Process & Decision Documentation

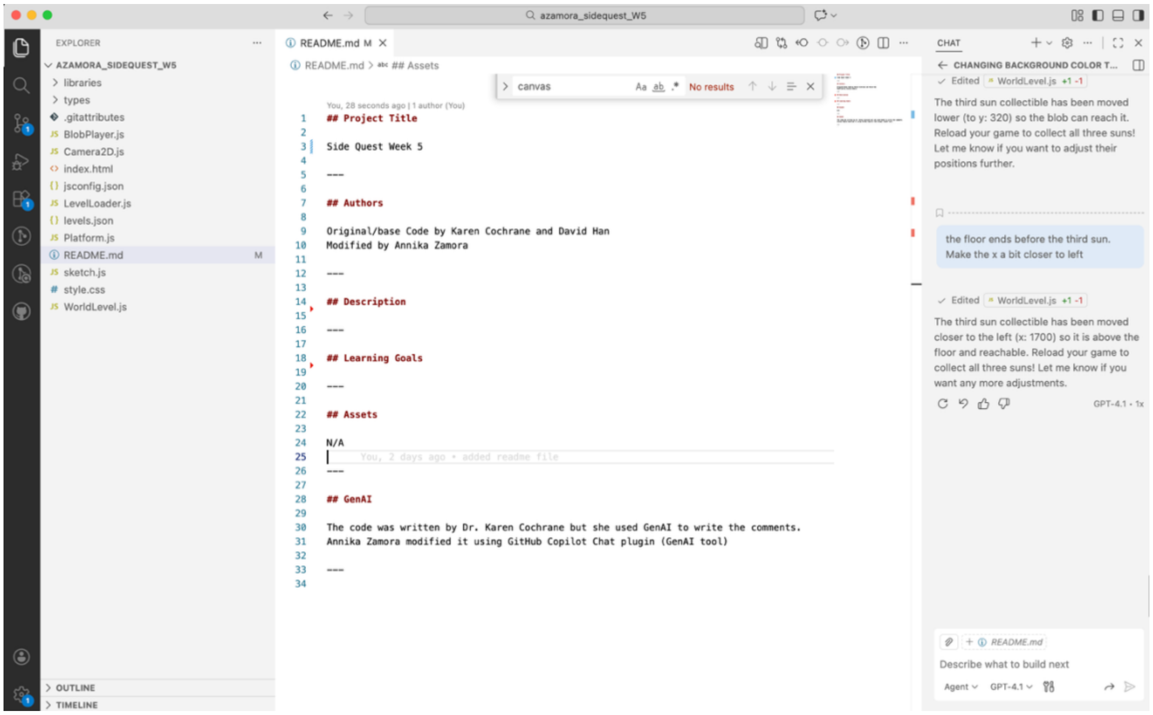
Project/Assignment Decisions

Side Quests and A4 (Individual Work)

- Since this is a side quest, I wanted to focus more on the structure of the code/functionality, specifically the concepts we learned in class for that week, rather than aesthetics and visual appeal
- I obviously want it to still look somewhat good, especially since I needed to portray an emotion, however, aesthetics was not my main priority
- This allowed me to shift my focus towards the meditative camera experience, which was a major part of the in-class lesson for this side quest week

