Object Oriented Programming 2 Lab Final Project Report
Alihan BOZKIR 151220182032

Uml Diagram:

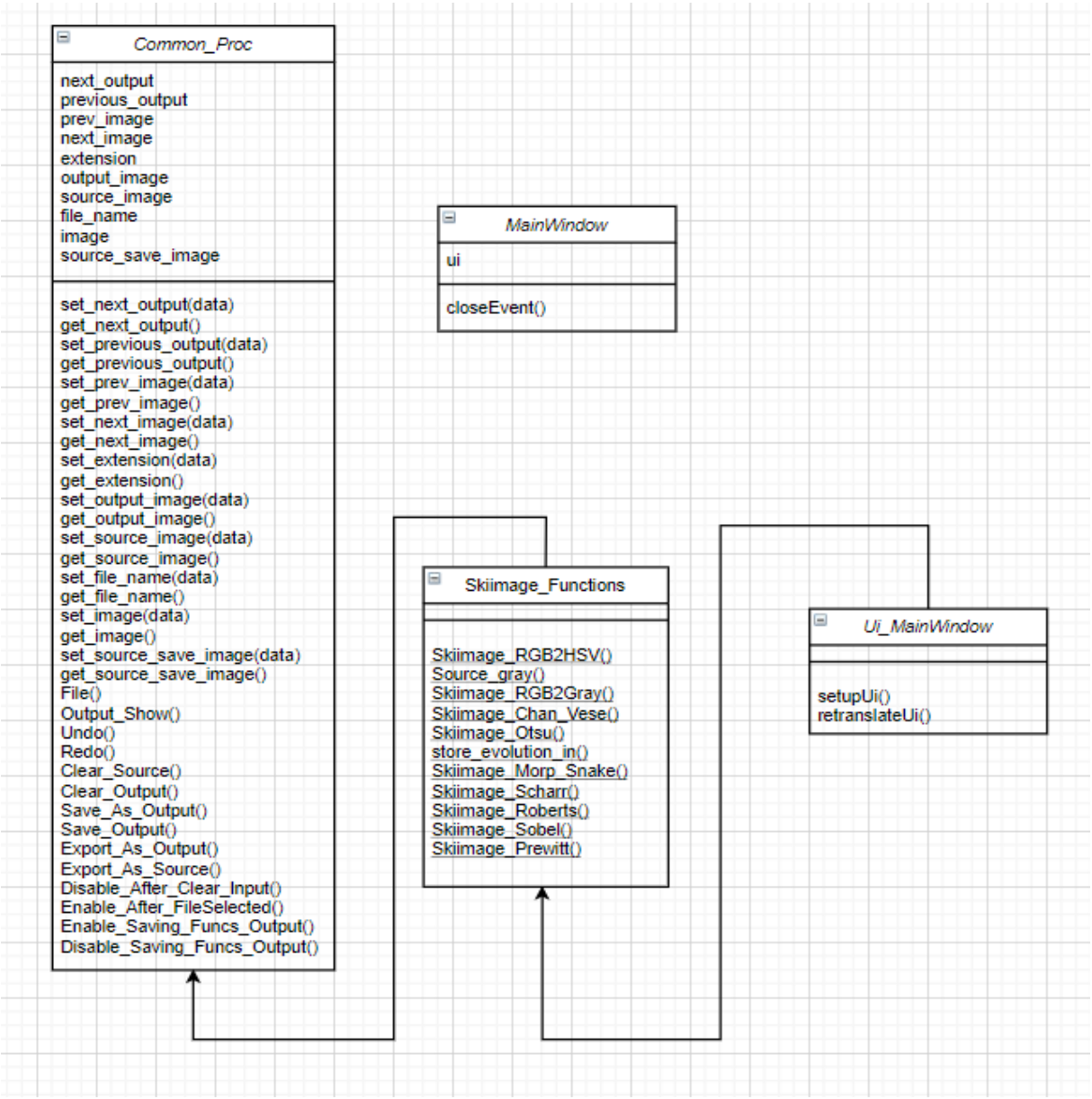

Figure 1 UML Diagram of the Lab Final

MainWindow Class:

Our MainWindow class acts as a runner. QMainWindow is the parent class of our MainWindow class. Our program activates the GUI of our software from the moment this class is run and brings it to the screen of our computer.

