

Process & Decision Document

Student: Qianhui (Kiki) Tan

Role: Creator

Project / Assignment Decisions

(Side Quests – Individual Work)

A key design decision in this assignment was to express **panic** primarily through movement and interaction rather than visual detail alone. I chose to exaggerate panic behavior in early iterations (e.g., high speed, strong jitter, unstable edge behavior) to clearly surface problems before refining them in later commits. This approach allowed me to make concrete, observable improvements—such as stabilizing wall collisions and improving emotional readability—rather than attempting to “design perfectly” from the start.

Role-Based Process Evidence

Entry Header

Name: Kiki Tan

Role(s): Designer & Developer

Primary responsibility for this work:

Designing and implementing emotional movement, interaction mechanics, and iterative refinement using p5.js.

Goal of Work Session

The goal was to redesign a simple animated blob so that its behavior clearly communicates panic, and to explore a small “mischief” interaction where the blob disrupts objects in its environment. I aimed to document this process through multiple GitHub commits that demonstrate exploration, failure, refinement, and polish.

Tools, Resources, or Inputs Used

- **p5.js** (JavaScript creative coding library)
- **GitHub** (version control and process evidence via commits)
- **ChatGPT (GenAI)** for coding and debugging support
- Manual testing and visual observation in the browser

GenAI Documentation

Date Used

2026 Jan 26

Tool Disclosure

ChatGPT (text-based large language model)

Purpose of Use:

GenAI was used primarily as coding support during development. This included rewriting and extending existing p5.js code, helping structure functions, suggesting parameter adjustments, and improving code comments for clarity. It was not used to generate the project concept or emotional intent.

Summary of Interaction:

Through iterative prompts, GenAI assisted with modifying and expanding my existing codebase. This included proposing alternative implementations for movement logic, boundary handling, and collision-based interactions, as well as rewriting code segments to improve readability and maintain consistent commenting style. GenAI also helped identify technical issues (such as edge-sticking or unstable motion) and suggest code-level fixes rather than conceptual ideas.

Human Decision Point(s):

All design and implementation decisions remained human-driven. I selected which code changes to integrate, manually tuned all parameters through testing, and chose when to intentionally retain or remove flawed behaviors to support iteration. GenAI-generated code was treated as a draft or reference and was frequently modified, simplified, or partially discarded based on observed behavior.

Integrity & Verification Note:

All GenAI-assisted code was reviewed and tested in the browser. Behavior was validated through repeated interaction and observation, and changes were cross-checked against assignment requirements. No code was included without manual verification and adjustment.

Scope of GenAI Use:

GenAI did not define the project's goals, emotional framing, or interaction design decisions. It did not independently generate the full codebase or finalize mechanics. Core logic, tuning, and evaluation were completed by me.

Limitations or Misfires:

Some suggested implementations required additional adjustment to fit the existing code structure or produced unintended side effects (e.g., overly strong reactions or mismatched visual cues). These cases required human judgement and iterative correction.

Summary of Process (Human + Tool)

The project progressed through a series of small, intentional commits: establishing an environmental baseline, exploring exaggerated panic behavior, refining motion for readability, adding a mischief mechanic, and finally balancing the system for clarity and presentation. GenAI supported ideation and reflection, while all evaluation, tuning, and final decisions were made through testing and observation.

Decision Points & Trade-offs

- **Exaggeration vs. Control:**
Early panic behavior was intentionally chaotic to surface problems before refinement.
 - **Expressiveness vs. Simplicity:**
Facial features were kept minimal to avoid over-design while still communicating emotion.
 - **Mischief Frequency vs. Readability:**
Object stealing was reduced and limited in duration to prevent visual overload.
-

Verification & Judgement

Decisions were evaluated through:

- Repeated manual testing
 - Comparing emotional readability across commits
 - Observing unintended behaviors (e.g., edge-sticking, overly cheerful expressions)
 - Re-checking assignment goals and bonus requirements
-

Limitations, Dead Ends, or Open Questions

This project prioritised panic over other emotional states. Additional work could explore how the same system adapts to emotions like curiosity or joy. More complex object behaviors were intentionally avoided to keep the system lightweight and focused on emotional communication.

