# Process & Decision Documentation

This document is used to make your design and development process visible. At this stage of your academic career, you are expected not only to produce finished work, but to articulate how decisions were made, how ideas changed, and how collaboration (for the assignments that include group work) unfolds

In professional and co-op contexts, employers do not only evaluate your final projects in your portfolio. They often ask candidates to explain their process, justify trade-offs, reflect on iteration, and describe their roles within a team.

You will need to submit a modification of this document for every group assignment (A1 – A3) and a shorter version for your individual assignments (Side Quests and A4).

For A1 – A3, this is a group document submitted once per group. Each group member must clearly document their own role and responsibilities. Different roles will naturally produce different design processes.

## Process Overview Visualization (Group Work Only)

Include one simple visual representation of your group's design process. This may take the form of a timeline, flow diagram, loop, branching sketch, or other format that reflects how your project actually unfolded.

This visualization does not need to be polished or visually refined. Hand-drawn sketches or screenshots are acceptable. For assignments that are extensions of previous assignments (A2 and A3), you may resubmit or reuse previous submissions with additional add-ones and context.

Your visualization must clearly show the following elements:

- Roles and responsibilities: Indicate which roles involved at different stages of the project.
- Who did what, and when: Show how work was distributed over time. This can be approximate (e.g., "early," "mid-project," "final week") and does not need exact dates.
- Key stages or phases of work: For example, for A1, can start with research.
- Decision points or pivots: Indicate moments where the group changed direction, simplified scope, or made a significant design decision.
- Iteration or feedback loops: Show where ideas were tested, revised, or reworked (e.g., after playtesting or feedback from instructors).

The purpose of this visualization is not to present an ideal or “correct” design process, but make your actual process visible including parallel work, false starts, and revisions.

### Reference examples (for inspiration only)

You may find it helpful to look at well-known design process visualizations you’ve likely seen before, such as:

- Nielsen Norman Group – Design Thinking Overview: https://media.nngroup.com/media/articles/attachments/Design-thinking-101-NNG.pdf
- Design Council – Double Diamond: https://www.designcouncil.org.uk/our-resources/the-double-diamond/

These examples are provided as references, not templates to follow. Your visualization does not need to match these models and will not be assessed based on adherence to any specific framework.

## Project/Assignment Decisions

Use this section to document key decisions that shape the direction, scope, or outcome of the assignment. The purpose is to make your judgement and reasoning visible, not to record every action taken.

What you include here should reflect decisions that mattered: moments where you chose one path over another, changed direction, or adapted to constraints.

The expected length and detail depend on the assignment.

### Side Quests and A4 (Individual Work)

Keep this section brief, typically 2 to 4 sentences.

Focus on:

- One significant decision or change you made
- Why you made it
- What effect it had on the work

Examples:

- Simplifying a mechanic so it functioned correctly

- Changing an approach after something failed
- Deciding not to pursue an idea due to time or technical limitations

You are not expected to document every alternative or iteration

### A1 – A3 (Group Work)

Provide more details, as decisions are shared, cumulative, and affect multiple roles.

Document decisions such as:

- Scope changes or feature cuts
- Design or analytical pivots
- Outcomes of playtesting or peer/instructor feedback
- Technical or coordination-related trade-offs
- Ethical, representational, or global contexts for design decisions

For each major decision, briefly include:

- What changed
- Why the decision was made
- Who was involved or responsible

This may be written in short paragraphs or bullet points.

This section is not a complete project history. It is a record of decisions that shape the work. Clarity, judgement, and relevance matter more than length.

## Role-Based Process Evidence

This section documents how your work developed over time and provides concrete evidence of your process. It is used to make your thinking, iteration, and decision-making visible, whether or not GenAI was used. You may reuse relevant process evidence across documents where appropriate, as long as it accurately reflects the work for that assignment.

For group assignments (A1 – A3), this section is completed collectively, with clearly attributed role-based entries. For the group, your entries should correspond to elements shown in your process overview visualization such as stages, iterations, or decision points.

For individual assignments (Side Quests and A4), this section is completed individually, using a simplified version of the same structure. For Side Quests, you may omit sections below where no meaningful decision-making occurred. For example, for the first side quest which you need to just upload a blank p5.js to GitHub, you must have to remove a significant amount of the process based evidence as not a lot of design decisions were made.

Examples of acceptable process evidence include (but are not limited to):

- Screenshots of video editing timelines
- Drafts of scripts or research notes
- Annotated sources
- Sketches or diagrams
- Early document outlines
- Prototype screenshots
- Code snippets or commits
- Before-and-after revisions

Project management artifacts (e.g., task boards, timelines, coordination) may be included here when they are relevant to understand how assignment-level decisions were made.

