Praktikum II
(Pengolahan Citra Digital)

(A3 – A7)

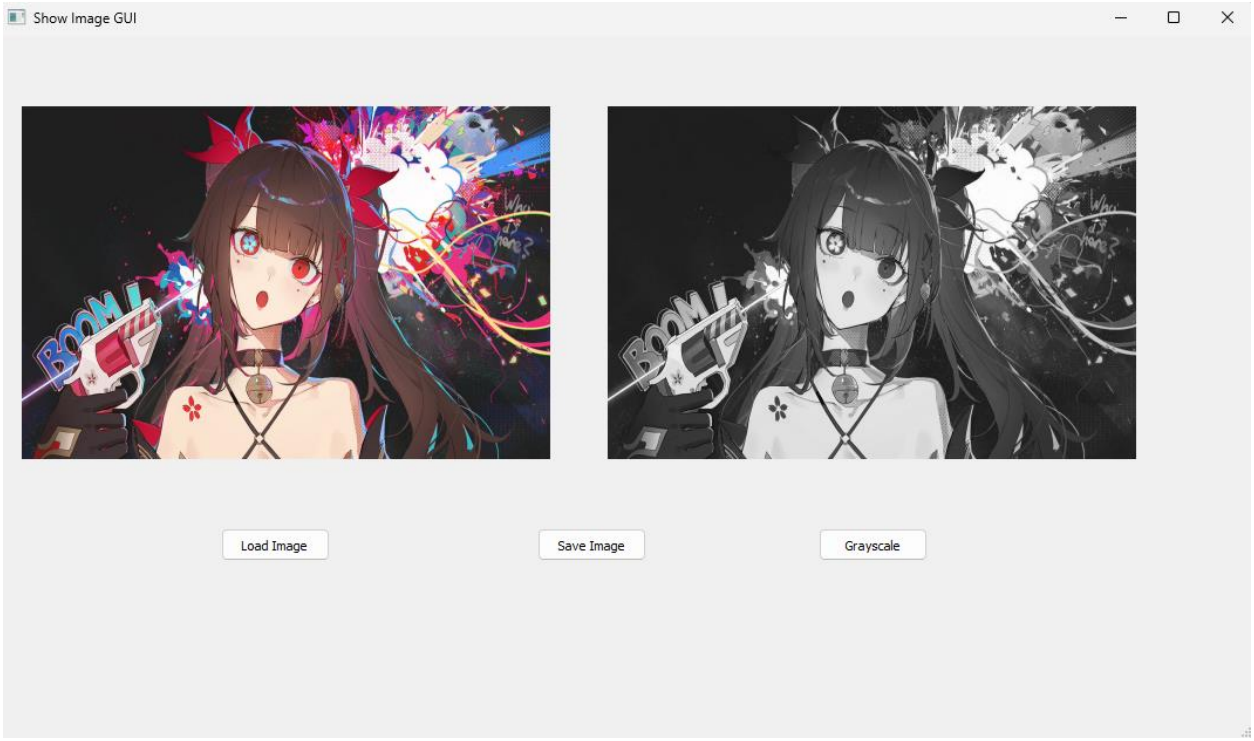
A3

```python
import cv2
import sys
import numpy as np
from PyQt5 import QtCore, QtWidgets
from PyQt5.QtCore import pyqtSlot, Qt
from PyQt5.QtGui import QImage, QPixmap
from PyQt5.QtWidgets import QMainWindow, QMessageBox
from PyQt5.uic import loadUi

2 usages
class ShowImage(QMainWindow):
    def __init__(self):
        super(ShowImage, self).__init__()
        loadUi( uifile: 'showgui.ui', self)
        self.image = None
        self.loadButton.clicked.connect(self.loadClicked)
        self.grayButton.clicked.connect(self.grayClicked)

    1 usage
    @pyqtSlot()
    def loadClicked(self):
        self.loadImage('BANG.jpeg')
```

```python
    1 usage
    def loadImage(self, flname):
        self.image = cv2.imread(flname)
        self.displayImage()

    2 usages
    def displayImage(self):
        qformat = QImage.Format_Indexed8

        if len(self.image.shape) == 3:  # row[0], col[1], channel[2]
            if self.image.shape[2] == 4:
                qformat = QImage.Format_RGBA8888
            else:
                qformat = QImage.Format_RGB888

        img = QImage(self.image.data, self.image.shape[1], self.image.shape[0], self.image.strides[0], qformat)
        img = img.rgbSwapped()

        self.imgLabel.setPixmap(QPixmap.fromImage(img))
        self.imgLabel.setAlignment(Qt.AlignHCenter | Qt.AlignVCenter)
```

```
    2 usages
    def displayImage(self, window=1):
        qformat = QImage.Format_Indexed8
        if len(self.image.shape) == 3:
            if self.image.shape[2] == 4:
                qformat = QImage.Format_RGBA8888
            else:
                qformat = QImage.Format_RGB888
        img = QImage(self.image, self.image.shape[1], self.image.shape[0], self.image.strides[0], qformat)
        img = img.rgbSwapped()

        if window == 1:
            self.imgLabel.setPixmap(QPixmap.fromImage(img))
            self.imgLabel.setAlignment(QtCore.Qt.AlignHCenter | QtCore.Qt.AlignVCenter)
            self.imgLabel.setScaledContents(True)

        if window == 2:
            self.hasilLabel.setPixmap(QPixmap.fromImage(img))
            self.hasilLabel.setAlignment(QtCore.Qt.AlignHCenter | QtCore.Qt.AlignVCenter)
            self.hasilLabel.setScaledContents(True)

