*Project Report*

**ELECTRICAL CIRCUIT SIMULATOR**

*Object Oriented Programming - IT3100E*

Group 11

*Luyen Minh Khanh 20214905*

*Do Dinh Kien 20214906*

*Le Trung Kien 20214907*

*Vu Duc Hung 20214902*

*Hanoi University of Science and Technology*

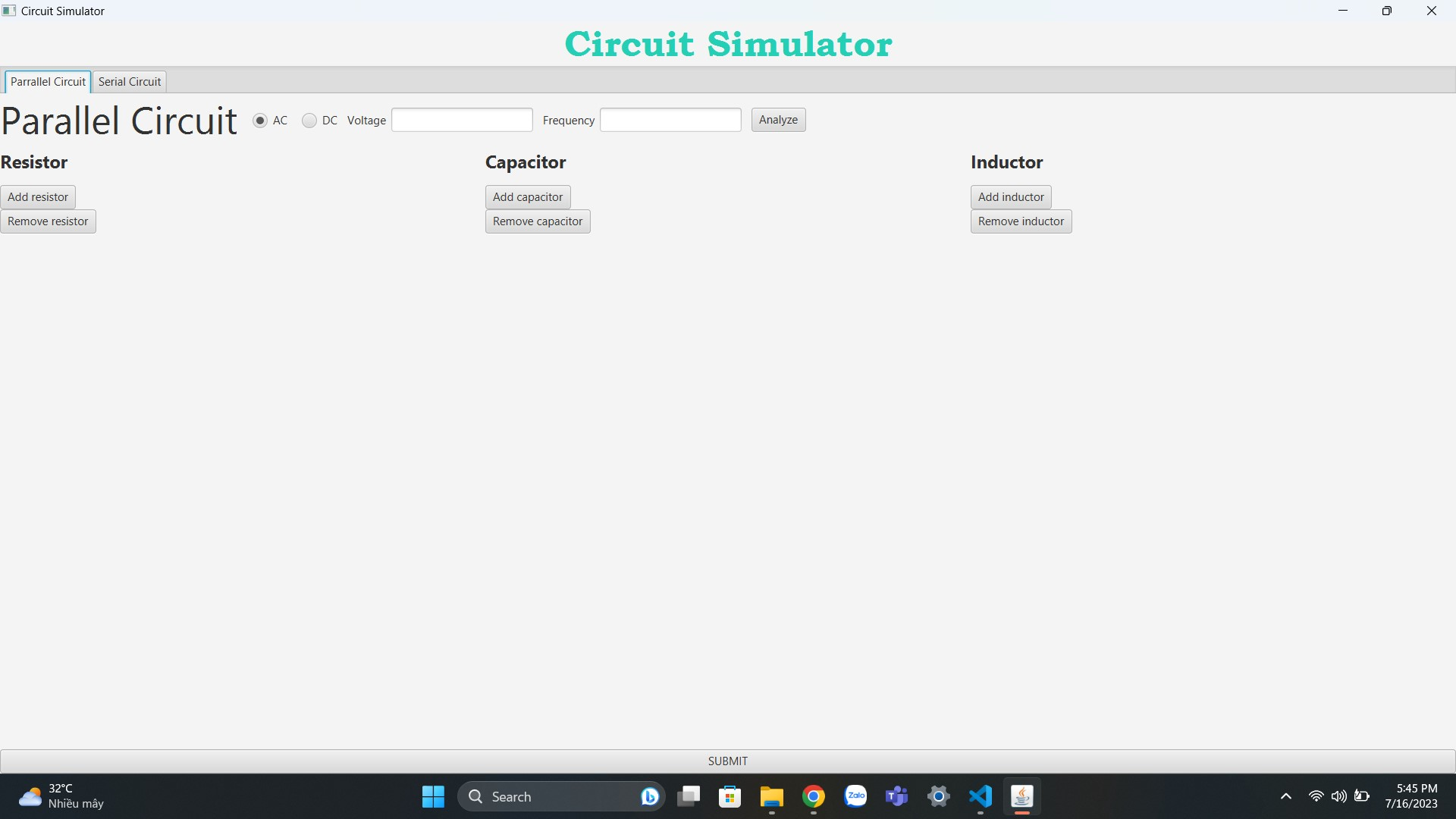
*School of Information and Communication Technology*

Table of contents

1. Introduction……………………………………………………………….
2. Assignment of members………………………………………………….
3. Project description……………………………………………………..…
4. Design……………………………………………………………………
5. Conclusion…………………………………………………………………
6. **Introduction**

Electrical circuits are a vital part of our everyday lives. They are used in everything from our homes to our cars to our computers. However, electrical circuits can be complex and difficult to understand. This is where electrical circuit simulators come in. These applications allow engineers and students to test and debug electrical circuits without having to build them in the real world.

In this project, we will build an electrical circuit simulator using Java and JavaFX. The user interface is shown as below.



*Figure 1. Circuit simulator interface.*

1. **Assignment of members**

The packages and classes are shown in the table below:

| Packages | Classes | Contents |
| --- | --- | --- |
| Circuit | *ElectricalCircuit.java* | Add, remove elements and check short circuits. |
| *ParallelCircuit.java* | Calculate equivalent resistance for parallel circuit. |
| *SerialCircuit.java* | Calculate equivalent resistance for serial circuit. |
| CircuitAnalysis | *CircuitAnalysis.java* | Analyze circuit and show the analysis table. |
| CircuitDiagram | *CircuitDiagram.java* | Draw circuit diagram. |
| *Test.java* | Testing. |
| ComplexNumber | *ComplexNumber.java* | Complex numbers calculating method and expression |
| Elements | *Capacitor.java* | Accessors and mutators for capacitors. |
| *Elements.java* | Generate constructors for electrical elements. |
| *Inductor.java* | Accessors and mutators for inductors. |
| *Resistor.java* | Accessors and mutators for resistors. |
| SourceType | *ACVoltageSource.java* | Accessors and mutators for AC voltage source. |
| *DCVoltageSource.java* | Accessors and mutators for DC voltage source. |
| *VoltageSource.java* | Generate constructors for voltage sources. |
| TableRowData | *TableRowData.java* | Constructors for analysis table. |

