# Progress Evaluation: Milestone 1

## Code Visualization

**Team Members**

- Curtice Gough cgough2019@my.fit.edu
- Joshua Hartzfeld jhartzfeld2020@my.fit.edu
- Catherine DiResta cdiresta2019@my.fit.edu

**Client/Advisor**

- Dr. Ryan Stansifer ryan@fit.edu

**Progress Matrix**

| Task | Completion % | Curtice | Joshua | Catherine |
|---|---|---|---|---|
| 1. Investigate tools | 100% | 33% | 33% | 33% |
| 2. “hello world” demos | 100% | 33% | 33% | 33% |
| 3. Requirement Document | 100% | 100% | 0% | 0% |
| 4. Design Document | 100% | 0% | 100% | 0% |
| 5. Test Plan | 100% | 50% | 0% | 50% |

**Task Summary**

1. Investigate tools

   After considering multiple options each, we decided on the following tools:

   **Collaboration**

   | | |
   |---|---|
   | Version control / Task calendar | GitHub |
   | Documents / Presentations | Google Docs |
   | Communication | Discord |

   **Technical**

   | | |
   |---|---|
   | GUI | PyQt5 |

| | |
|---|---|
| Code tracing backend | Traceprinter |
| Target language | Java |

2. “Hello World” demos

   Curtice demonstrated basic code tracing using Traceprinter. Joshua constructed a bare-bones GUI application in PyQt5. Catherine wrote a Java program featuring a couple of our supported data structures.

3. Requirement Document

   Curtice began to write the document initially, but had to rewrite it after receiving clarification from Dr. Stansifer.

4. Design Document

   After completion of the Requirement Document, Joshua used the information as a basis for his system design.

5. Test Plan

   Curtice and Catherine worked together to write the Test Plan according to the features and functionality defined in the Requirement Document.

**Team Member Contribution**

- Curtice Gough

  Curtice assigned tasks to each team member, including himself. His primary tasks were writing the requirements document and deciding on a code tracing method. After accomplishing both of these tasks, he put together a simple demo of the code tracing backend “Traceprinter”, and helped Catherine finish the Design document.

- Joshua Hartzfeld

  Joshua wrote the entire Design document by himself, occasionally checking in with Curtice to ensure his design worked well with the requirements. After completion of the Design document, he decided on PyQt5 as the GUI frontend technology and constructed a simple demo to showcase some of its features.

- Catherine DiResta

  Catherine worked closely with Curtice throughout the process of accomplishing Milestone 1’s tasks. The decision to target Java in our visualization software was influenced by Curtice’s discovery of Traceprinter. After deciding on the target language and writing a simple demo, Catherine worked alongside Curtice to finish writing the Test Plan.

**Milestone 2 Task Matrix**

| Task | Curtice | Josh | Catherine |
|---|---|---|---|
| 1. GUI groundwork | 20% | 60% | 20% |
| 2. Traceprinter JSON parsing | 60% | 20% | 20% |
| 3. GUI Testing | 33% | 33% | 33% |
| 4. Example Java programs | 20% | 20% | 60% |

**Task Summary**

1. GUI groundwork

   During Milestone 2, the goal for this task will be to construct the main GUI window and separate it into the sections laid out in the Design Document. The Data Structures View takes priority here, as the first set of structure diagrams will be developed as a part of the following Milestone. Once the basic layout has been constructed, we will implement a custom PyQt5 Widget to represent a data structure diagram.

2. Traceprinter JSON parsing

   During Milestone 2, we will write a program that serves to convert Traceprinter's JSON output into a Python dictionary. This dictionary will be used as a basis for constructing objects that represent each data structure present in the target code, and keeping track of how the data within them changes.

3. GUI Testing

   This task involves testing the functionality of each GUI element as it is developed. The tests themselves will be conducted according to the Test Plan. Each test will be conducted on Linux, Windows, and MacOS systems to ensure cross-platform compatibility.

4. Example Java programs

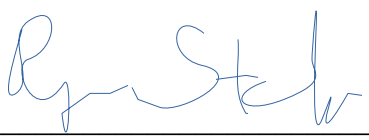
   Writing example Java programs will improve our testing ability as well as the user-friendliness of the final product. When testing, we can use these programs as placeholders for traced code. We can also provide these programs as samples to show users some examples of the type of code they can write for use in the visualizer. These programs will include examples of supported and non-supported data structures.

**Client Meeting Dates**

- 25 September 2023
- 29 September 2023

**Faculty Advisor Feedback**

1. Investigate tools:
2. “Hello world” demos:
3. Requirement document:
4. Design document:
5. Test plan:

Faculty Advisor Signature:  Date: 2 Oct 2023