```

```
    1 usage
    @pyqtSlot()
    def grayClicked(self):
        try:
            if self.image is not None:
                H, W = self.image.shape[:2]
                gray = np.zeros( shape: (H, W), np.uint8)
                for i in range(H):
                    for j in range(W):
                        gray[i, j] = np.clip(
                            0.299 * self.image[i, j, 0] + 0.587 * self.image[i, j, 1] + 0.114 * self.image[i, j, 2],  a_min: 0,
                            a_max: 255)
                self.image = gray
                self.displayImage(2)
        except Exception as e:
            QMessageBox.critical(self, "Error", str(e))



app = QtWidgets.QApplication(sys.argv)
window = ShowImage()
window.setWindowTitle('Show Image GUI')
window.show()
sys.exit(app.exec_())
```



A4

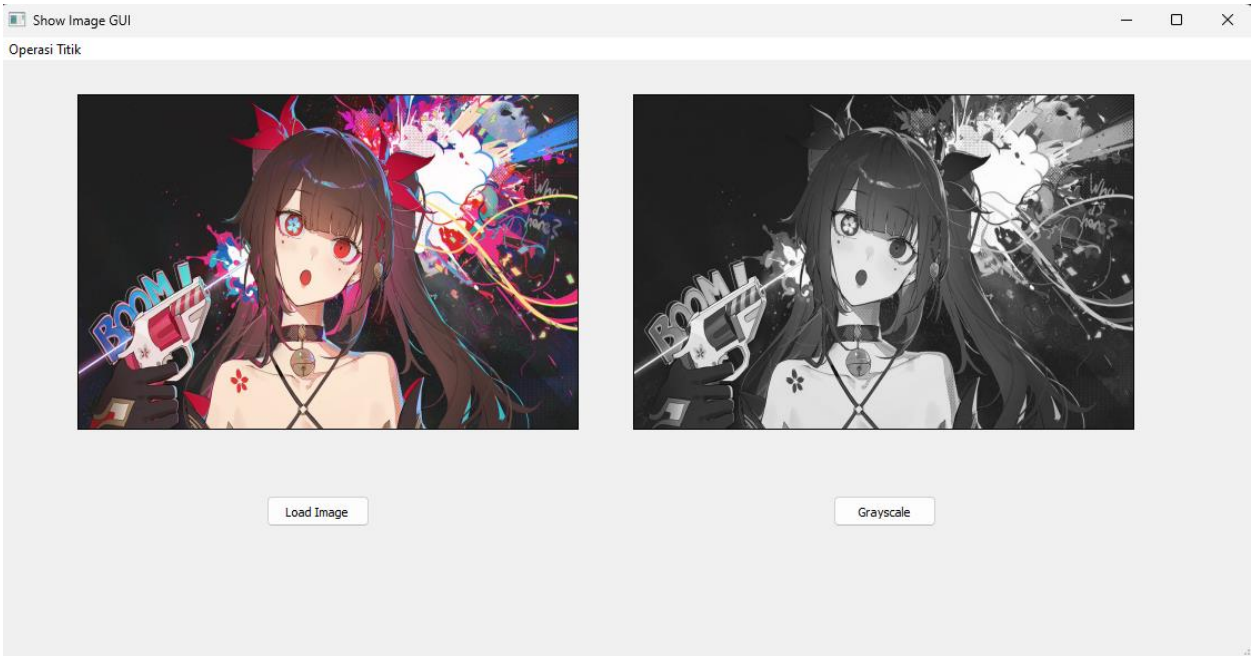
```python
import cv2
import sys
import numpy as np
from PyQt5 import QtCore, QtWidgets
from PyQt5.QtCore import pyqtSlot, Qt
from PyQt5.QtGui import QImage, QPixmap
from PyQt5.QtWidgets import QMainWindow, QMessageBox
from PyQt5.uic import loadUi

2 usages
class ShowImage(QMainWindow):
    def __init__(self):
        super(ShowImage, self).__init__()
        loadUi( uifile: 'showgui.ui', self)
        self.image = None
        self.loadButton.clicked.connect(self.loadClicked)
        self.grayButton.clicked.connect(self.grayClicked)
        self.actionOperasi_Pencerahan.triggered.connect(self.brightness)

    1 usage
    @pyqtSlot()
    def loadClicked(self):
        self.loadImage('BANG.jpeg')
```

