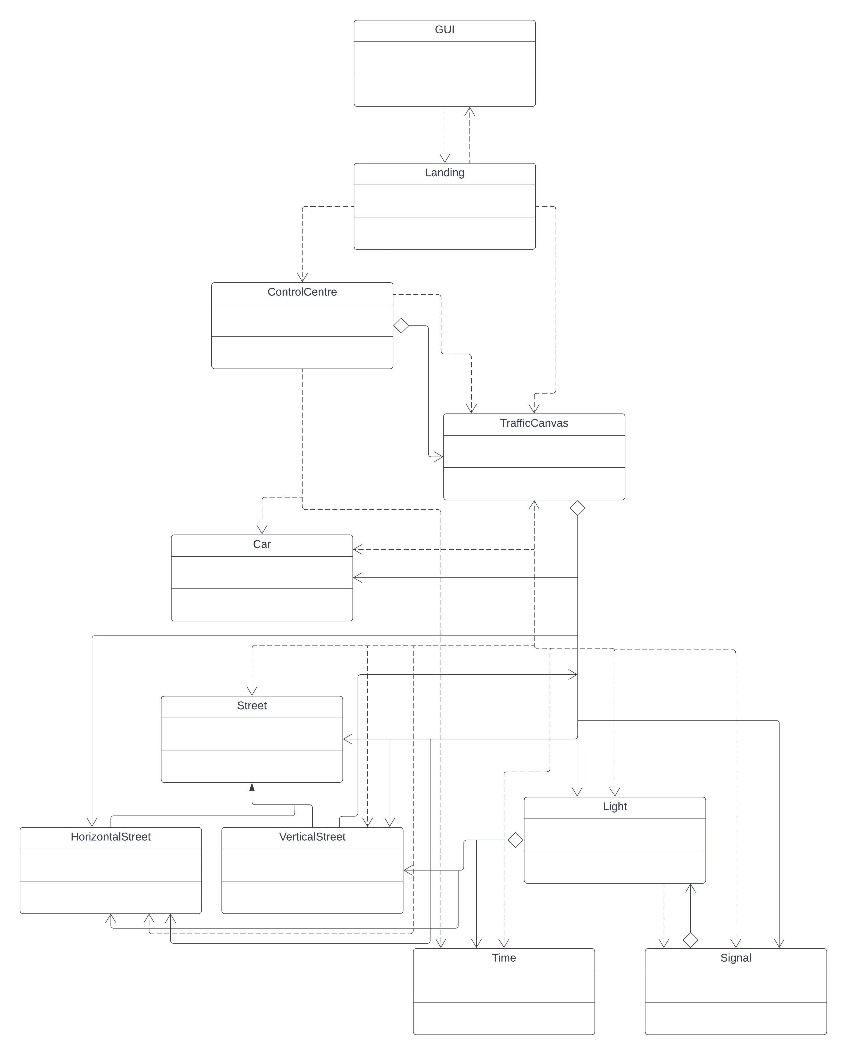
Oluwaseyi Ariyo

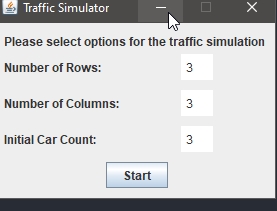
**Traffic Simulator**

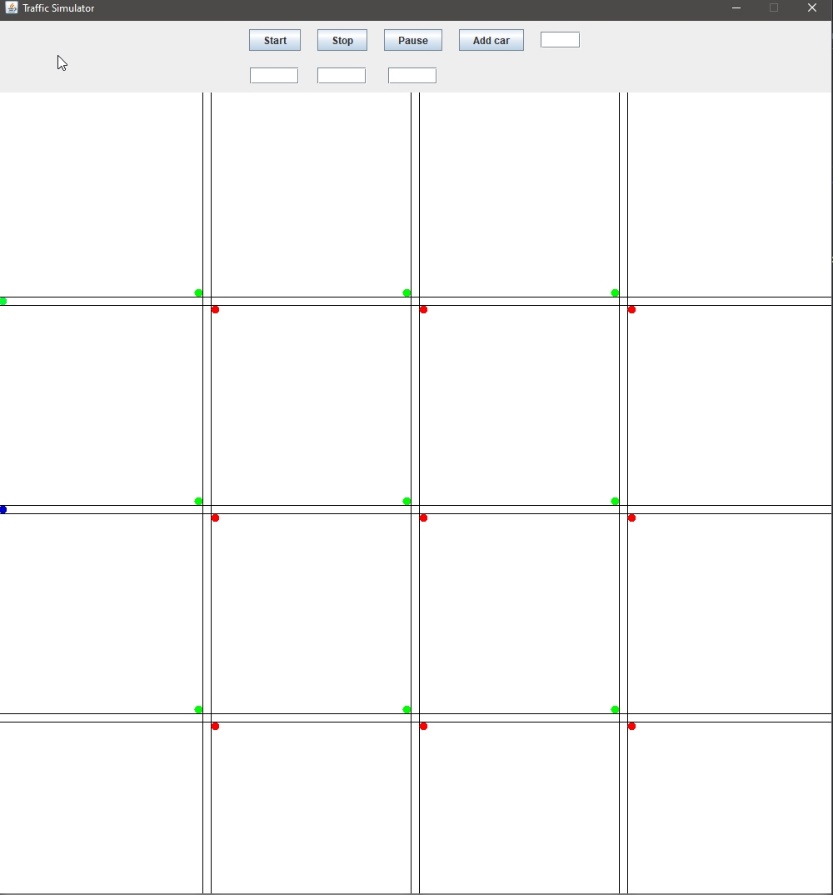
Approach

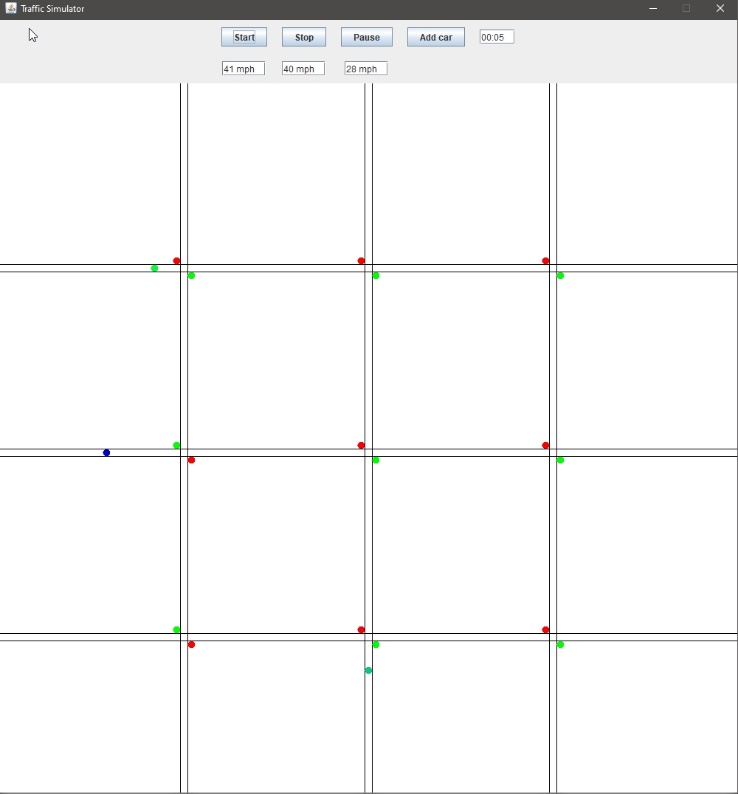
I initially had a difficult time visualising what I wanted the GUI to look like and therefore function. I discussed with other classmates how to design something simple as to not commit to a project that would take too much time to complete. This resulted in the structure that’s here. Since the GUI components are more robust compared to the last project, I made each component its own object and worked through connecting the swing components to the object classes and providing functionality.