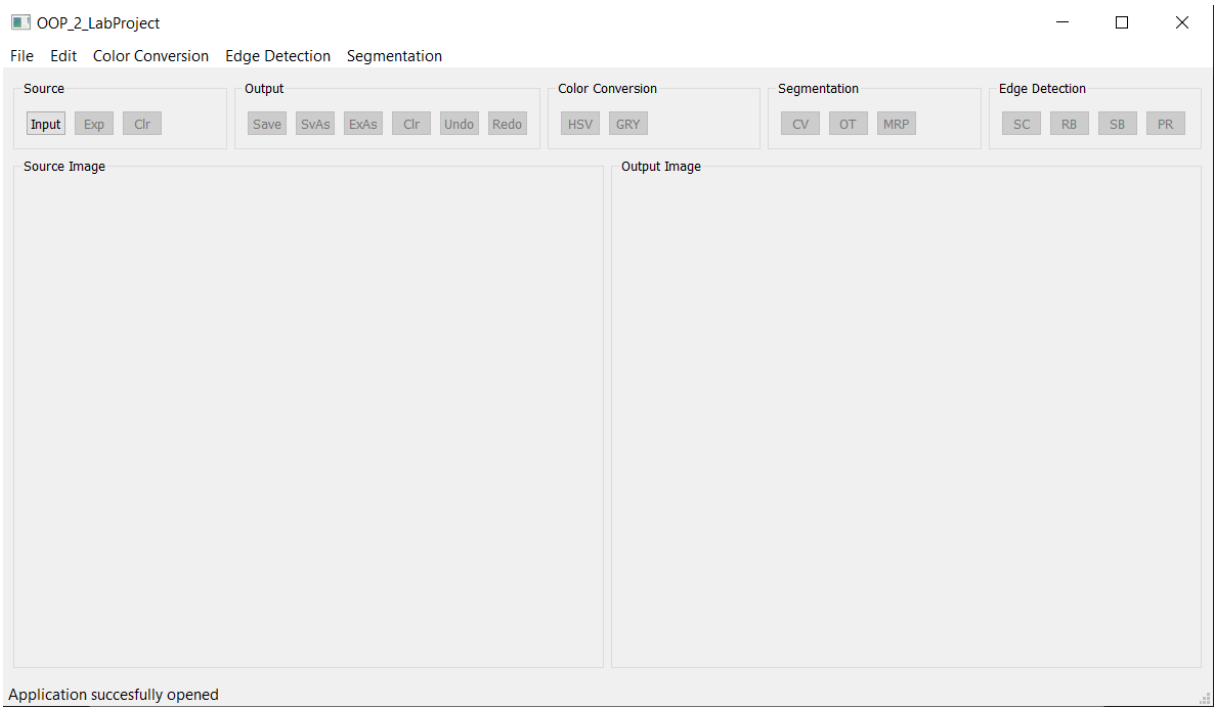

Figure 2 GIU of our software

Common_Proc Class:

Common_Proc class contains all the attributes required for our program and their set and get methods. Although our Class is a direct parent class of Skiimage_Functions class in the program, its main purpose is to contain every method that is done jointly, regardless of the work done in the program. Since our Common_Proc Class has an indirect parent class to the Ui_MainWindow class, we can manipulate the signal and slot functions from the Common_Proc Class to ensure changes in the interface when any operation is made.

