Alexandria University
Faculty of Engineering
Computer and Systems Engineering

Lab 5

Producer Consumer Simulation Web Application

Name	***ID***
Mariam Mohamed Ahmed	***19017274***
Yara Hossam Abdelaziz	***19016871***
Nada Mohamed Ibrahim	***19016782***
Nourhan Ahmed Arafa	***19016812***

Video link:

https://drive.google.com/drive/folders/12wpaE1QWnI-nICP6nRl33wqbkATqbT1d?usp=sharing

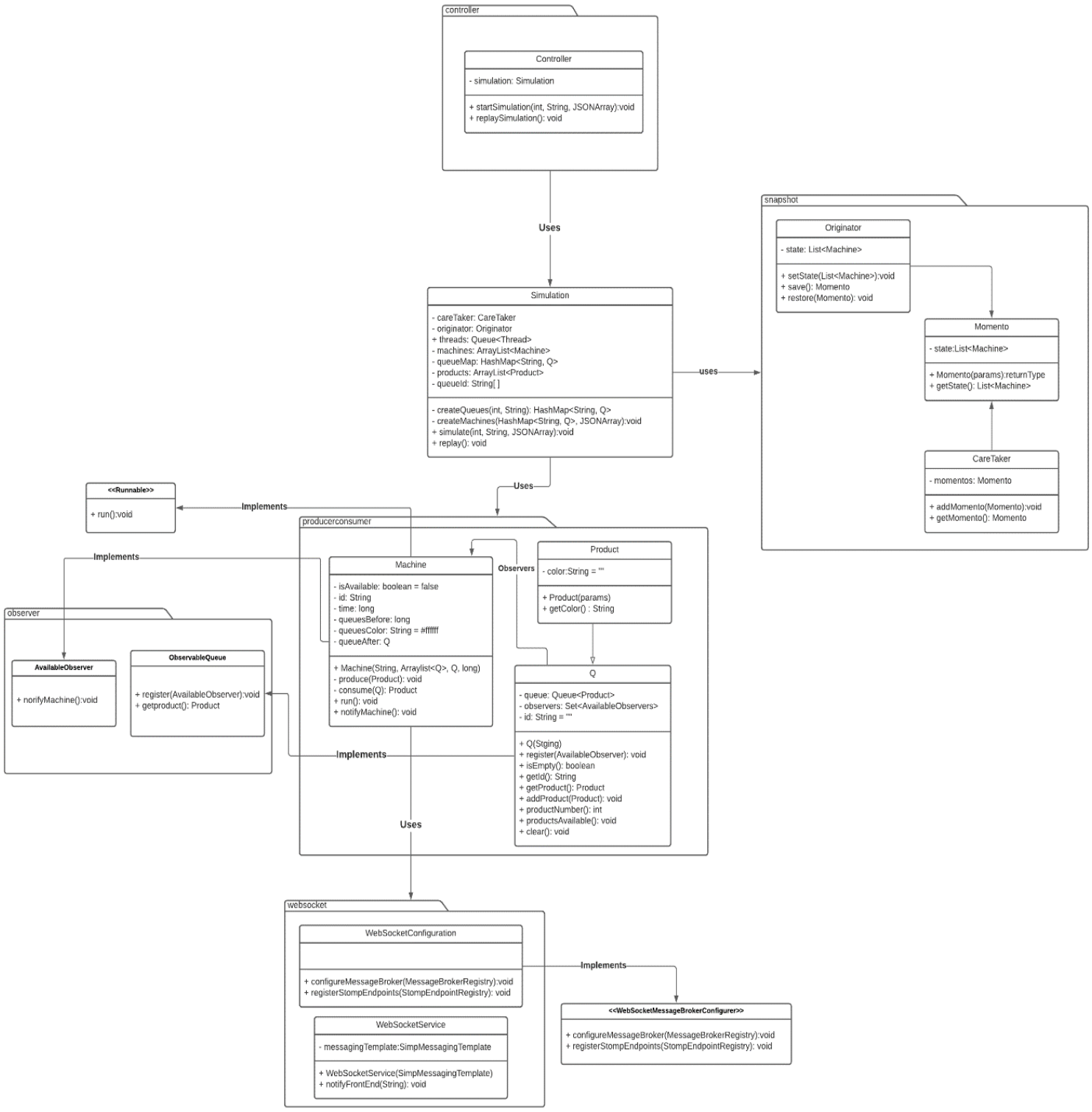
How to run the code

- Front-End
    1. Install angular cli : npm install - g @angular/cli
    2. Create new angular project: ng new producerConsumer
    3. Navigate to your local directory: cd producerConsumer
    4. Replace the existing src file with the project src file
    5. Run server: ng serve -open

- Back-End
    1. Open a command prompt window and go to the directory where you saved the java program (ProducerConsumerProgram.java): cd (directory)
    2. To compile code: javac ProducerConsumerProgram.java
    3. To run program: java ProducerConsumerProgram