Role-Based Process Evidence

Editing the readme file



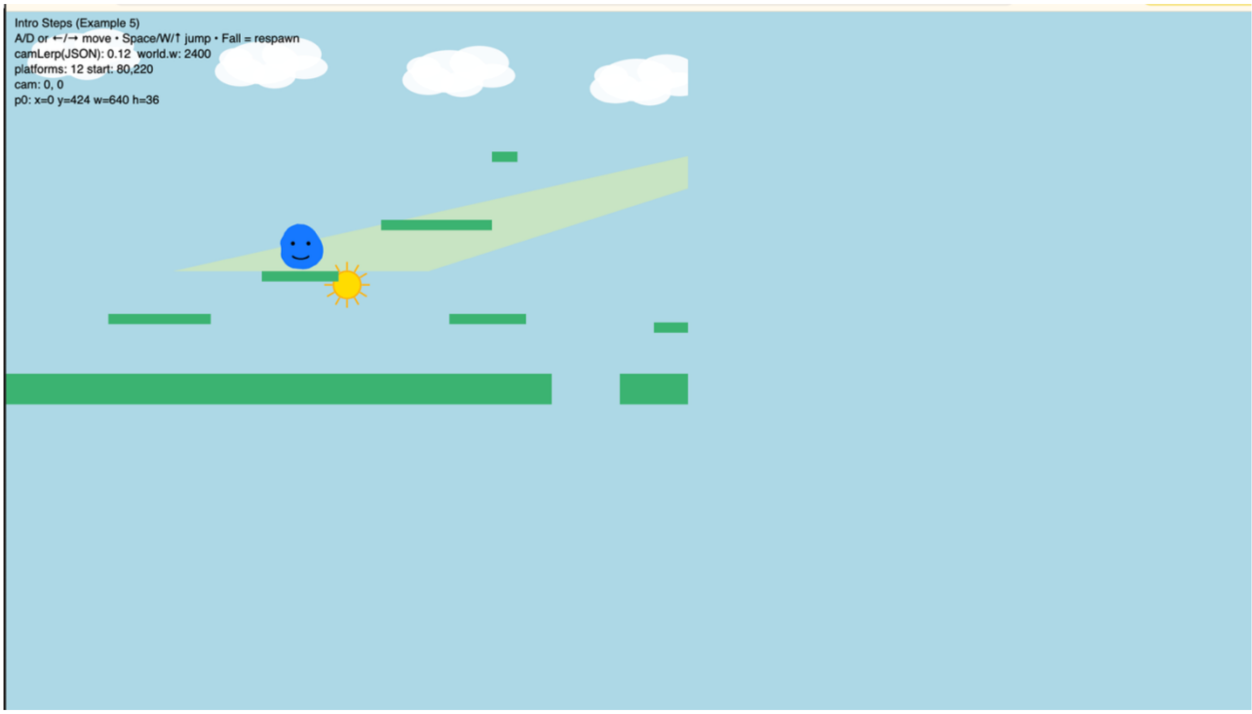
```
## Project Title

Side Quest Week 5

---

## Authors

Original/base Code by Karen Cochrane and David Han
Modified by Annika Zamora

---

## Description

---

## Learning Goals

---

## Assets

N/A

---

## GenAI

The code was written by Dr. Karen Cochrane but she used GenAI to write the comments.
Annika Zamora modified it using GitHub Copilot Chat plugin (GenAI tool)

---

```

The third sun collectible has been moved lower (to y: 320) so the blob can reach it. Reload your game to collect all three suns! Let me know if you want to adjust their positions further.

the floor ends before the third sun. Make the x a bit closer to left

The third sun collectible has been moved closer to the left (x: 1700) so it is above the floor and reachable. Reload your game to collect all three suns! Let me know if you want any more adjustments.

Testing how the code functions after every change/prompt made



```
Intro Steps (Example 5)
A/D or ←/→ move • Space/W/↑ jump • Fall = respawn
camLerp(JSON): 0.12  world.w: 2400
platforms: 12 start: 80,220
cam: 0, 0
p0: x=0 y=424 w=640 h=36
```

Goal of Work Session

- Modifying the example code using the built-in visual studio code GenAI tool
- Create detailed and specific prompts to meet the requirements of the side quest

Tools, Resources, or Inputs Used

- Visual Studio Code CoPilot addon
- LEARN week 5 code example

GenAI Documentation

If GenAI was used (keep each response as brief as possible):

Date Used: February 20 2026

Tool Disclosure: Visual Studio Code CoPilot addon

Purpose of Use: The overall use was to modify the starter code to follow the prompt of the week 5 side quest requirements. More specifically, the purpose of use was to be able to input ideas of what I wanted the code to display and execute, and have an output of written and working code to demonstrate those ideas.

Summary of Interaction: I came up with each idea/intended output I wanted GenAI to produce → I created then inputted the prompt → the GenAI tool modified the code to demonstrate what I inputted → I tested the code after each prompt was completed → I made changes to the output or kept the modifications from the GenAI output based on how satisfied I was with what the GenAI tool output

Human Decision Point(s): Most of the time, I prompt for an idea through multiple prompt inputs, however, when I notice that the idea isn't working out the way I initially intended for it to, or if I change my mind about a concept, I usually back up a few steps by rejecting all changes of that section I am trying to complete, so that I can start from scratch (in that section of code, not the entire project) and refine my prompts based on what errors I had previously.