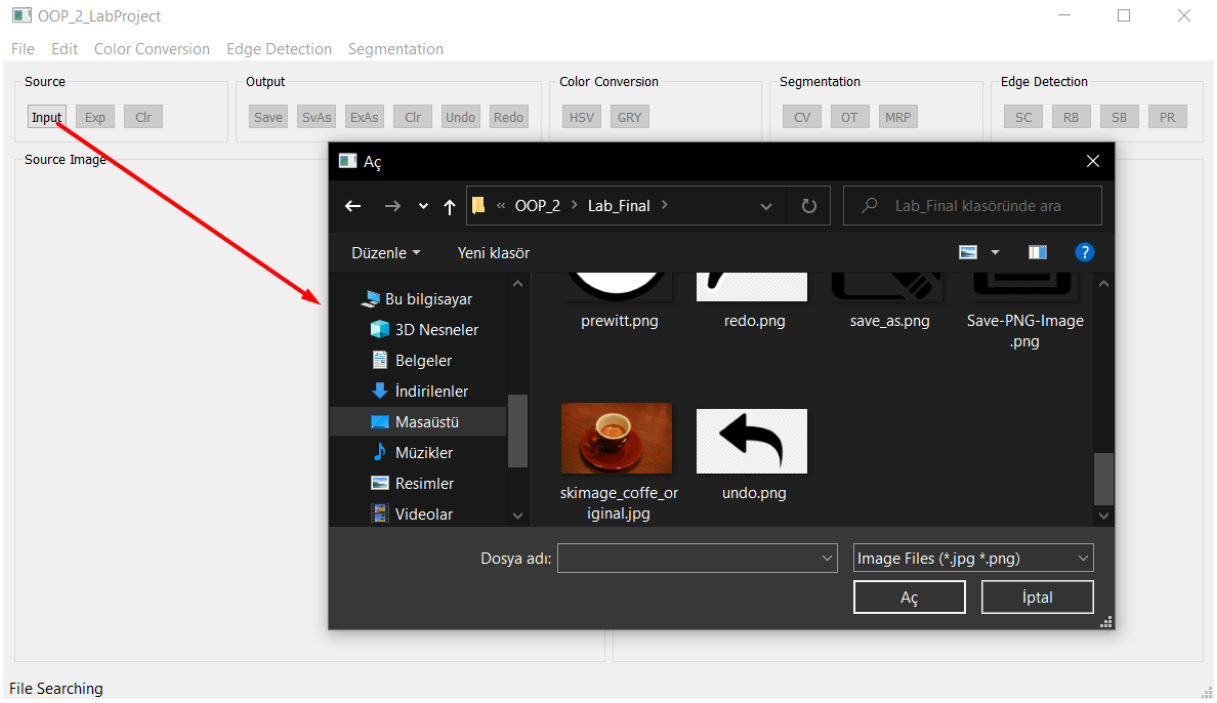

Figure 3 Opening data selector window

Clustering Class:

While our Skiimage_Functions class is the parent class of the Ui_MainWindow class, the parent class of our Skiimage_Functions class is the Common_Proc class. After obtaining the necessary attributes with the get functions from the Common_Proc Class of our Class, after interacting with the necessary buttons and actions in our interface, we applied the methods in 3 different processing windows to our image data [Color Conversion(RGB to HSV or RGB to GrayScale), Segmentation(Chan_veese, Otsu, Morpghological Snakes) and Edge Detection(Scharr, Roberts, Sobel, Prewitt)] our image is manipulated. After the operations are finished, the data is sent to the Common_Proc Class to be printed on the interface after it is updated with the set functions.

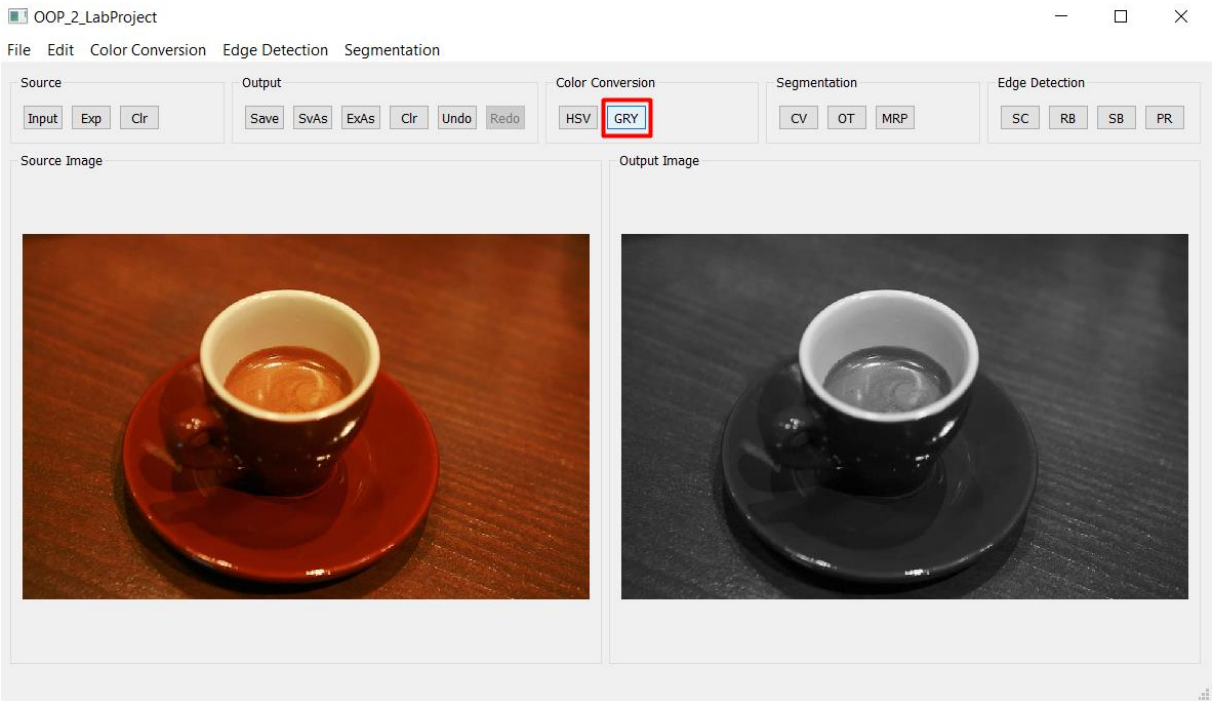

Figure 4 An example of Color Conversion (RGB to GrayScale) method

Ui_MainWindow Class:

Our Ui_MainWindow class contains the software of our interface. Skiimage_Functions class is the parent classes of our Ui_MainWindow class. Our Class is obtained by converting our interface from the .ui extension to .py extension after our interface is made with the Qt Designer program. By using the lambda: structure for the necessary signal and slot operations for our Class obtained in this way, with the structures on the screen (Buttons, Actions, etc.). When interacting, it is ensured what to do in the next step, and changes to our interface by interacting with our necessary Classes.