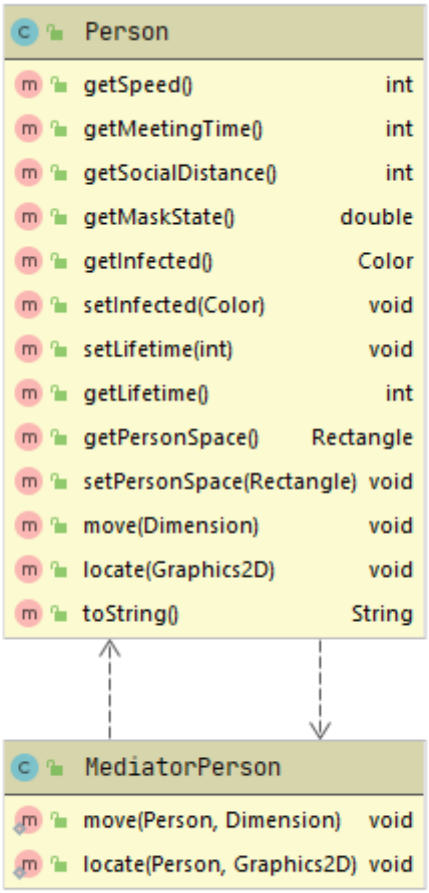
T. Bayram KOMUT

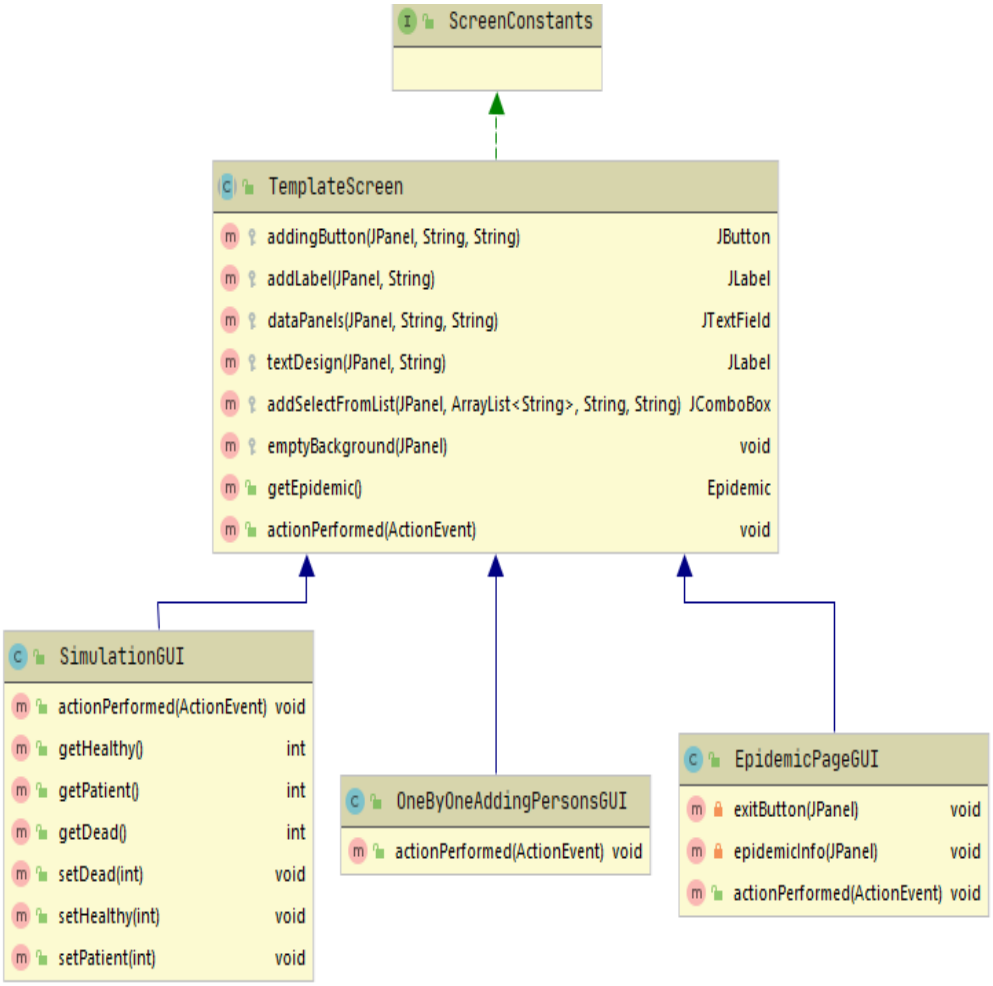
FINAL PROJECT REPORT

Class Diagrams Of The Project And Relationship Between Classes

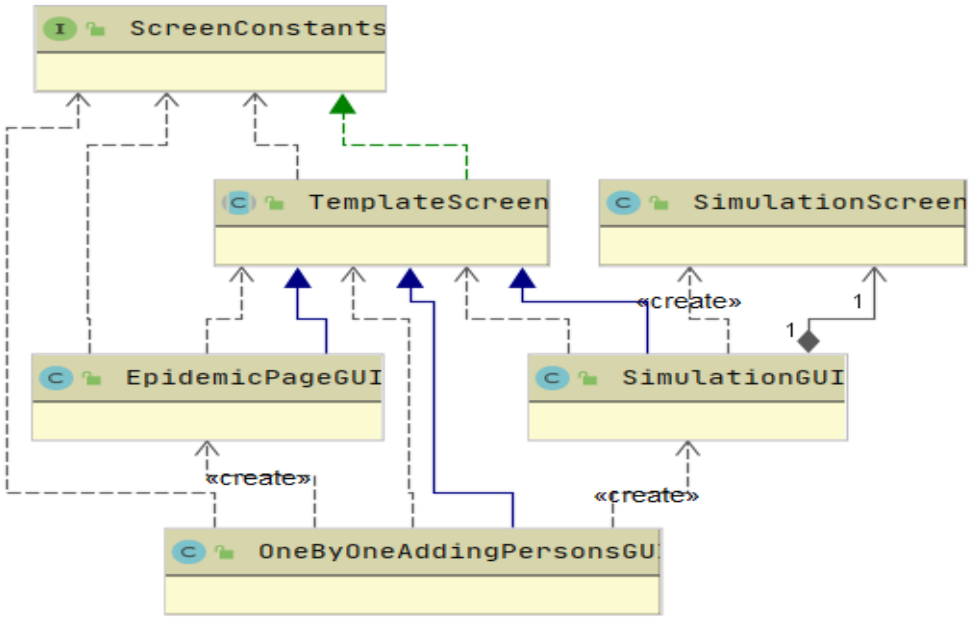
Mediator Design Pattern



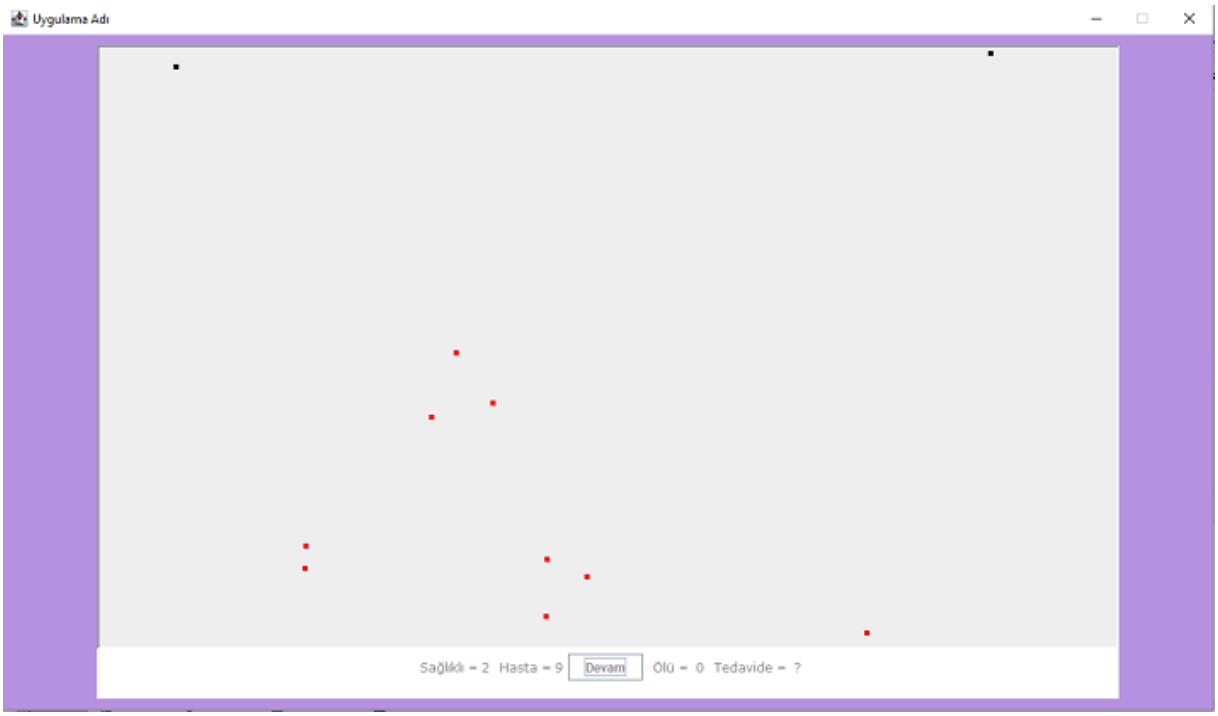
Mediator design pattern is used between these two classes. In fact, here the person class uses the mediator person class. Mediator person class only calls getter setter methods. Therefore it looks that way.



This is the general design of the GUI. There is an interface at the top. This interface can be used if other pages are added to the simulation program in the future. Template Screen class is created to keep general theme color and size settings in this program and frequently used methods on each page. The get Epidemic method catches our eye directly, Epidemic variable is subject of the our project. Below are other pages in the simulation program using this general template.



This diagram shows the connections between classes. There is a Has-a relationship between simulation classes. The reason is that the 1000x600 screen in our GUI is the SimulationScreen class. So it's part of it. Simulation GUI, on the other hand, is the entire simulation window, that is, not just the 1000x600 screen, but the stop, continue button and the labels showing the numbers.



Here the 1000x600 gray part is the SimulationScreen class, extends from JPanel, it is not a frame. Below it are 1 button and 4 labels, together it creates a frame, ie SimulationGUI class.

About The Screen Pause And Continue With Multithreading

I create a new thread object and give the SimulationScreen class as a parameter to this thread object. This class should implement the Runnable interface and provide the run method. The event in pausing and resuming actually takes place within this run method. This method includes an endless loop until the thread dies. In addition to there are two if statements to continue and stop of the simulation.

Inside these two if statements, it starts and pauses a timer object we use to update the screen. Thus, the screen is paused and continued. The type of the variable we use with these if statements must be volatile. Because it provides convenience, eliminates the need for communication between two threads.