Integrity & Verification Note: After every change that the GenAI tool made, I checked by running the code. This was especially important since sometimes the GenAI would say they did what I prompted for, but didn't actually modify the code to do so. This process was also important since sometimes the GenAI interprets the prompts differently than I had intended for. Testing after each prompt helped break down the code into sections, making it easier to focus on one thing at a time. It also made it easier to fix errors and bumps in the code/GenAI output.

Scope of GenAI Use: Before even creating prompts for GenAI, I came up with an overall idea of what I wanted the code to look like in its final product. I chose the emotion (joy) to demonstrate, and decided what features I wanted to add in to show that (light blue background colour, clouds, sun ray, smiley face on blob, etc.). This brainstorming process done beforehand, was completed without the use of any GenAI tools.

Limitations or Misfires: Creating the sun ray was the most time-consuming and frustrating part. When making the sun ray, the tool had a hard time executing what I had hoped for, meaning I had to spend a lot of time prompting for small changes within the sun ray. It had a hard time understanding what I meant by the bottom floor of the game, as it drew the sun ray to extend past the game canvas where you could see it stick out, it drew it too high above the bottom floor of the game screen leaving a gap, and also matched the sun ray to extend to each platform height. It also wouldn't centre the sun ray around the blob, so I had to experiment with the width of the shape, and the shape itself.

Summary of Process (Human + Tool)

Describe what you did, focusing on process rather than outcome. This may include:

- Testing/running the code after every change GenAI made

- Worked through the code in small sections to allow for my prompts to be highly detailed and focused on one aspect at a time
- Based each prompt on what did or didn't work in the previous prompts

Decision Points & Trade-offs

Describe one or two key decisions you made:

- Decided to prioritize functionality over aesthetics
- I wanted to focus on the core course concepts of that week, which were camera systems
- Since visuals are important within this concept, I still made sure it looked nice, however I chose a simpler design, rather than a highly detailed one
- This allowed me to learn, adapt, and apply this new concept in my own example in a simple way, where I can then apply it in a more detailed way in future projects

Verification & Judgement

Explain how you evaluated whether your decision or change was appropriate:

- Decided to keep or change each modification that GenAI made by running the code, then comparing it to the initial idea I had in mind when I created the prompts
- Used what I learned in previous lectures to compare with in terms of course concepts: Does the code demonstrate the concepts we learned in class? (In this case, the moving background)

Limitations, Dead Ends, or Open Questions

- I originally wanted the sun ray to stay touching the floor the entire time
    - However, I had a hard time getting it to work, even after lots of prompt refinement
    - I decided to change/simplify it and have the bottom of the sun ray match the y-level of the bottom of the blob instead of the floor, which ended up looking and working out better

Appendix

Input: make the background light blue

Output: The background is now set to light blue. Refresh your page to see the change. Let me know if you want a different shade or further adjustments!

Input: the background of the canvas where the ball is should be light blue

Output: The canvas background where the ball is has been updated to light blue. This is done by changing the level's background color in levels.json. Reload your game to see the effect! If you want a different shade or want this for all levels, let me know.

Input: make the platforms and the floor green to look like grass

Output: The platforms and floor are now green to resemble grass. Refresh your game to see the new look! If you want a different shade of green or more details, just let me know.

Input: put white clouds at the top

Output: White clouds have been added at the top of the canvas. Reload your game to see them! If you want more clouds or different shapes, just let me know.

Input: put more clouds until the end of the level. Add more ellipses to each cloud

Output: There are now many more clouds stretching across the entire level, and each cloud is fuller with more ellipses. Reload your game to see the enhanced sky! Let me know if you want even more variety or specific cloud arrangements.

Input: the clouds appear to be shaking vertically. Make sure they are in the same y position

Output: The clouds will now stay at fixed y positions and no longer shake vertically. Reload your game to see the stable clouds! Let me know if you want further adjustments.

Input: put a sun at the end of the level. Make it look like there is a ray of sunshine always beaming on the ball, coming from the sun

Output: A sun now appears at the end of the level, and a sunbeam always shines on the ball as it moves. Reload your game to see the effect! Let me know if you want to adjust the sun or the beam style.

Input: put the sun more in the centre of the level. make the ray of sun wider.

