**Functional Requirements**:

Runs Assembly-Like code: Takes BasicML, machine like code, and runs it like a basic computer might.

Main Window GUI: Contains the main file viewport, console, and toolbar

Toolbar GUI: Contains main function buttons such as File Load/Save/Run/Delete/Etc.

File load button: allow user to click the file destination through their Operating system

Run System button: allow the user to run the file selected from a button in the GUI

Main Window doesn’t close/crash after run: Allow multiple runs of the same file or many different files in a single “session”

File viewer: A viewport for the user to see the loaded file that is ready to be run

File editor: The viewport is accessible to the user and allows the user to make modifications to their file

File saver button: Once the edits are ready to be saved, allow the user to save the file changes and keep the modifications.

File deletion button: Allow the user to delete a file from the available folder

File step-through/debugger: Divide the file into stepwise functions and display to the user which line is about to be run and what the last line’s output was after it was run.

Step-through/Debugger STOP button: Halt the current line of code being run through the step-through function and leave the debugger.

Console input: Allow user to type input codes into the console through a clean GUI box part of the main screen

Console input Enter button: Allow the user to click a button to enter the code instead of the enter key on the keyboard.

Console output: Display for the user the output of the machine during and after the code is run.

**Non-Functional Requirements**:

Efficient: The program is not slow and can perform functions and arithmetic with speed.

Usable: The program is set up in an intuitive way that is easy to understand and does not crash.

Pretty: The program is aesthetic but does not sacrifice too much efficiency or usability.