Matlab Project

Simulating a Single Phase Transformer

Prepared by

Name: Shourov Joarder
Id: 190656

BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY

Department of Electrical and Electronic Engineering

Project Report on
"Simulating a Single Phase Transformer"

Course No: EEE 212

Course Title : Numerical Techniques Laboratory

Submitted to:

Dr. Hafiz Imtiaz	**Nurun Nahar Erina**
Professor , Department of EEE	PT, Department of EEE

Submitted by:

Shourov Joarder **Id: 1906156**

Date of Submission: 13/02/2022

Introduction:

The purpose of this Matlab project is to create an interface to simulate a single phase transformer. The graphical interface was built with **GUI** which is a feature of Matlab Graphical User Interface Development Environment or **GUIDE**. In this project the user will give the short circuit and the open circuit test data and the program will output the equivalent transformer circuit both referred to both primary and secondary sides. Also our program will calculate the voltage regulation, efficiency and different losses in a single phase transformer. Here is the user interface.

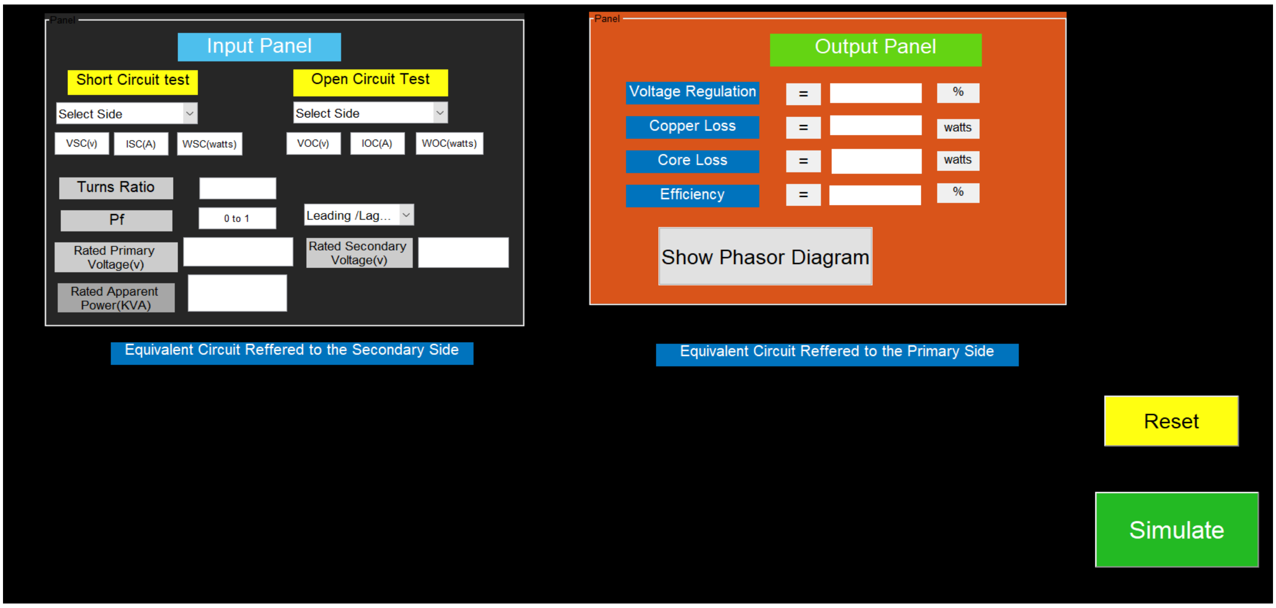


Fig: The User Interface

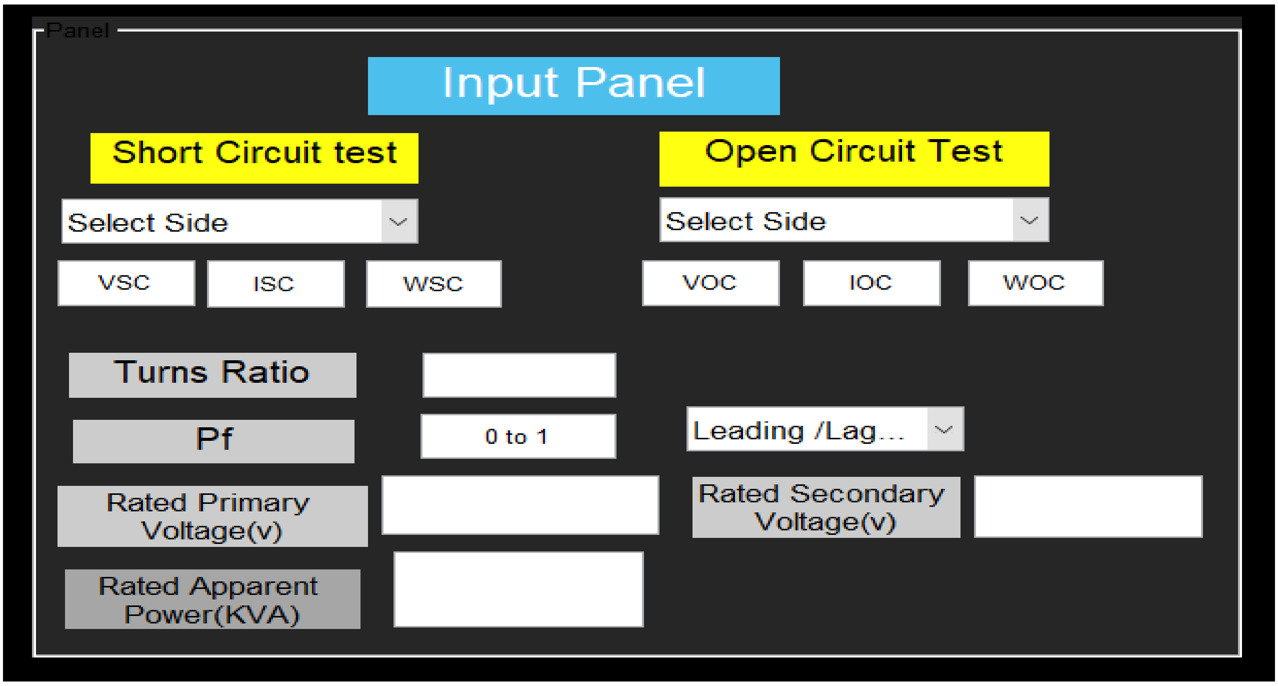
Project Description

Input Panel:

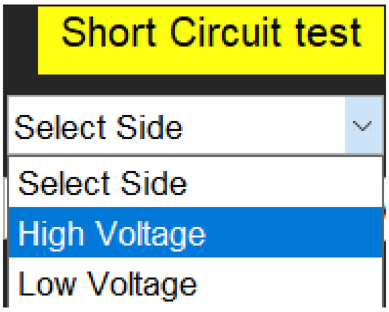
At first the user will interact with the input panel of the graphical interface. The input panel is at the top left of the corner of the main interface. Here there are two main parts…

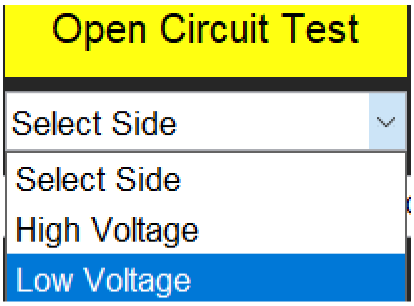
- Short Circuit Test
- Open Circuit Test

The input panel:

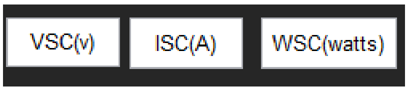


In both the short and open circuit test sides there are two options for the user whether the test was done on the High Voltage side or on the Low Voltage side. The user can select the side from the pop up menu.

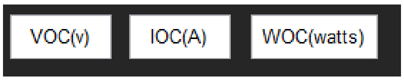




After selecting the referred side the user will give the short circuit and open circuit data as the input.



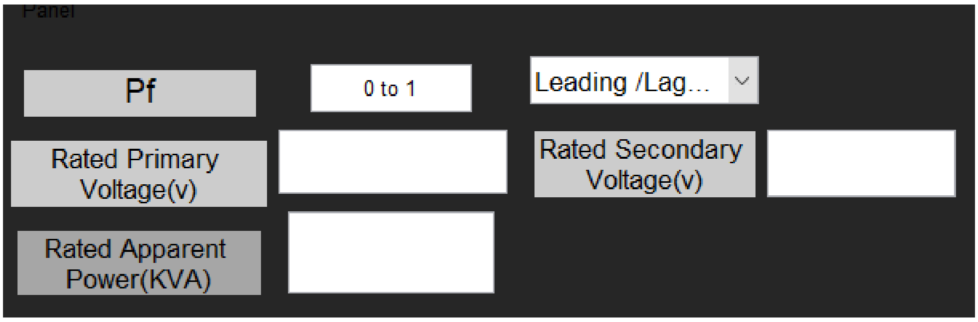
Short circuit test data



Open circuit test data

These data should be given in SI units , volts, amperes and watts.

Then the user will give the rest of the inputs to the system through the input panel. Such as primary and secondary voltages , rated power and pf



Main Code and Calculation Section:

The main calculation happens after the user clicks the 

button in the interface.

After clicking on the simulate button the program goes to the callback function of this button and calculates everything needed.

Simulate Button Callback function:

```
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
Vp=str2num(get(handles.rpv,'string'))%getting the rated primary voltage
global Vs
Vs=str2num(get(handles.rsv,'string'))%getting the rated secondary voltage
b=Vp/Vs %%turns ratio
% short circuit data
Wsc=str2num(get(handles.wsc,'string'));
Isc=str2num(get(handles.isc,'string'));
Vsc=str2num(get(handles.vsc,'string'));
% open circuit data
Voc=str2num(get(handles.voc,'string'));
Ioc=str2num(get(handles.ioc,'string'));
Woc=str2num(get(handles.woc,'string'));
%short circuit calcl
thetasc=acos(Wsc/(Vsc*Isc));
zsc=Vsc/Isc;
[Req,Xeq]=pol2cart(thetasc,zsc);
thetaoc=acos(Woc/(Voc*Ioc));
zoc=Voc/Ioc;
```

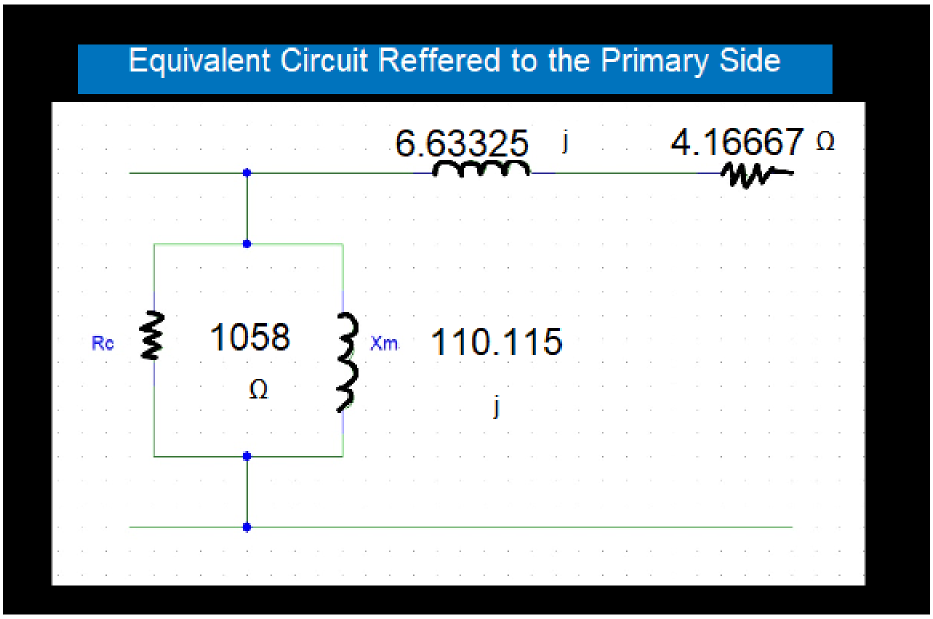
```
[R,X]=pol2cart(-thetaoc,1/zoc);
Rc=abs(1/R);
Xm=abs(1/X);
sideval_sc=get(handles.sc,'value');
switch sideval_sc
    case 2
        fprintf('Short ckt test was done in High voltage side')
        Reqs=Req/b^2
        Xeqs=Xeq/b^2
    case 3
        fprintf('Short ckt test was done in low voltage side')
        Reqs=Req
        Xeqs=Xeq
    otherwise
        fprintf('Select High or Low voltage')
end
sideval_oc=get(handles.oc,'value');
switch sideval_oc
    case 2
        fprintf('Short ckt test was done in High voltage side')
        Rcs=Rc/b^2
        Xms=Xm/b^2
    case 3
        fprintf('Short ckt test was done in low voltage side')
        Rcs=Rc
        Xms=Xm
    otherwise
        fprintf('Select High or Low voltage')
end
fprintf('Equivalent ckt reffered to the secondary side\n')
disp(['Reqs =',num2str(Reqs)])
disp(['Xeqs =',num2str(Xeqs)])
disp(['Rcs =',num2str(Rcs)])
disp(['Xms =',num2str(Xms)])
%Equivalent ckt reffered to the secondary side
fprintf('Equivalent ckt reffered to the secondary side\n')
Reqp=Reqs*b^2
Xeqp=Xeqs*b^2
Rcp=Rcs*b^2
Xmp=Xms*b^2
set(handles.reqs,'string',Reqs)
set(handles.xeqs,'string',Xeqs)
set(handles.rcs,'string',Rcs)
set(handles.xms,'string',Xms)
set(handles.reqp,'string',Reqp)
set(handles.xeqp,'string',Xeqp)
set(handles.rcp,'string',Rcp)
set(handles.xmp,'string',Xmp)
global Vps
%Calculation for pf
pf=str2num(get(handles.pf,'string')) %getiing the pf value
S=1000*str2num(get(handles.rap,'string')) %getting the rated apparent power
% Vp=str2num(get(handles.rpv,'string'))%getting the rated primary voltage
% global Vs
```

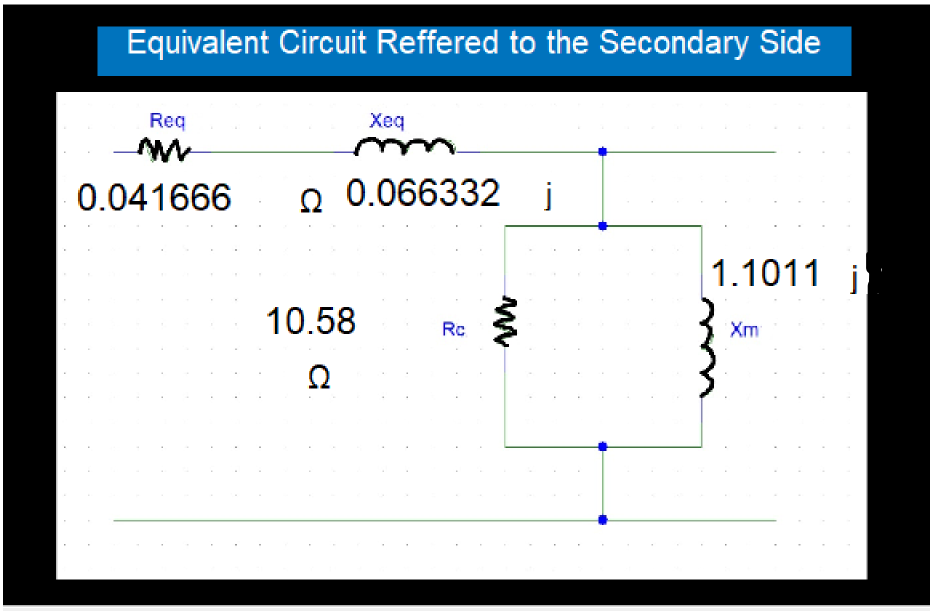
```
% Vs=str2num(get(handles.rsv,'string'))%getting the rated secondary voltage
flag=get(handles.L,'value') %for identefying lagging or leading
switch flag
    case 2
        fprintf('Lagging')
    case 3
        fprintf('Leading')
    otherwise
        fprintf('Select lagging or leading')
end
%load pf angle
global pf_angle
if(flag==2) %lagging
    pf_angle=acos(pf)
else
    pf_angle=-acos(pf)
end
% Always assume secondary er voltage er angle 0
global Is
Is=((S)/Vs)*exp(-j*pf_angle) %if flag==2 taile lagging and current er angle
neg
% determining Vps at this pf
Vps=Vs+Reqs*Is+Is*j*Xeqs
VR=((abs(Vps)-Vs)/Vs)*100
Copper_loss=(abs(Is))^2*Reqs
Core_loss=((abs(Vps))^2)/Rcs
Pw=S*cos(pf_angle)
Efficiency=(Pw)/(Copper_loss+Core_loss+Pw)*100
set(handles.vr,'string',VR)
set(handles.cu,'string',Copper_loss)
set(handles.core,'string',Core_loss)
set(handles.eta,'string',Efficiency)
set(handles.uipanel7,'visible','off')
axes(handles.axes1);
imshow('s1.jpg')
axes(handles.axes2);
imshow('p1.jpg')
```

This is the main code section where the whole calculation happens.

