Producer/Consumer Simulation Program

Team members:

Name: عمرو عصام محمد الخطيب

ID: 19016113

Name: محمد خالد محمد علي

ID: 19016354

Name: محمود محمد حسانين مرعي

ID: 19016574

Name: محمد احمد محمد الجريتلي

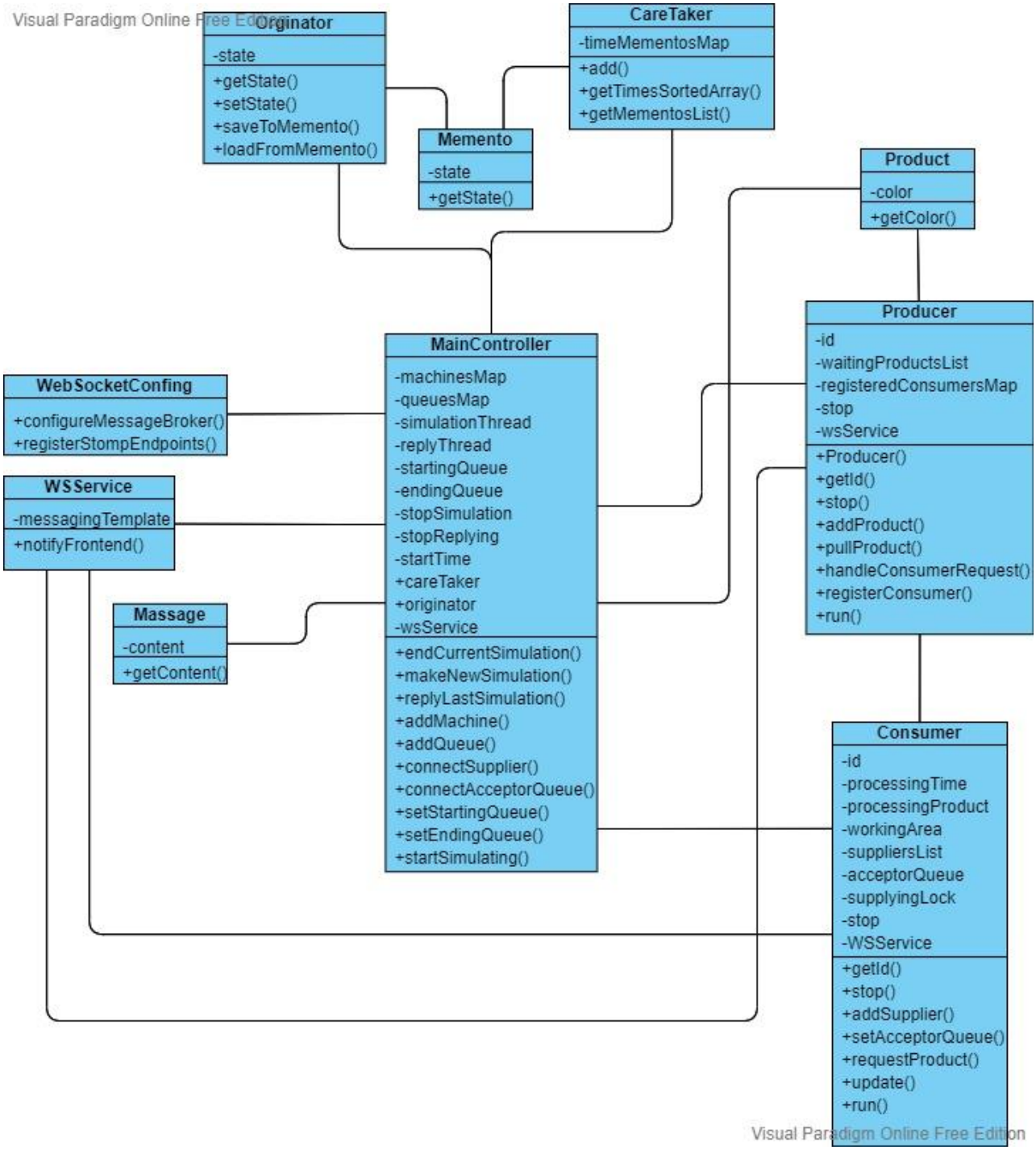
ID: 19016292

How to run the code:

Extract the zip file, open “BackEnd/Producer-Consumer” using IntelliJ IDEA run the main function at “com.mixtoler.producerconsumer.producerconsumerapplication” class Open “FrontEnd/ Producer-Consumer” using VS Code, make sure you have added “node_modules” and “.angular” folder at “FrontEnd/ Producer-Consumer”, open a terminal execute “ng serve”, open google chrome and access the URL “http://localhost:4200/”.

