PROJECT 3 – PROJECT PLAN

Complete each section of this project plan completely and professionally. This should be a living document that gets updated as appropriate. You should upload this document to your private channel on Teams so all members have access to it. You should also upload a PDF version to the documents folder in your GitHub repository. Each version you upload to GitHub should have the version number appended to the name of the file.

Section 1 – Product Overview – This should be a more refined version of the original elevator pitch.

Generative art is one of the most exciting ways to introduce CS to beginners. I've run generative art workshops in the past to students who had prior programming experience, and it was the best in terms of learning, engagement, and collaboration. Unfortunately, it's inaccessible if people don't have prior programming experience, as people get stuck in the programming and environment setup rather than the actual artwork.

We're building a programming language for generative art that makes it easier for beginners to get started and focus on the art itself. We abstract away recursion by framing it as evolution, which we think is significantly more intuitive. In the end, we'll have a web editor with programming capabilities for the user, and a way for them to run their code in real time.

Section 2 - User Personas: A detailed summary of your main target user groups. How many are there? Niche audience or mass market? What are each one's characteristics (age, lifestyle etc.) and key goals.

Our main target user group consists of beginners who want to learn the basics of programming with visual progress markers. Our users are typically younger students (8-18 years old) who would appreciate the visual applications of our programming language as they learn to create code projects for the first time. This product isn't geared towards experienced artists or those who want to break into digital art, as the operations we support aren't those that appeal to digital artists.

Section 3 – User Stories - Short, simple descriptions of a feature told from the user perspective. As a <type of user>, I want <some goal> so that <some reason>. This will become your backlog and your feature list.

As a beginner programmer, I want a programming language for generative art so that it is easy and intuitive for me to be able to create art through programming. I want to be able to create shapes, add colors, and manipulate the canvas so that I can use minimal code to easily create an art piece. I also want to be able to incorporate more complicated concepts, such as recursion to remove redundancy while coding, but the programming language should enable me to use recursive and other advanced programming concepts easily since I don't fully understand the details of them yet.

Section 4 – Development Tools – Must be something all team members can access at school and agree upon. Must be something where the source code can be tracked with GitHub.

We are going to be using Rust in IntelliJ IDEA with the IntelliJ Rust plugin. We chose to use Rust because of its memory safety, performance, and cross-platform support since some of us will be working on different platforms. In building a programming language, we wanted a low-level unmanaged language that could perform well, while also maintaining memory safety, and Rust fits both of those criteria. Despite its unconventional type-system, we think it's a good opportunity to learn a new language to build a project with. With respect to source control, our code will be tracked on GitHub, and we will utilize GitHub Actions to automate the Rust workflow.

Section 5 – Minimum Viable Product (MVP) - What is the key minimum feature set that you will need to build to make your product successful.

Our MVP would have basic functional parser and interpreter for our programming language which can be used to generate basic art pieces like fractals. The minimum viable product should support basic shapes, and colors, and the manipulation of these parameters on an evolutionary basis. Even if the language is not yet simplified or super beginner friendly, it should at a minimum be able to produce art based on generations rather than recursion.

Minimum Feature Set:

- Syntax, easy definitions, and rules for how code in our language should be written.
- Geometry, handling basic shapes and visual elements as well as some basic properties/operations like transformations, rotations, and scaling.
- Interface, a basic command line interface for the user to run their code.
- Generations, the ability to manipulate visual elements on a generational/evolutionary basis