**Project Plan**

**Domain** - Programming paradigm

**Programming paradigm** - is a fundamental approach or style of organizing, designing, and structuring computer programs. It includes a number of principles, concepts, and practices that guide the development process. Different programming paradigms offer distinct methodologies for expressing computational logic, data manipulation, and system interactions.

**Language Overview**

**Basic Computation**: The DSL is designed to express computations as a series of data transformations through functional composition. The core computational model is based on Category Theory, leveraging concepts like monads, and arrows to represent and compose operations.

A **monad** is a design pattern in functional programming that provides a structured way to sequence computations.

**Basic control structures**: Control flow is primarily determined by the sequence of transformations within the pipeline. Users can specify the order and dependencies of operations, and the DSL automatically manages the flow of data through the pipeline.

**Input and Output**: Programs in the DSL require data as input, typically organized into a structure compatible with the defined pipeline. The output is the result of applying the pipeline to the input data, producing transformed or processed data.

**Error Handling**: Errors may occur during data conversion or function composition. The language can communicate errors to the user through error messages.

**Implementation Plan**: The team will delineate the specific syntax and structure for creating pipelines within the DSL, clarifying how pipes will be instantiated and manipulated. This involves defining the conventions for composing functions or operations into a cohesive pipeline.

**Teamwork Plan**: A team involves a collaborative and inclusive approach to project tasks, ensuring that each team member is involved in various aspects of the project, including design, implementation, estimating, and documentation.