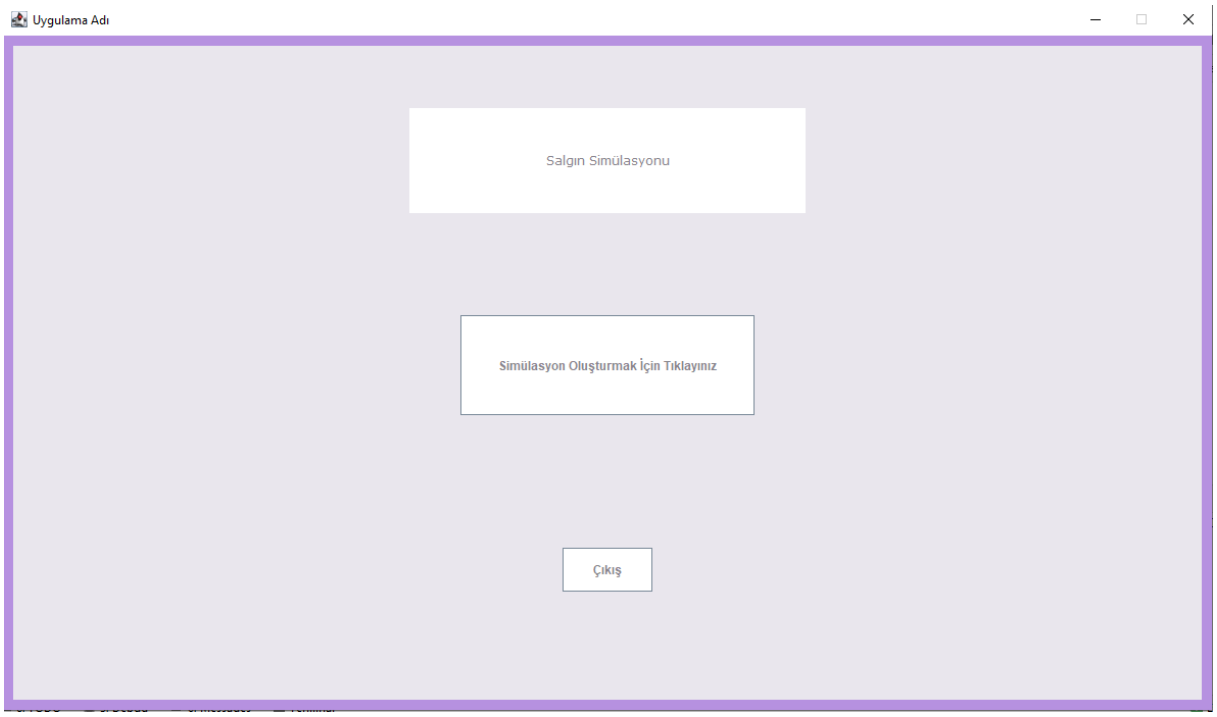
```
panel = new SimulationScreen(persons, epidemic,healthy,patient,dead);
Thread thread = new Thread(panel,  name: "Simulation Thread");
```

SimulationScreen object is given to the thread as a parameter.

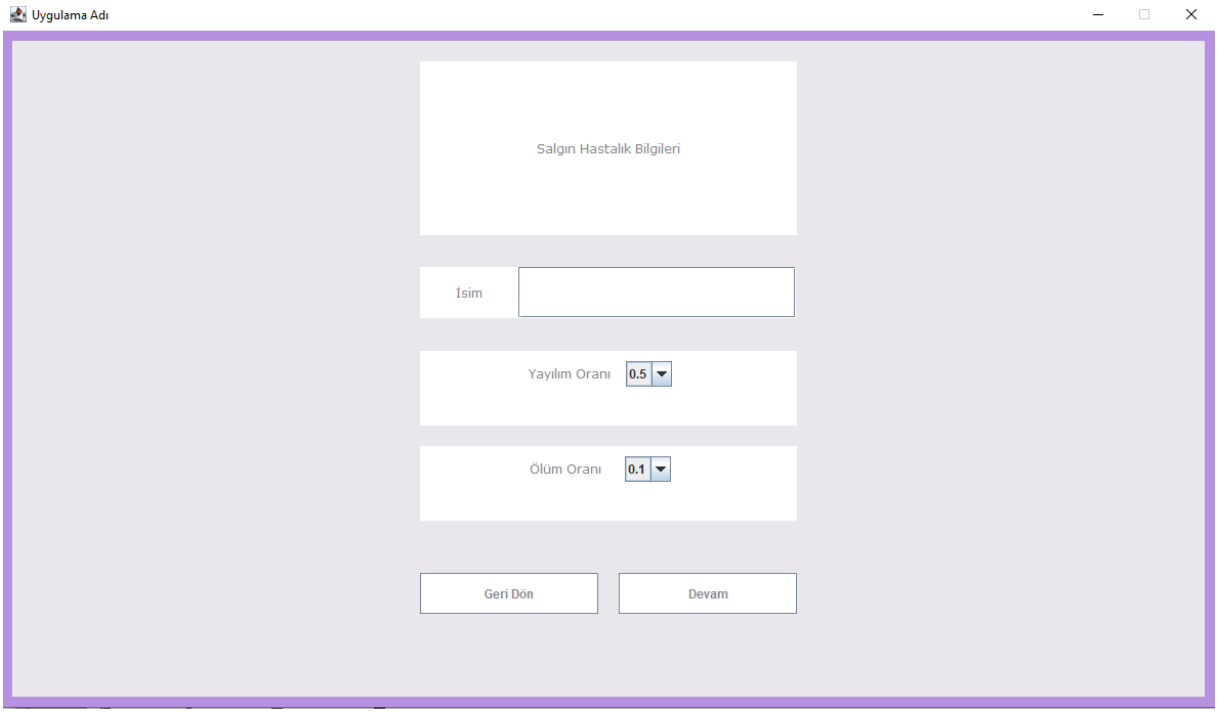
```
public void run() {
    while(kill) {
        if (state) {
            //  System.out.println("Devam Ediyor.");
            h1.setText("Sağlıklı = " + healthy);
            p1.setText("Hasta = " + (patient + 1)) ;
            d1.setText("Ölü =  " + (dead - 1)) ;
            timer.start();
        }
        if (!state) {
            // System.out.println("Duraklatıldı.");
            timer.stop();
        }
    }
}
```

The method to run in the Simulation Screen class, "pauses and runs the timer."

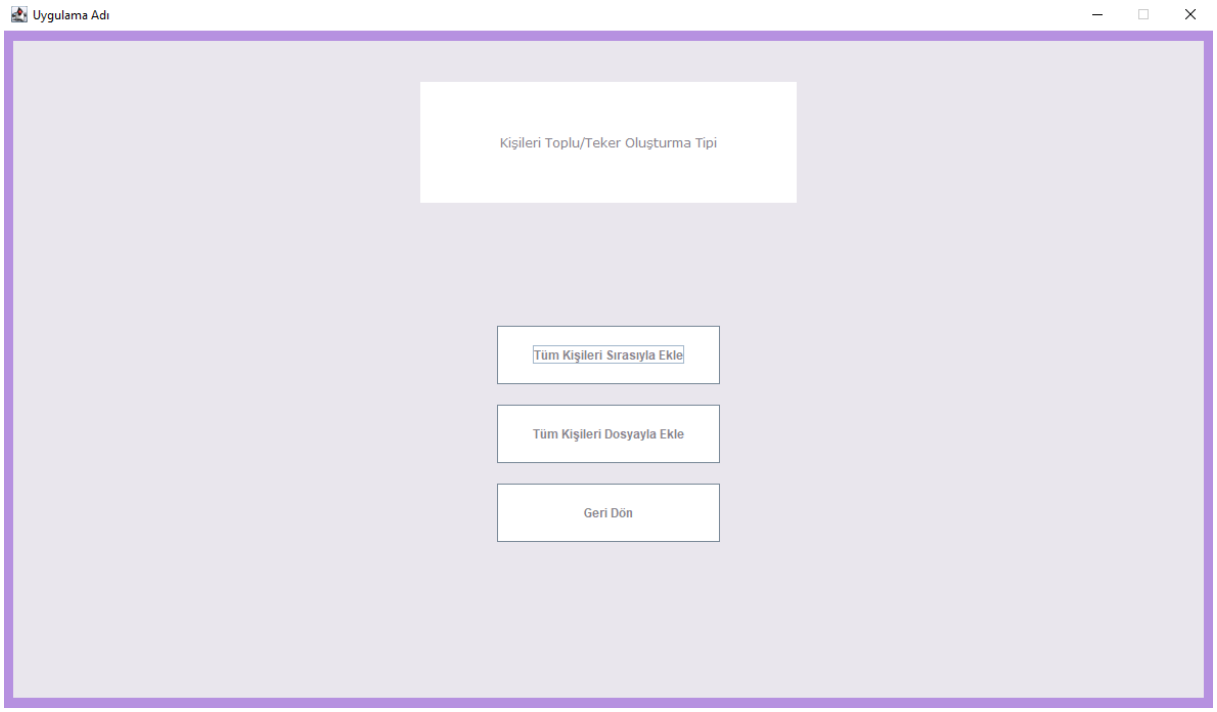
Example of The Running Simulation Program