*Table 1. Packages and classes.*

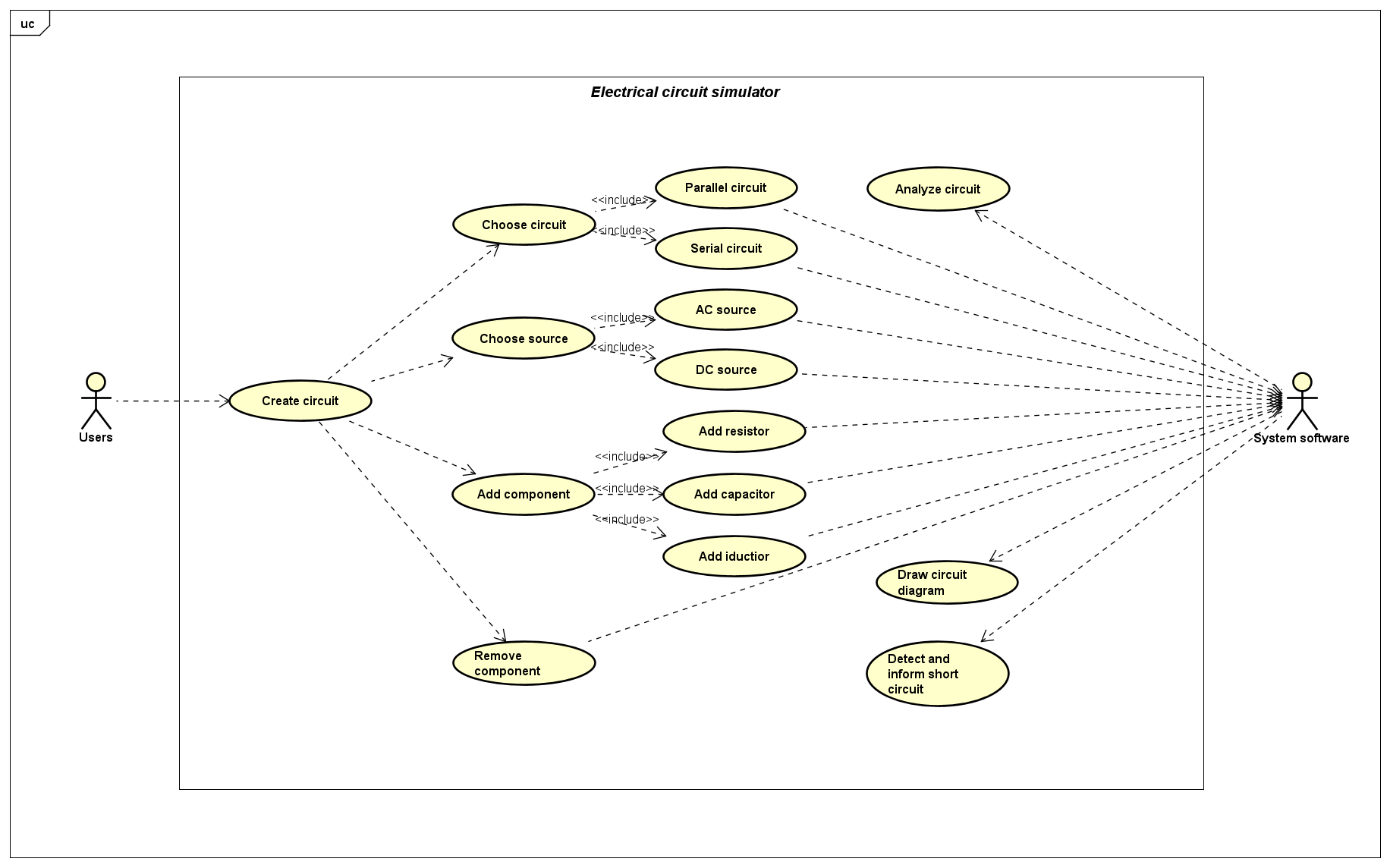
Beside the above classes, the source code also contains other class:

* *App.java:* contains the main() method to run the simulator.
* *CircuitController.java:* the controller class.
* *GUI.fxml:* FXML document for GUI programming.

1. **Project description**
2. Project requirements

This project requires some platforms and toolkits in order to set up and run the simulator:

* Java Development Kit (JDK): a collection of tools used to develop and run Java applications.
* JavaFX Runtime: the runtime environment for JavaFX applications.
* A code editor: Any code editor can be used to develop JavaFX applications (Eclipse, NetBeans, Visual Code Studio…).

1. Use case diagram

*Figure 2. Use case diagram.*

* For users:

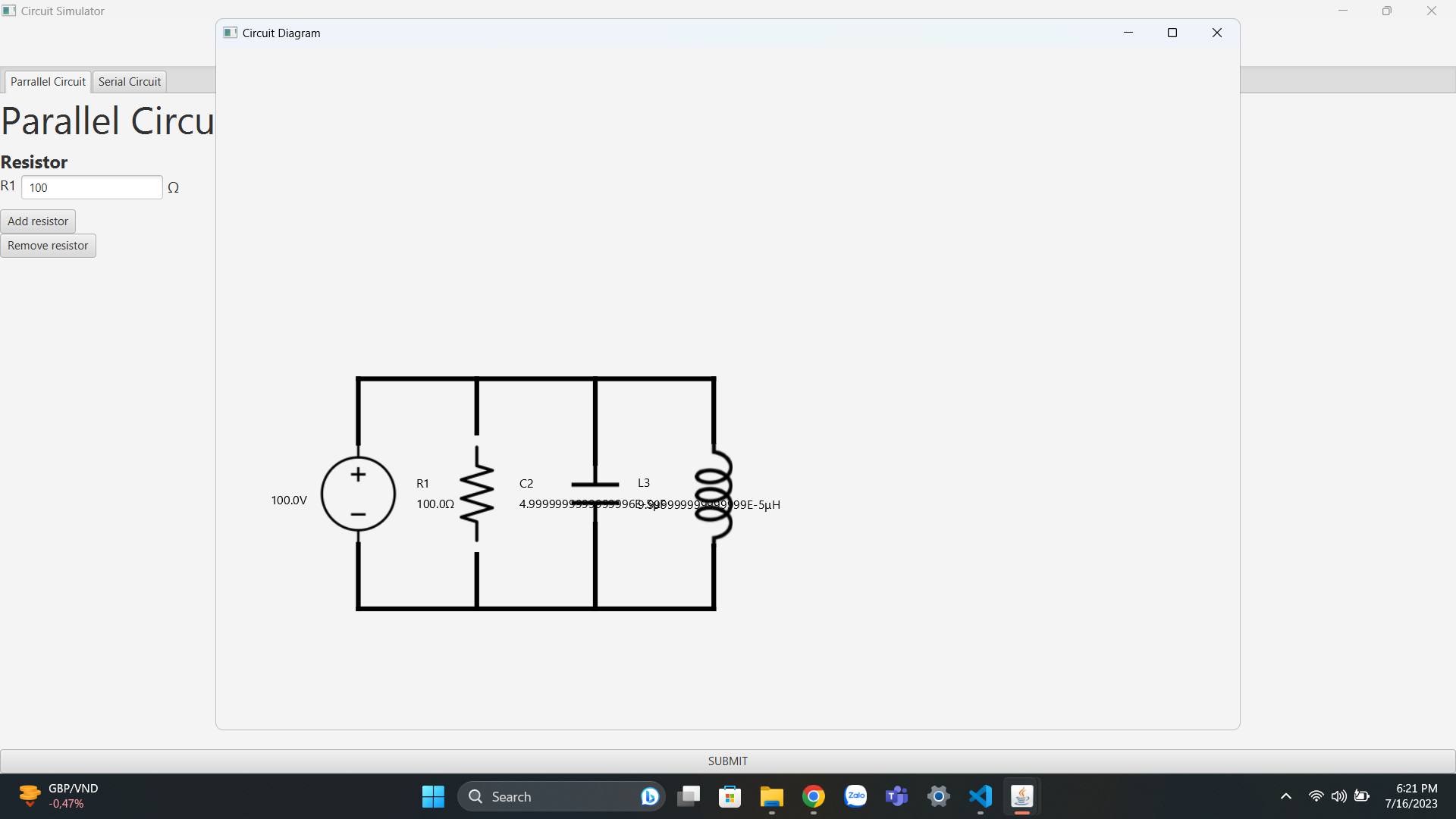
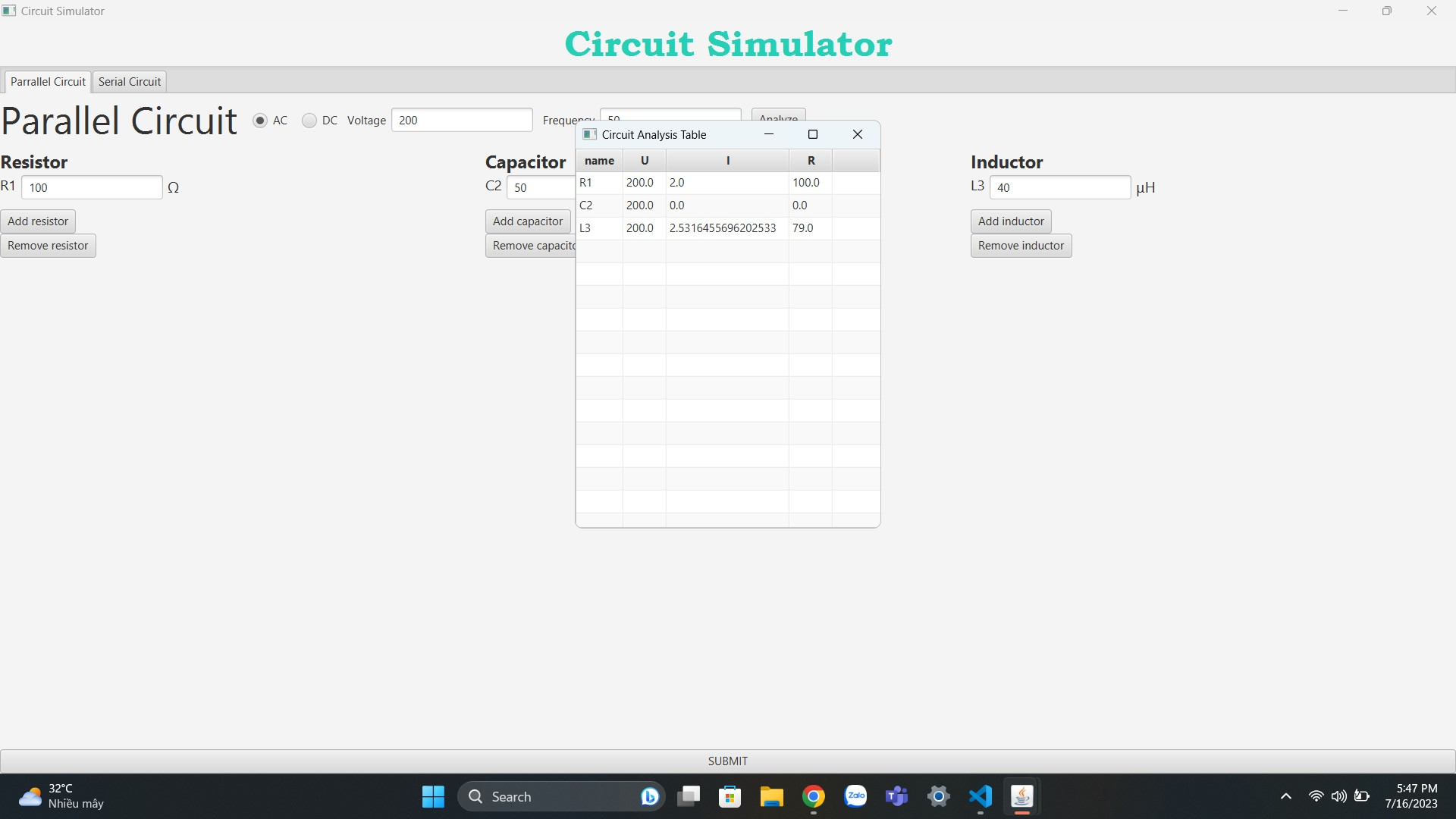
| The simulator demonstrates two types of circuits: parallel circuit and serial circuit. Users can choose one or two tabs in the navigation bar, like in the image beside. |  |
| --- | --- |

Then, the user can start to construct a circuit by adding elements (resistor, inductor, capacitor, voltage source) and their values.

The user can also remove the elements he or she does not want to keep in the circuit with the remove button. Then the user can press the *submit* button when finishing constructing the circuit.

* System software:

After the user presses the *submit* button, the simulator will draw the corresponding circuit diagram, show the circuit analysis table and calculate the equivalent resistance of the circuit.