If GenAI was used to support planning, coordination, or role management, this should be noted briefly here as part of the process evidence. Deeper reflection on how GenAI affected teamwork, fairness, and coordination over time belongs in the GenAI Reflections (A4).

## Entry Header

Name:

Role(s):

Primary responsibility for this work:

### *Goal of Work Session*

Briefly describe what you were trying to accomplish during this phase of the assignment.

Examples:

- Drafting an initial script section
- Refining pacing in the video edit
- Revising a mechanic after playtesting

- Narrowing or reframing research examples
- Debugging or simplifying a feature

Tools, Resources, or Inputs Used

- GenAI tools (if used)
- Lecture Notes
- Teammates
- Prior drafts or code
- Playtesting feedback
- External references (in-text citations with square brackets, ex. [1])

### *GenAI Documentation*

Everyone must complete this section. If not GenAI was used, write, “No GenAI used for this task.” When GenAI is not used, process evidence should still demonstrate iteration, revision, or development over time.

Because GenAI can closely mimic human-created work, instructors or TAs may occasionally request additional process evidence to confirm non-use. This may include original working files (e.g., an illustrator file), intermediate drafts, or a brief check-in with a TA to walk through your process.

These requests are not an assumption of misconduct. They are part of ensuring academic integrity in an environment where distinguishing between human-created and AI-generated work is increasingly difficult.

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

**Date Used**: Feb 2nd

**Tool Disclosure**: Claude Opus 4.5

**Purpose of Use**: Implementation

**Summary of Interaction**: The tool contributed the majority of the code, I only made small changes to styling

**Human Decision Point(s):** I created the design and imported it using Figma MCP. I monitored when styling was inaccurate to my design and asked for alterations.

**Integrity & Verification Note**: I checked for accuracy by constantly monitoring the live server and changes.

**Scope of GenAI Use**: GenAI did not design the interactions, styling, or concept of the rpoject

**Limitations or Misfires**: The tool did alignment poorly when translating from Figma to responsive web.

### *Summary of Process (Human + Tool)*

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

- Design, creative ideation, layout, interactions

### *Decision Points & Trade-offs*

Describe one or two key decisions you made:

- I decided to use a "gift a clover" theme and made the UI. I thought of the clover falling interaction. I thought of the flow, and next steps after clicking the button

### *Limitations, Dead Ends, or Open Questions*

Note anything that:

- Claude started adding interactions that I didn't ask for which I found frustrating and had to remove
- Claude was not able to add a background-blur in P5JS

## Appendix

Please include a full transcript of your conversation with the GenAI. GenAI transcripts are used for transparency and verification only and are not assessed for writing quality or completeness.

**Note**: If GenAI is used in another language, you must include both the original language and a translated transcript in the appendix

Implement this design from figma using the MCP. Make sure the clover-hand is centered.

Enter the card with an ease-in-out when hovering that slides in from the right side. Have particles come from the clover on hover. After clicking the email button, clovers fall in from the sky and the screen says “CLOVER SENT TO [EMAIL]”

● Let me check the current state of the project and any Figma design that was being referenced.

● Read 3 files (ctrl+o to expand)

● Let me check the conversation transcript to see what task was in progress.

● Searched for 1 pattern, read 1 file (ctrl+o to expand)

● I can see the Figma file being used. Let me fetch the design again to see what state we're at and what might still need to be done.

```
● claude.ai figma - get_screenshot (MCP)(fileKey: "qvbIIWCoKLE8LEuW2wr1jJ", nodeId: "83:2", clientLanguages: "javascript,html,css", clientFrameworks: "p5js")
 ⎿ [Image]
```

```
● Framelink_Figma_MCP - get_figma_data (MCP)(fileKey: "qvbIIWCoKLE8LEuW2wr1jJ", nodeId: "83:2")
  ⎿ metadata:
     name: Bookmarks on twitter
     components: {}
    … +221 lines (ctrl+o to expand)
```

● I can see the Figma design. The button text should be "ONWARDS, WITH LUCK" but the current implementation has "BLOW WITH THE WIND ~". Let me fix that to match the design.

```
● Update(index.html)
 ⎿ Added 1 line, removed 1 line
    328         <h1 class="card-title">GIFT A CLOVER</h1>
329        <div class="card-fields">
```

```
    330           <input type="email" class="email-input" id="emailInput"
placeholder="RECIPIENT EMAIL">
    331 -         <button class="submit-btn">BLOW WITH THE WIND ~</button>
    331 +          <button class="submit-btn">ONWARDS, WITH LUCK</button>
    332         </div>
    333       </div>
    334     </div>
```

● Now let me also check if the background color and watermark match the Figma design. The Figma shows rgba(208, 255, 233, 0.6) for the card background and a watermark image.