On the first page, there are only two buttons called start program or exit program.

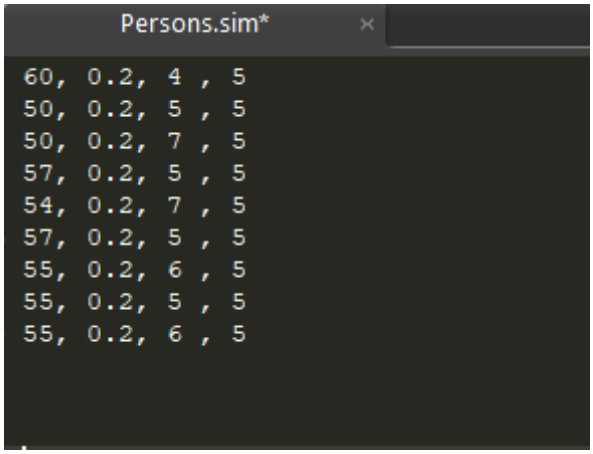


The second page is designed to get information about the epidemic.



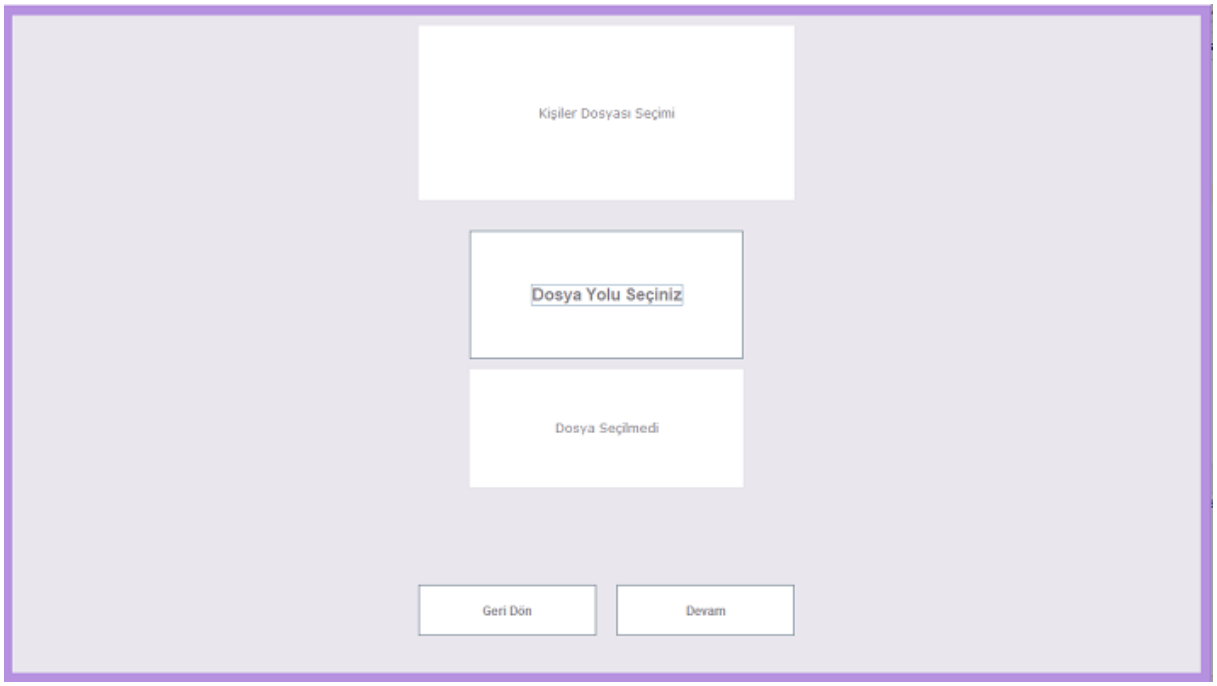
The third page, which is a part of Question 1 in the project, is the page about adding people one by one or multiple. If clicking add one by one, the number of people in the population is first obtained.