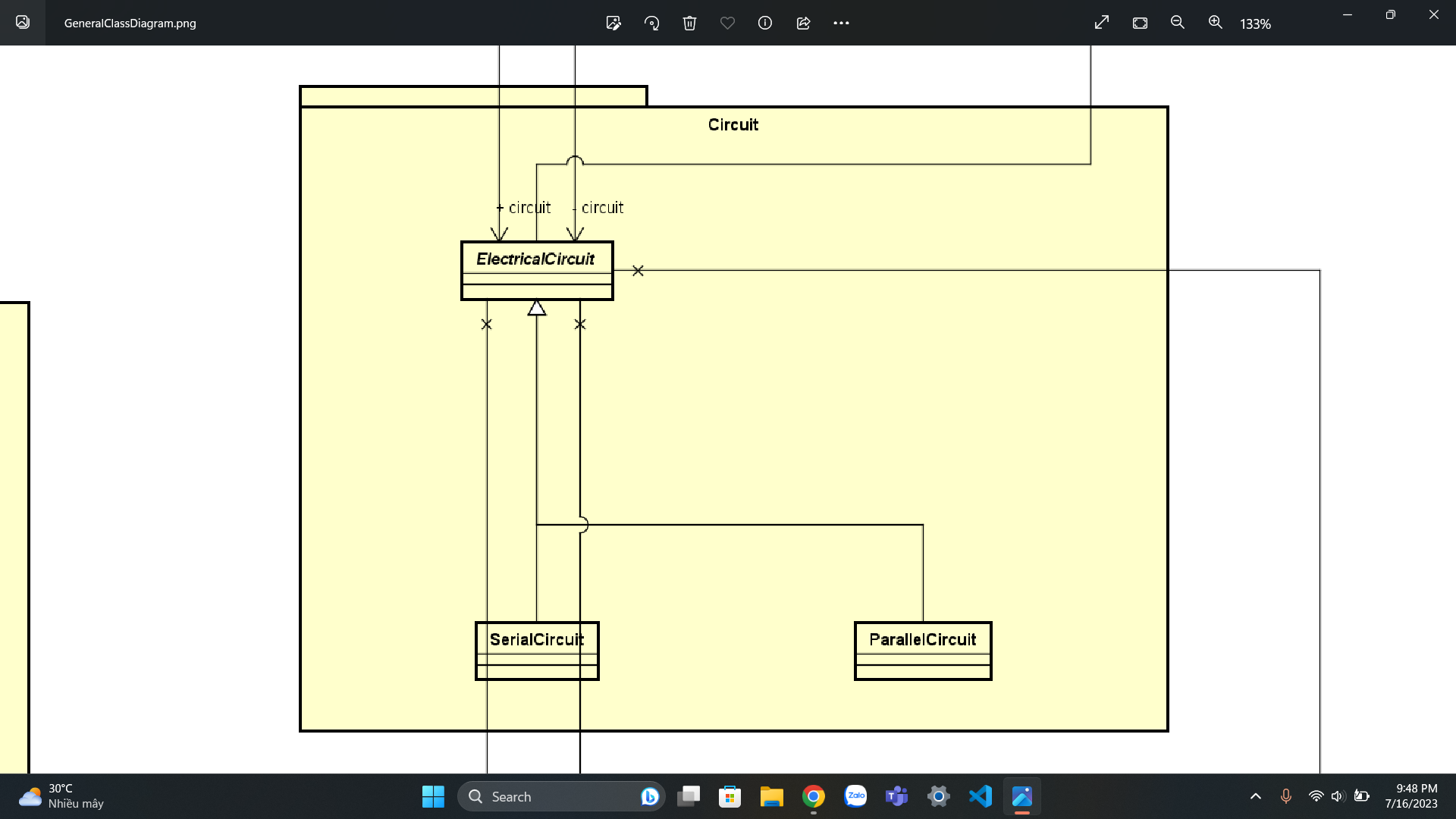
*Figure 3. Example for analysis table and circuit diagram.*

The system will also detect and announce the user if a short circuit occurs and ask them to change the element causing it. In this case, the circuit analysis table will be empty.

1. **Design**

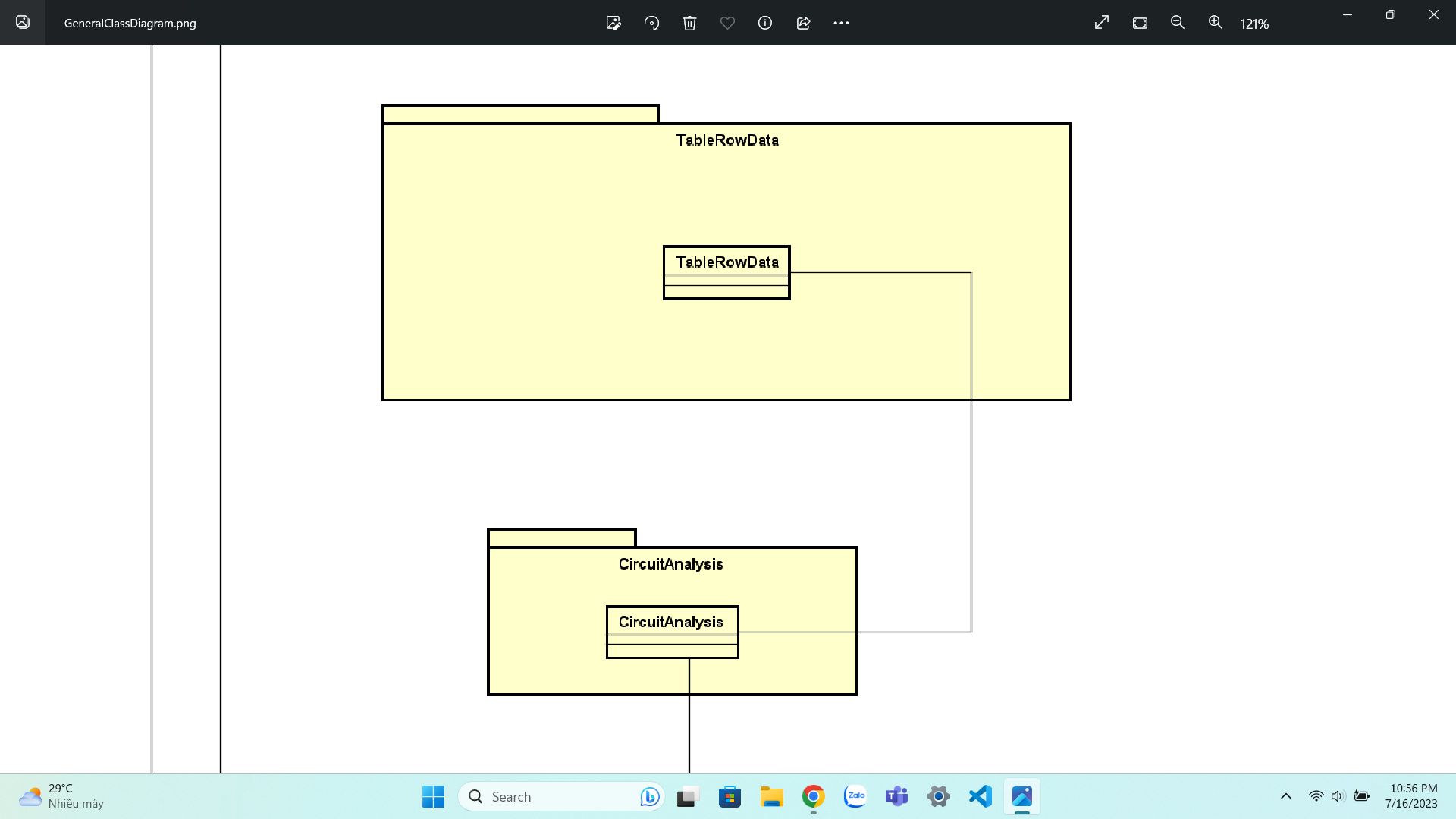
- Circuit package:

The two classes *ParallelCircuit.java* and *SerialCircuit.java* extend the class *ElectricalCircuit.java.* They also import the classes *Elements.java* and *VoltageSource.java* to compute equivalent resistance and other values for each element and source case.

**

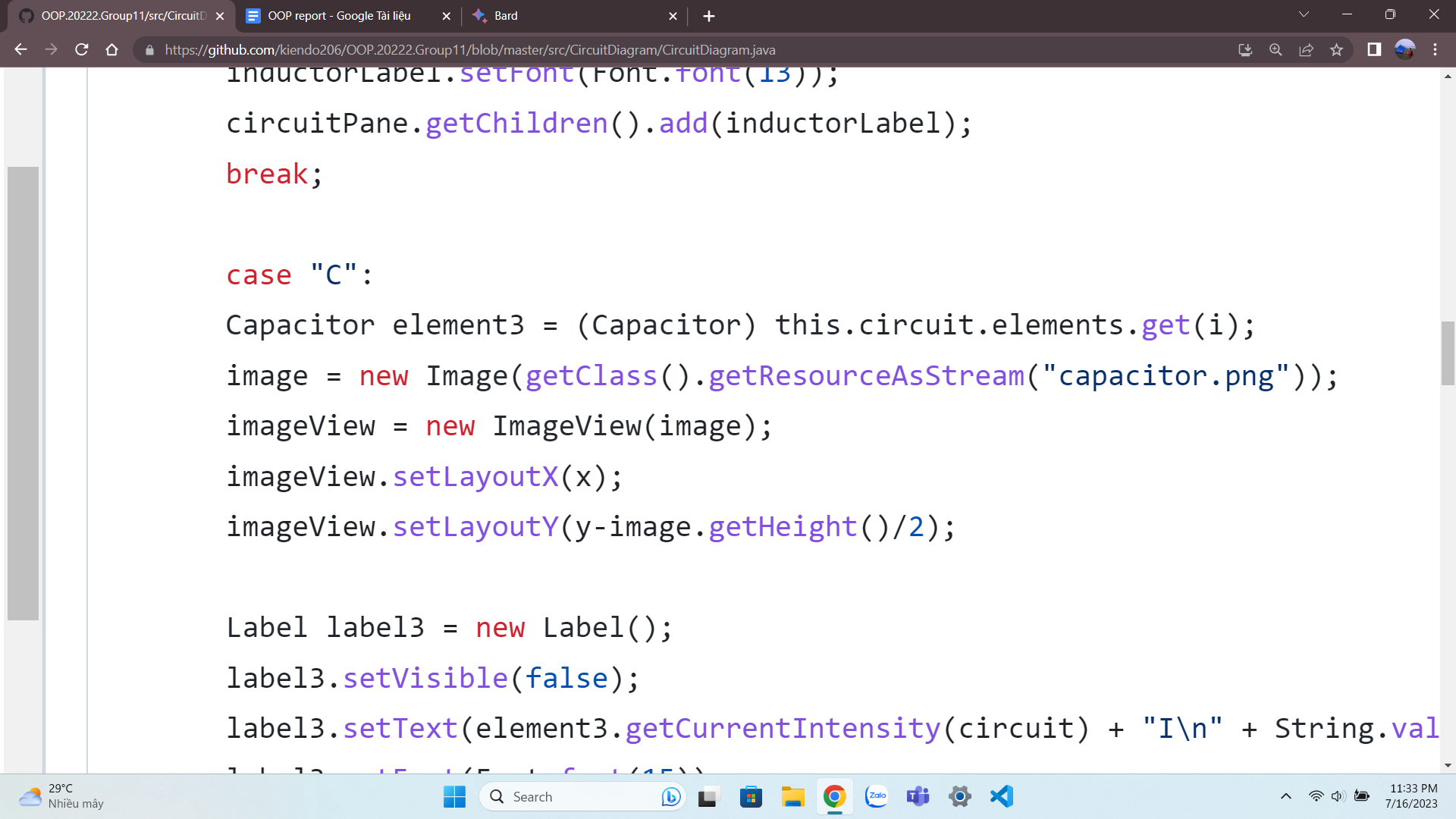
- *CircuitAnalysis.java* and *TableRow.java:*

The class *TableRow.java* is imported to create the analysis table. Then the class *CircuitAnalysis.java* will add the electrical value and then show the table to users.