● Update(index.html)
 ⎿ Added 1 line, removed 1 line

```
    115      left: 0;
116      width: 100%;
    117      height: 100%;
    118 -     background: rgba(208, 255, 233, 0.3);
    118 +      background: rgba(208, 255, 233, 0.6);
    119      backdrop-filter: blur(16px);
    120      -webkit-backdrop-filter: blur(16px);
    121      border-radius: 4px;
```

● Update(index.html)

  ⎿ Error editing file

● Searched for 1 pattern (ctrl+o to expand)

```
● Update(index.html)
 ⎿ Added 1 line, removed 1 line

   290         M 182 12 L 182 26 L 196 26 L 196 12 Z
291          M 346 12 L 346 26 L 360 26 L 360 12 Z

   292         M 503 12 L 503 26 L 517 26 L 517 12 Z

   293 -      " fill="rgba(208, 255, 233, 0.35)"/>

   293 +       " fill="rgba(208, 255, 233, 0.6)"/>

   294       <!-- Dashed border for outer shape -->

   295       <path d="

   296         M 4 0
```

● Let me check if the watermark is properly implemented. Looking for the watermark element.

● Searched for 1 pattern (ctrl+o to expand)

● The watermark is missing. Let me add it based on the Figma design. I'll add the watermark image from the assets folder.

```
● Read 1 file (ctrl+o to expand)

● Update(index.html)

  ⎿ Added 10 lines

   141      overflow: hidden;
142     }

   143

   144 +   .card-watermark {

   145 +     position: absolute;

   146 +     top: -74px;
```

```
147 +     right: -163px;

148 +     width: 370px;

149 +     height: 370px;

150 +     opacity: 0.28;

151 +     pointer-events: none;

152 +   }

153 +

154     .card-content {

155       position: relative;

156       z-index: 1;
```

● Now let me add the watermark image element to the HTML.

● Read 1 file (ctrl+o to expand)

● Update(index.html)
⎿ Added 3 lines

```
333        <rect x="503" y="12" width="14" height="14" stroke="white" stroke-width="1"
stroke-dasharray="2 2" fill="none"/>                                    334      </svg>

335

336 +     <!-- Watermark -->

337 +     <img src="assets/clover-watermark.png" alt="" class="card-watermark">

338 +

339      <div class="card-inner">

340        <div class="card-content">

341          <h1 class="card-title">GIFT A CLOVER</h1>
```

● Done. I've updated the implementation to match the Figma design:

1. Button text: Changed from "BLOW WITH THE WIND ~" to "ONWARDS, WITH LUCK"
2. Background color: Updated to rgba(208, 255, 233, 0.6) to match the Figma fill
3. Watermark: Added the clover watermark image positioned at top: -74px, right: -163px, size: 370x370, opacity: 0.28

The card now matches Frame 87 from the Figma design.

✻ Worked for 2m 11s

❯ add sparkle particles when the clover is hovered that float upwards.

● Read 1 file (ctrl+o to expand)

● Update(sketch.js)
⎿ Added 3 lines

```
13  let sentEmail = '';
14  let showSuccessMessage = false;
15
16 +// Sparkle particles state
17 +let sparkles = [];
18 +
19  function preload() {
20    handImg = loadImage('assets/hand-clover.png');
21    leafImg = loadImage('assets/leaf-left.png');
```

● Update(sketch.js)
⎿ Added 18 lines

```
   });
 }

+function spawnSparkle() {
+  // Spawn sparkle within the clover hover bounds
+  let x = handBounds.x + random(handBounds.w);
+  let y = handBounds.y + random(handBounds.h * 0.6);
+
+  sparkles.push({
+    x: x,
+    y: y,
+    size: random(3, 8),
+    opacity: 255,
+    speedY: random(-1.5, -3),
+    speedX: random(-0.5, 0.5),
+    rotation: random(TWO_PI),
+    rotationSpeed: random(-0.1, 0.1),
+    fadeRate: random(3, 6)
+  });
+}
+
 function draw() {
   // Background color
   background(45, 56, 64); // #2D3840
```