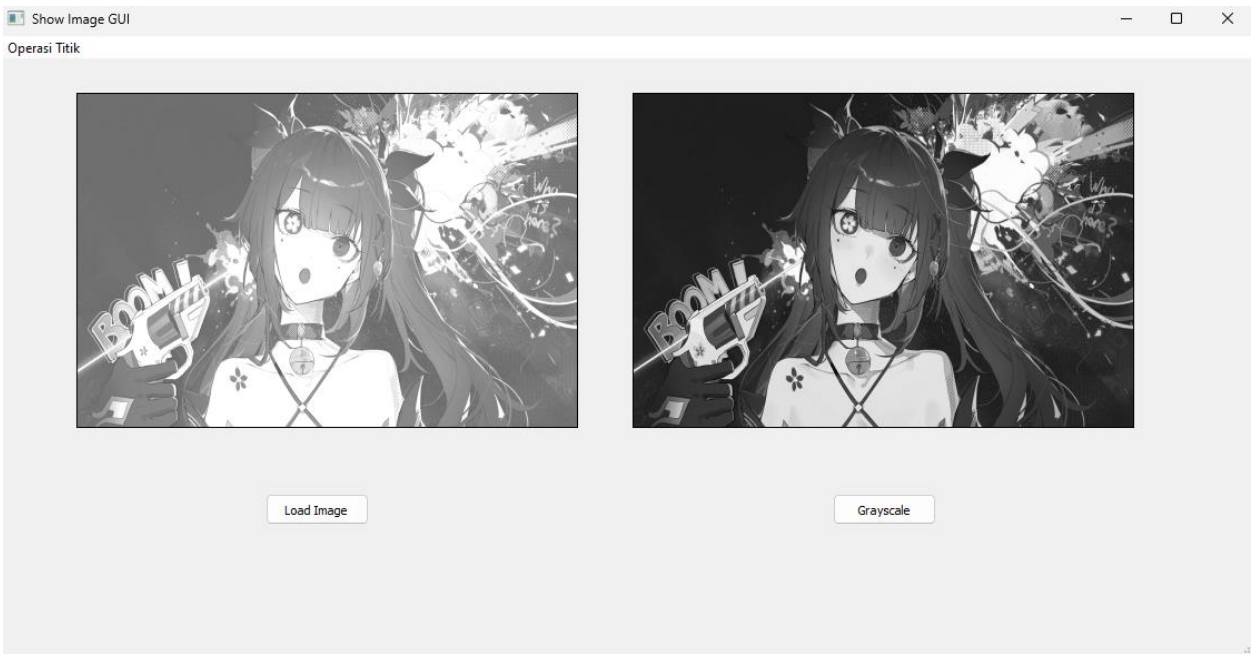
```
1 usage
def loadImage(self, flname):
    self.image = cv2.imread(flname)
    self.displayImage()

3 usages
def displayImage(self):
    qformat = QImage.Format_Indexed8

    if len(self.image.shape) == 3:  # row[0], col[1], channel[2]
        if self.image.shape[2] == 4:
            qformat = QImage.Format_RGBA8888
        else:
            qformat = QImage.Format_RGB888

    img = QImage(self.image.data, self.image.shape[1], self.image.shape[0], self.image.strides[0], qformat)
    img = img.rgbSwapped()

    self.imgLabel.setPixmap(QPixmap.fromImage(img))
    self.imgLabel.setAlignment(Qt.AlignHCenter | Qt.AlignVCenter)
```

```
1 usage
def grayClicked(self):
            H, W = self.image.shape[:2]
            gray = np.zeros( shape: (H, W), np.uint8)
            for i in range(H):
                for j in range(W):
                    gray[i, j] = np.clip(
                        0.299 * self.image[i, j, 0] + 0.587 * self.image[i, j, 1] + 0.114 * self.image[i, j, 2],  a_min: 0,
                        a_max: 255)
            self.image = gray
            self.displayImage(2)

1 usage
def brightness(self):
    try:
        self.image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)
    except:
        pass
```

```
58
59          H, W = self.image.shape[:2]
60          brightness = 80
61          for i in range(H):
62              for j in range(W):
63                  a = self.image.item(i, j)
64                  b = np.clip(a + brightness, a_min: 0, a_max: 255)
65                  self.image.itemset((i, j), b)
66
67          self.displayImage(1)
68
        3 usages
69      def displayImage(self, window=1):
70          qformat = QImage.Format_Indexed8
71          if len(self.image.shape) == 3:
72              if self.image.shape[2] == 4:
73                  qformat = QImage.Format_RGBA8888
74              else:
75                  qformat = QImage.Format_RGB888
76          img = QImage(self.image, self.image.shape[1], self.image.shape[0], self.image.strides[0], qformat)
77          img = img.rgbSwapped()
78
```

```
78
79          if window == 1:
80              self.imgLabel.setPixmap(QPixmap.fromImage(img))
81              self.imgLabel.setAlignment(QtCore.Qt.AlignHCenter | QtCore.Qt.AlignVCenter)
82              self.imgLabel.setScaledContents(True)
83
84          if window == 2:
85              self.hasilLabel.setPixmap(QPixmap.fromImage(img))
86              self.hasilLabel.setAlignment(QtCore.Qt.AlignHCenter | QtCore.Qt.AlignVCenter)
87              self.hasilLabel.setScaledContents(True)
88
89  app = QtWidgets.QApplication(sys.argv)
90  window = ShowImage()
91  window.setWindowTitle('Show Image GUI')
92  window.show()
93  sys.exit(app.exec_())
```