UML & Code Design:

UML:



Design Patterns:

Producer/Consumer DP have been used to manage the process of queues supplying the machines and vise versa, a queue puts one product on a machine blocking queue (workingArea) and notify that machine so it continues its “run” method, releases its wait and start processing that product, when the machine finishes processing a product it add that product to its “acceptorQueue” and notify that queue in order for it to check registered waiting machines list and supply them with that product.

Observer DP have been used by the producer consumer DP so when a machine finishes processing a product, it sends notification to its suppliers queue "requestProduct", if the queue have waiting products the machine take one product from that queue, else the machine is registered to the queue so when the queue receives a product it call the "update" method of that machine, then the machine check itself, if it was unoccupied it request that product from the queue, else it ignores that update.

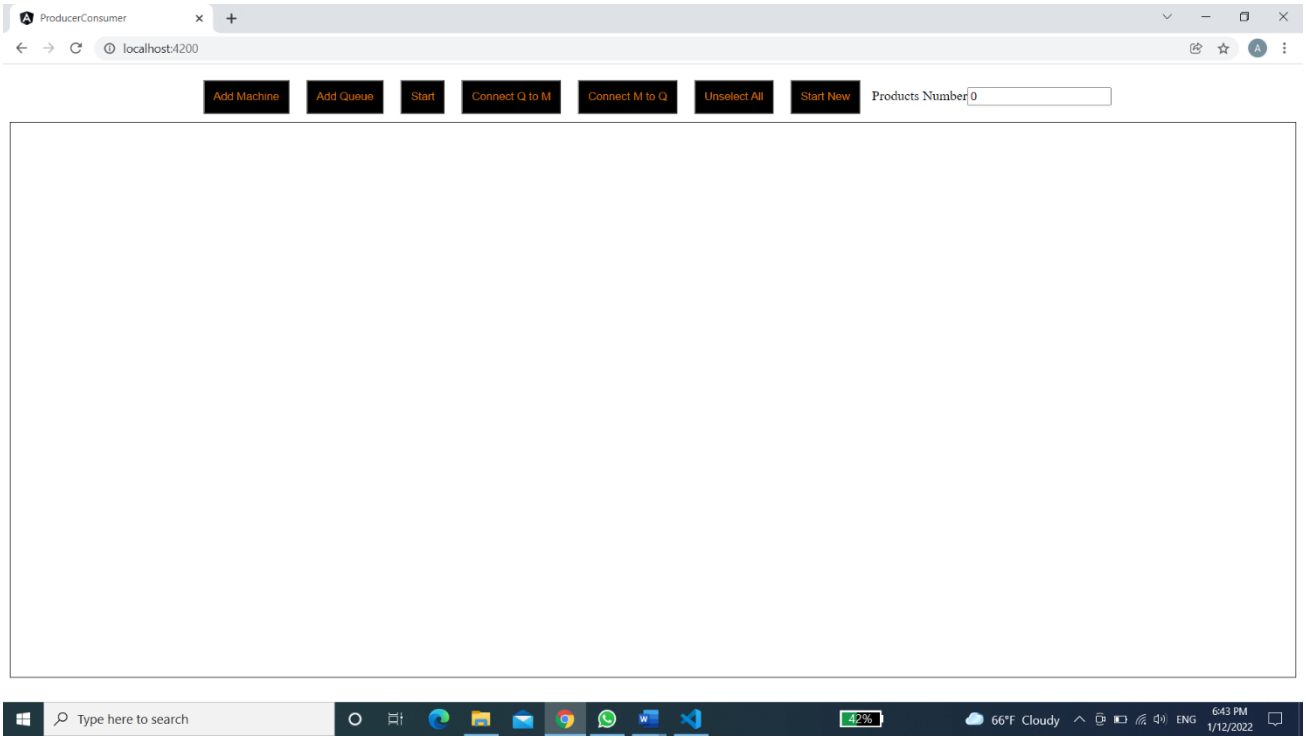
Snapshot DP have been used to store and retrieve the states of machines, queues, and main simulation loop (machine process a "product color", machine finishes a product, queue got a product, queue supplied a product, and simulation ended), the states are stored in a "hashMap" its keys are the time in which the states happened and its values are "arrayList" containing all the states which have been happened in that time corresponding time.