Appendix — Full GenAI Conversation Transcript (Simulated)

User

I'm starting a p5.js side quest where I need to redesign a simple animated shape to express a specific emotion. The assignment asks for both emotional movement and an optional mischief mechanic involving objects in a small map.

I already have a basic blob animation that uses Perlin noise to “breathe,” but it doesn’t express any emotion yet. I want to build on this code instead of starting from scratch, and I need to keep a clear record of how the project evolves over time.

Here is my initial `sketch.js` code that I’ll be working from:

```
// Object representing a soft animated blob
let blob = {
  // Position of the blob (centre of the shape)
  x: 240,
  y: 160, // centre of the canvas

  // Base size and shape resolution
  r: 28, // Base radius of the blob
  points: 48, // Number of vertices around the circle (higher = smoother)

  // Shape deformation settings
  wobble: 8, // Maximum amount the edge can move in or out
  wobbleFreq: 0.8, // Controls how lumpy or smooth the blob looks

  // Time values for animation
  t: 0, // Time input for noise()
  tSpeed: 0.01, // How fast the blob "breathes"
};

function setup() {
  createCanvas(480, 320);
  noStroke();

  // Text settings for on-screen instructions
  textFont("sans-serif");
  textSize(14);
}

function draw() {
  background(240);

  // --- Animate over time ---
  // Increment time so noise() changes smoothly every frame
  blob.t += blob.tSpeed;

  // --- Draw the blob ---
  fill(20, 120, 255);
```

```

beginShape();

for (let i = 0; i < blob.points; i++) {
  const a = (i / blob.points) * TAU;

  const n = noise(
    cos(a) * blob.wobbleFreq + 100,
    sin(a) * blob.wobbleFreq + 100,
    blob.t,
  );

  const r = blob.r + map(n, 0, 1, -blob.wobble, blob.wobble);
  vertex(blob.x + cos(a) * r, blob.y + sin(a) * r);
}

endShape(CLOSE);

fill(0);
text("Blob breathing via noise().", 10, 18);
}

```

My goal is to transform this neutral animation into something that clearly communicates **panic** through movement and interaction, rather than adding narrative or detailed graphics. I want to proceed incrementally so each change is easy to document and evaluate.

GenAI

Thanks for sharing the starting point. Your existing blob code is a solid foundation because it already has organic motion through noise.

A good next step is to avoid touching this blob logic right away and instead build structure around it—such as setting up the page layout and a simple environment—so later behavioral changes are easier to isolate and compare. Proceeding incrementally will also make your process documentation much clearer.

User

I want to start clean and show process.

For the first commit, I want to **set up the HTML layout first**, then establish a baseline environment in `sketch.js`. No interaction yet. Can you help?

GenAI

Yes. Starting with `index.html` is a good idea—it makes the project feel intentional and clearly staged.

You can center the canvas, load p5.js from a CDN, and add a caption indicating this is an early process build.

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Panic Blob (Process Build)</title>

    <!-- Load p5.js from CDN -->
    <script
src="https://cdn.jsdelivr.net/npm/p5@1.9.0/lib/p5.min.js"></script>

    <style>
      /* Simple page reset so the canvas sits cleanly */
      body {
        margin: 0;
        font-family: sans-serif;
        background: #f0f0f0;
        display: grid;
        place-items: center;
        height: 100vh;
      }
      main {
        width: 480px;
      }
      .caption {
        font-size: 12px;
        color: #333;
        margin-top: 8px;
      }
    </style>
  </head>

  <body>
    <main>
      <!-- p5 will inject the canvas here -->
      <script src="sketch.js"></script>
      <div class="caption">
        Build log version. Commit 1: environment + objects only.
      </div>
    </main>
  </body>
</html>
```

User

Nice. Now in the same commit I want to update `sketch.js` so there's a small map and objects, but the blob animation itself should stay unchanged.

GenAI

That's a good baseline decision. You can keep the existing noise-based blob and only add environment elements.

Here are the key additions:

```
// --- Small map objects ---
// These are props for the blob to interact with later
let objects = [];
function setup() {
  createCanvas(480, 320);
  noStroke();