Show Image GUI
Operasi Titik
BOOM!
BOOM!
Load Image
Grayscale

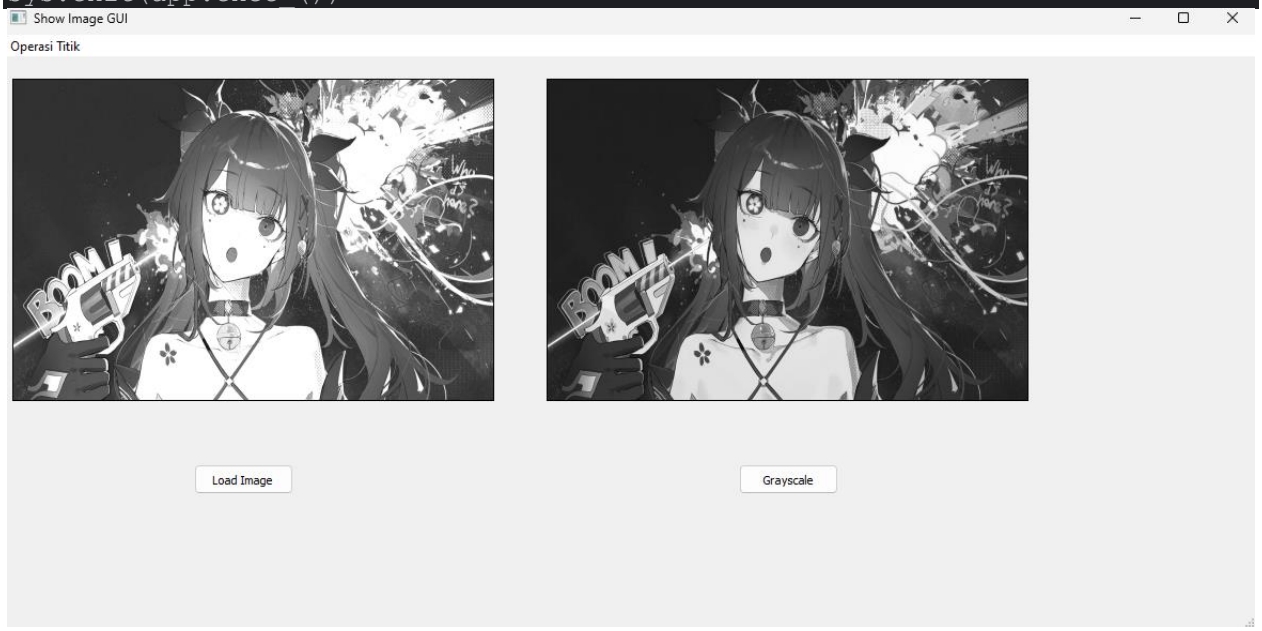

Show Image GUI
Operasi Titik
BOOM!
BOOM!
Load Image
Grayscale

A5

```
import cv2
import sys
import numpy as np
from PyQt5 import QtCore, QtWidgets
from PyQt5.QtCore import pyqtSlot, Qt
from PyQt5.QtGui import QImage, QPixmap
from PyQt5.QtWidgets import QMainWindow, QMessageBox
from PyQt5.uic import loadUi

class ShowImage(QMainWindow):
    def __init__(self):
        super(ShowImage, self).__init__()
        loadUi('showgui.ui', self)
        self.image = None
        self.loadButton.clicked.connect(self.loadClicked)
        self.grayButton.clicked.connect(self.grayClicked)
        self.actionOperasi_Pencerahan.triggered.connect(self.brightness)
        self.actionSimple_Contrast.triggered.connect(self.contrast)

    @pyqtSlot()
    def loadClicked(self):
        self.loadImage('BANG.jpeg')

    def loadImage(self, flname):
        self.image = cv2.imread(flname)
        self.displayImage()

    def displayImage(self):
        qformat = QImage.Format_Indexed8

        if len(self.image.shape) == 3:  # row[0], col[1], channel[2]
            if self.image.shape[2] == 4:
                qformat = QImage.Format_RGBA8888
            else:
                qformat = QImage.Format_RGB888

        img = QImage(self.image.data, self.image.shape[1],
self.image.shape[0], self.image.strides[0], qformat)
        img = img.rgbSwapped()

        self.imgLabel.setPixmap(QPixmap.fromImage(img))
        self.imgLabel.setAlignment(Qt.AlignHCenter | Qt.AlignVCenter)

    def grayClicked(self):
                H, W = self.image.shape[:2]
                gray = np.zeros((H, W), np.uint8)
                for i in range(H):
                    for j in range(W):
                        gray[i, j] = np.clip(
                            0.299 * self.image[i, j, 0] + 0.587 *
self.image[i, j, 1] + 0.114 * self.image[i, j, 2], 0,
                            255)
                self.image = gray
                self.displayImage(2)
```

```
    def brightness(self):
        try:
            self.image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)
        except:
            pass

        H, W = self.image.shape[:2]
        brightness = 50
        for i in range(H):
            for j in range(W):
                a = self.image.item(i, j)
                b = np.clip(a + brightness, 0, 255)
                self.image.itemset((i, j), b)

        self.displayImage(1)

    def contrast(self):
        try:
            self.image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)
        except:
            pass

        H, W = self.image.shape[:2]
        contrast = 1.6
        for i in range(H):
            for j in range(W):
                a = self.image.item(i, j)
                b = np.clip(a * contrast, 0, 255)
                self.image.itemset((i, j), b)

        self.displayImage(1)
    def displayImage(self, window=1):
        qformat = QImage.Format_Indexed8
        if len(self.image.shape) == 3:
            if self.image.shape[2] == 4:
                qformat = QImage.Format_RGBA8888
            else:
                qformat = QImage.Format_RGB888
        img = QImage(self.image, self.image.shape[1], self.image.shape[0],
self.image.strides[0], qformat)
        img = img.rgbSwapped()

        if window == 1:
            self.imgLabel.setPixmap(QPixmap.fromImage(img))
            self.imgLabel.setAlignment(QtCore.Qt.AlignHCenter |
QtCore.Qt.AlignVCenter)
            self.imgLabel.setScaledContents(True)

        if window == 2:
            self.hasilLabel.setPixmap(QPixmap.fromImage(img))
            self.hasilLabel.setAlignment(QtCore.Qt.AlignHCenter |
QtCore.Qt.AlignVCenter)
            self.hasilLabel.setScaledContents(True)

app = QtWidgets.QApplication(sys.argv)
window = ShowImage()
```

```
window.setWindowTitle('Show Image GUI')
window.show()
sys.exit(app.exec_())
```

