1. Each team member (without consulting with other team members) will create their own SRS document containing 15 functional requirements and 3 non-functional requirements.

2. Submit these documents WITHOUT NAMES to the scrum master assigned to this sprint.

3. The scrum master will divide the team into two groups and assign half of the documents to each section.

4. Each sub-team will discuss the documents they've been given and "merge" them into one document. (The merged documents should still have at least 15 functional requirements and 3 non-functional requirements in the end but can have more if you find the submitted requirements distinct and valuable enough apart from each other. Don't just copy and paste one into another, though -- you'll need to discuss how to add/combine/rewrite the contents of both files to create the new one so it is internally consistent without repeats, redundancy, or contradiction between all the requirements.)

5. Each sub-team submits its merged document back to the scrum master.

6. As a team, meet and examine the resulting documents, and create one final SRS document based on your group's discussion and feelings about the results of the previous exercises. (The final document should have at least the same 15/3 content as before after any merging/rewrites, but it can have more. As before, make sure your final document is consistent after the merge with no repeats, redundancy, or contradictions)

7. Submit the final document and the 6-7 working documents created in the earlier steps. (One per team member, plus one per subgroup, in addition to the final document)

Functional Requirements:

1. System interface must be GUI with labeled buttons
2. Application accepts files on the users system
3. Accumulator must update after every command
4. Application will accept multiple files without needing to be reset
5. Application will display error messages and halt upon encountering an error
6. Application must differentiate between negative and positive numbers
7. Application must perform integer division without decimal carries
8. Application must accept user input
9. Application must validate all input and halt upon encountering invalid input
10. Application must perform addition
11. Application must perform subtraction
12. Application must perform multiplication
13. Application must have storage for up to 99 BasicML words
14. Application must accept all BasicML commands
15. Application must differentiate between BasicML commands and regular words

Non-Functional Requirements:

1. If an improper input is entered, the application will display an error in red text.
2. There will be a red quit button below the input bar
3. Application will display current status of task performed