  // Text settings for on-screen instructions
  textFont("sans-serif");
  textSize(14);

  // --- Create a small map of objects ---
  // We only draw them for now (no interaction yet)
  for (let i = 0; i < 12; i++) {
    objects.push(makeObject());
  }
}
// --- Environment: draw walls ---
function drawWalls() {
  // Simple border frame makes a small map
  fill(220);
  rect(0, 0, width, 10);
  rect(0, height - 10, width, 10);
  rect(0, 0, 10, height);
  rect(width - 10, 0, 10, height);
}
// --- Objects: draw helper ---
function drawObjects() {
  // Objects are simple dots for now
  fill(60);
  for (const o of objects) {
    ellipse(o.x, o.y, o.r * 2, o.r * 2);
  }
}
```

The blob drawing code itself should remain unchanged.

User

I ran it. It's pretty static and boring, but I assume that's fine.

GenAI

Yes—that's expected. This commit is about establishing a reference point, not interaction.

Commit 2 — Panic Movement (First Rough Pass)

User

Next step: I want the blob to panic when the mouse gets close. I'm okay if this first version is exaggerated and a bit messy.

GenAI

We can add velocity and treat the mouse as a threat. A simple approach is to apply a repulsion force when the cursor is nearby.

Key additions might look like this:

```
blob.vx = 0;
blob.vy = 0;
blob.maxSpeed = 6;
blob.fearRadius = 120;
// --- Panic movement ---
let dx = blob.x - mouseX;
let dy = blob.y - mouseY;
let d = dist(blob.x, blob.y, mouseX, mouseY);

if (d < blob.fearRadius) {
  let force = map(d, 0, blob.fearRadius, 0.6, 0);
  blob.vx += (dx / d) * force;
  blob.vy += (dy / d) * force;
}
blob.x += blob.vx;
blob.y += blob.vy;

// Temporary boundary constraint (intentionally rough)
blob.x = constrain(blob.x, 20, width - 20);
blob.y = constrain(blob.y, 20, height - 20);
```

This version prioritizes emotional intensity over control.

User

Yeah... it definitely panics, but it sticks to the walls and feels kind of broken.

GenAI

That's expected with this approach. This gives you a clear reason to refine boundary handling in the next iteration.

Commit 3 — Refining Panic and Fixing Edge Behavior

User

I want it to still feel panicked, but not like a physics bug. What should I change?

GenAI

The biggest improvement is switching from constraints to bounce-based walls and adding damping.

```
// --- Wall collision with bounce ---
if (blob.x < 20 || blob.x > width - 20) {
  blob.vx *= -0.85;
}
if (blob.y < 20 || blob.y > height - 20) {
  blob.vy *= -0.85;
}
// Damping for control
blob.vx *= 0.95;
blob.vy *= 0.95;
```

This usually reads as frantic motion instead of a bug.

User

This feels much better. I also added a simple face so the emotion reads faster—but the mouth kind of looks like a smile.

GenAI

That's a common visual misread. Curved mouths often look friendly. For panic, an open mouth shape usually works better.

Commit 4 — Mischief Mechanic + Facial Correction

User

I fixed the mouth to an open shape. Now I want the bonus: the blob should bump objects and sometimes steal smaller ones.

GenAI

You can keep this lightweight. First, give objects velocity so bumps are visible.

```
o.vx += dx * 0.05;  
o.vy += dy * 0.05;
```

For stealing:

```
if (o.r < 14 && random() < 0.02) {  
    o.stolen = true;  
}  
if (o.stolen) {  
    o.x = blob.x + cos(frameCount * 0.1) * 30;  
    o.y = blob.y + sin(frameCount * 0.1) * 30;  
}
```

This version can be chaotic—you can balance it later.

User

Yeah it's fun but everything gets stolen really fast.

GenAI

That's useful feedback. Once you know it works, you can tune frequency and duration.

Commit 5 — Balance and Polish

User

I reduced the steal chance and added a timer so stolen objects drop back. That feels much better.

GenAI

That's a solid polish pass. Limiting duration keeps the environment readable.

```
o.life--;  
  
if (o.life <= 0) {  
    o.stolen = false;  
}
```