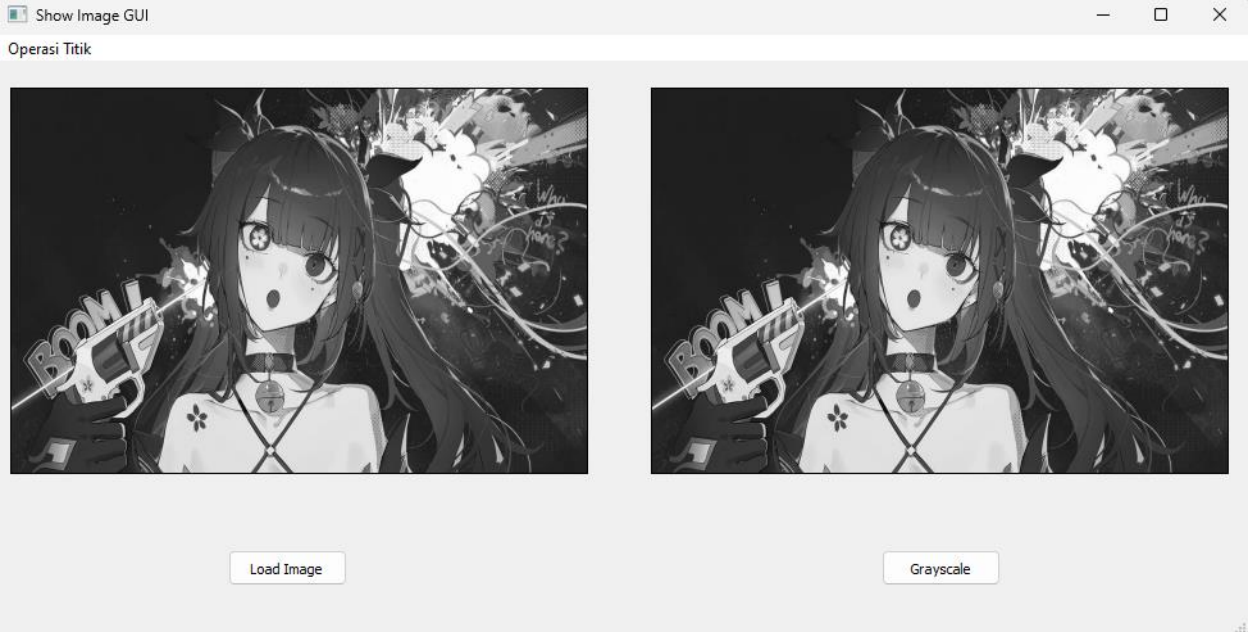


A6

```
import cv2
import sys
import numpy as np
from PyQt5 import QtCore, QtWidgets
from PyQt5.QtCore import pyqtSlot, Qt
from PyQt5.QtGui import QImage, QPixmap
from PyQt5.QtWidgets import QMainWindow, QMessageBox
from PyQt5.uic import loadUi

class ShowImage(QMainWindow):
    def __init__(self):
        super(ShowImage, self).__init__()
        loadUi('showgui.ui', self)
        self.image = None
        self.loadButton.clicked.connect(self.loadClicked)
        self.grayButton.clicked.connect(self.grayClicked)
        self.actionOperasi_Pencerahan.triggered.connect(self.brightness)
        self.actionSimple_Contrast.triggered.connect(self.contrast)

self.actionContrast_Streching.triggered.connect(self.contrastStreching)

    @pyqtSlot()
    def loadClicked(self):
        self.loadImage('BANG.jpeg')

    def loadImage(self, flname):
        self.image = cv2.imread(flname)
        self.displayImage()
```

```
    def displayImage(self):
        qformat = QImage.Format_Indexed8

        if len(self.image.shape) == 3:  # row[0], col[1], channel[2]
            if self.image.shape[2] == 4:
                qformat = QImage.Format_RGBA8888
            else:
                qformat = QImage.Format_RGB888

        img = QImage(self.image.data, self.image.shape[1],
self.image.shape[0], self.image.strides[0], qformat)
        img = img.rgbSwapped()

        self.imgLabel.setPixmap(QPixmap.fromImage(img))
        self.imgLabel.setAlignment(Qt.AlignHCenter | Qt.AlignVCenter)

    def grayClicked(self):
                H, W = self.image.shape[:2]
                gray = np.zeros((H, W), np.uint8)
                for i in range(H):
                    for j in range(W):
                        gray[i, j] = np.clip(
                            0.299 * self.image[i, j, 0] + 0.587 *
self.image[i, j, 1] + 0.114 * self.image[i, j, 2], 0,
                            255)
                self.image = gray
                self.displayImage(2)

    def brightness(self):
        try:
            self.image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)
        except:
            pass

        H, W = self.image.shape[:2]