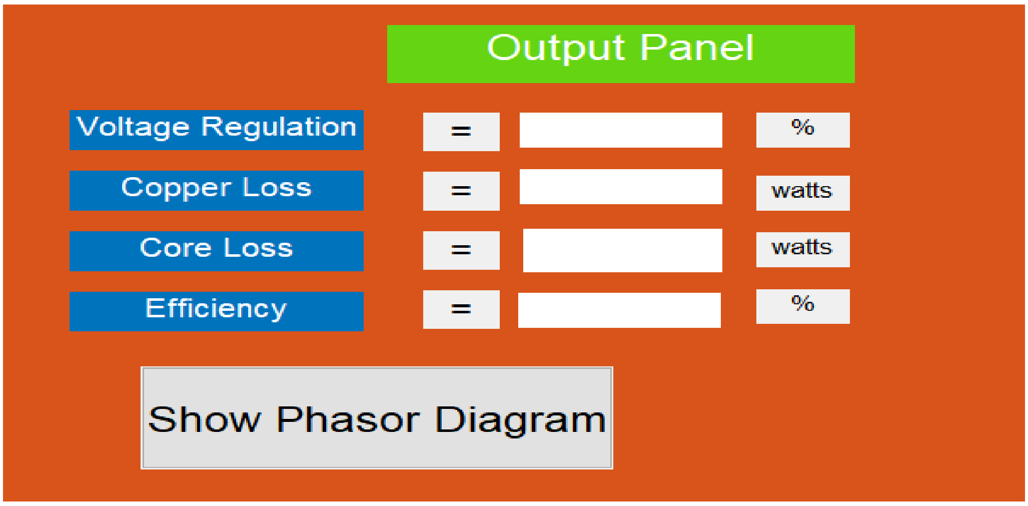
Output Panel:

At first the program will show the equivalent circuits referred to both the primary and secondary sides.





The output panel at the right of the screen shows the voltage regulation, efficiency and different losses.



In this output panel there is a button Show Phasor Diagram .If the uses clicks on this button, a figure will pop up which will show the phasor diagram of the Vs,Is and Vsp.

Show Phasor Diagram <u>Button Callback function:</u>

```
%% Phasor Diagram

function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Vs Is pf_angle Vps
figure
phasediagram(abs(Vs),0,'r','Vs')
phasediagram(abs(Vps),angle(Vps),'g','Vps')
phasediagram(abs(Is),0-pf_angle,'b','Is')
```

<u>Phase Diagram Dunction:</u>

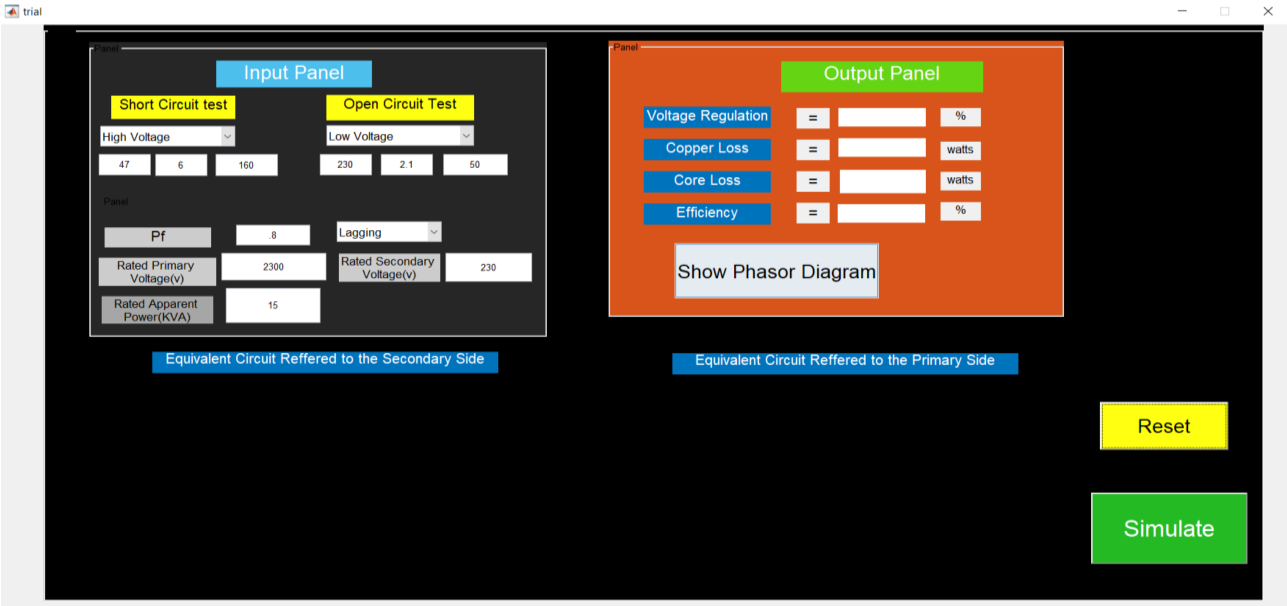
```
function []=phasediagram(mag,angl,s,s1)
disp('Hi')
x=linspace(0,mag,1000);
y=angl*ones(1,length(x));
title('Phase Diagram reffered to the Secondary Side')
polarplot(y,x,s,'DisplayName',s1,'LineWidth',3,'MarkerSize',10)
legend
hold on
rlim('auto')
thetalim([-90 90])
end
```

Reset Button:

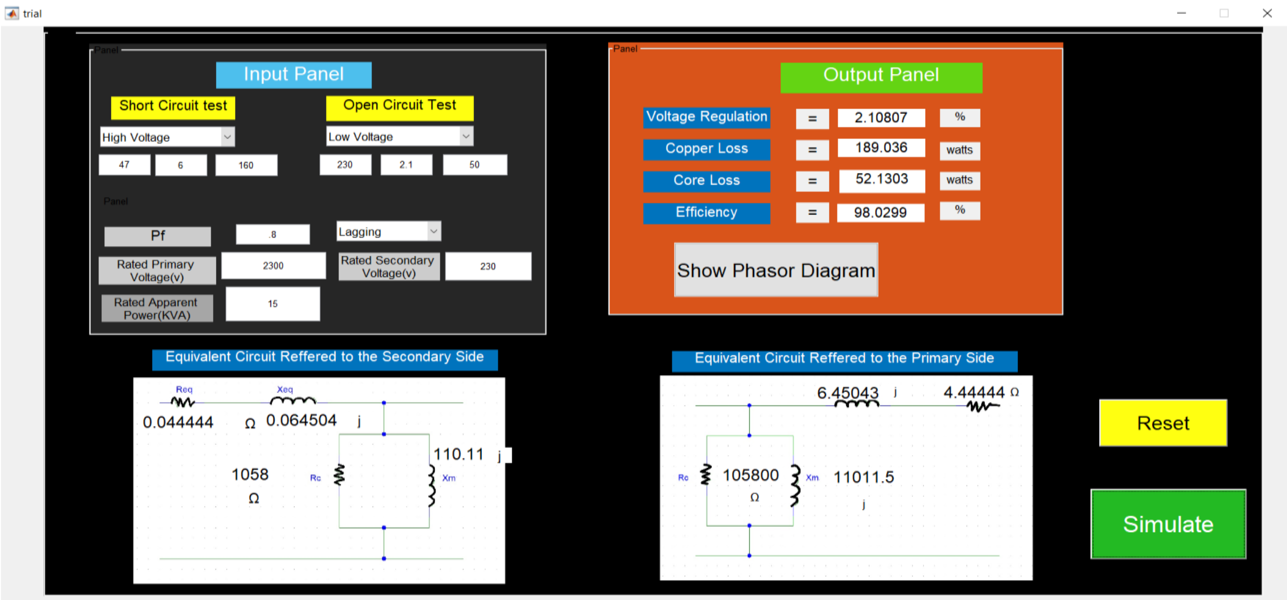
After clicking on the reset button the interface gets back to the normal mode and the user can give another input in the interface.

