**User Stories**

User Story 1:

As a computer science student, I need a software simulator so that I can learn and understand the basic operations of a simple virtual machine, including loading and storing values in memory, performing arithmetic operations, and branching to different parts of the program, enhancing my learning, and preparing me for advanced topics in computer science.

User Story 2:

As a computer science teacher, I need a software simulator so that I can demonstrate the fundamental concepts of computer architecture and programming to my students in an interactive and hands-on manner, allowing them to write and execute programs on a virtual machine and observe the changes in memory and the accumulator.

User Story 3:

As a power user of this program, I want to be able to choose the primary and secondary colors that are displayed in the GUI so that I keep the program looking fresh and “new” instead of looking at the same plain colors whenever I’m using the simulator.

User Story 4:

As a user of this program, I want to be able to run programs in either format (4-digit or 6-digit) so that I do not have to worry about converting all my old files into the new format.

User Story 5:

As a teacher using UVSim to teach my students, I want to be able to open multiple program editor windows at once, so that I can quickly alternate between different programs and make changes to them.

**Use Cases**

Branch

Actor: Processor

System: UV Sim

Goal: Jump to different instruction

1. Processor reads BRANCH instruction
   1. Error: branch address is negative
      1. System stops loading program, outputs error message, and terminates
2. Processor loads program counter with branch address
3. Program continues starting at new address immediately

Branchneg

Actor: Processor

System: UV Sim

Goal: Jump to different instruction if value in accumulator is negative

1. Processor reads BRANCHNEG instruction
   1. Error: branch address is negative
      1. System stops loading program, outputs error message, and terminates
2. Processor reads accumulator
3. If accumulator contains negative value, processor loads program counter with branch address
4. Program continues at address in program counter

Branchzero

Actor: Processor

System: UV Sim

Goal: Jump to different instruction if value in accumulator is zero

1. Processor reads BRANCHZERO instruction
   1. Error: branch address is negative
      1. System stops loading program, outputs error message, and terminates
2. Processor reads accumulator for a positive zero value
3. Processor reads accumulator for a negative zero value
4. If accumulator contains any zero value, processor loads program counter with branch address
5. Program continues from address in program counter

Halt

Actor: Processor

System: UV Sim

Goal: Terminate execution of instructions

1. Error: Program does not contain any halt instructions
   1. System does not load program, outputs error message, and terminates
2. Processor reaches HALT instruction
3. System immediately terminates execution of instructions

Addition

Actor: Processor

System: UV Sim

Goal: Add a value from the accumulator

1. The program begins iterating through its memory contents
2. An addition operation is reached in the memory – “3016”
3. The system retrieves the value in memory location 16
4. The accumulator is incremented by the retrieved value

Subtraction

Actor: Processor

System: UV Sim

Goal: Subtract a value from the accumulator

1. The program begins iterating through its memory contents
2. A subtraction operation is reached in the memory – “3199”
3. The system retrieves the value in memory location 99
4. The accumulator is decremented by the retrieved value

Multiplication

Actor: Processor

System: UV Sim

Goal: Subtract a value from the accumulator

1. The program begins iterating through its memory contents
2. A multiplication operation is reached in the memory – “3345”
3. The system retrieves the value in memory location 45
4. The accumulator is multiplied by the retrieved value

Division

Actor: Processor

System: UV Sim

Goal: Subtract a value from the accumulator

1. The program begins iterating through its memory contents
2. A division operation is reached in the memory – “3209”
3. The system retrieves the value in memory location 09
4. The accumulator is divided by the retrieved value

Load

Actor: System

System: UVSim

Goal: Load a word from memory into the accumulator

1. System reads the load instruction into program
2. System runs program and reaches load instruction
3. System reads operand from instruction
4. System loads word from memory address specified by operand into accumulator

Store

Actor: Processor

System: UVSim

Goal: Store the value in the accumulator into a memory location

1. Processor reads the store instruction into program
2. Processor reads the value currently in the accumulator.
3. Processor stores the value from the accumulator into the specified memory location.
4. Execution continues with the next instruction in the program.

Read

Actor: Student (end user)

System: UVSim

Goal: Read a word from the keyboard into a specific location in memory.

1. Create file with read instruction

2. Load a file with instructions into the program

3. System reads the file and expects a user input

4. User types an input

5. Program accepts the input and saves it into a memory location specified by the instruction

Write

Actor: Student (end user)

System: UVSim

Goal: Write a word from a specific location in memory to screen location

1. Create file with read instruction

2. Load a file with instructions into the program

3. Program reads the file and expects a memory location

4. Program retrieves value at that location

5. Program displays the value

Save File

Actor: Student (end user)

System: UVSim

Goal: Save the currently loaded program into memory

1. User presses the Save button

2. A “Save As...” dialog box opens from the host computer

3. The file extension is automatically set to be “.txt”

4. The user chooses a filename

5. The user presses “Save” in the dialog box and the txt program file is saved

Multiple Editor Windows

Actor: Student (end user)

System: Program Editor Window

Goal: Open and edit multiple program (files) at once

1. User opens that program editing window
2. User opens a 2nd or 3rd programming window
3. User alternates between all windows open, making edits to each
4. User can choose to run any of the open programs at any given time

Color Scheme

Actor: Student (end user)

System: UVSim

Goal: Change the color scheme of the program

1. The user opens the theme.json file
2. The user makes changes to the file to update the color scheme of the program
3. Upon rerunning the program, the user can see the changes made to the color scheme

File Format Flexibility

Actor: Student (end user)

System: UVSim

Goal: Convert any 4-digit programs into 6-digit programs so that they can still run on UVSim

1. The user imports a file from his computer that uses all 4-digit words
2. The UVSim automatically converts the file into the equivariant 6-digit program
3. The program runs as expected in the updated format