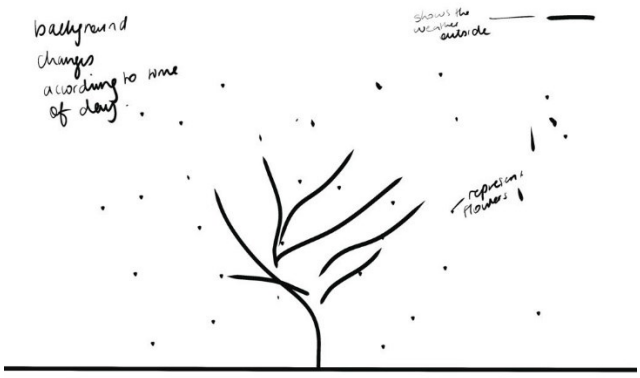
Generative Artwork: Reflection & Design Thinking

Debugging & Refinement Reflection

Starting out

When I started this project, I wanted to create a banner that felt alive. I didn't just want a static image — I wanted something that could respond to real-world changes and let users interact with it. I was really drawn to the idea of mixing nature with code: a tree that grows using recursion, a background that reflects the weather, and the ability to plant flowers by clicking.

I didn't have it all figured out from the start, but I began building one feature at a time — and learning as I went.



Class exercises that helped me get started

To be honest, I wouldn't have known how to even begin without the exercises we did in class. Seeing how we could use random(), noise(), and recursion in P5.js really helped me understand the basics of generative visuals. Some of the sketch examples — like drawing patterns, animating shapes, or using functions like mousePressed() — made me realise how I could break this down and build my own version step by step.

Those smaller tasks gave me the confidence to start experimenting, and I kept referring back to them when I got stuck. They also helped me get comfortable working inside the P5.js editor, writing and testing functions, and slowly editing them into something more personal.

Making weather feel dynamic

Originally, I used wttr.in for live weather, but it kept showing "sunny" no matter what — which made the background feel fake. That pushed me to find something better.

I switched to **Open-Meteo**, which gives real weather data using codes. I used console.log() to figure out the structure of the response, and then I mapped the codes to simple states like clear, cloudy, and rainy. It took a bit of trial and error, but once it worked, it felt way more authentic.

Understanding recursion for the tree

The tree is drawn using **recursion**, and at first, I didn't fully get how it worked. The branches were going in weird directions and looked completely off. But after playing around with the angles and scaling, I started to see how each layer fit together.

Renaming the function to drawFractalTreeRecursive() helped me understand what it was actually doing, and made it easier to manage. Watching the tree evolve and take shape as I tweaked the values was one of the most satisfying parts of the whole project.

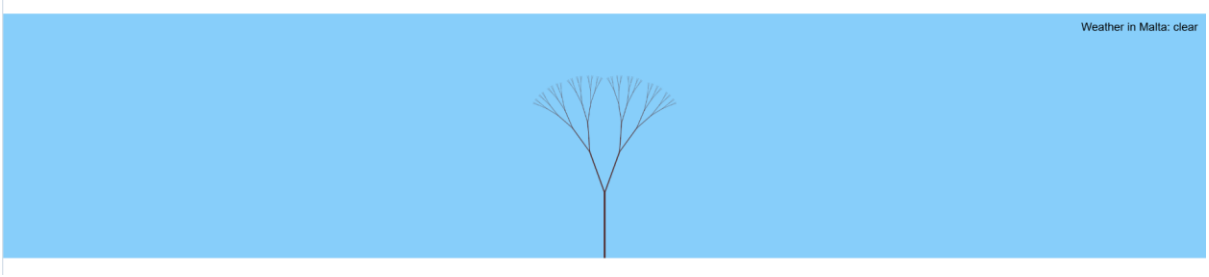
Making interaction feel meaningful

I wanted users to be able to interact with the banner, so I added a flower-planting feature. At first, it was just small circles with random colours — and they didn't really stand out. I used map() to vary the size and adjusted the colour palette to make them more vibrant and noticeable.

