**SRS**:

1. The system shall parse the contents of an input test file, extracting BasicML instructions formatted as signed 4-digit numbers or signed 6-digit numbers.
2. The system shall be able to detect if the test file utilizes 4 digit or 6 digits words and perform accordingly.
3. The system shall extract the operation defined by the first two digits of the word.
4. The system shall extract the memory location as defined by the last two digits of the word, determining the memory address where the operation should be applied or from where data should be accessed.
5. The user will be able to interact with the program through a graphical user interface (GUI).
6. Through the GUI, the user will be able to load a file.
7. The program shall have the capability to save the registers in the GUI to a file that the user can name and choose the directory of.
8. The GUI will allow the user to load a text file from a directory of their choice.
9. The GUI will allow the user to edit, copy, and paste registers into the system before running.
10. The GUI will display any errors in the system through a pop up window visible to the user without the console.
11. The GUI will have a default color scheme of #4C721D upon first launch of the program.
12. The GUI will allow the user to change the color of the GUI with any HEX code.
13. The GUI will allow the user to launch multiple windows of the GUI, without creating multiple instances of the program.
14. The system shall load the extracted instructions into the main memory, starting at position 00.
15. The system shall handle overflow words (5+ digits) by only utilizing the first 4 given digits of the word, if the test file contains only 4 digit words.
16. The system shall handle overflow words (7+ digits) by only utilizing the first 6 given digits of the word, if the test file contains only 6 digit words.
17. The user will be able to start the BasicML program running from the GUI.
18. The system shall have 250 usable memory registers.
19. The user will be able to clear the accumulator after the BasicML program has finished running.
20. The user will be able to clear the file being run, restoring the contents of each register to 0.
21. The system shall have the capability to save the current accumulator value in a specified memory location.
22. The system shall incorporate an accumulator into which operations shall be loaded into before executing and will store the result of.
23. Once the BasicML program has finished running, the GUI will display the current value in the accumulator.
24. The system shall implement the LOAD operation, defined as 20 in BasicML vocabulary, which shall transfer data from the specified memory location into the accumulator.
25. The system shall implement the STORE operation, defined as 21 in BasicML vocabulary, which shall transfer data from the accumulator into the specified location in memory.
26. The system shall implement the READ operation, defined as 10 in BasicML vocabulary, which shall write a word from the keyboard into the specified location in memory.
27. The system shall implement the WRITE operation, defined as 11 in BasicML vocabulary, which shall write a word from the specified location in memory to the screen.
28. The system shall implement the ADD operation, defined as 30 in BasicML vocabulary, which shall add a word from the specified location in memory to the word currently stored in the accumulator.
29. The system shall implement the SUBTRACT operation, defined as 31 in BasicML vocabulary, which shall subtract a word in the specified location in memory from the word currently in the accumulator.
30. The system shall implement the DIVIDE operation, defined as 32 in BasicML vocabulary, which shall divide the word in the accumulator by the word in the specified location in memory.
31. The system shall implement the MULTIPLY operation, defined as 33 in BasicML vocabulary, which shall multiply the word in the specified memory location to the word in the accumulator.
32. The system shall implement the BRANCH operation, defined as 40 in BasicML vocabulary, which shall unconditionally branch to the specified location in memory and continue execution.
33. The system shall implement the BRANCHNEG operation, defined as 41 in BasicML vocabulary, which shall branch to the specified location in memory if the word in the accumulator is negative.
34. The system shall implement the BRANCHZERO operation, defined as 42 in BasicML vocabulary, which shall branch to the specified location in memory if the word in the accumulator is zero.
35. The system shall implement the HALT operation, defined as 43 in BasicML vocabulary, which shall end the program.

**Non-functional Requirements**:

1. All functions will individually run in no more than 1 second.

2. All buttons will be labeled with a one to two word description of their function.

3. The system shall display word prompts that specify the actions the user must take.