● Update(sketch.js)

```
 ⎿ Added 44 lines
   238   }
239  }
   240
   241 +function drawSparkles() {
   242 +  noStroke();
   243 +
   244 +  for (let i = sparkles.length - 1; i >= 0; i--) {
   245 +    let s = sparkles[i];
   246 +
   247 +    push();
   248 +    translate(s.x, s.y);
   249 +    rotate(s.rotation);
   250 +
   251 +    // Draw a 4-pointed star sparkle
   252 +    fill(255, 255, 255, s.opacity);
   253 +
   254 +    // Main sparkle shape
   255 +    beginShape();
   256 +    let points = 4;
   257 +    let outerRadius = s.size;
   258 +    let innerRadius = s.size * 0.3;
   259 +    for (let j = 0; j < points * 2; j++) {
   260 +      let angle = (j * PI) / points - HALF_PI;
   261 +      let r = j % 2 === 0 ? outerRadius : innerRadius;
   262 +      vertex(cos(angle) * r, sin(angle) * r);
+    }
+    endShape(CLOSE);
+
+    // Add a small glow effect
+    fill(255, 255, 255, s.opacity * 0.3);
+    ellipse(0, 0, s.size * 1.5, s.size * 1.5);
+
+    pop();
+
+    // Update sparkle
+    s.y += s.speedY;
+    s.x += s.speedX;
+    s.rotation += s.rotationSpeed;
+    s.opacity -= s.fadeRate;
+
+    // Remove faded sparkles
+    if (s.opacity <= 0) {
+      sparkles.splice(i, 1);
+    }
+  }
+}
+
 function drawSuccessMessage() {
   push();
 
```

● Now let me add the sparkle drawing and spawning to the draw loop and hover check.

```
● Update(sketch.js)
 ⎿  Added 8 lines

    112    // Draw hand image with opacity
113    drawHand();

    114
115 +  // Draw sparkles (behind and around the clover)

    116 +  drawSparkles();

    117 +

    118 +  // Spawn sparkles while hovering clover

    119 +  if (isHoveringClover && !showSuccessMessage && random() < 0.4) {

    120 +    spawnSparkle();

    121 +  }

    122 +

    123    // Draw clover rain

    124    if (isRaining) {

    125      drawCloverRain();
```

❯ remove clover on card

● Done with sparkles. Now removing the clover watermark from the card.

```
● Update(index.html)
 ⎿  Removed 3 lines

    333        <rect x="503" y="12" width="14" height="14" stroke="white" stroke-width="1"
stroke-dasharray="2 2" fill="none"/>                                    334      </svg>
```

```
   335

   336 -     <!-- Watermark -->

   337 -     <img src="assets/clover-watermark.png" alt="" class="card-watermark">

   338 -

   336      <div class="card-inner">

   337       <div class="card-content">

   338         <h1 class="card-title">GIFT A CLOVER</h1>
```

● Update(index.html)

⎿ Removed 9 lines

```
   141      overflow: hidden;
142     }

   143

   144 -   .card-watermark {

   145 -     position: absolute;

   146 -     top: -74px;

   147 -     right: -163px;

   148 -     width: 370px;

   149 -     height: 370px;

   150 -     opacity: 0.28;

   151 -     pointer-events: none;

   152 -   }

   144

   145    .card-content {

   146      position: relative;
```

● Done. I've:

1. Added sparkle particles - When hovering over the clover, 4-pointed star sparkles spawn and float upwards with a fading glow effect
2. Removed the clover watermark from the card

✻ Sautéed for 1m 1s

❯ clover fall is laggy. can you fix?

● The lag is likely from rendering emoji text repeatedly. Let me optimize by pre-rendering the clover to a graphics buffer.

● Read 1 file (ctrl+o to expand)

● Update(sketch.js)

```
  ⎿ Added 3 lines

    16  // Sparkle particles state
17  let sparkles = [];

    18

    19 +// Pre-rendered clover for performance

    20 +let cloverGraphic;

    21 +

    22  function preload() {

    23    handImg = loadImage('assets/hand-clover.png');

    24    leafImg = loadImage('assets/leaf-left.png');
```

● Update(sketch.js)

```
  ⎿ Added 6 lines
```

```
    29   cardContainer = document.getElementById('cardContainer');
30   frameRate(60);

    31

    32 +  // Pre-render clover emoji to graphics buffer for performance

    33 +  cloverGraphic = createGraphics(50, 50);

    34 +  cloverGraphic.textSize(40);

    35 +  cloverGraphic.textAlign(CENTER, CENTER);

    36 +  cloverGraphic.text('🍀', 25, 25);

    37 +

    38   // Add hover listeners to card

    39   if (cardContainer) {

    40     cardContainer.addEventListener('mouseenter', () => {


● Update(sketch.js)

  ⎿ Added 6 lines, removed 3 lines

    226  }
227

    228  function drawCloverRain() {

    229 -  textSize(40);

    230 -  textAlign(CENTER, CENTER);

    229 +  imageMode(CENTER);

    230

    231    for (let i = clovers.length - 1; i >= 0; i--) {

    232      let c = clovers[i];

   ...

    235      push();
    236      translate(c.x, c.y);
    237      rotate(c.rotation);
    238 -     text('🍀', 0, 0);
    238 +     // Use pre-rendered clover graphic instead of text for performance
    239 +     let scale = c.size / 40;
    240 +     image(cloverGraphic, 0, 0, 50 * scale, 50 * scale);
    241      pop();
    242
    243      // Update position
   …
    249      }
    250    }
    251
    252 +  imageMode(CORNER);
    253 +
    254    // Keep spawning clovers while raining
    255    if (clovers.length < 15 && random() < 0.2) {
    256      spawnClover();
```

