Process & Decision Documentation

GBDA 302 - Week 3 Side Quest The Mysterious Forest: Interactive Branching Story

Entry Header

Name: Jimin Kim **WatID:** j243kim **Role(s):** Developer / Designer (Individual Assignment) **Primary responsibility for this work:** Building an interactive branching story with multiple game states, implementing a karma tracking system, and creating atmospheric visual designs for each scene.

Goal of Work Session

The goal was to transform the professor's Example 01 multi-screen architecture into an interactive narrative game inspired by *Dixit*. Specifically, I aimed to:

- Create a branching story with multiple scenes across separate JavaScript files
- Implement meaningful player choices that affect the narrative outcome
- Add a karma tracking system (BONUS) that persists across scenes and determines the ending
- Design atmospheric backgrounds for each scene to enhance the storytelling experience

Tools, Resources, or Inputs Used

- **GenAI tools:** Claude (Anthropic) via Claude Code CLI
- **Lecture Notes:** Week 3 tutorial on game states, UI, and menus
- **Prior code:** Professor's Example 01 architecture (start.js, instructions.js, game.js, win.js, lose.js, main.js)
- **External references:** p5.js documentation [1]

GenAI Documentation

Date Used: January 29, 2026

Tool Disclosure: Claude (Anthropic) - Claude Opus 4.5 via Claude Code CLI

Purpose of Use:

- Generating initial code structure following the professor's Example 01 architecture
- Creating scene background decorations's basic structure (trees, fireflies, treasure chest, traveler figure)
- Debugging layout issues (button positioning, background sizing)
- Iterating on visual designs based on my feedback

Summary of Interaction: Claude generated the initial file structure with 7 screen files (start.js, instructions.js, scene1.js, scene2_good.js, scene2_bad.js, ending_good.js, ending_bad.js) plus the main.js router. It also created the karma tracking system and visual elements for each scene. When I previewed the results, I provided feedback to adjust specific elements like the signpost design, ground positioning, and button text sizing.

Human Decision Point(s):

1. **Story Theme Selection:** I chose "The Mysterious Forest" as the narrative theme and directed the moral choice framework (helping vs. ignoring others)
2. **Visual Feedback Loop:** I reviewed each scene's background and changed specific changes with AI helps :
   - Removed decorative border from instructions page
   - Adjusted tree positions to reach canvas bottom
   - Redesigned the crossroads signpost multiple times until satisfied
   - Moved the forest spirit orb position in scene2_bad
   - Shortened button labels to fit within button boundaries
3. **Karma Threshold:** I accepted the karma >= 1 threshold for the good ending

Integrity & Verification Note: I verified all code by running it via Live Server and visually inspecting each screen. I tested the branching logic by playing through multiple paths to ensure karma tracked correctly and endings triggered appropriately. All p5.js functions were cross-referenced with official documentation.

Scope of GenAI Use: GenAI contributed to:

- Initial code structure and routing logic
- Background decoration code (shapes, animations)
- Button styling and hover effects
- README content structure

GenAI did NOT contribute to:

- Story concept and narrative decisions
- Visual design direction and aesthetic choices
- Final adjustments based on preview feedback
- Testing and verification of functionality

Limitations or Misfires:

- The signpost design required multiple iterations; initial versions had misaligned elements
- Some background elements didn't extend to canvas edges and needed manual correction
- Button text was too long for the button width in some scenes, requiring label shortening

---

Summary of Process (Human + Tool)

1. **Initial Setup:** Started with the professor's Example 01 code and identified the file structure pattern
2. **Architecture Planning:** Directed Claude to create the branching story structure with karma tracking
3. **Code Generation:** Claude generated all screen files following the established pattern
4. **Preview & Iterate:** Used Live Server to preview each screen and identified visual issues
5. **Refinement Cycle:** Provided specific feedback for each issue (signpost position, ground sizing, button labels) and Claude made targeted edits
6. **Testing:** Played through all story paths to verify branching logic and karma calculations

7. **Documentation:** Created README and this process document

The development followed an iterative cycle where I would preview, identify issues, describe the problem, and receive targeted fixes. This continued until all visual elements met my expectations.

Decision Points & Trade-offs

Decision 1: Karma System Design

Options considered:

- Simple binary good/bad path based on first choice only
- Cumulative karma score affecting final ending (chosen)
- Multiple stat tracking (karma + health + trust)

What changed: Implemented a single karma variable that accumulates across all choices.

Why: A cumulative system creates more meaningful replayability - players can make one "bad" choice and still achieve a good ending if they make positive choices later. This aligns with the theme of redemption and consequences.

Decision 2: Visual Simplification

Options considered:

- Complex animated characters with multiple frames
- Simple shape-based characters (chosen)
- No character representations, text-only

What changed: Used p5.js primitive shapes (ellipses, triangles, beginShape) for all visual elements.

Why: Shape-based graphics are easier to iterate on and debug. They also ensure the game runs smoothly without external asset loading issues, which is important for GitHub Pages deployment.

Verification & Judgement

- **Playtesting:** Played through all 4 possible story paths (Help→Share, Help→Keep, Ignore→Humble, Ignore→Demand) to verify correct karma calculations and ending triggers
- **Visual Review:** Checked each screen in browser to ensure backgrounds fill the canvas and buttons are clickable
- **Code Review:** Verified that all function names in main.js router match the functions defined in each screen file
- **Assignment Criteria Check:** Confirmed the project meets Week 3 requirements:
  - Multiple game states across multiple files
  - Branching decision tree structure
  - BONUS: Karma stat tracking with ending unlock

Limitations, Dead Ends, or Open Questions

Limitations:

- No sound effects implemented (planned for future improvement)
- Character designs are basic shape compositions rather than detailed sprites
- Story has only 2 decision points per path; could be expanded for deeper branching

Dead Ends:

- Initially tried to create a more complex signpost with 3D perspective effect, but simplified to flat arrow signs for clarity
- Considered adding a third "neutral" ending but decided two endings (good/bad) were sufficient for the assignment scope

Open Questions:

- Would the karma system benefit from displaying the actual numerical value, or is the +/- indicator sufficient for player feedback?
- Could the story themes be made more relevant to empathy and inclusivity as emphasized in the course?

References

[1] McCarthy, L. (2024). *p5.js Reference*. The Processing Foundation. Retrieved January 29, 2026, from https://p5js.org/reference/

Appendix

Note: A complete transcript of the GenAI conversation is available upon request. The conversation included approximately 15-20 exchanges covering:

1. Initial project setup and file creation
2. Scene background structure generation
3. Multiple rounds of visual refinement (signpost, ground layers, button sizing)

Key prompts included:

- "Build an interactive story that unfolds through multiple game states and files, branching like a small decision tree. BONUS: Track a player stat (e.g., karma) across scenes and unlock endings based on it."
- "give me some redesign ideas about the Scene 1 background decorations to make them prettier and more noticeable"
- "which code line do I need to adjust the right sign locatec at the top of the brown line?"

The full transcript demonstrates an iterative feedback loop where I directed design decisions and Claude implemented specific code changes.