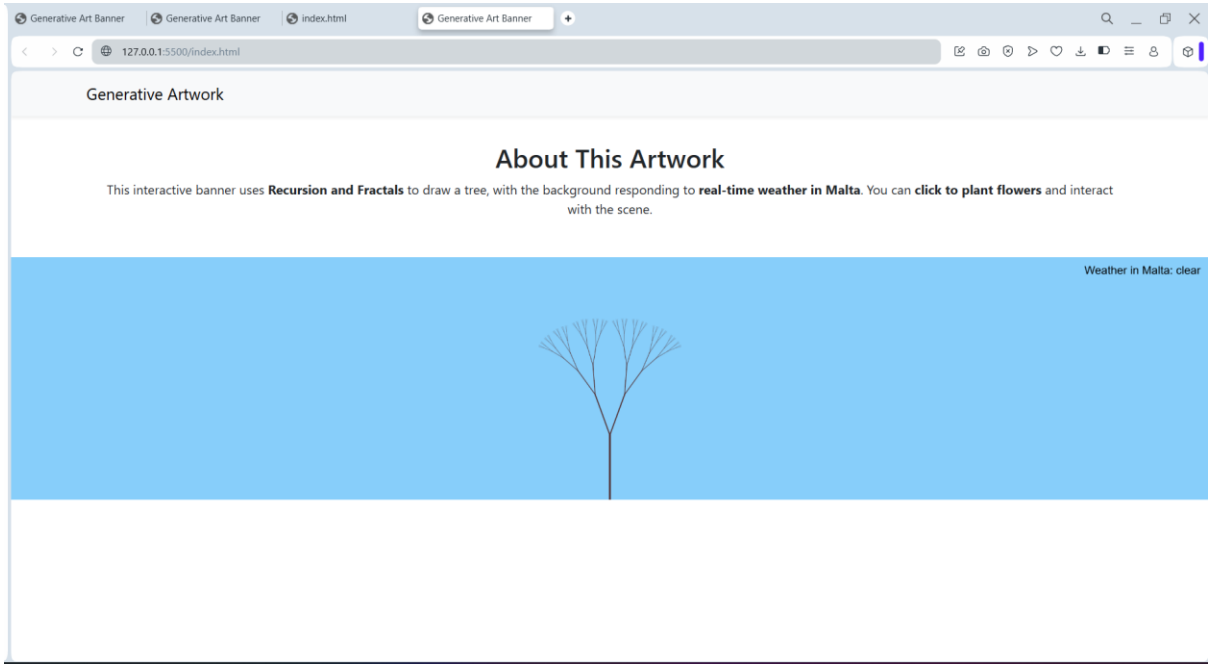
I also added noLoop() to stop the canvas from constantly redrawing, so now it only updates when a flower is added or the weather loads. That made it run much smoother.

Cleaning up the layout

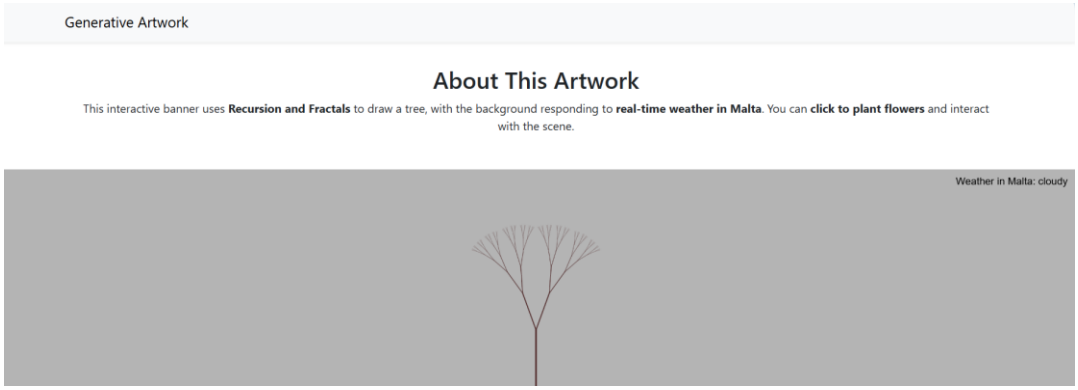
Once the artwork worked technically, I realised the page still looked really plain. I added **Bootstrap** to give the layout some structure — a navbar, cleaner spacing, and a proper header section to hold the canvas. That small design update made the whole thing feel more polished and complete.