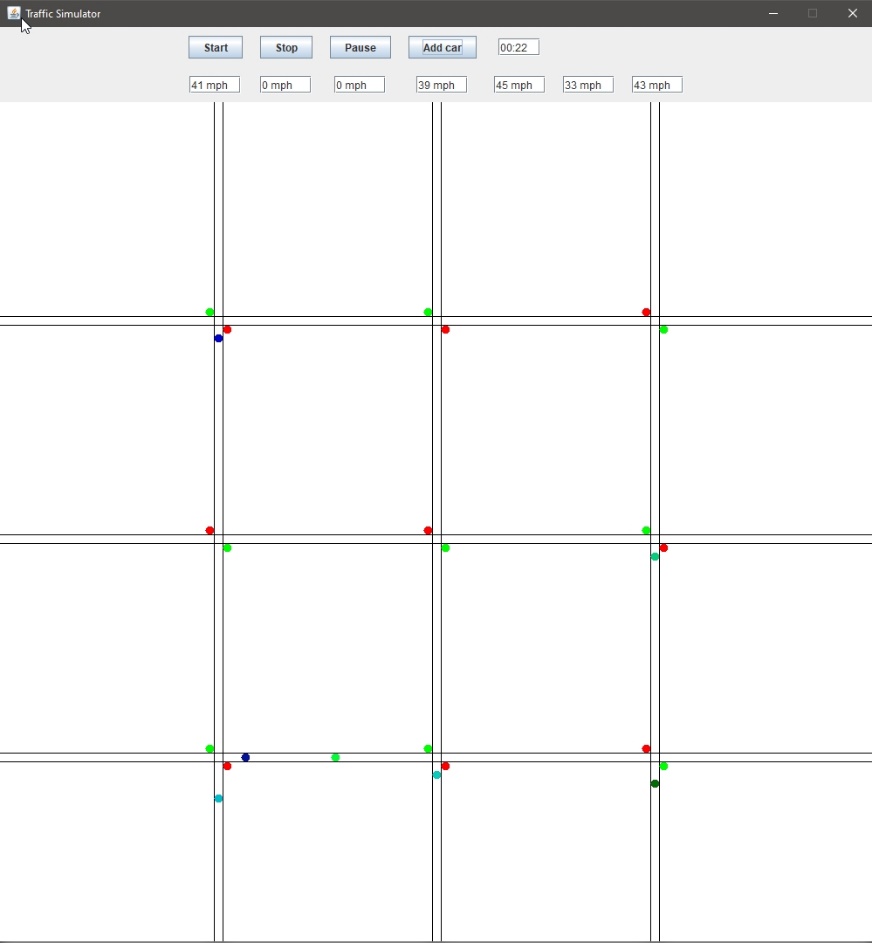
UML

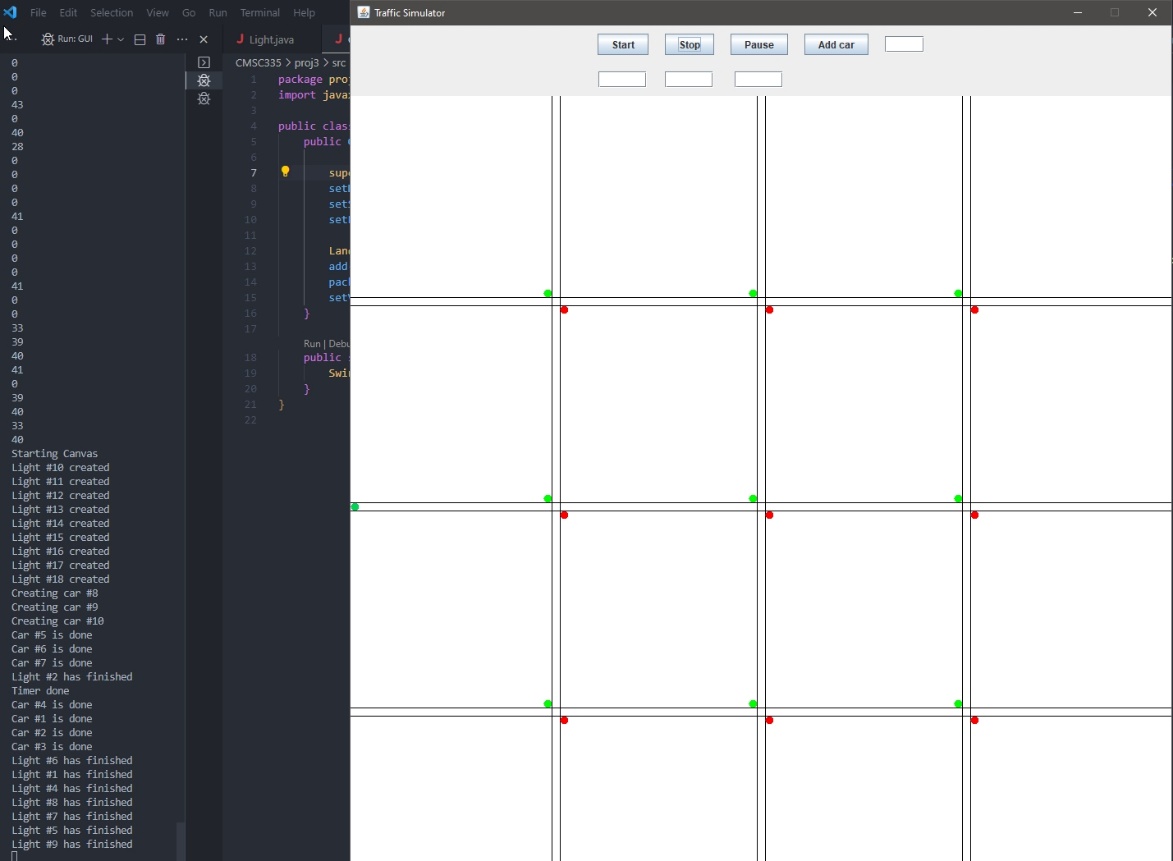
Test Plan

Test 1: Ensuring that the landing page works

Test 2: Upon clicking start, loads the appropriate amount of horizontal streets, vertical streets, and cars (some cars start in the same position and overlap one another).

Test 3: Timer begins after hitting the start button and cars start to move. Car speed is displayed and updated.

Test 4: A car is added upon clicking the add car button. For every additional car added, its speed is also displayed.

Test 5: Pauses the program after using the pause button and resumes the program after using the start button(impossible to show through a screenshot). Upon stopping the program using the stop button, clears the traffic canvas and resets the amount of cars to what was initially entered on the landing page. Also clears the speed of each car and displays the speed of the number of cars initially entered.



Lessons Learned

I learned quite a bit over the course of this project. I struggled a bit with figuring out how to reconcile the format of the GUI with the different approaches that I can take with concurrency implementation. Using the doInBackground, publish, and process methods were integral in figuring out how to implement the car objects concurrently. Breaking up the components to better organise the project solidified the idea of how objects and components interact and can build robust and scalable applications.