        brightness = 50
        for i in range(H):
            for j in range(W):
                a = self.image.item(i, j)
                b = np.clip(a + brightness, 0, 255)
                self.image.itemset((i, j), b)

        self.displayImage(1)

    def contrast(self):
        try:
            self.image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)
        except:
            pass

        H, W = self.image.shape[:2]
        contrast = 1.6
        for i in range(H):
            for j in range(W):
                a = self.image.item(i, j)
                b = np.clip(a * contrast, 0, 255)
                self.image.itemset((i, j), b)
```

```
        self.displayImage(1)

    def contrastStreching(self):
        try:
            self.image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)
        except:
            pass

        H, W = self.image.shape[:2]
        minV = np.min(self.image)
        maxV = np.max(self.image)

        for i in range(H):
            for j in range(W):
                a = self.image.item(i, j)
                b = float(a - minV) / (maxV - minV) * 255
                self.image.itemset((i, j), b)

        self.displayImage(1)

    def displayImage(self, window=1):
        qformat = QImage.Format_Indexed8
        if len(self.image.shape) == 3:
            if self.image.shape[2] == 4:
                qformat = QImage.Format_RGBA8888
            else:
                qformat = QImage.Format_RGB888
        img = QImage(self.image, self.image.shape[1], self.image.shape[0],
self.image.strides[0], qformat)
        img = img.rgbSwapped()

        if window == 1:
            self.imgLabel.setPixmap(QPixmap.fromImage(img))
            self.imgLabel.setAlignment(QtCore.Qt.AlignHCenter |
QtCore.Qt.AlignVCenter)
            self.imgLabel.setScaledContents(True)

        if window == 2:
            self.hasilLabel.setPixmap(QPixmap.fromImage(img))
            self.hasilLabel.setAlignment(QtCore.Qt.AlignHCenter |
QtCore.Qt.AlignVCenter)
            self.hasilLabel.setScaledContents(True)

app = QtWidgets.QApplication(sys.argv)
window = ShowImage()
window.setWindowTitle('Show Image GUI')
window.show()
sys.exit(app.exec_())
```

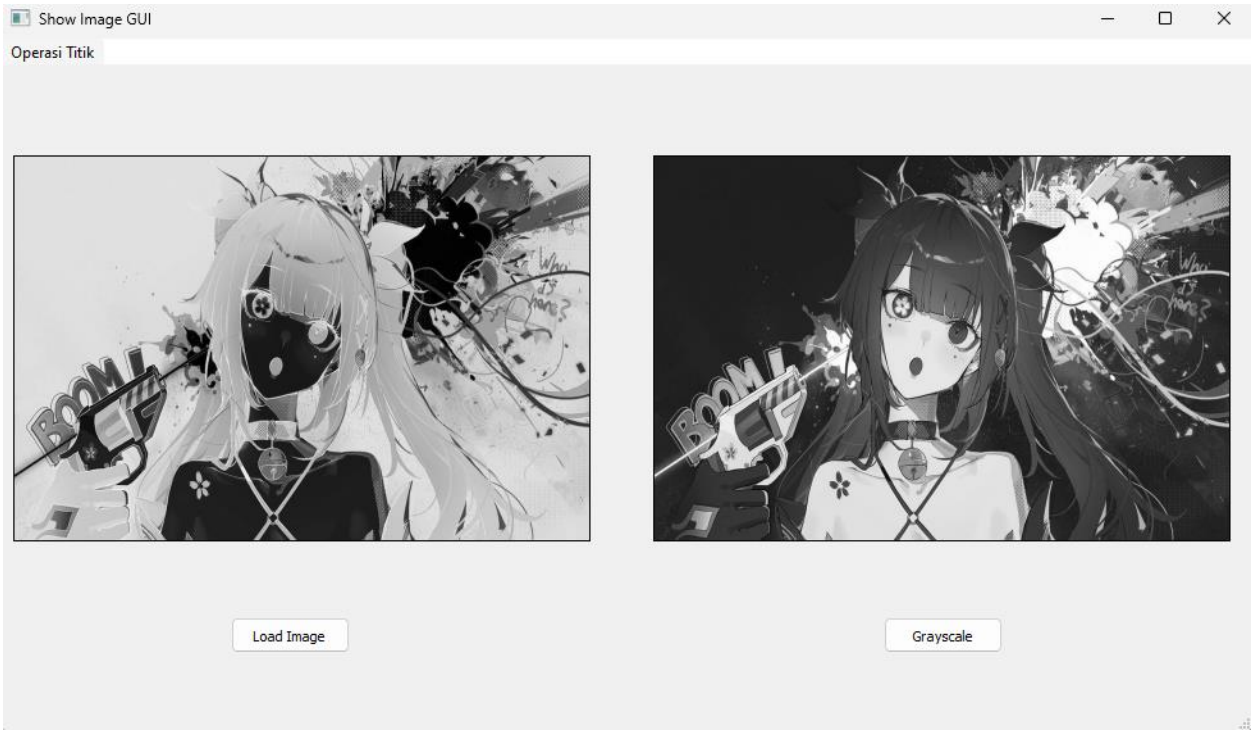


A7

```
import cv2
import sys
import numpy as np
from PyQt5 import QtCore, QtWidgets
from PyQt5.QtCore import pyqtSlot, Qt
from PyQt5.QtGui import QImage, QPixmap
from PyQt5.QtWidgets import QMainWindow, QMessageBox
from PyQt5.uic import loadUi

class ShowImage(QMainWindow):
    def __init__(self):
        super(ShowImage, self).__init__()
        loadUi('showgui.ui', self)
        self.image = None
        self.loadButton.clicked.connect(self.loadClicked)
        self.grayButton.clicked.connect(self.grayClicked)
        self.actionOperasi_Pencerahan.triggered.connect(self.brightness)
        self.actionSimple_Contrast.triggered.connect(self.contrast)

self.actionContrast_Streching.triggered.connect(self.contrastStreching)
        self.actionNegative_Image.triggered.connect(self.negativeImage)

    @pyqtSlot()
    def loadClicked(self):
        self.loadImage('BANG.JPEG')

    def loadImage(self, flname):
        self.image = cv2.imread(flname)
        self.displayImage()
```

```
    def displayImage(self):
        qformat = QImage.Format_Indexed8

        if len(self.image.shape) == 3:  # row[0], col[1], channel[2]
            if self.image.shape[2] == 4:
                qformat = QImage.Format_RGBA8888
            else:
                qformat = QImage.Format_RGB888

        img = QImage(self.image.data, self.image.shape[1],
self.image.shape[0], self.image.strides[0], qformat)
        img = img.rgbSwapped()

        self.imgLabel.setPixmap(QPixmap.fromImage(img))
        self.imgLabel.setAlignment(Qt.AlignHCenter | Qt.AlignVCenter)

    def grayClicked(self):
                H, W = self.image.shape[:2]
                gray = np.zeros((H, W), np.uint8)
                for i in range(H):
                    for j in range(W):
                        gray[i, j] = np.clip(
                            0.299 * self.image[i, j, 0] + 0.587 *
self.image[i, j, 1] + 0.114 * self.image[i, j, 2], 0,
                            255)
                self.image = gray
                self.displayImage(2)

    def brightness(self):
        try:
            self.image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)
        except:
            pass

        H, W = self.image.shape[:2]
        brightness = 50
        for i in range(H):
            for j in range(W):