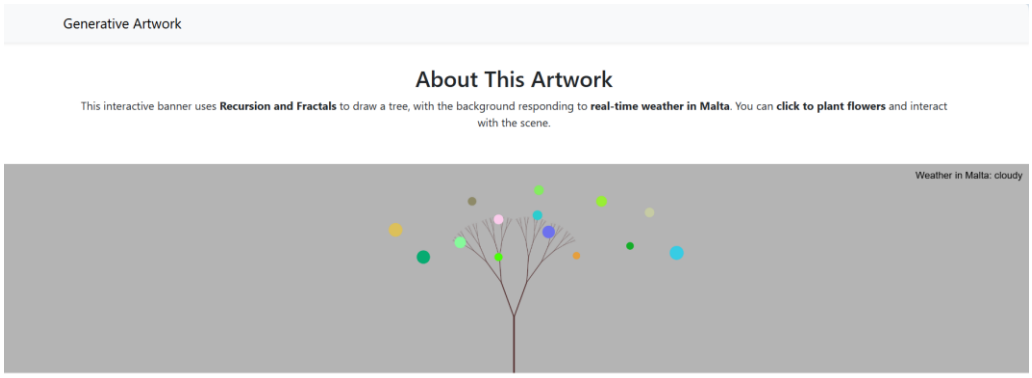
Canvas updates with live weather



Screenshot taken when it was cloudy in Malta.



User clicks plant flowers represented through circles.



Chosen Technique: Recursion and Fractals
This project uses recursion to draw a fractal tree. The recursive function generates branching structures that shrink and rotate based on angles, creating a natural-looking tree that evolves algorithmically.

System Flow Overview:

- **Weather Input**: Real-time weather data is fetched using Open-Meteo's API.
- **Weather Mapping**: Weather codes are mapped to clear, cloudy, or rain conditions.
- **Background Change**: The canvas background color dynamically reflects current weather.
- **Fractal Tree**: A recursive function drawFractalTreeRecursive() is used to draw the tree.
- **User Interaction**: Users can click on the banner to "plant" randomly colored flowers.
- **Performance Optimization**: noLoop() is used to prevent unnecessary redraws for better efficiency.

Landing Page Design

The landing page was created using HTML5 and styled with Bootstrap 5 for responsive layout. It includes:

- A **navigation bar** for branding.
- A **header section** where the p5.js canvas is embedded dynamically.
- A **content section** describing the artwork.

The canvas is styled to scale across different devices, and Bootstrap classes ensure layout consistency across screen sizes. Additional CSS styles were used to enhance alignment and spacing.

Critical Reflection

Why this kind of work matters

This project made me realise that generative art isn't just about making visuals with code — it's about building systems that react to data and people. That's what I tried to do here: combine recursion (for the tree), real-time weather (for the sky), and mouse interaction (for the flowers) to make something that feels dynamic and alive.

Where it could go

Even though this is a small project, it could easily be expanded into:

- A **live website banner** that reflects local weather
- A **digital art installation** that shows changing weather from around the world

- A **learning tool** for teaching recursion or environmental data in a more visual, interactive way

It shows that even with basic tools, you can create something personal, interactive, and responsive.

Looking forward

Generative art is becoming a big part of things like web design, branding, immersive media, and even AI art. This project helped me get a better sense of how all those elements come together. It also showed me that with just some basic coding knowledge and a bit of curiosity, I can make something creative and unique using tools like P5.js.