● Done. I've optimized the clover rain by:

1. Pre-rendering the clover emoji to a graphics buffer once during setup
2. Using image() instead of text() to draw clovers - much faster since it skips emoji text rendering each frame

This should eliminate the lag since emoji rendering is expensive and was happening for every clover every frame.

✻ Crunched for 42s

❯ mkae the top text 12px

● Update(index.html)

```
  ⎿  Added 1 line, removed 1 line
    71
72     .header span {
    73       font-family: 'Open Runde', sans-serif;
    74 -     font-size: 16px;
    74 +     font-size: 12px;
    75       font-weight: 500;
    76       color: white;
    77       letter-spacing: -0.05em;
```

● Done. Header text is now 12px.

❯ make the particles green to match the clover. some are spawning too far to the left.

● Update(sketch.js)

```
  ⎿  Added 3 lines, removed 3 lines
     94  }
95
     96  function spawnSparkle() {
     97 -  // Spawn sparkle within the clover hover bounds
     98 -  let x = handBounds.x + random(handBounds.w);
     99 -  let y = handBounds.y + random(handBounds.h * 0.6);
```

```
   97 +  // Spawn sparkle within a tighter area around the clover center
   98 +  let x = handBounds.x + handBounds.w * 0.3 + random(handBounds.w * 0.4);
   99 +  let y = handBounds.y + random(handBounds.h * 0.5);
  100
  101    sparkles.push({
  102      x: x,

● Update(sketch.js)
  ⎿ Added 3 lines, removed 3 lines
  268      translate(s.x, s.y);
269      rotate(s.rotation);
  270
271 -    // Draw a 4-pointed star sparkle
  272 -    fill(255, 255, 255, s.opacity);
  271 +    // Draw a 4-pointed star sparkle in green
  272 +    fill(117, 159, 107, s.opacity); // #759F6B - clover green
  273
  274      // Main sparkle shape
  275      beginShape();
 ...
  284      endShape(CLOSE);
  285
  286      // Add a small glow effect
  287 -    fill(255, 255, 255, s.opacity * 0.3);
  287 +    fill(180, 230, 160, s.opacity * 0.3);
  288      ellipse(0, 0, s.size * 1.5, s.size * 1.5);
  289
```

```
    290    pop();

❯ remove right-leaf

● Done with green particles. Now removing the right leaf.

● Update(sketch.js)
 ⎿ Removed 10 lines
    167   image(leafImg, 0, 0, leftLeafW, leftLeafH);
168   pop();
    169
    170 -  // Right leaf - positioned off-screen to the right and top
    171 -  push();
    172 -  let rightLeafScale = max(height / 870, 1);
    173 -  let rightLeafW = 1012 * rightLeafScale;
    174 -  let rightLeafH = 925 * rightLeafScale;
    175 -  tint(255, 100);
    176 -  translate(width - rightLeafW * 0.25, -rightLeafH * 0.27);
    177 -  rotate(radians(150.49));
    178 -  image(leafImg, 0, 0, rightLeafW, rightLeafH);
    179 -  pop();
    170
    171   blendMode(BLEND);
    172   pop();

● Done. I've:
```

1. Made sparkles green (#759F6B) to match the clover
2. Tightened spawn area to center around the clover
3. Removed the right leaf

❯ move the particle spawn a little bit more to the right

```
● Update(sketch.js)
  ⎿ Added 1 line, removed 1 line
     95
    96  function spawnSparkle() {
     97   // Spawn sparkle within a tighter area around the clover center
     98 -  let x = handBounds.x + handBounds.w * 0.3 + random(handBounds.w * 0.4);
     98 +  let x = handBounds.x + handBounds.w * 0.45 + random(handBounds.w * 0.4);
     99   let y = handBounds.y + random(handBounds.h * 0.5);
    100
    101   sparkles.push({
```

● Done. Shifted the spawn area to the right.