                a = self.image.item(i, j)
                b = np.clip(a + brightness, 0, 255)
                self.image.itemset((i, j), b)

        self.displayImage(1)

    def contrast(self):
        try:
            self.image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)
        except:
            pass

        H, W = self.image.shape[:2]
        contrast = 1.6
        for i in range(H):
            for j in range(W):
                a = self.image.item(i, j)
                b = np.clip(a * contrast, 0, 255)
                self.image.itemset((i, j), b)
```

```
        self.displayImage(1)

    def contrastStreching(self):
        try:
            self.image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)
        except:
            pass

        H, W = self.image.shape[:2]
        minV = np.min(self.image)
        maxV = np.max(self.image)

        for i in range(H):
            for j in range(W):
                a = self.image.item(i, j)
                b = float(a - minV) / (maxV - minV) * 255
                self.image.itemset((i, j), b)

        self.displayImage(1)

    def negativeImage(self):
        if self.image is not None:
            negative_img = 255 - self.image
            self.image = negative_img
            self.displayImage()

    def displayImage(self, window=1):
        qformat = QImage.Format_Indexed8
        if len(self.image.shape) == 3:
            if self.image.shape[2] == 4:
                qformat = QImage.Format_RGBA8888
            else:
                qformat = QImage.Format_RGB888
        img = QImage(self.image, self.image.shape[1], self.image.shape[0],
self.image.strides[0], qformat)
        img = img.rgbSwapped()

        if window == 1:
            self.imgLabel.setPixmap(QPixmap.fromImage(img))
            self.imgLabel.setAlignment(QtCore.Qt.AlignHCenter |
QtCore.Qt.AlignVCenter)
            self.imgLabel.setScaledContents(True)

        if window == 2:
            self.hasilLabel.setPixmap(QPixmap.fromImage(img))
            self.hasilLabel.setAlignment(QtCore.Qt.AlignHCenter |
QtCore.Qt.AlignVCenter)
            self.hasilLabel.setScaledContents(True)

app = QtWidgets.QApplication(sys.argv)
window = ShowImage()
window.setWindowTitle('Show Image GUI')
window.show()
sys.exit(app.exec_())
```


Show Image GUI
Operasi Titik
BOOM!
BOOM!
Load Image
Grayscale