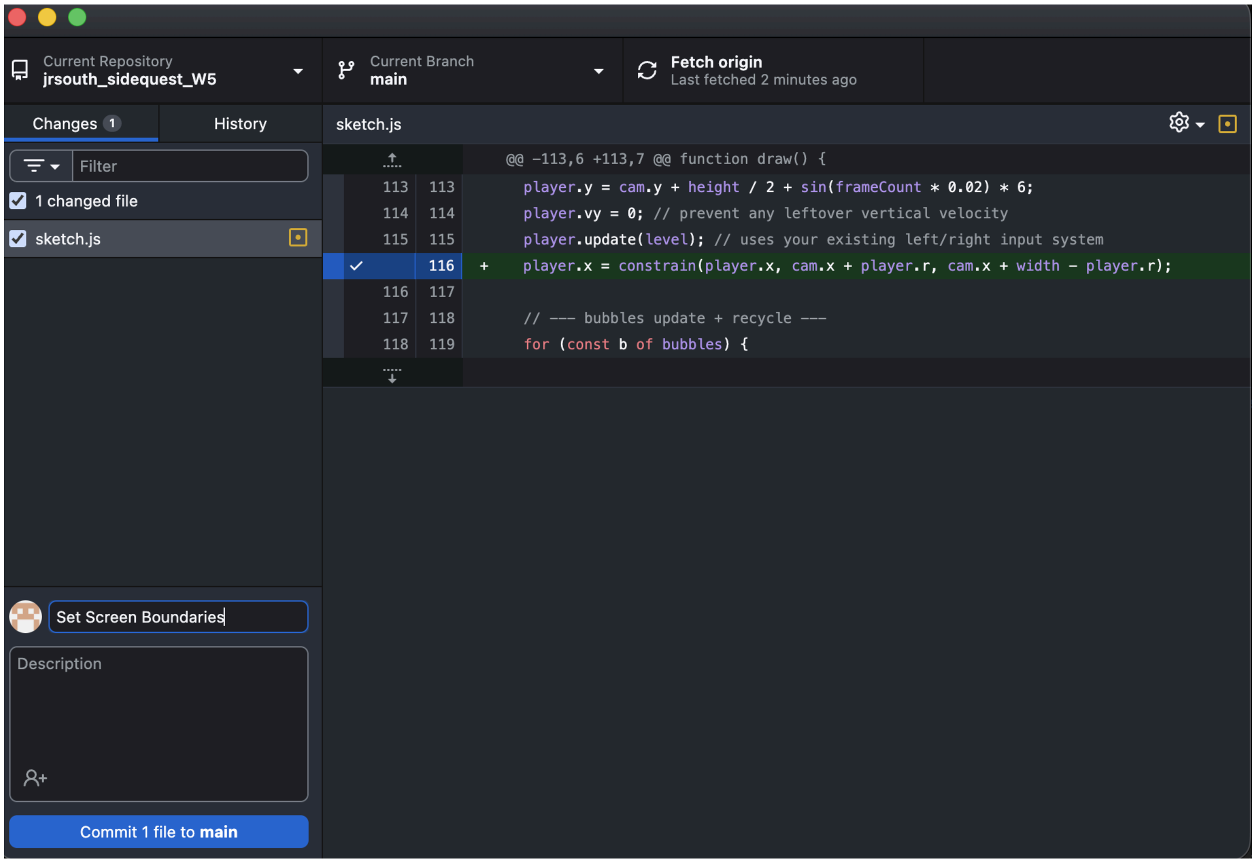
- Modify my underwater scrolling p5.js game with the following changes:
    - Increase the number of bubbles so the scene feels more populated.
    - Make the camera descend faster.
    - Display lives and depth more prominently in the top corner.
    - Make the downward camera motion feel more obvious and continuous.
    - Keep the structure modular and only adjust what is necessary.

What I Added On My Own:

- Wanted to add a starting splash screen and an ending splash screen
- Didn't want him to be able to go off-screen, so made canvas boundaries



Setting Screen Constraints



Current Repository
jrsouth_sidequest_W5

Current Branch
main

1 changed file
sketch.js

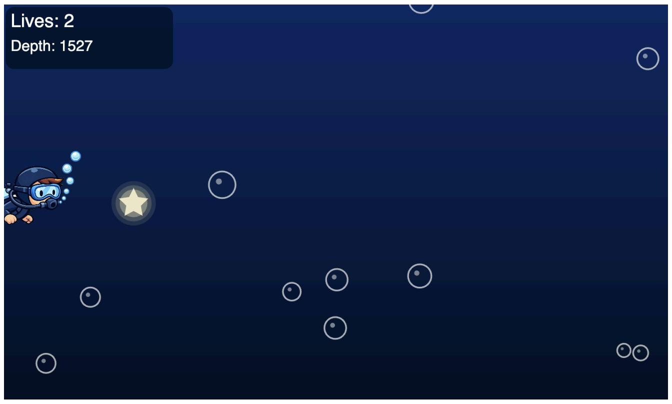
```
@@ -113,6 +113,7 @@ function draw() {
      player.y = cam.y + height / 2 + sin(frameCount * 0.02) * 6;
      player.vy = 0; // prevent any leftover vertical velocity
      player.update(level); // uses your existing left/right input system
+     player.x = constrain(player.x, cam.x + player.r, cam.x + width - player.r);

      // --- bubbles update + recycle ---
      for (const b of bubbles) {
```