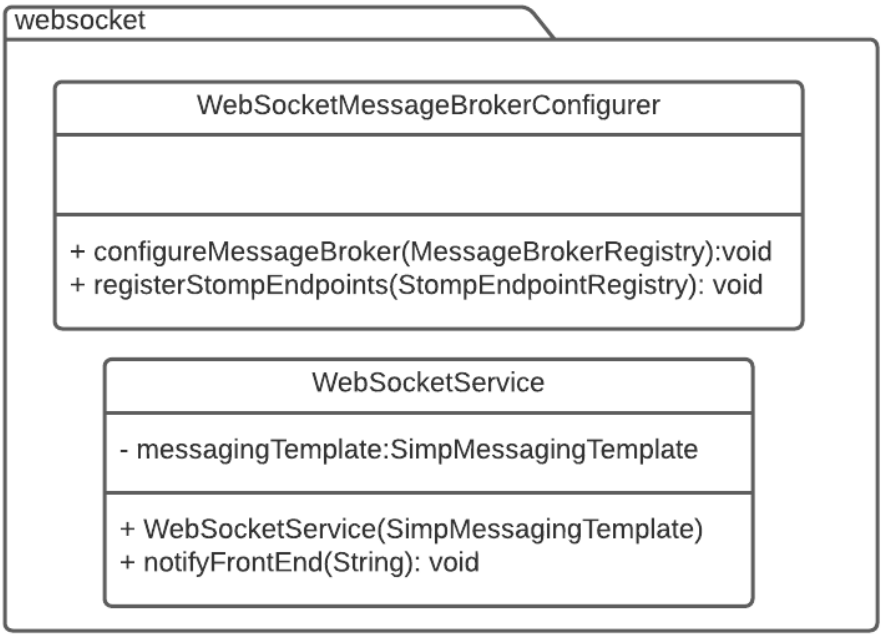
UML Diagrams

- Back-end class diagram:

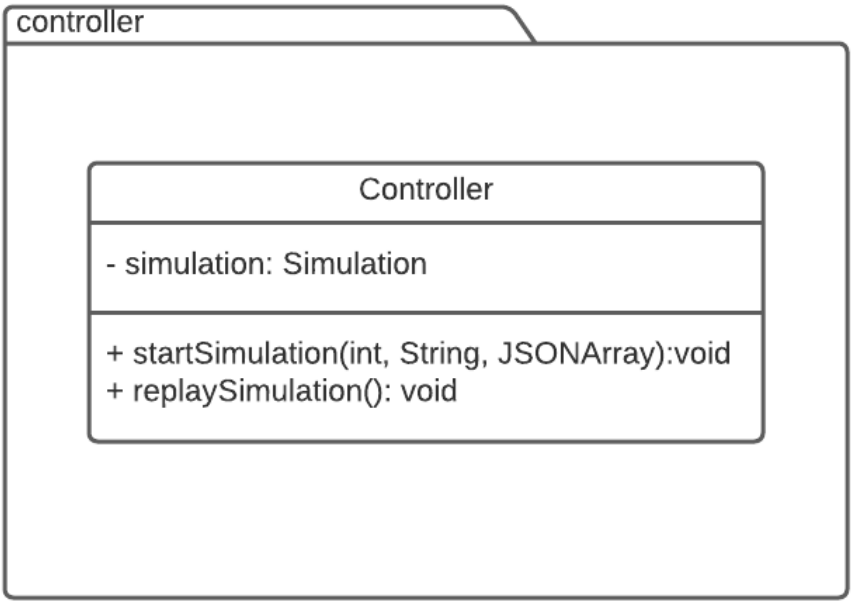


Each package separately:

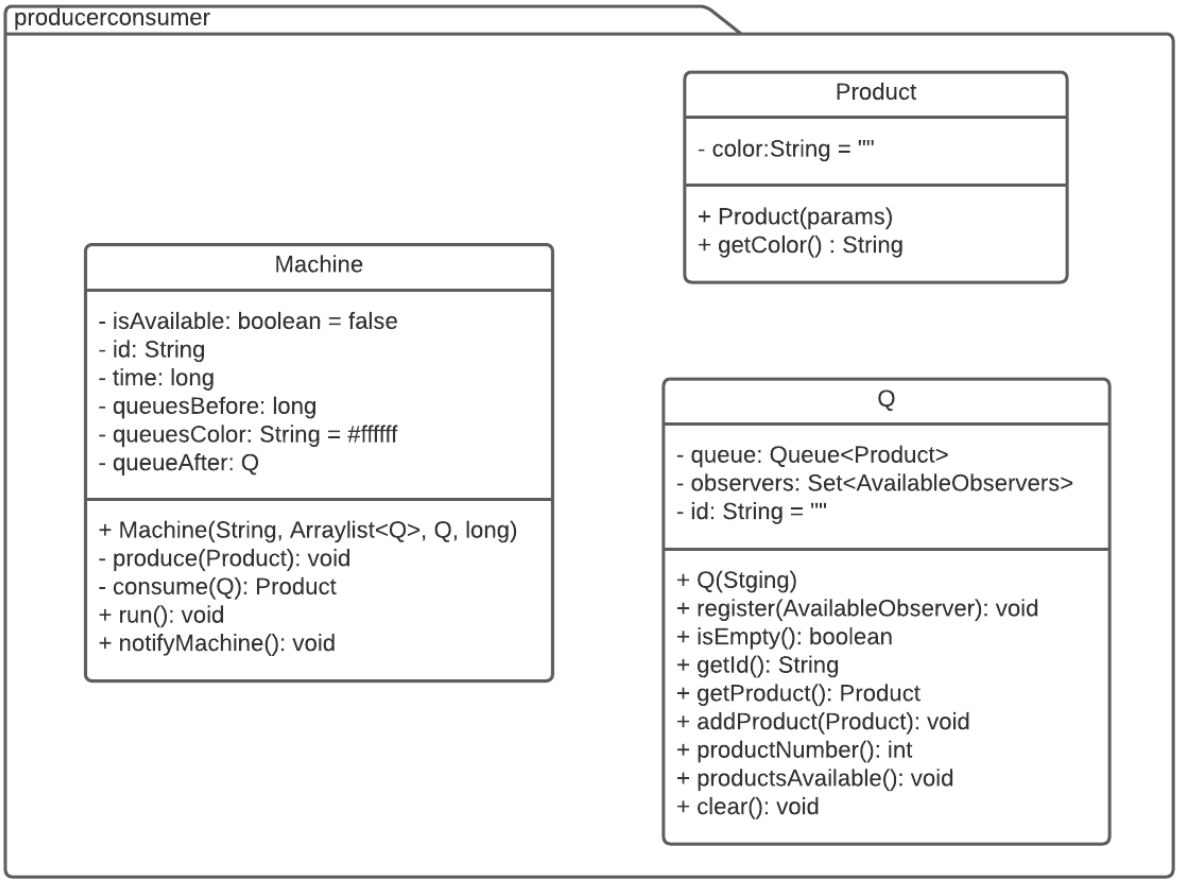
Websocket:



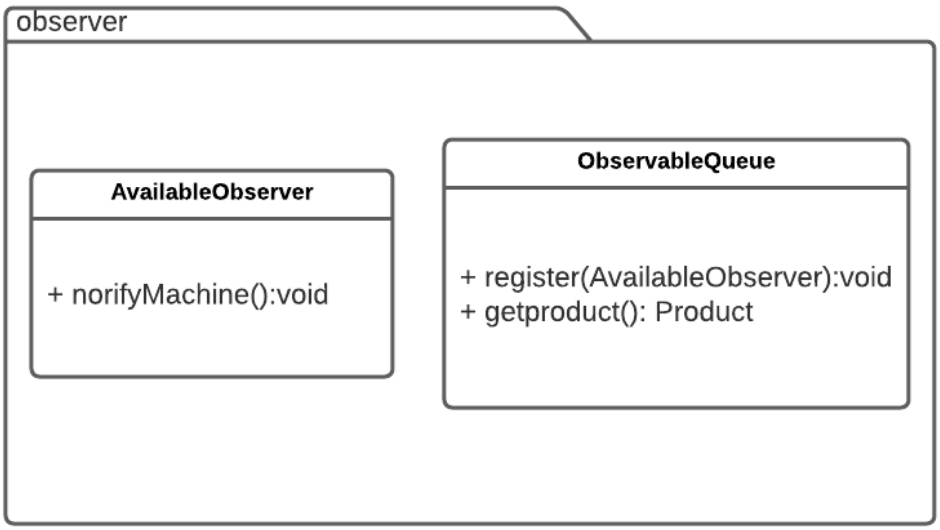
Controller:



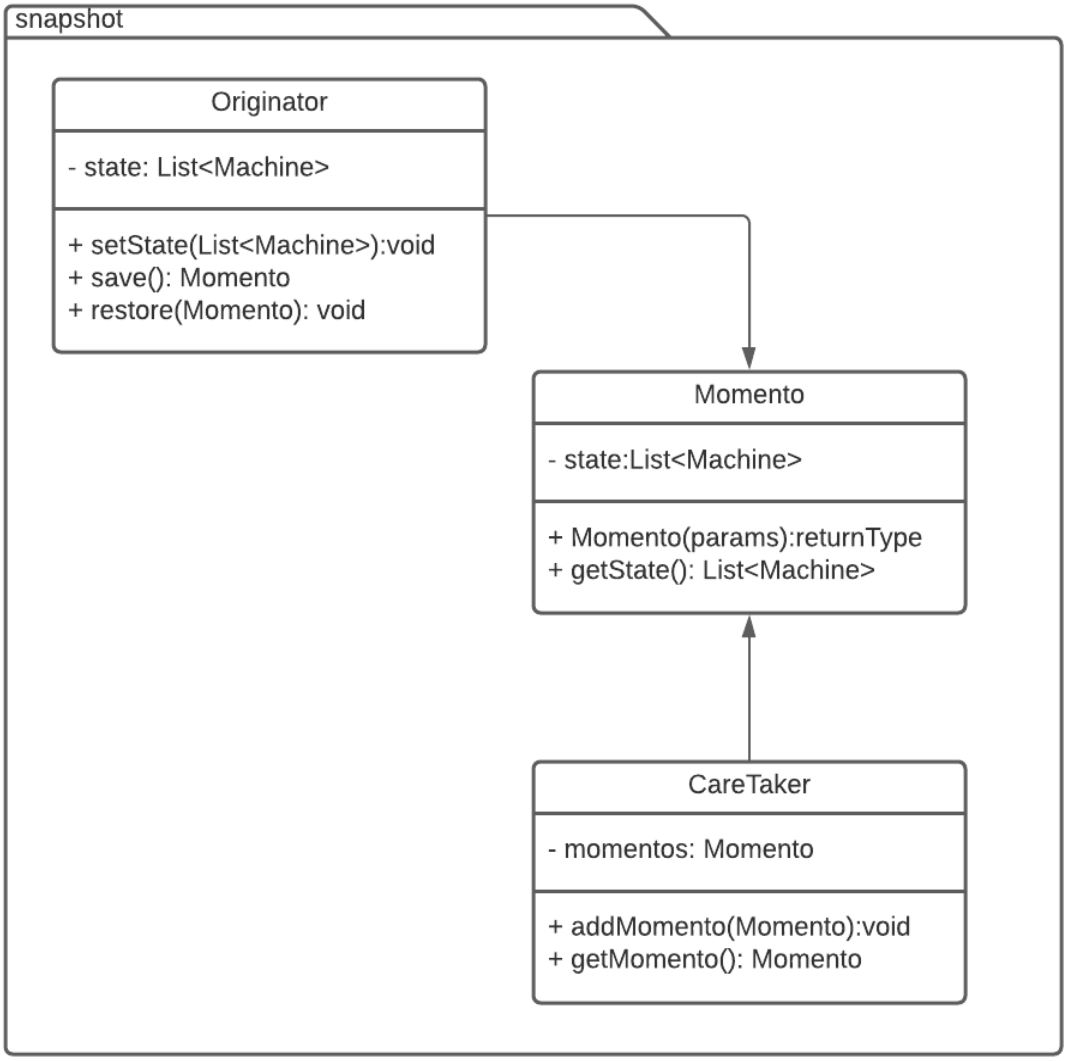
ProducerConsumer:



