# Project Description

Li, Qinyang / Nguyen, Kathy M. / Wang, Binren / Zhang, Qiuyu

## Source Code

**In ClassProject/src/main/java/com/app/classproject/model:**

**Cache.java**: Models a fully associated, unified cache using a LinkedList data structure. 16 cache lines and FIFO policy.

**Instructions.java**: Contains definitions of program instructions. Each instruction takes a 16-bit binary input.

**Computer.java**: Define the virtual computer, could initialize registers and memory, preload and run program.

**ComputerUI.java**: Links architecture models to UI.

**Memory.java**: Models a 16-bit memory unit. Includes functions to translate from a base 2 binary array to base 10 integer and parse the components of an instruction.

**Pipeline.java:** Models a simple 3-stage pipeline. Stage 1- Fetch instruction. Stage 2- Decode instruction and load register values. Stage 3- Execute instruction, memory access, and write back.

**Register.java**: Models a program register and provides a constructor to create all required types of registers.

**In ClassProject/src/main/java/com/app/classproject/controller:**

**InterfaceController.java**: Display the user interface

**NewActionController.java**: Controls requests submitted by the front-end, execute the action and return the result.

**In ClassProject/src/main/resources/templates:**

ProjectUI.html: The GUI which user can interact with(It is a thymeleaf template working for Spring Boot so still part of the Java program!!)

## GUI

Our GUI uses the Java Spring Boot Framework.

As the program progresses, the GUI displays the current value of various registers and status of the machine.

The GUI also provides several switches for user to execute instructions or modify the machine components.

## Running the Program

Please refer to Readme for details.

## Trap Code Routines

Trap code routine 1 prints “0”, routine 2 prints “1”, …, routine 16 prints “15”.

## Description of Our Pre-loaded Program

It sets some register values as well as memory values and loads some instructions (including all Load/Store Instructions in 5 types)

See loadProgram(), loadTestProgramOne (), loadTestProgramTwo() in Computer.java for details.

In Test Program 2, the required input word should be typed in character by character. Use one or space to indicate “finished inputting the word”. For example, if you want to input word “see”, then input “s”, “e”, “e” and a white space in sequence.