Functional Requirements

1. The system shall support loading program instructions from a file with the ".uv" extension.

2. The system shall validate each instruction format according to the UVSim instruction set architecture guidelines.

3. Upon encountering an input instruction, the system shall prompt the user for input and store the provided value in the specified memory location.

4. When executing a load instruction, the system shall retrieve the value from the specified memory location and store it in the accumulator register.

5. The system shall perform arithmetic operations (addition, subtraction, multiplication, division) on the accumulator register and the value retrieved from memory based on the instruction provided.

6. For branching instructions, the system shall update the instruction pointer to the specified memory location if the branch condition is met.

7. The system shall halt program execution when a halt instruction is encountered, indicating the end of program execution.

8. In case of encountering invalid instructions or memory addresses, the system shall display descriptive error messages indicating the nature of the error and its location in the program.

9. During input operations, the system shall handle invalid user inputs, providing clear instructions for valid input format.

10. After each operation, the system shall accurately update the accumulator and memory registers to reflect the result of the operation.

11. The system shall provide real-time feedback to the user, displaying the result of each operation executed.

12. Throughout program execution, the system shall maintain a current address pointer, indicating the address of the instruction being executed.

13. The system shall ensure sequential execution of program instructions, moving to the next instruction after each operation is completed.

14. During input operations, the system shall provide clear instructions to the user, guiding them on the format and type of input expected.

15. The system shall implement error-checking mechanisms to ensure the precision of memory addresses and data, preventing data corruption during program execution.

Non-Functional Requirements

1. The system's user interface shall be designed following principles of minimalism, with intuitive controls and concise instructions to facilitate ease of use.

2. The system's response time to user interactions shall be under 1 second, ensuring efficient execution of instructions and a seamless user experience.

3. The system's reliability shall be demonstrated by achieving at least 99.9% uptime during continuous operation, ensuring consistent and accurate execution of programs.