Observer:



Snapshot:


snapshot
Originator
- state: List<Machine>
+ setState(List<Machine>):void
+ save(): Momento
+ restore(Momento): void
Momento
- state:List<Machine>
+ Momento(params):returnType
+ getState(): List<Machine>
CareTaker
- momentos: Momento
+ addMomento(Momento):void
+ getMomento(): Momento

Design Patterns

- Producer Consumer DP: used to implement the simulation flow.
- Observer DP: used to get queues updates and register machines to empty queues.
- Snapshot DP: used to allow replaying the previous simulation.

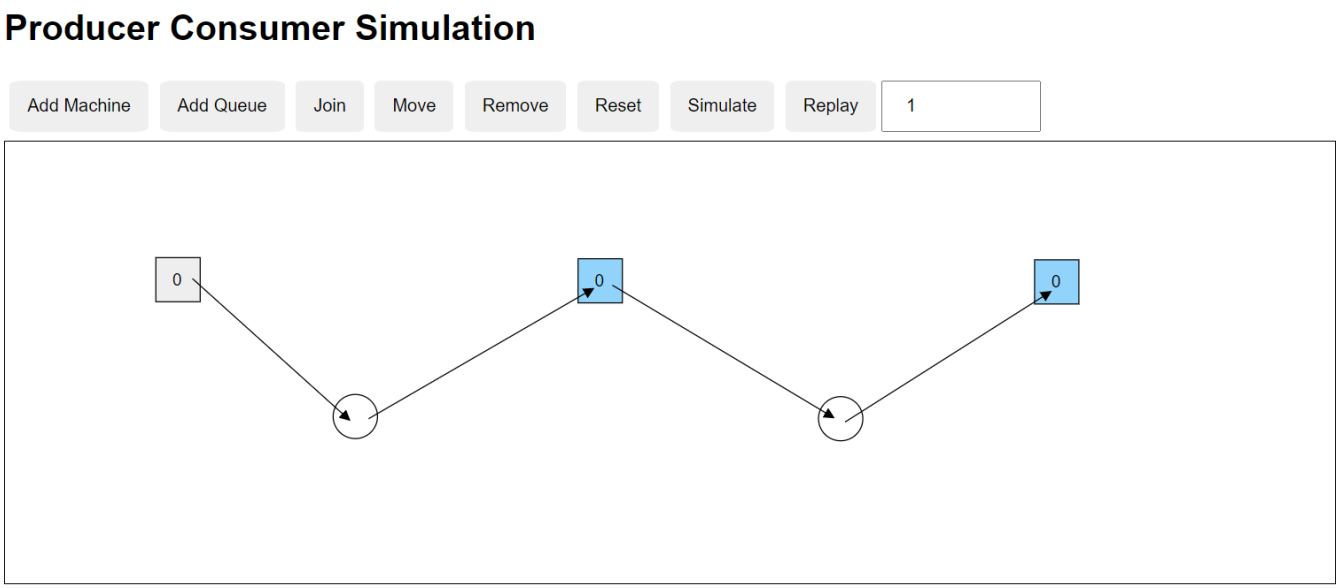
Design Decisions

- Websocket is used to open a two-way interactive communication session between the user's browser and a server.
- On websocket connection, server messages are sent to display the simulation changes on the client side.
- First Queue drawn is the starting queue and it’s marked by a different color than other queues.
- Each machine can have multiple queues before it (input queues) but must have only one output queue.
- Machine serve time is generated randomly for each machine. The generated time is bounded between 1s and 5s.
- Product color is generated randomly for each product.
- Every product has its own color that will keep it from start till the end and each machine will change its color to the product’s color being processed by it then change back to white (default color).

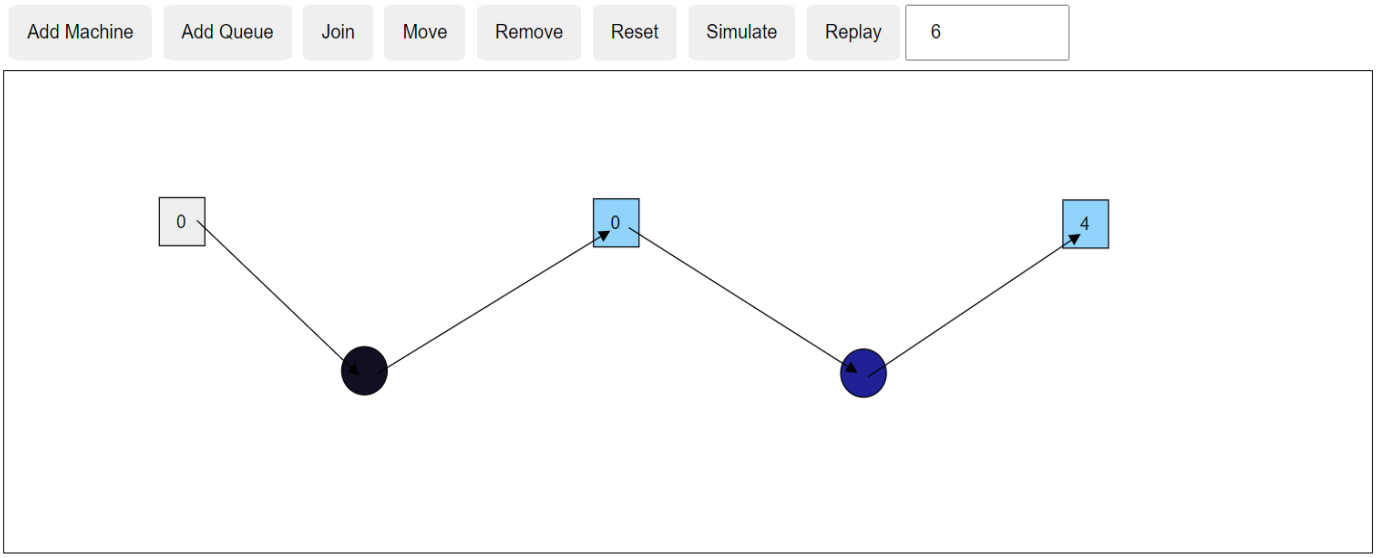
- Each queue displays its size.
- Each machine is running and processing its products on a separate thread different from other machines' processing threads.
- After the simulation ends, the user can start a new simulation or replay the previous simulation.
- Handled user mistakes:
    1. Prevention of multiple output queues to the machine.
    2. Line should connect between a machine and a line. Any other connection (e.g a machine to another) displays an error.
    3. On simulation start, all machines should be connected to queues and all queues should be connected to the graph. If any machine/queue was left without a connection, an error pops up.
    4. On moving/removing items, only unjoined machines/queues can be moved.
    5. Starting queue can't be removed. For any change, user should move it instead.

Snapshots of the UI

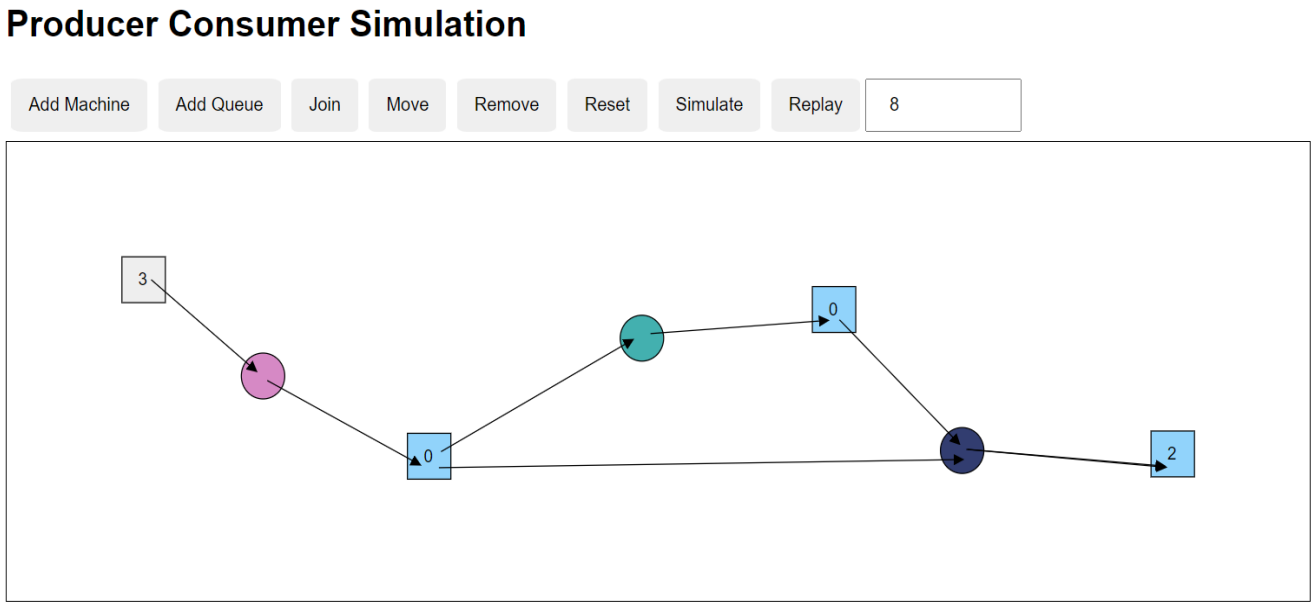
- The UI (Starting queue is marked with a different color than other queues as shown)