If multiple add is clicked, the program retrieves a list of contacts according to a certain format from a file, there is a certain format in this list, the person properties according to that format are read. There is a sample format for the file that I have specified as .sim (simulation) in this project. I will tell about now.

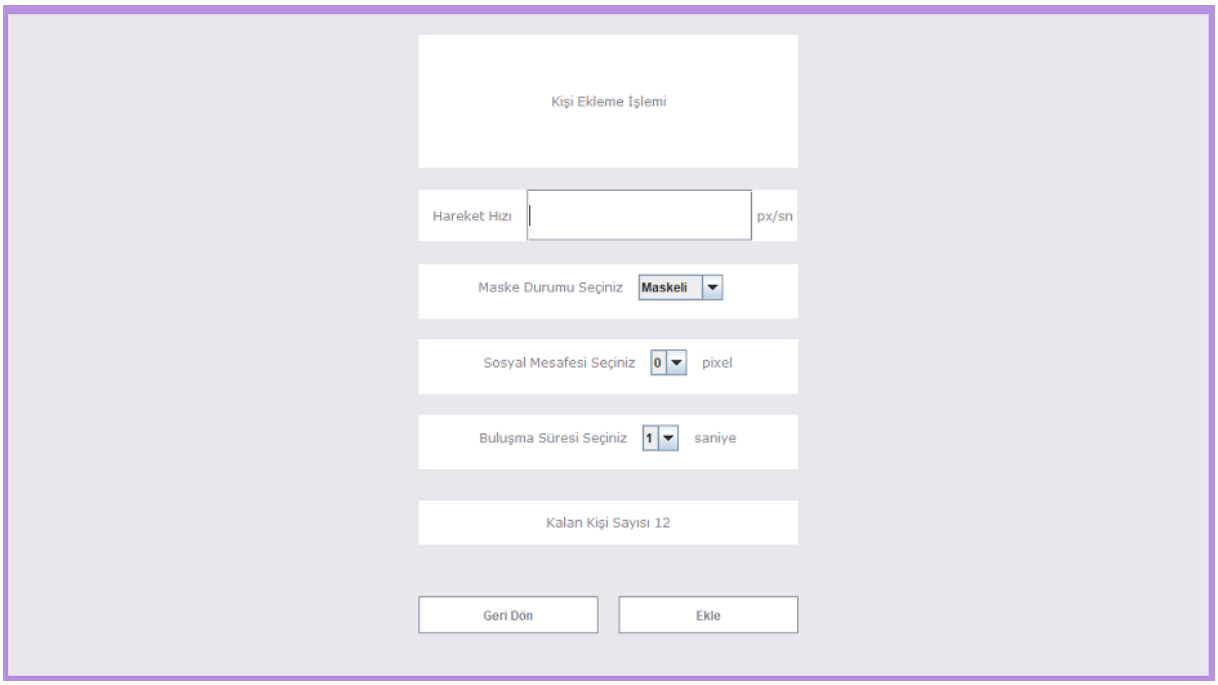


(speed, mask, social distance, meeting time)

