Laboratory Activity No. 5	
Introduction to Event Handling in GUI Development	
Course Code: CPE009	**Program:** BSCPE
Course Title: Object-Oriented Programming	**Date Performed:**
Section:	**Date Submitted:**
Name:	**Instructor:**

1. Objective(s):

This activity aims to familiarize students on how to implement event handling in a GUI Application.

2. Intended Learning Outcomes (ILOs):

The students should be able to:
2.1 Identify the main components in a GUI Application
2.2 Create a simple GUI Application using PyQt5 Widgets

3. Discussion:

A Graphical User Interface (GUI) application is a program that the user can interact with through graphics (windows, buttons, text fields, checkboxes, images, icons, etc..) such as the Desktop GUI of Windows OS by using a mouse and keyboard unlike with a Command-line program or Terminal program that support keyboard inputs only.

In PyQt, event handling is composed of a signal, and a slot PyQt5 widget(buttons, labels, comboBox) which can emit a signal, an alert that something has happened for example: The button was clicked, label was hovered, a selection was made in the combo box. Signals do not do anything on their own, but they can be connected to functions where codes can be called once the signal occurs. So the signal will be connected to a slot (PyQt5 Widget).

A Signal is emitted when something of potential interest happens. A slot is a Python callable. If a signal is connected to a slot then the slot is called when the signal is emitted. If a signal is not connected, then nothing happens. The code (or component) that emits the signal does not know or care if the signal is being used.

The signal/slot mechanism has the following features.

- A signal may be connected to many slots.
- A signal may also be connected to another signal.
- Signal arguments may be any Python type.
- A slot may be connected to many signals.
- Connections may be direct (ie. synchronous) or queued (ie. asynchronous).
- Connections may be made across threads.
- Signals may be disconnected.

Source: PyQt5 Documentation https://www.riverbankcomputing.com/static/Docs/PyQt5/signals_slots.html

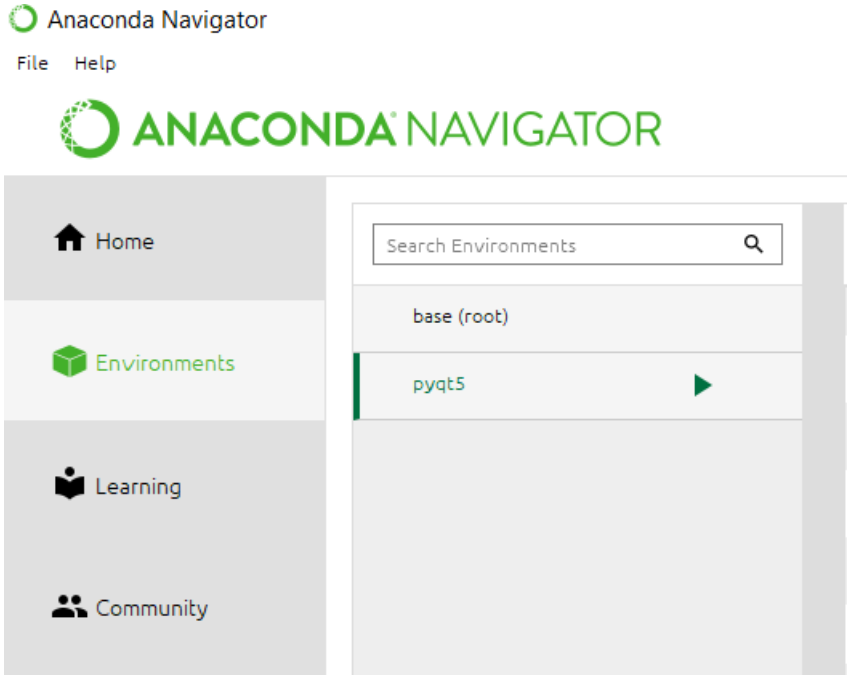
4. Materials and Equipment:

Desktop Computer with Anaconda Python
Windows Operating System

5. Procedure:

Using the pyqt5 Virtual Environment

1. Open the Anaconda Navigator and go to Environments
2. Select the pyqt5 virtual environment that you created.



Event Handling

1. Create a folder named oopfa1<lastname>_lab9
2. Open your Anaconda Navigator and select Visual Studio Code or Spyder IDE.
3. Open that folder in your editor and create a file named **gui_buttonclicked.py** then copy the code as shown:

```python
import sys
from PyQt5.QtWidgets import QWidget,QApplication, QMainWindow, QPushButton
from PyQt5.QtGui import QIcon

class App(QWidget):

    def __init__(self):
        super().__init__() # initializes the main window like in the previous one
        # window = QMainWindow()
        self.title= "PyQt Button"
        self.x=200 # or left
        self.y=200 # or top
        self.width=300
        self.height=300
        self.initUI()

    def initUI(self):
        self.setWindowTitle(self.title)
        self.setGeometry(self.x,self.y,self.width,self.height)
        self.setWindowIcon(QIcon('pythonico.ico'))

        # In GUI Python, these buttons, textboxes, labels are called Widgets
        self.button = QPushButton('Click me!', self)
        self.button.setToolTip("You've hovered over me!")
        self.button.move(100,70)  # button.move(x,y)

        self.show()


if __name__ == '__main__':
    app = QApplication(sys.argv)
    ex = App()
    sys.exit(app.exec_())
```

1. Run the program and observe the output.
2. Add an import to the top. **from PyQt5.QtCore import pyqtSlot**
3. Add a new the following code:

```
    def initUI(self):
        self.setWindowTitle(self.title)
        self.setGeometry(self.x,self.y,self.width,self.height)
        self.setWindowIcon(QIcon('pythonico.ico'))

        # In GUI Python, these buttons, textboxes, labels are called Widgets
        self.button = QPushButton('Click me!', self)
        self.button.setToolTip("You've hovered over me!")
        self.button.move(100,70)  # button.move(x,y)
        self.button.clicked.connect(self.on_click)

        self.show()

    @pyqtSlot()
    def on_click(self):
        print('You clicked me!')

if __name__ == '__main__':
    app = QApplication(sys.argv)
    ex = App()
    sys.exit(app.exec_())
```

4. Run the code and observe the output at the terminal.

Adding a Message Box

1. Copy the gui_buttonclicked.py and place the contents in a new file named gui_messagebox.py and modify the on_click method with the code shown:

```
@pyqtSlot()
def clickMe(self):
    buttonReply = QMessageBox.question(self,"Testing Response","Do you like PyQt5?",
                                       QMessageBox.Yes | QMessageBox.No, QMessageBox.Yes)
    if buttonReply == QMessageBox.Yes:
        QMessageBox.warning(self, "Evaluation", "User clicked Yes", QMessageBox.Ok,QMessageBox.Ok)

    else:
        QMessageBox.information(self, "Evaluation", "User clicked No", QMessageBox.Ok, QMessageBox.Ok)
```

2. Run the program and observe the output when the button is clicked.

6. Supplementary Activity:

Task

Using the simple Account Registration system created in the previous laboratory activity. Create the functionality that will register the account by saving it to a Database using sqlitedict or a .csv or .txt file.

The GUI program should not allow the registration to proceed if there is a field that has an empty value and notify of the missing values using a message box. Once the registration is successful a message should appear that would inform the user that the registration was successful. Use the appropriate symbol in making the message box.

Questions

1. What are the other signals available in PyQt5? (give at least 3 and describe each)

   _____
   _____

2. Why do you think that event handling in Python is divided into signals and slots?

   _____
   _____

3. How can message boxes be used to provide a better User Experience or how can message boxes be used to make a GUI Application more user-friendly?

   _____
   _____

4. What is Error-handling and how was it applied in the task performed?

   _____
   _____

5. What maybe the reasons behind the need to implement error handling?

   _____
   _____

7. Conclusion:

8. Assessment Rubric: