

Process & Decision Documentation

Project/Assignment Decisions

Key Design Decision: Implementing a Score-Based Branching Story System

I implemented a score tracking system that accumulates points based on player choices throughout the story. Each decision adds different point values (+5, +10, +15, +20), creating meaningful weight to player actions. The final score (threshold: 30 points) determines which of two endings the player receives. This design transforms the interactive story from a simple linear narrative into a consequential experience where choices matter and players can replay to explore different outcomes.

Role-Based Process Evidence

Name: Rini Lu

Role(s): Individual contributor

Primary responsibility: Game architecture design and code implementation

Goal of Work Session

Create a multi-screen interactive story game using p5.js with state management, decision-tree branching, and a player score tracking system (bonus feature).

Tools, Resources, or Inputs Used

- Week 3 lecture slides on game states and architecture
- Week 3 example code (win/lose game structure)
- p5.js documentation
- Live Server for testing and debugging

GenAI Documentation

Date Used: February 1, 2026

Tool Disclosure: Claude Sonnet 4.5

Purpose of Use: Brainstorming story concepts and double-checking code structure

Summary of Interaction: I discussed potential story themes with Claude and asked for feedback on whether my multi-file architecture followed the Week 3 pattern correctly.

Claude helped me brainstorm the forest setting and suggested that a score-based system would be simpler to implement than tracking three separate stats (health, trust, karma).

The assistant also reviewed my file structure to confirm that I was correctly routing draw() and mousePressed() calls through main.js.

Human Decision Point(s): I independently chose the magical forest theme and wrote all story content myself. I decided to use a single score variable instead of multiple stats for simplicity. I determined the point values for each choice (+5, +10, +15, +20) and set the 30-point threshold for the good ending through testing. I also decided on the three-scene structure and wrote all the scene descriptions and choice text without AI assistance.

Scope of GenAI Use: GenAI was used only for brainstorming initial concepts and verifying that my code structure matched the Week 3 architecture pattern. I independently implemented all code files (main.js, start.js, scene1.js, scene2.js, scene3.js, good_ending.js, bad_ending.js), wrote all story content, designed the user interface with buttons and score display, and debugged all functionality through testing in Live Server.

Summary of Process (Human + Tool)

- Reviewed Week 3 examples to understand state management and routing patterns
 - Brainstormed story theme and setting with Claude
 - Designed three-scene structure with branching choices
 - Implemented score tracking system and point values
 - Created all seven game files following Week 3 architecture
 - Asked Claude to double-check code structure against requirements
 - Tested and refined gameplay in Live Server until satisfied
-

Appendix: GenAI Transcript Summary

12:30 PM

I want to make an interactive story for Week 3 Side Quest. Can you help me brainstorm a theme?

12:31 PM

Great! Here are some simple themes that would work well:

- Magical forest exploration
- Space station mystery
- Desert island survival

The magical forest would be easiest because you can create interesting choices like "take the bright path vs dark path" and "trust the stranger vs go alone." Would you like to go with that theme?

12:35 PM

Yes, magical forest sounds good. Should I track multiple stats like health, trust, karma?

12:36 PM

For the bonus feature, a single score variable would be simpler and still demonstrate the data tracking concept. You can add different point values for choices (+5, +10, +15, +20) and use a threshold like 30 points to determine good vs bad ending. This is easier to balance than three separate stats.

1:15 PM

I've written my code with 7 files (main.js, start.js, scene1-3.js, good/bad_ending.js). Can you double-check that I'm following the Week 3 architecture correctly?

1:16 PM

Let me review your structure:

✓ You have currentScreen variable in main.js ✓ setup() and draw() are in main.js ✓ Each scene has drawX() and XMousePressed() functions ✓ Input routing works through main.js ✓ State transitions use currentScreen = "newScreen" ✓ Script loading order is correct (scenes → main.js last)

Yes, your architecture perfectly matches the Week 3 pattern! You're correctly implementing:

- The router pattern from lecture
- File separation (one screen per file)
- State management with currentScreen variable
- Input routing through main.js

Your code structure is exactly what the assignment requires. Great work following the architecture!