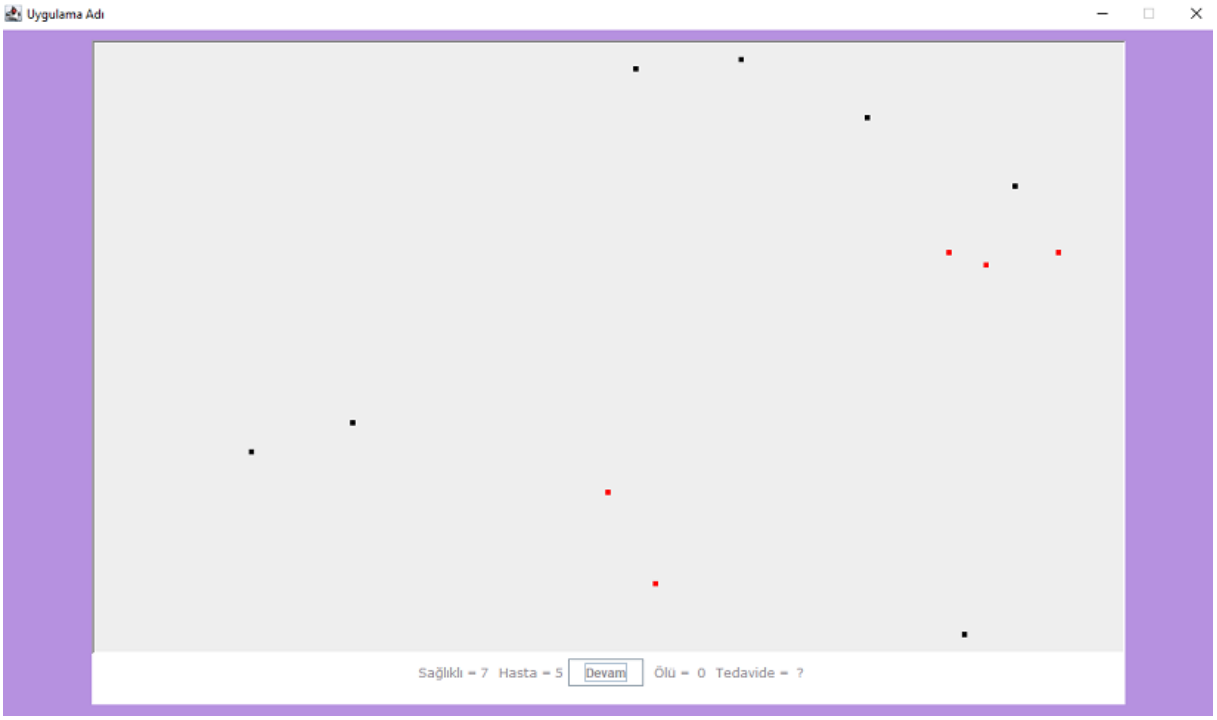
That's how I chose the format. There are 4 entries about the person. These are, in turn, its speed, mask status, social distance and meeting time. I have specified the mask status in terms of double type. Like 0.2 without mask, like 1 with mask. The rest is the same.



In multiple additions, we click the button to select a file and from there we select the appropriate file in the specified format. I will send it with homework as an example. Bugs can happen every now and then, it's rare.



The upper screen comes when we choose to add one by one. Here we select the values of each person and click the add button and add. It writes and updates the number of people to be added, just above the buttons.



We chose everything and finally we reach our simulation page. Here, it shows a 1000x600 screen, it is stopped and continued with the button below on this screen and updates the information.

Deficiencies in The Project And General Evaluation

There is nothing about the hospital in the project, so there is a question mark on the label written as treatment in the simulation screen.

I thought of drawing graphics, but while searching the net, I was afraid I'd cause -100 and chose not to.

I extracted the JAR file, but it didn't work, it probably won't work. I am sorry for this. The project doesn't work from jar, due to an error while creating jar.

Since there is no hospital, the simulation ends when there is no sick person left. Otherwise simulation always run. I calculate the dead, but I do not delete them from the canvas, I paint them blue. Meanwhile, the black squares represent healthy people and the red squares represent sick people.