Design Decisions:

The machines are the consumers, and they are the observer (does their suppliers have products or not), the queues are the producers, and they are the observable (when they got a product, they update the registered machines).

UI:

This is the UI of the program

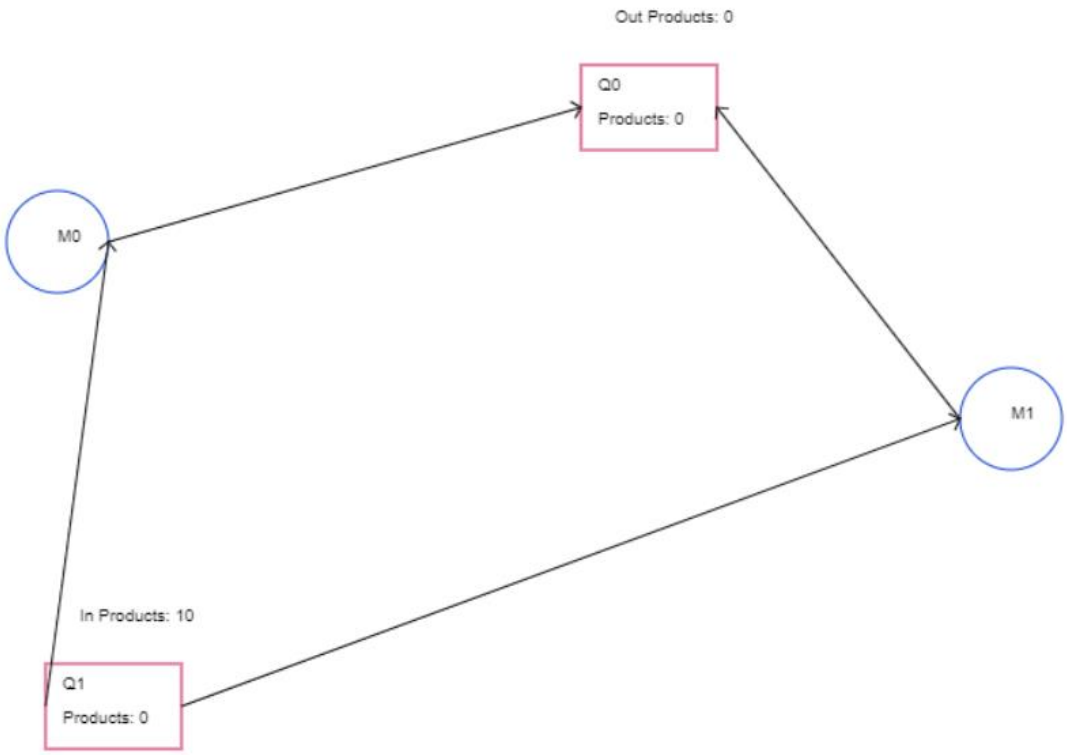


To add a queue or a machine you simply click on add machine/queue and it gets drawn on a random location on the screen and you can drag it and drop it anywhere you want.

To connect a machine and a queue you need click on them – a selected queue or machine will have a black circle inside it indicating that this machine/queue is selected- and then click on connect Q to M to let the queue supply the machine and click on connect M to Q to let the queue accept products from the machine.

Enter the number of products in the input bar and, select the start and the end queue and then press start to see the simulation.

After the simulation has ended you can either start a new simulation or replay last simulation.



Number of products in a queue will be written inside of it and the number of products that will go to the first queue will be written above it, and the number of finished products will be updated through the simulation and will be written above the end queue.