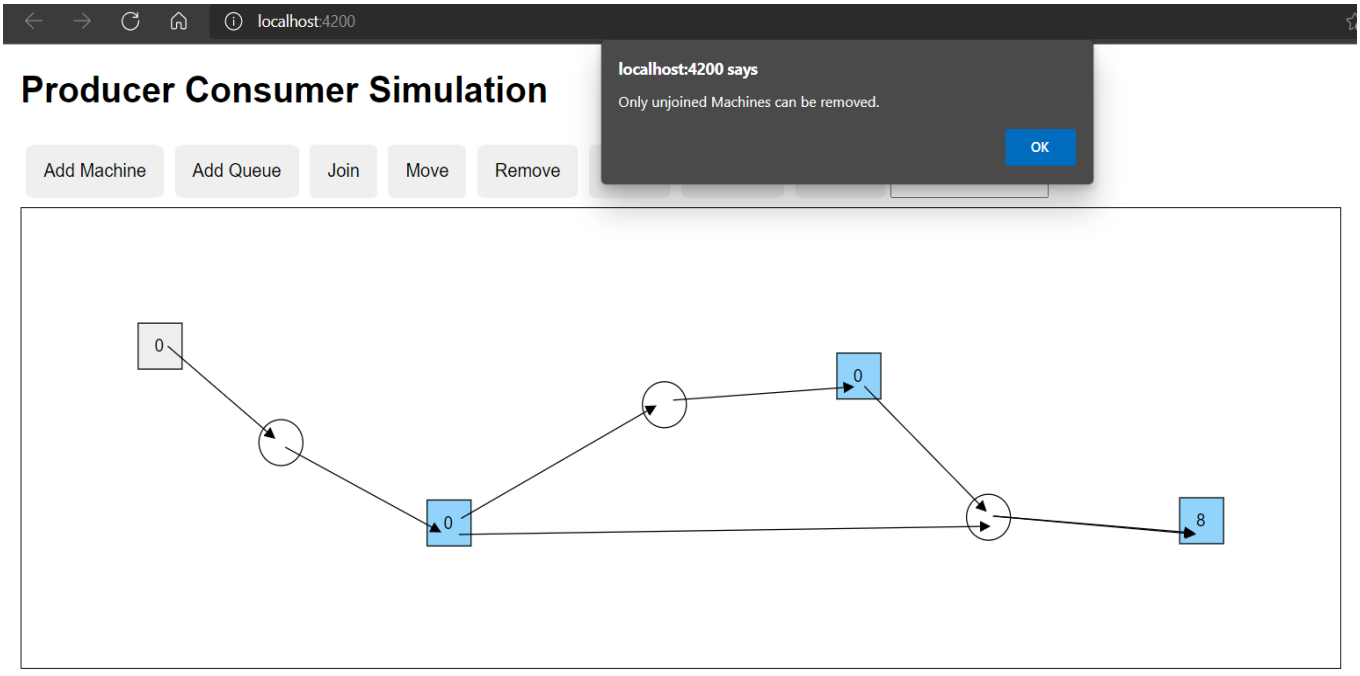
- Ongoing simulation



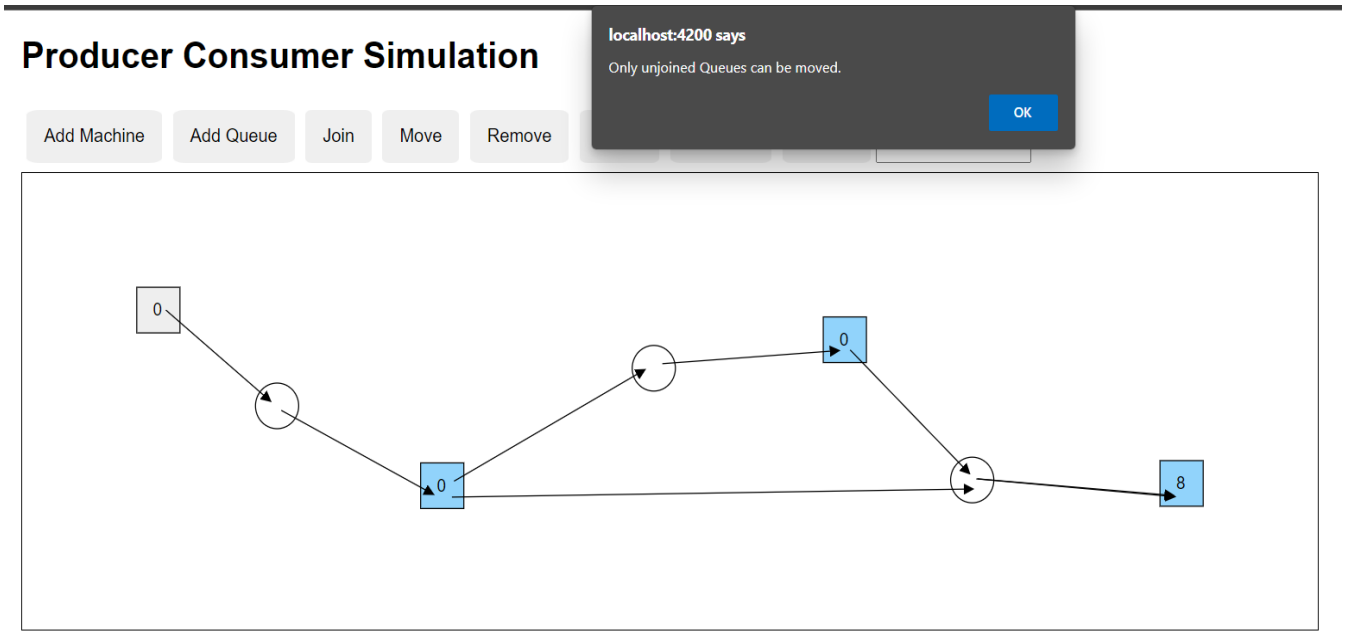
- Each queue displays its size as shown