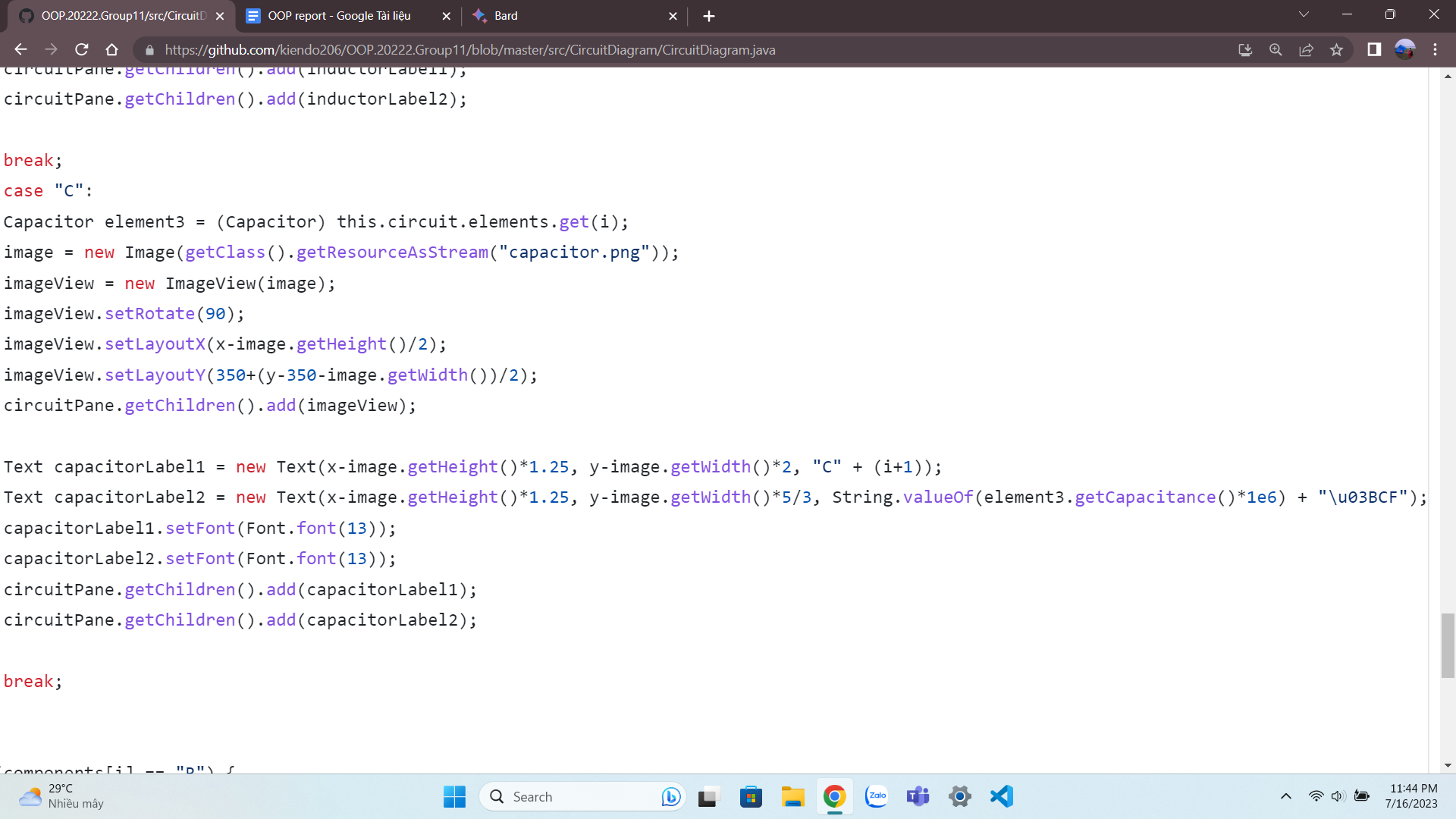
**

- *CircuitDiagram.java:*

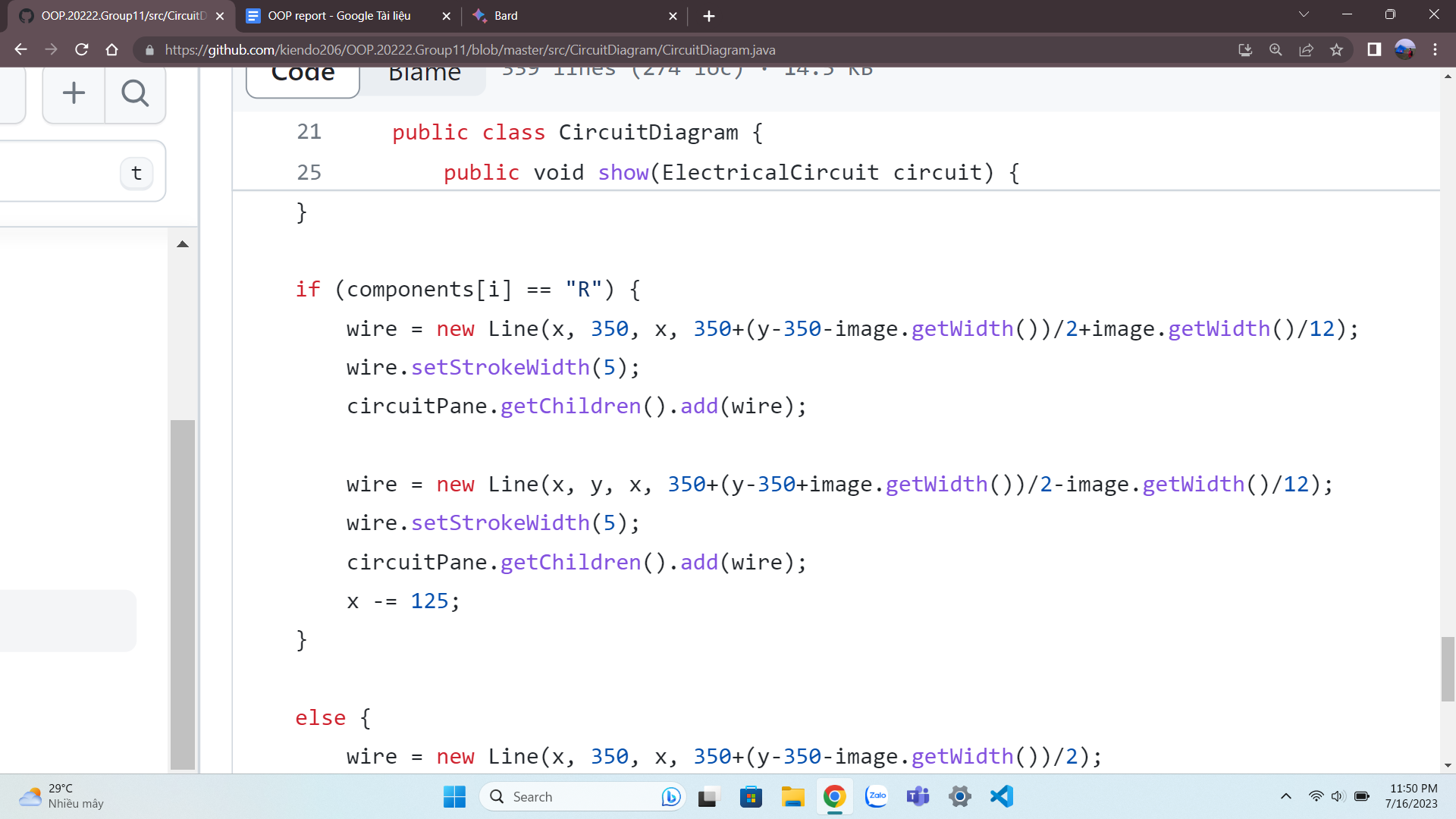
This is the method to draw the corresponding circuit diagram and show it to users.