Output: The sun is now positioned near the center of the level, and the sunbeam shining on the ball is much wider. Reload your game to see the updated effect! Let me know if you want further adjustments.

Input: make the ray touch the floor

Output: The sunbeam now extends all the way to the very bottom of the world, not just to the platform floor. Reload your game to see the updated effect! Let me know if you need any more changes.

Input: extend the ray to the bottom floor, not the platorm floor

Output: The sunbeam now extends all the way to the very bottom of the world, not just to the platform floor. Reload your game to see the updated effect! Let me know if you need any more changes.

Input: sun beam does not touch the bottom yet

Output: The sunbeam now extends to the very bottom of the canvas, ensuring it always touches the visible floor. Reload your game to see the effect! Let me know if you need any more adjustments.

Input: height of the sun ray goes under the floor, also does not go over the ball

Output: The sunbeam now starts at the sun, passes over the ball, and stops exactly at the floor (not below it). It will always cover the ball as it moves. Reload your game to see the improved sunray! Let me know if you need further tweaks.

Input: there is a rectangle under the triangle part of the sun ray. Extend the sunray to touch the bottom floor

Output: The sunray is now a triangle from the sun to the left and right of the ball at the bottom floor, with no rectangle underneath. The beam always touches the floor. Reload your game to see the corrected sunray! Let me know if you want any more adjustments.

Input: what is the y level of the bottom of the sun ray

Output: The y level of the bottom of the sun ray is set to the world's height, which is 360 (from levels.json: "world": { "w": 2400, "h": 360, ... }). So, the sun ray touches y = 360. If you want it to be a different value, let me know!

Input: what is the y level of the bottom of the canvas

Output: The y level of the bottom of the canvas is 480, since your canvas is created with createCanvas(VIEW_W, VIEW_H) and VIEW_H is set to 480. So, the bottom of the visible area is y = 480.

Input: make the y level of the bottom of the sun ray 480

Output: The bottom of the sun ray is now set to y = 480, matching the bottom of the canvas. Reload your game to see the sunray reach the very bottom! Let me know if you need any more changes.

Input: make the sun ray a bit wider to cover the blob the entire time

Output: The sun ray is now wider (beam width set to 300) to ensure it always covers the blob as it moves. Reload your game to see the improved coverage! Let me know if you want it even wider or need more adjustments.

Input: match the sun ray to the bottom of the blob instead

Output: The sun ray now ends exactly at the bottom of the blob (player), matching its movement. Reload your game to see the sunbeam always touch the bottom of the blob! Let me know if you want any more adjustments.

Input: put a smiley face on the blob

Output: A smiley face has been added to the blob, with two eyes and a curved smile. Reload your game to see the happy blob! Let me know if you want to adjust the face or add more features.

Input: every time the blob jumps, make the face a happy open mouth

Output: Now, every time the blob jumps, its face will show a happy open mouth. When it's not jumping, it will display the regular smile. Reload your game to see the animated face! Let me know if you want to tweak the expression further.

Input: place mini suns for the player to collect. Spread 3 around the map so that players can find them and touch them to collect it

Output: Three collectible mini suns are now spread around the map. When the player touches a mini sun, it disappears (collected). Reload your game to find and collect them! Let me know if you want to display a score or change their locations.

Input: make sure that the sun disappears when the blob touches the sun at any point

Output: Now, the mini sun will disappear as soon as any part of the blob touches any part of the sun (using bounding circle collision). This ensures reliable collection. Reload your game to see the fix! Let me know if you want to add a collection sound or score.

Input: the blob sits on top of the first sun and doesnt disappear. Make sure the sun rays count as collision, not just the circle part of the sun

Output: Now, the sun collectible will disappear if the blob touches any part of the sun, including the rays (using a larger collision radius). The blob will no longer be able to sit on top of the sun without collecting it. Reload your game to see the fix! Let me know if you need further adjustments.

Input: The third sun is not reachable. Make sure the blob is able to reach it.

Output: The third sun collectible has been moved lower (to y: 320) so the blob can reach it. Reload your game to collect all three suns! Let me know if you want to adjust their positions further.

Input: the floor ends before the third sun. Make the x a bit closer to left

Output: The third sun collectible has been moved closer to the left (x: 1700) so it is above the floor and reachable. Reload your game to collect all three suns! Let me know if you want any more adjustments.