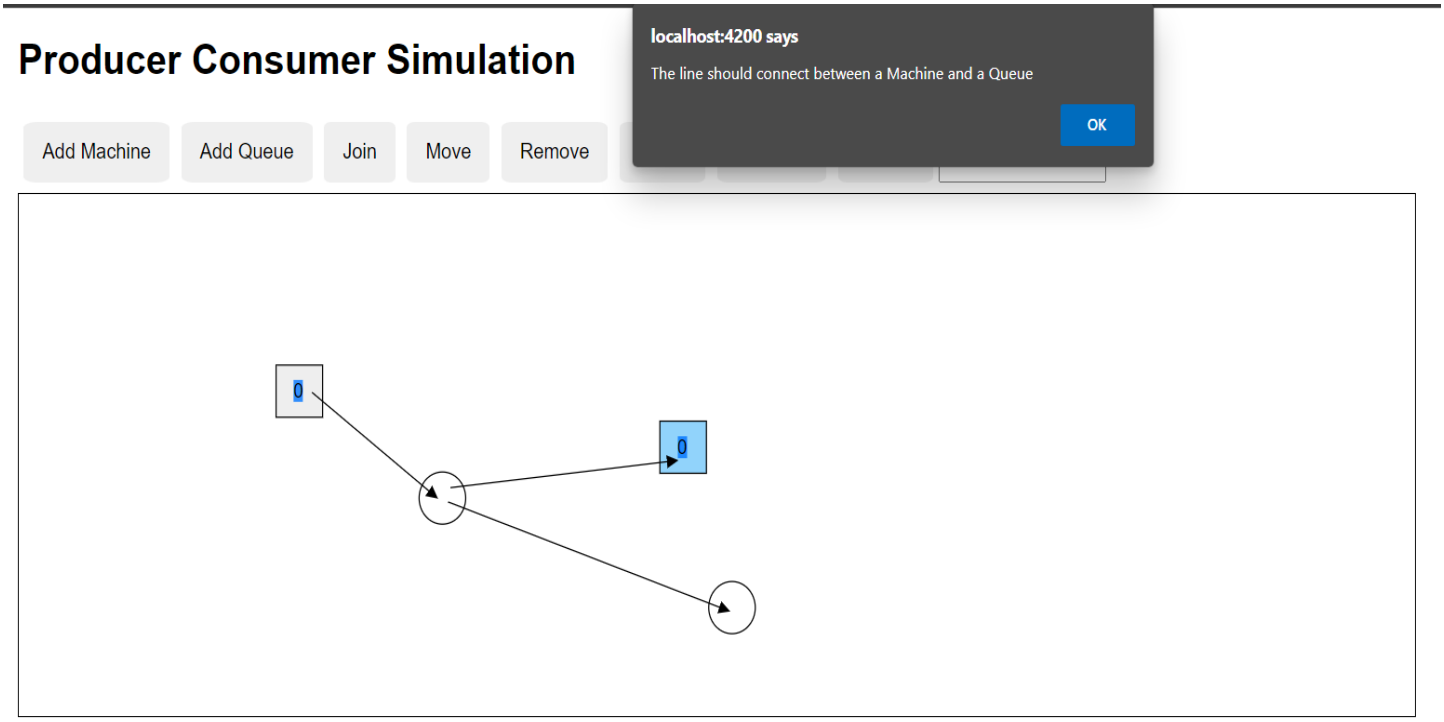
- Error (removing joined machine)



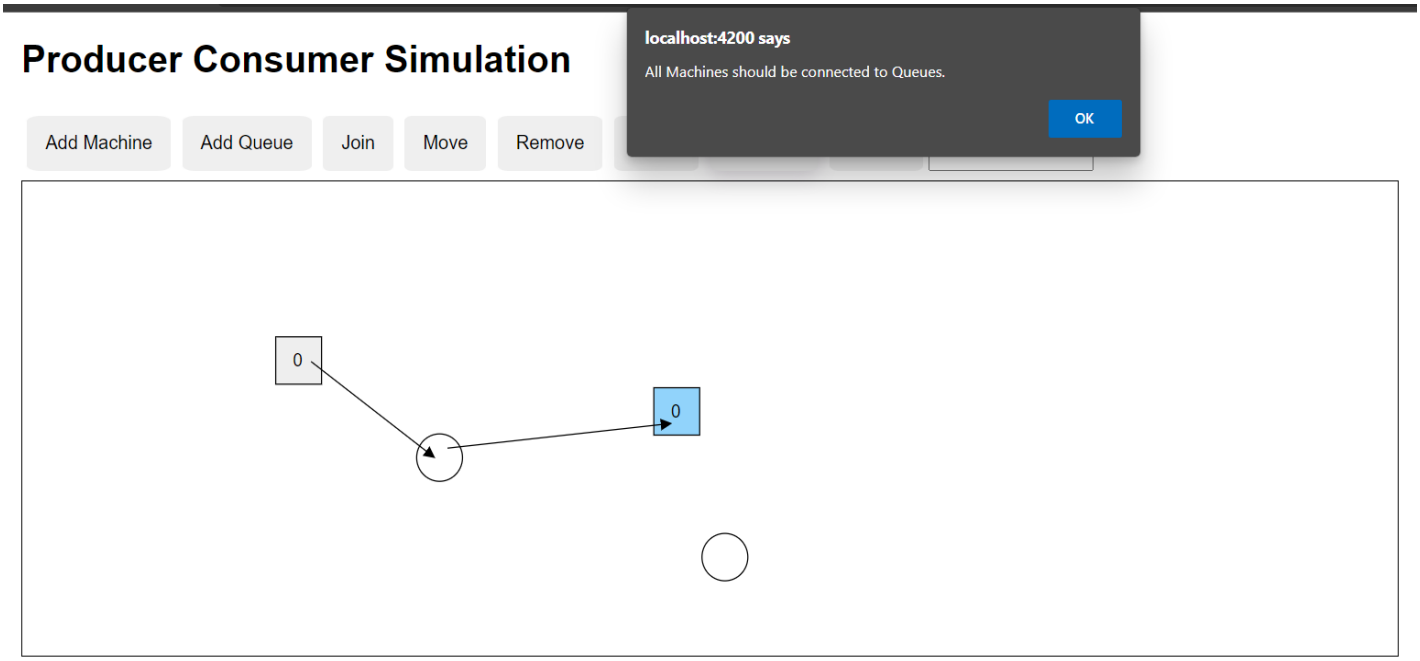
- Error (moving joined queue)