Demonstrating a test case

Taking the imput:

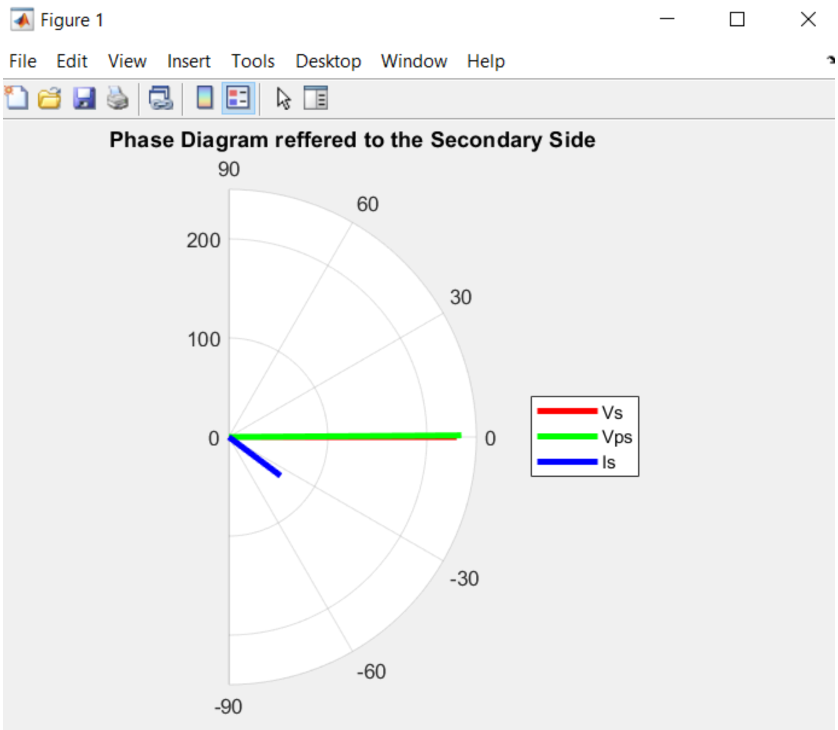


After clicking on the  button:

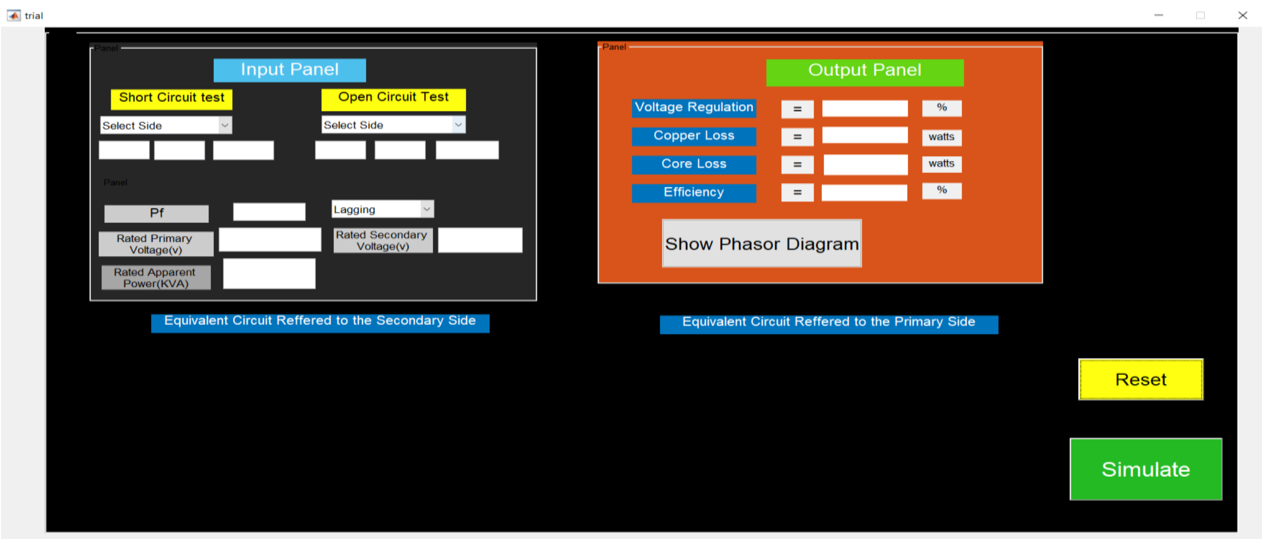


The output panel shows all the outputs that were asked to compute.

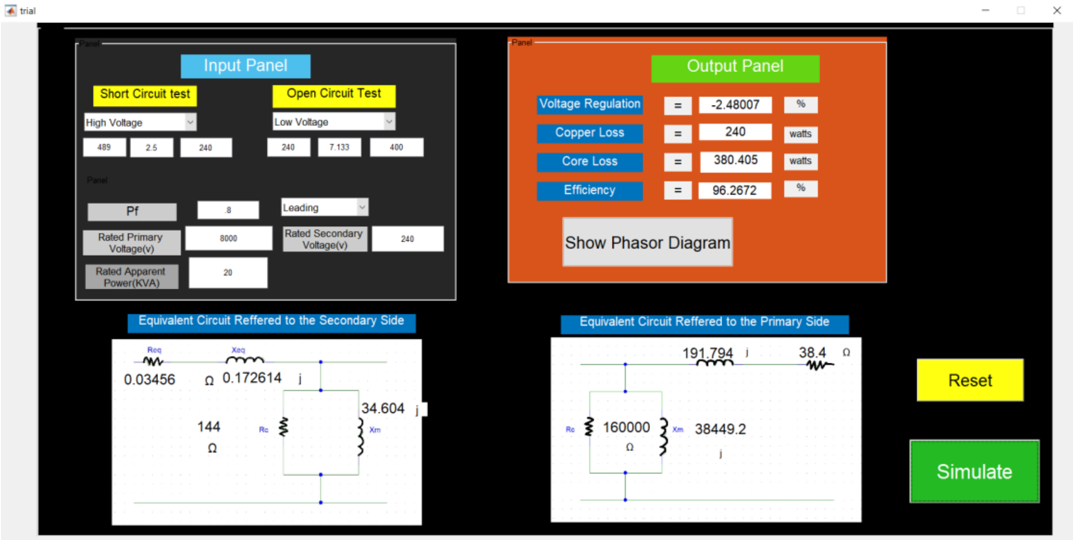
After clicking on the Show Phasor Diagram button the program will open a new figure and show the phasor diagram.



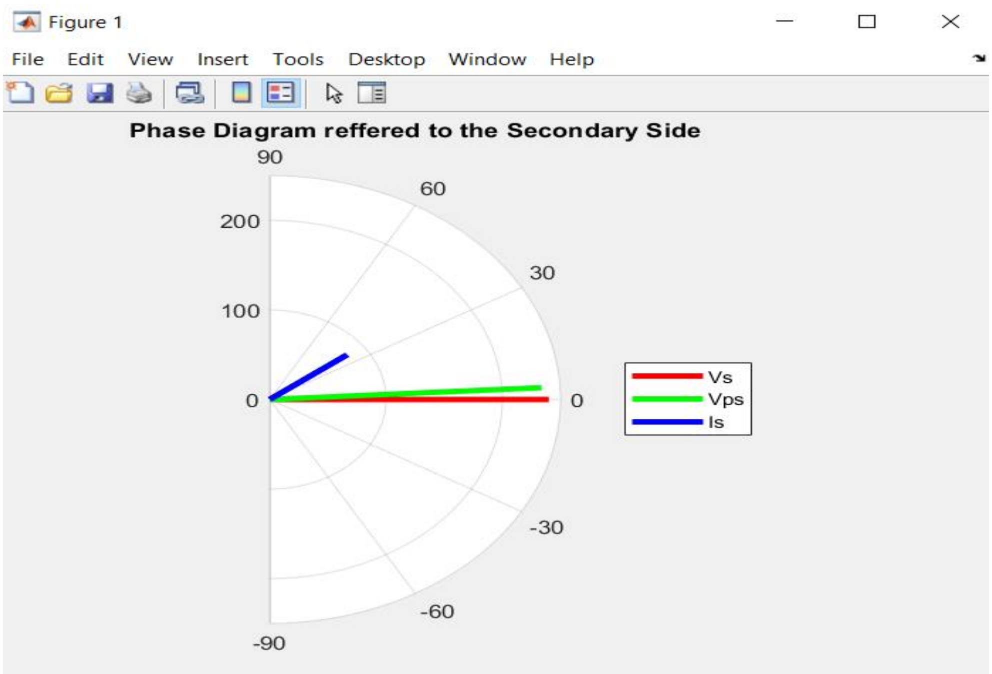
After clicking on the Reset button the program will reset automatically and the user will be able to give new inputs.



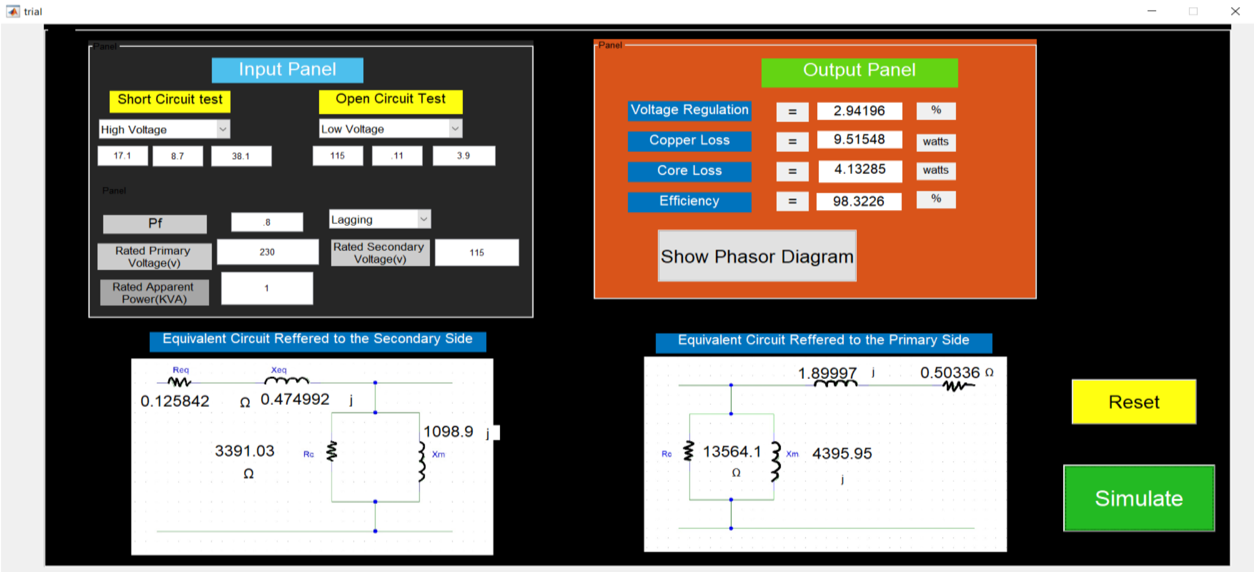
Test case-2:



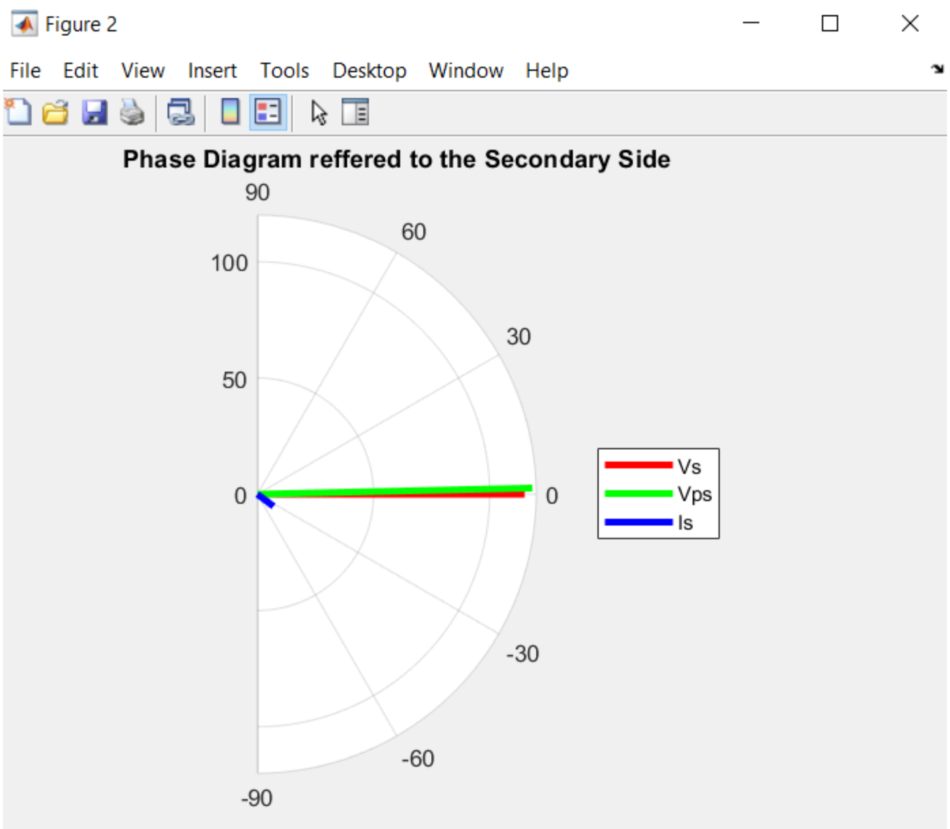
Phasor diagram:



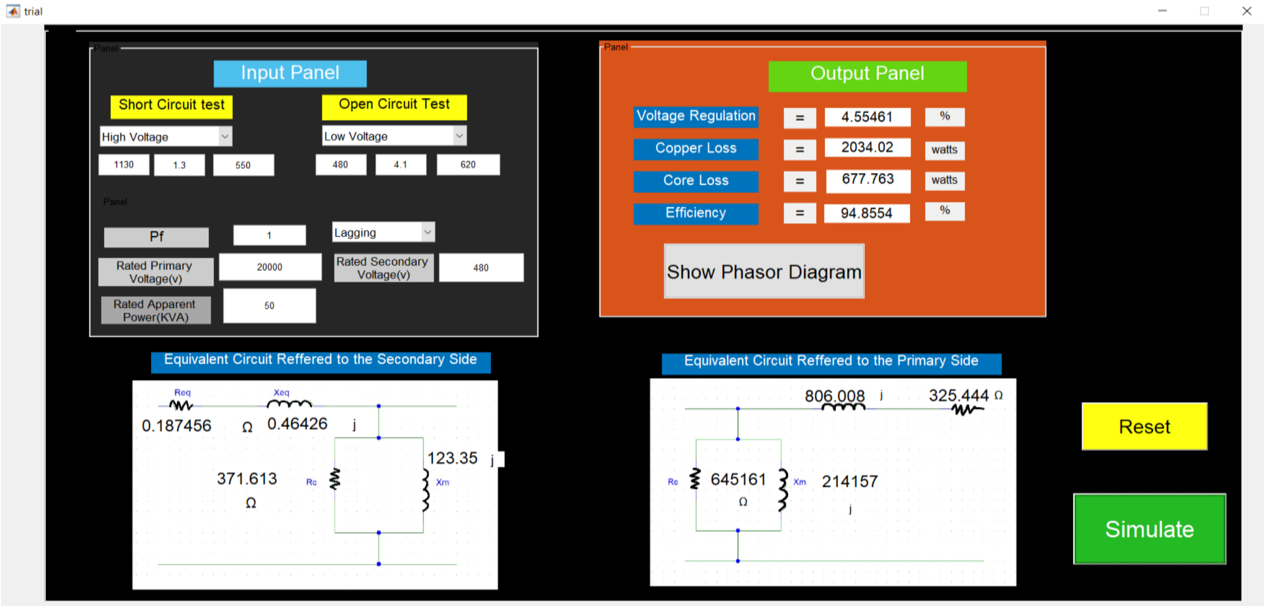
Test case-3:



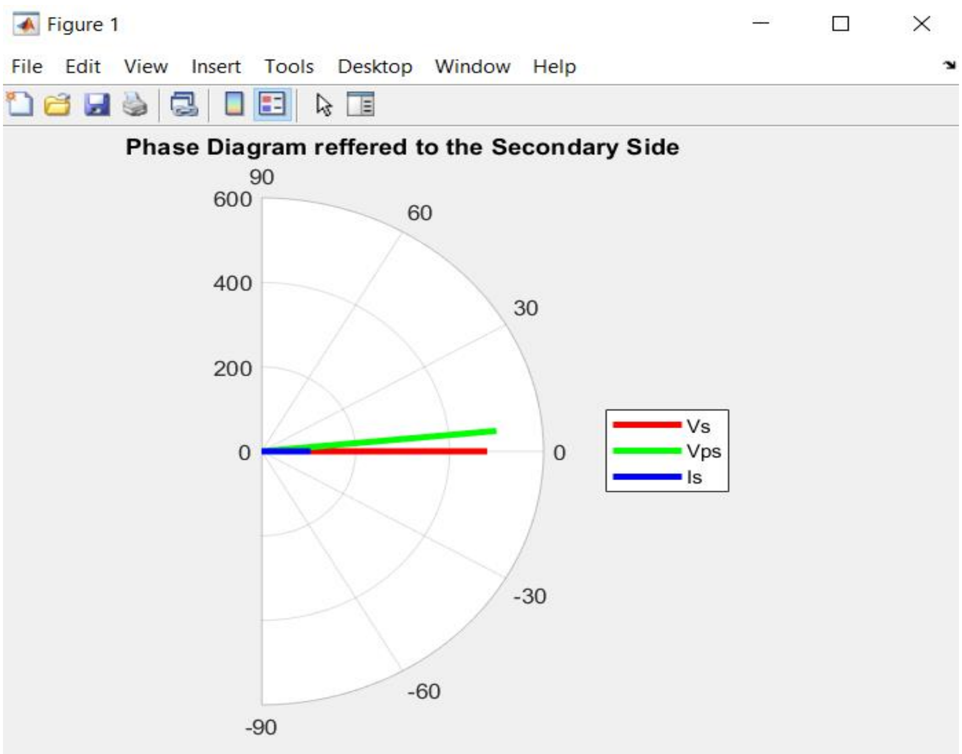
Phasor diagram:



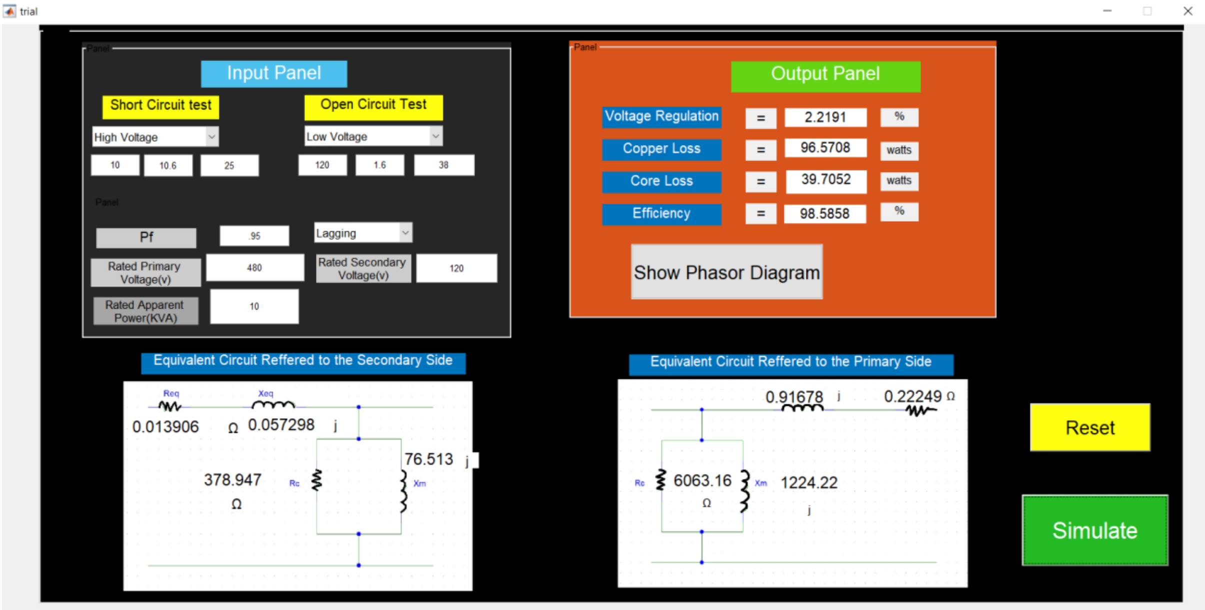
Test case-4:



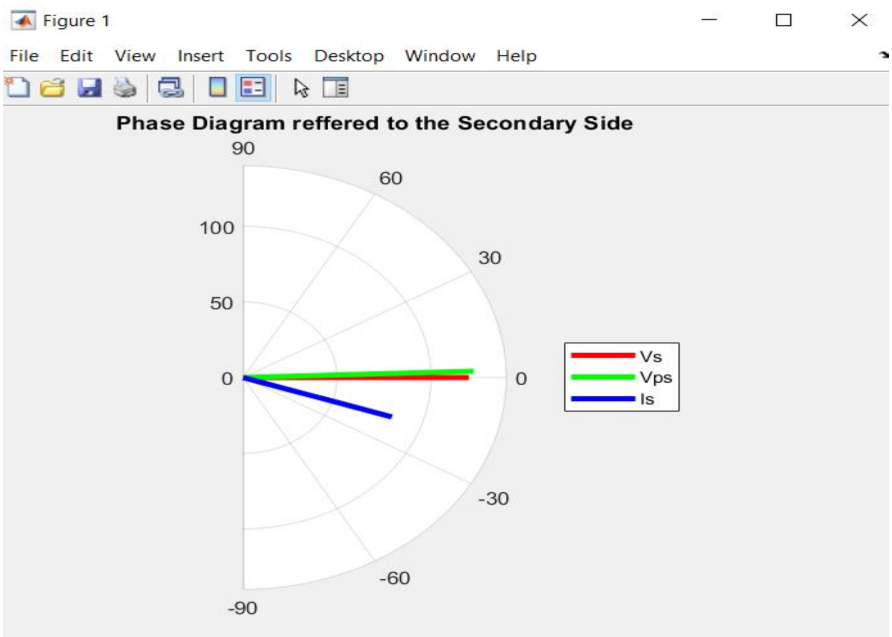
Phasor diagram:



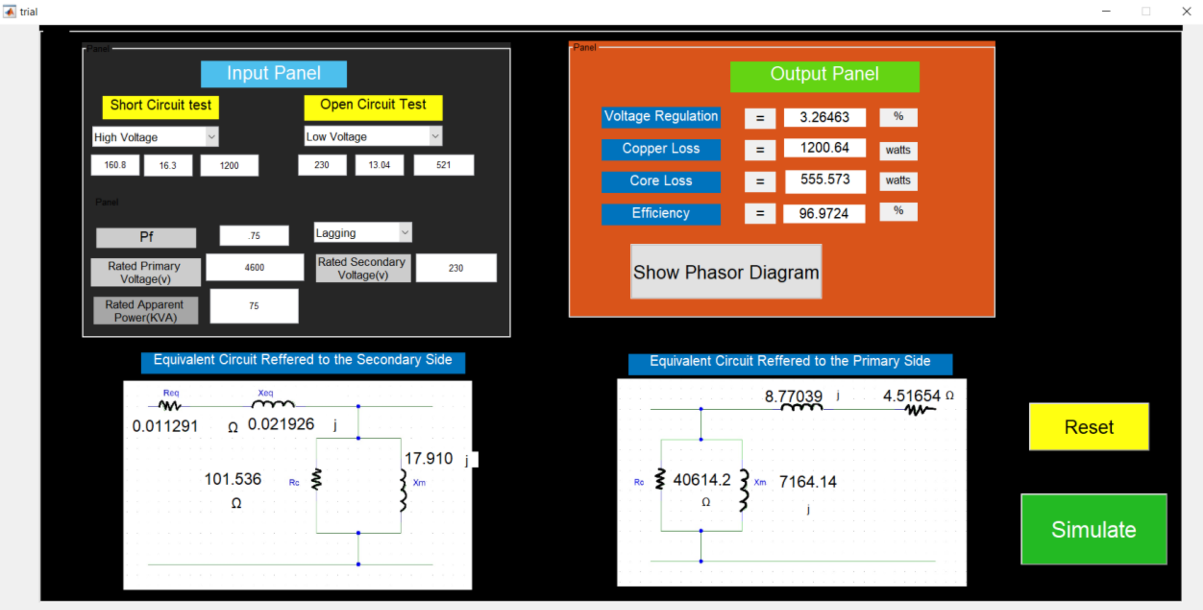
Test case-5:



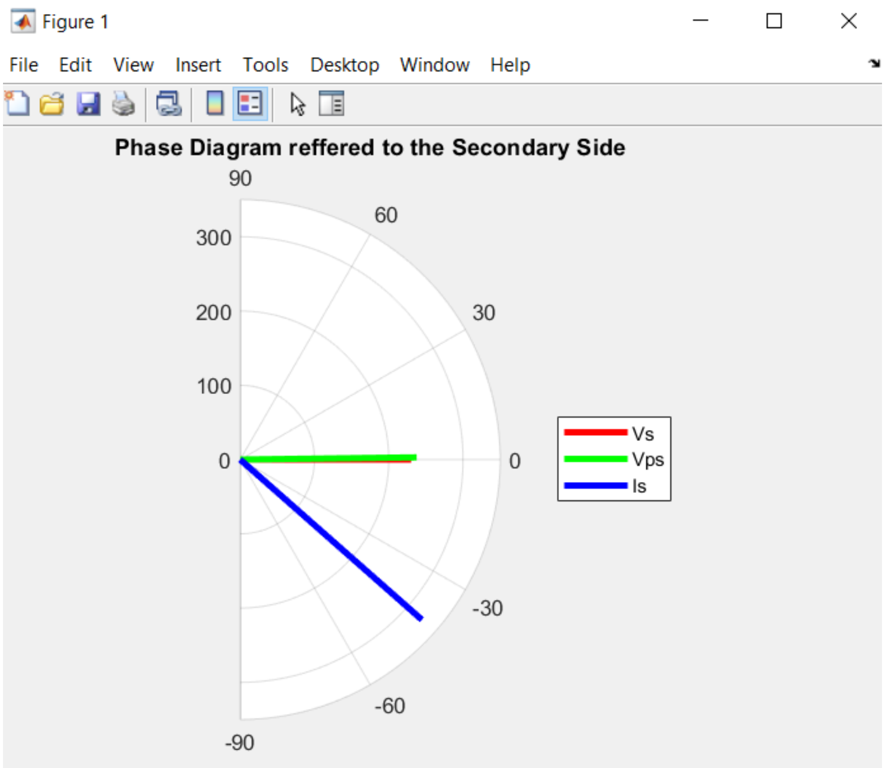
Phasor diagram:



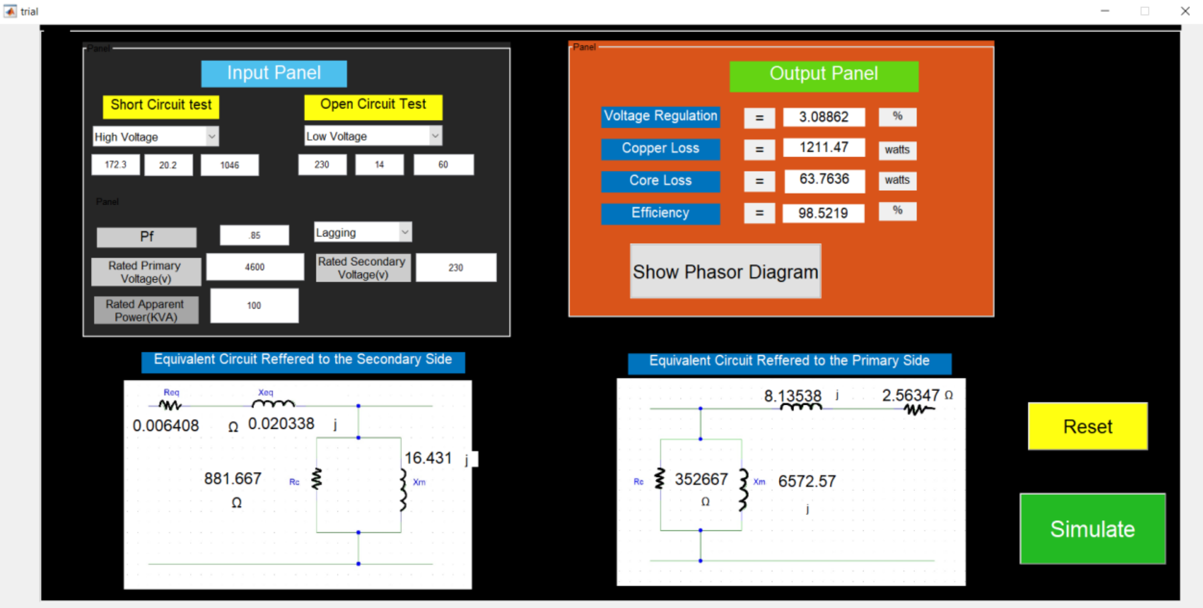
Test case-6:



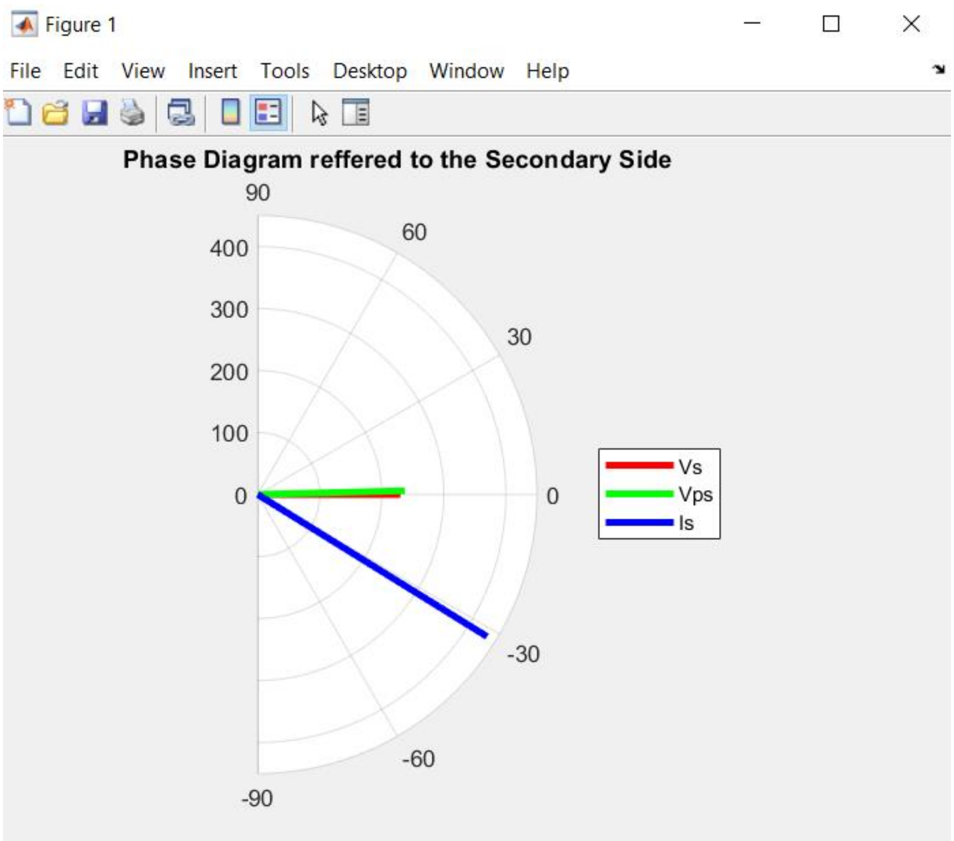
Phasor diagram:



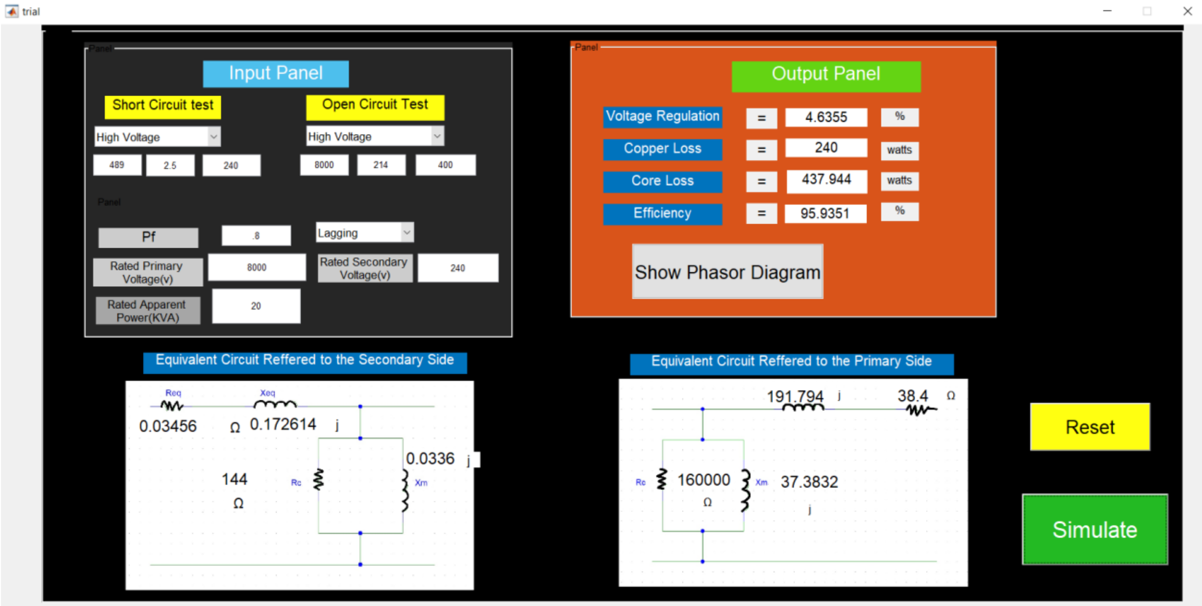
Test case-7:



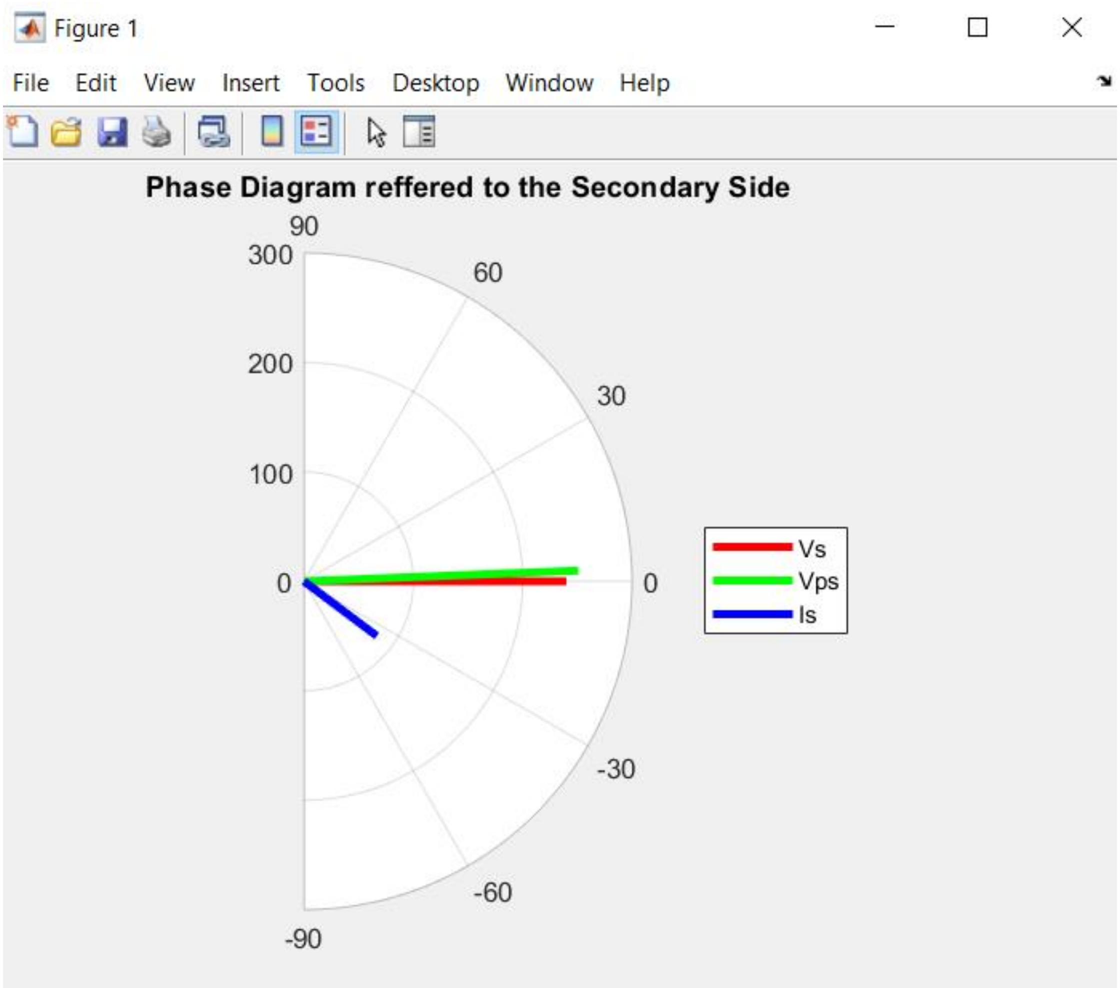
Phasor diagram:



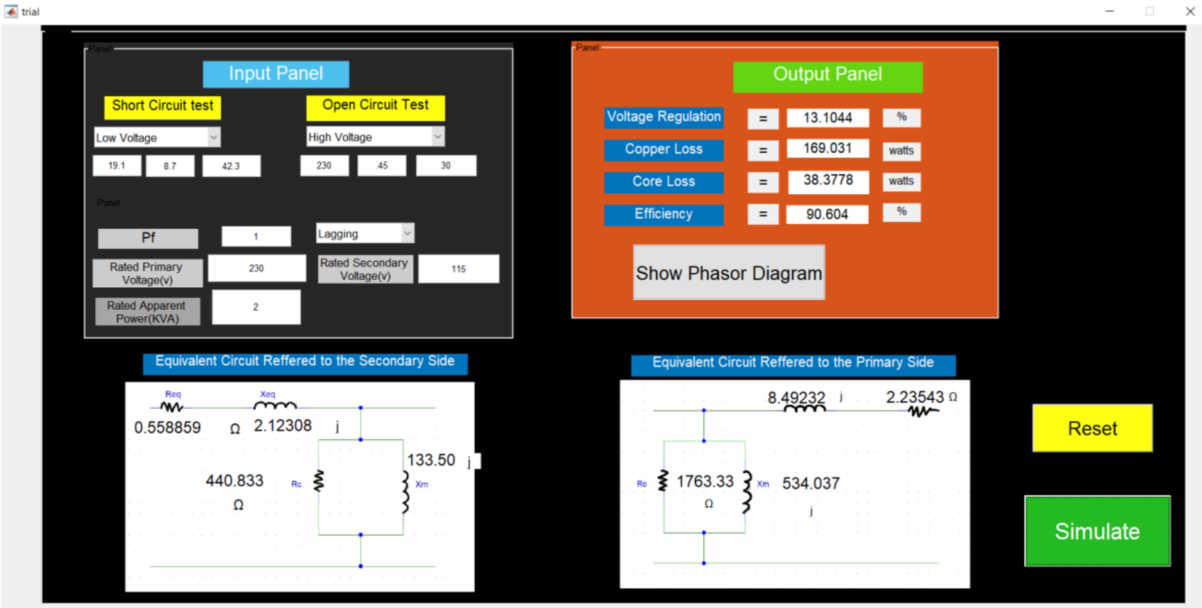
Test case-8:



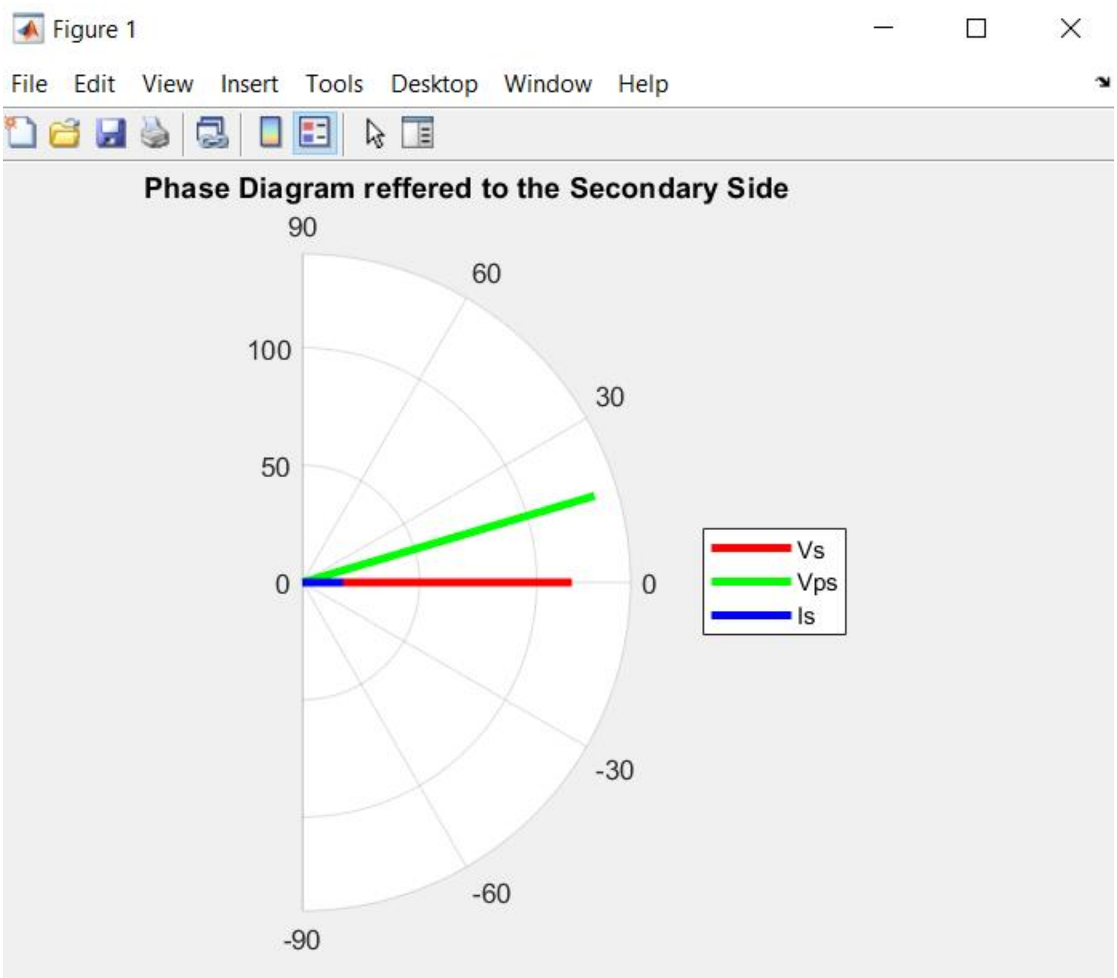
Phasor diagram:



Test case-9:



Phasor diagram:



Conclusion:

The main purpose of this project was to build a presentable graphical user interface that will simulate a single phase transformer from the given inputs. We have already showed how our project works and how the interface gives the corresponding outputs. But surely this graphical interface could have been much more improved. It is a continuous developing process, given enough time we can improve the user interface and make it more user interactive and user friendly.