Looking into github you can see the flow of the project. It took about a week of work and thinking about how to start. In the end I opted to make it object oriented such that the simulator is a class.
Then I was wondering how to address the registers and so on. The best option for the memory and the program and registers was to use a map as the map can be addressed through a key. So for registers I could just read their name through the line and if the user decides to use something such as gp, s1, s2, t0, t1, etc.. I could just look for that in another map and if it is present, I just take its original name and access the registers, otherwise I would know the user is using a register that does not exist.
And in memory, instead of having a huge array or vector i could just use a map and store anything i want in it and access the values using the addresses. Same for the program, after reading it from the text file i just go over it to find any labels simultaneously checking if the user made any errors which would cause the program to terminate early on.
Then I store the program in a map whose keys are the addresses of each line in the program. The first line has the address of the program which is initialized using the data text file along with the preloaded variables in memory.
It took a lot of time during the first week just thinking about how to approach the project, planning wise. There was a lot of thinking and theoretical work until I decided on a structure and built a base to be able to divide the work.
Afterwards we decided to divide the work such that each instruction can have a function based on its format. Some functions may have had the same format in risc-v but implementation wise it was different. For example, the load instructions and jalr may have the i format but they are different from all the other instructions which is why they ended up in the i0 function. Because the instruction had a different line structure.
Some problems arised here and there such as the labels, I forgot to pass them by reference which caused the b instructions to act funny, but they were not so hard to trace.
All in all the project took a week's worth of work and planning, and the updates and process can be seen over the github repo attached.

Github: BasmalaAB/RARS_simulator (github.com)