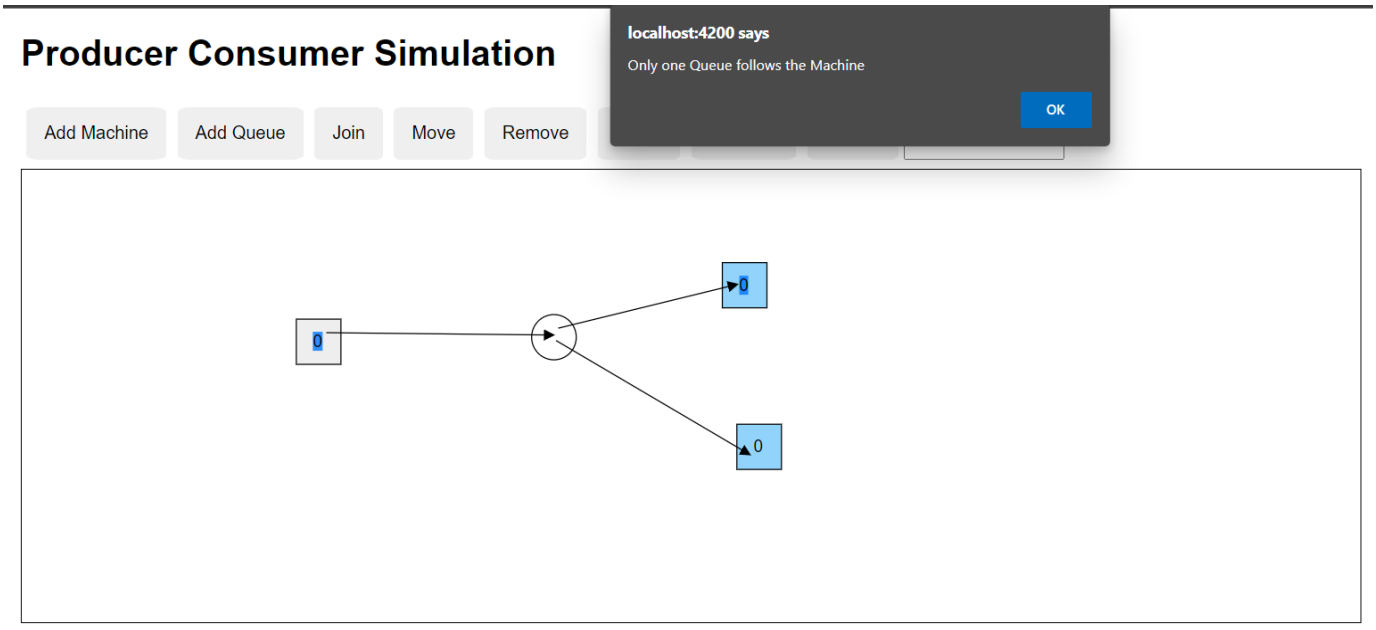
- Error (joining a machine to another machine)



- Error (starting simulation with an unjoined (not connected) machine)



- Error (connecting multiple output queues to a machine)



How-To-Use

- Add Items
  - Machine
    - Click the button (‘Add Machine’).
    - Click on any point on the board to add the circle that represents the machine.
    - The default color of the machines is (“white”).
  - Queue
    - Click the button (‘Add Queue’).
    - Click on any point on the board to add the rectangle that represents the queue.
    - All the queues have (‘deep sky blue’) color, while the start queue has (‘gray’) color.
  - Join
    - Click the button (‘Join’).
    - Select the start point, then drag until the second point.
    - You only can join a queue with a machine any other joined items won’t be accepted (ex. A queue with a queue, a machine with a machine or a queue or a machine with any nothing).
- Move Items
  - Queues & Machines

        - You can move any non-joined machine or queue.
        - Click the button (‘Move)
        - Drag the item you want to move to the new position.
    - Join
        - You can’t do any operation on the Joins.
- Remove Items
    - Queues & Machines
        - You can remove any non-joined machine or queue.
        - Click the button (‘Remove’).
        - Select the item you want to remove.
    - Join
        - You can’t do any operation on the joins.
- Replay
    - Click the button(‘Replay’) to repeat the operation.
- Simulate
    - Click the button(‘Simulate’) to simulate the operation.
    - The default number of products is 1.

References used to implement websockets:

1. https://youtube.com/playlist?list=PLXy8DQl3058PNFvxOgb5k52Det1DGLWBW
2. https://www.javaguides.net/2019/06/spring-boot-angular-8-websocket-example-tutorial.html?m=1