The above code creates an ImageView object to display the image of a capacitor. The Image object is created with the path to the image *capacitor.png* as the argument. The ImageView object is then positioned on the Pane object using the setLayoutX() and setLayoutY() methods.

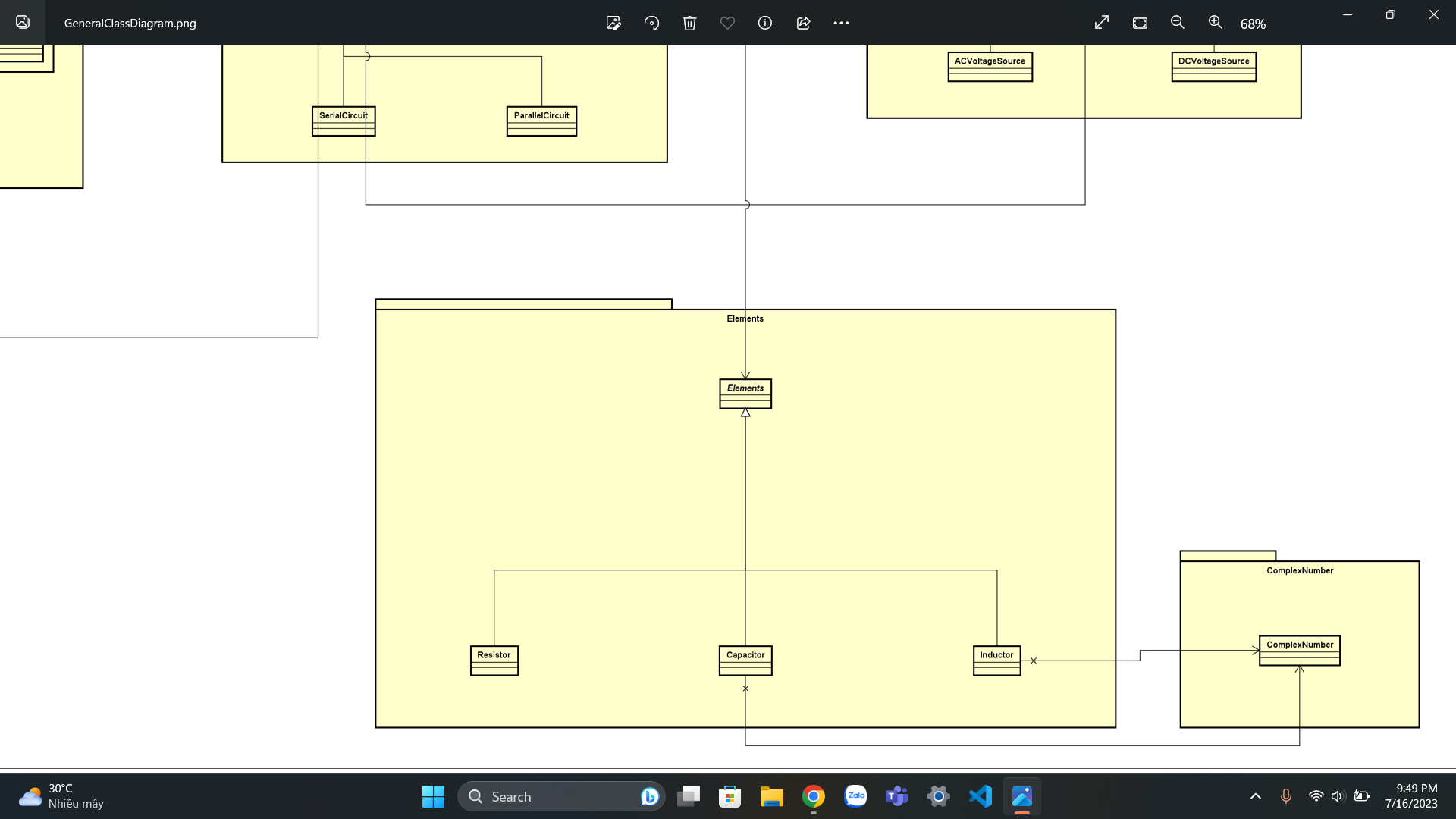


After that we create a Text object to display the name and value of the resistor. The Text object is created with the capacitor's name and value as the arguments. The similar method is applied for the inductor and resistor.



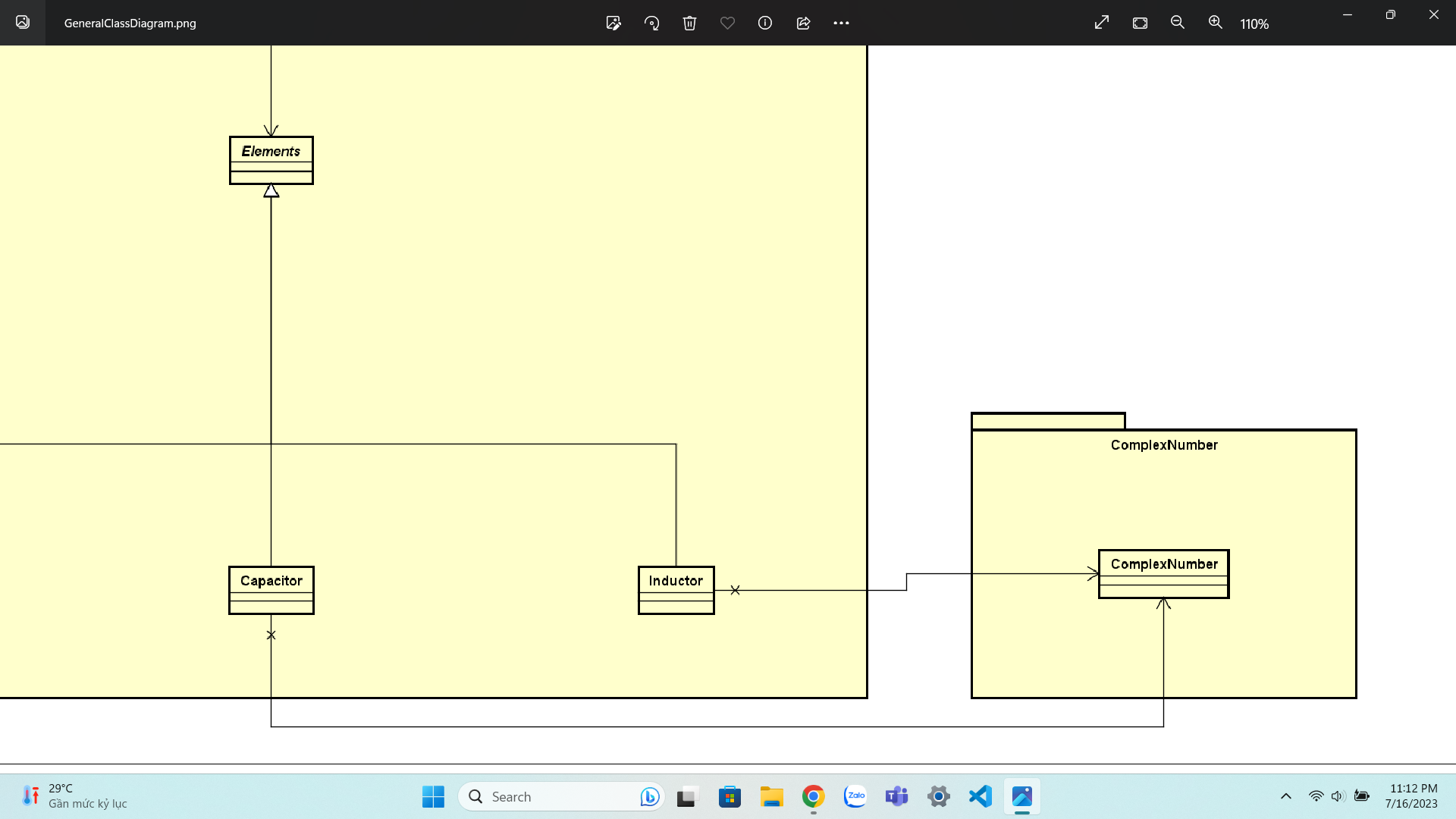
For wire, we create a Line object to represent the connection between the two elements. The Line object is created with the start and end points of the line as the arguments. It is then given a 5-pixel stroke width and added to the Pane object.