Set Screen Boundaries

Realized there's not a lot of discovery involved,
- Wanted to add stars that would be discovered and give life back

Prompt:
Modify my existing underwater scrolling p5.js game to include glowing collectible stars.
- Stars are small glowing symbols placed throughout the world.
- They appear occasionally (not too frequently).
- They pulse softly to feel magical/meditative.
- When the diver touches a star, the star disappears, and the player gains 1 life.
- Cap lives at a maximum of 3
- Keep the system modular and consistent with my existing Bubble system.
- Show exactly what code to add and where (new class if needed).
- Do not rewrite my entire project; only add what is necessary.



GenAI Documentation

Date Used: Feb 13, 2026

Tool Disclosure: ChatGPT 5.2 (OpenAI)

Purpose of Use: To understand how to modify my existing modular camera system, implement collision detection cleanly, add game states, and debug image loading and state logic. Also, to generate an image.

Summary of Interaction: All creative decisions came from my own brainstorming. I used GenAI to explain where to place code and how to integrate features into my current file structure. It helped me structure the state system (start, play, gameover), adjust camera behaviour, and properly constrain the diver to the screen.

Human Decision Point(s): I chose the underwater theme, the pacing, the use of stars as discoveries, and the balance between hazards and calm movement. I decided how often stars appear and how the life system works. All tuning and aesthetic decisions were made by me.

Integrity & Verification Note: All changes were manually implemented and tested in p5.js. I tested:

- That the world scrolls smoothly downward
- That bubbles correctly remove a life
- That stars correctly restore a life
- That lives do not exceed the cap
- That start and game over screens function properly
- That the diver cannot leave the screen

I played through multiple times to ensure pacing felt calm rather than chaotic.

Scope of GenAI Use: GenAI was used to clarify coding structure and debug implementation issues. It did not generate the core idea, theme, or interaction concept. I provided detailed prompts and decided what to implement.

Limitations or Misfires: At first, the game-over screen appeared twice due to state logic timing. I resolved this by restructuring the draw loop and separating state checks. I also initially had issues with image loading and positioning before properly placing the image inside preload() and adjusting screen-space drawing.

Summary of Process (Human + Tool)

I began with a simple idea: a diver descending underwater with bubbles rising. I gradually layered systems on top of that: camera motion, collision, lives, stars, and game states. GenAI helped explain how to structure the changes, but all creative and pacing decisions were made by me.

Decision Points & Trade-offs

I chose to keep the diver mostly centred so the camera becomes the focus. I also chose to add stars to balance the hazard system and strengthen the "discovery" bonus requirement. I avoided adding too many mechanics because I wanted the experience to remain reflective, not arcade-like.

Verification & Judgement

After implementation, I confirmed that:

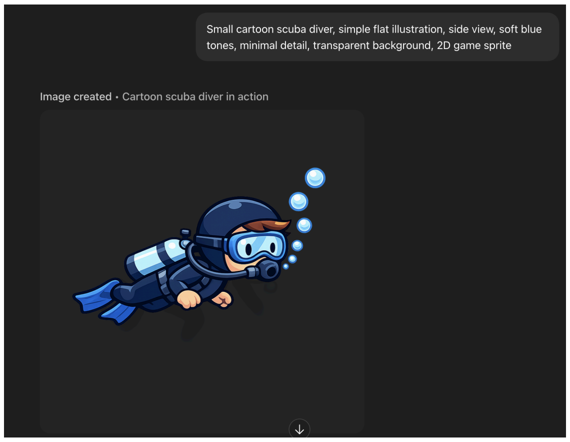
- The world is larger than the screen
- The camera scrolls smoothly
- Motion and pacing create a calm descent
- Interactive objects support discovery
- Game states function correctly

Limitations, Dead Ends, or Open Questions

Balancing calm pacing with hazard interaction required iteration. Too many bubbles made the experience feel stressful. I adjusted density and scroll speed to maintain a reflective tone. If expanded further, I would explore adding subtle audio or deeper visual layers to enhance immersion.

Appendix

https://chatgpt.com/share/698eb2e8-99cc-8007-9a45-f081c0780992



Small cartoon scuba diver, simple flat illustration, side view, soft blue tones, minimal detail, transparent background, 2D game sprite

Image created • Cartoon scuba diver in action