- Elements packages:

**

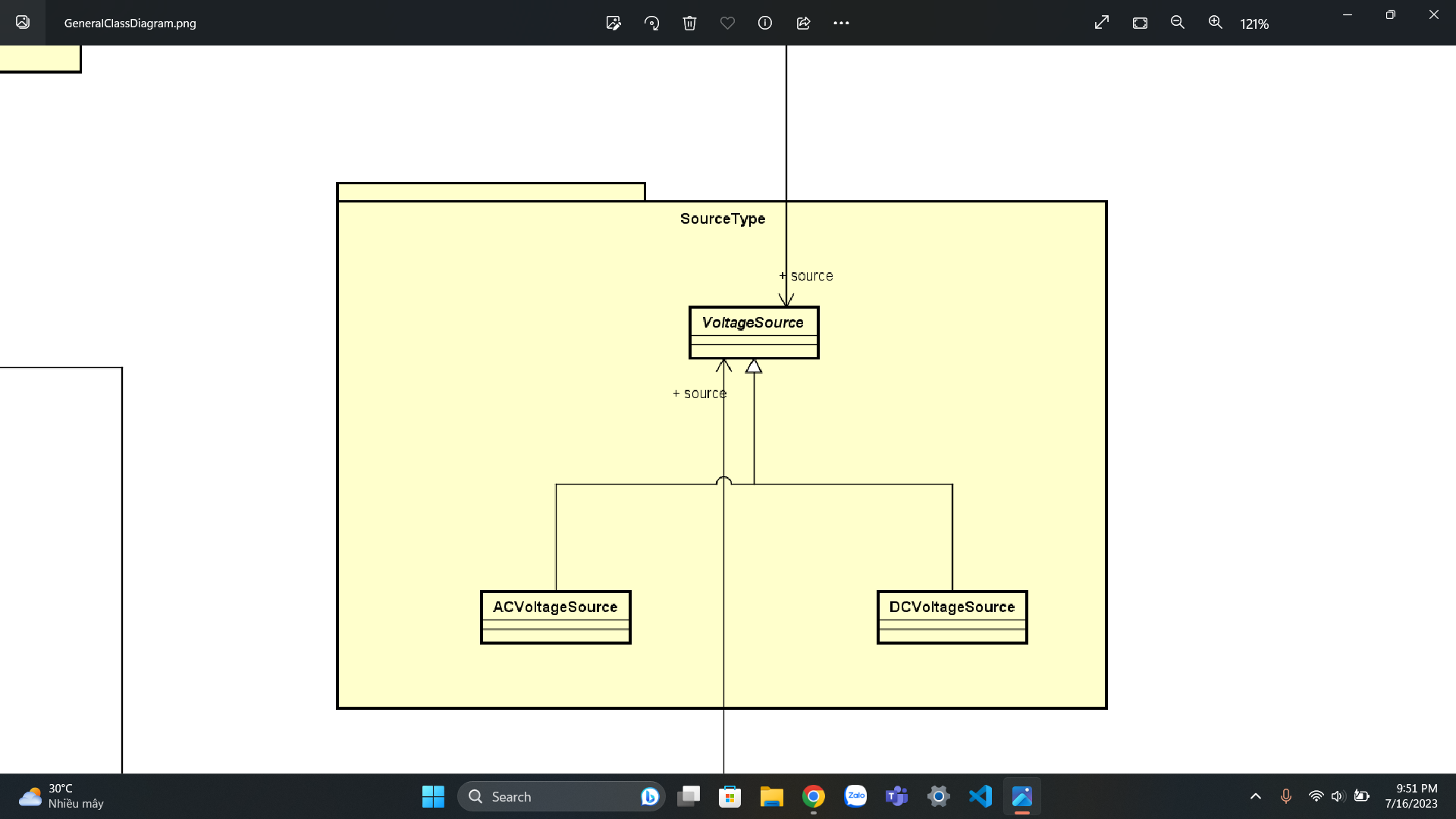
The three classes *Capacitor.java*, *Inductor.java*, *Resistor.java* extend the class *Elements.java.*

The classes *Capacitor.java* and *Inductor.java* also import the ComplexNumber.java so that their resistance values can be represented in complex number form.

**

- SourceType:

The two classes *ACVoltageSource.java* and *DCVoltageSource.java* extends the class *VoltageSource.java*

**

1. **Conclusion**

In this project, we have built an electrical circuit simulator using Java and JavaFX. The user interface is a graphical user interface that allows us to interact with the simulator. It allows users to create and simulate electrical circuits by adding components to the circuit, setting the values of the components, and analyzing the circuit.

The simulator is a valuable tool for learning about electrical circuits. It allows us to experiment with different circuits and see how they behave. The simulator can also be used to troubleshoot electrical problems such as short circuits.

In the future, we can extend the simulator to include more features. For example, we can add more electrical elements such as diodes, bulbs, … We can also improve the quality of values calculation or circuit diagram drawing.